CICS® Transaction Server for OS/390®

# CICS RACF Security Guide

*Release 3*

SC33-1701-33

CICS® Transaction Server for OS/390®

# CICS RACF Security Guide

*Release 3*

SC33-1701-33

**Note!**
Before using this information and the product it supports, be sure to read the general information under "Notices" on page ix.

**Third edition (November 2000)**

This edition applies to Release 3 of CICS Transaction Server for OS/390, program number 5655-147, and to all subsequent versions, releases, and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

This book is based on the CICS RACF Security Guide for CICS Transaction Server for OS/390 Release 2, SC33-1701-33. Changes from that edition are marked by vertical lines to the left of the changes.

The CICS Transaction Server for OS/390 Release 2 edition remains applicable and current for users of CICS Transaction Server for OS/390 Release 2, and may be ordered using its order number, SC33-1701-33.

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

At the back of this publication is a page entitled "Sending your comments to IBM". If you want to make comments, but the methods described are not available to you, please address them to:

IBM United Kingdom Laboratories, Information Development,
Mail Point 095, Hursley Park, Winchester, Hampshire, England, SO21 2JN.

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

# Contents

# Notices

This information was developed for products and services offered in the U.S.A. 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 (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

**The following paragraph does not apply in 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 publication 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.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM United Kingdom Laboratories, MP151, Hursley Park, Winchester, Hampshire, England, SO21 2JN. Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

## Programming interface information

This book is intended to help you use the IBM Resource Access Control Facility to provide security for CICS.

This book documents General-use Programming Interface and Associated Guidance Information provided by CICS.

General-use Programming Interfaces allow the customer to write programs that obtain the services of CICS. General-use Programming Interface and Associated Guidance Information is identified where it occurs, as follows:

**General-use programming interface information**

...

**End of General-use programming interface information**

General-use programming interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service.

Product-sensitive programming Interface and Associated Guidance Information is also provided.

Product-sensitive programming interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of CICS. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-sensitive programming interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it it to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service.

Product-sensitive programming Interface and Associated Guidance Information is identified where it occurs, as follows:

**Product-sensitive Programming Interface information**

...

**End of Product-sensitive Programming Interface information**

## Trademarks

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

| | | |
|---|---|---|
| AIX | CICS/VSE | OS/2 |
| AT | DB2 | OS/390 |

| | | |
|---|---|---|
| BookManager | ESA/390 | OpenEdition |
| CICS | IBM | RACF |
| CICS OS/2 | IMS | S/370 |
| CICS/ESA | MVS/ESA | SP |
| CICS/MVS | MVS/XA | VSE/ESA |
| CICS/VM | NetView | VTAM |

Windows NT is a trademark of Microsoft Corporation in the United States, or other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

# Preface

## What this book is about

This book is about using the IBM® Resource Access Control Facility (RACF) to provide security for CICS.

### Who this book is for

This book is intended for security administrators responsible for controlling access to resources used by CICS®. These resources are used by CICS terminals, users, or transactions in CICS regions, and by CICS application programs running in those regions. The book will also be of interest for CICS system programmers who may need to communicate their requirements to the security administrator for their installation.

## What you need to know to understand this book

It is assumed that you have a good working knowledge of RACF® facilities. It is also assumed that you know something about the types of resource owned and controlled by CICS.

Although this book shows many RACF command examples, it assumes that you have access to the *OS/390 Security Server (RACF) Security Administrator's Guide* and that you know how to issue TSO commands (or use ISPF panels to perform equivalent functions).

## How to use this book

The parts and chapters of this book are self-contained. Use an individual part or chapter where it contains information about the particular task you are engaged in. For example, see "Part 4. Customization" on page 219 if your task is to customize your CICS security processing.

### Determining if a publication is current

IBM regularly updates its publications with new and changed information. When first published, both hardcopy and BookManager softcopy versions of a publication are usually in step. However, due to the time required to print and distribute hardcopy books, the BookManager version is more likely to have had last-minute changes made to it before publication.

Subsequent updates will probably be available in softcopy before they are available in hardcopy. This means that at any time from the availability of a release, softcopy versions should be regarded as the most up-to-date.

For CICS Transaction Server books, these softcopy updates appear regularly on the *Transaction Processing and Data Collection Kit* CD-ROM, SK2T-0730-xx. Each reissue of the collection kit is indicated by an updated order number suffix (the -xx part). For example, collection kit SK2T-0730-06 is more up-to-date than SK2T-0730-05. The collection kit is also clearly dated on the cover.

Updates to the softcopy are clearly marked by revision codes (usually a "#" character) to the left of the changes.

## Notes on terminology

In general, this book uses the term CICS without qualification to refer to the CICS element of CICS Transaction Server for OS/390. However, when it is necessary to distinguish between particular CICS versions, we use the following abbreviations:

**"CICS/OS/VS"**
is used for IBM Customer Information Control System/Operating System/Virtual Storage.

**"CICS/MVS®"**
is used for IBM Customer Information Control System/Multiple Virtual Storage.

**"CICS/ESA®"**
is used for IBM Customer Information Control System/Enterprise Systems Architecture.

Other abbreviations for CICS releases used in this book are as follows:

- For CICS/OS/VS Version 1 Release 7-CICS/OS/VS 1.7
- For CICS/MVS Version 2 Release 1 and subsequent modification levels -CICS/MVS 2.1
- For CICS/MVS Version 3 Release 1.1-CICS/MVS 3.1.1
- For CICS/ESA Version 3 Release 2.1-CICS/ESA 3.2.1
- For CICS/ESA Version 3 Release 3 -CICS/ESA 3.3
- For CICS/ESA Version 4 Release 1 -CICS/ESA 4.1
- For CICS Transaction Server for OS/390® Version 1 Release 1-CTS 1.1

"RACF" is used for Resource Access Control Facility.

- "RACF 2.1" refers to RACF Version 2 Release 1.
- "RACF 2.2" refers to RACF Version 2 Release 2.

"RACF" refers to any supported release of Version 2.1 or higher.

"RACF 2.1", or "RACF 2.2" refers to content specific to that release or later.

"MVS" is used for the operating system, which can be either an element of OS/390 or MVS/Enterprise System Architecture System Product (MVS/ESA SP).

For definitions of security-related CICS and RACF terms used in this book, see "Glossary" on page 331.

# CICS Transaction Server for OS/390

| | |
|---|---|
| *CICS Transaction Server for OS/390: Planning for Installation* | GC33-1789 |
| *CICS Transaction Server for OS/390 Release Guide* | GC34-5352 |
| *CICS Transaction Server for OS/390 Migration Guide* | GC34-5353 |
| *CICS Transaction Server for OS/390 Program Directory* | GI10-2506 |
| *CICS Transaction Server for OS/390 Licensed Program Specification* | GC33-1707 |

# CICS books for CICS Transaction Server for OS/390

| | |
|---|---|
| **General** | |
| *CICS Master Index* | SC33-1704 |
| *CICS User's Handbook* | SX33-6104 |
| *CICS Transaction Server for OS/390 Glossary* (softcopy only) | GC33-1705 |
| **Administration** | |
| *CICS Transaction Server for OS/390 Installation Guide* | GC33-1681 |
| *CICS System Definition Guide* | SC33-1682 |
| *CICS Customization Guide* | SC33-1683 |
| *CICS Resource Definition Guide* | SC33-1684 |
| *CICS Operations and Utilities Guide* | SC33-1685 |
| *CICS Supplied Transactions* | SC33-1686 |
| **Programming** | |
| *CICS Application Programming Guide* | SC33-1687 |
| *CICS Application Programming Reference* | SC33-1688 |
| *CICS System Programming Reference* | SC33-1689 |
| *CICS Front End Programming Interface User's Guide* | SC33-1692 |
| *CICS C++ OO Class Libraries* | SC34-5455 |
| *CICS Distributed Transaction Programming Guide* | SC33-1691 |
| *CICS Business Transaction Services* | SC34-5268 |
| **Diagnosis** | |
| *CICS Problem Determination Guide* | GC33-1693 |
| *CICS Messages and Codes* | GC33-1694 |
| *CICS Diagnosis Reference* | LY33-6088 |
| *CICS Data Areas* | LY33-6089 |
| *CICS Trace Entries* | SC34-5446 |
| *CICS Supplementary Data Areas* | LY33-6090 |
| **Communication** | |
| *CICS Intercommunication Guide* | SC33-1695 |
| *CICS Family: Interproduct Communication* | SC33-0824 |
| *CICS Family: Communicating from CICS on System/390* | SC33-1697 |
| *CICS External Interfaces Guide* | SC33-1944 |
| *CICS Internet Guide* | SC34-5445 |
| **Special topics** | |
| *CICS Recovery and Restart Guide* | SC33-1698 |
| *CICS Performance Guide* | SC33-1699 |
| *CICS IMS Database Control Guide* | SC33-1700 |
| *CICS RACF Security Guide* | SC33-1701 |
| *CICS Shared Data Tables Guide* | SC33-1702 |
| *CICS Transaction Affinities Utility Guide* | SC33-1777 |
| *CICS DB2 Guide* | SC33-1939 |

## CICSPlex SM books for CICS Transaction Server for OS/390

| | |
|---|---|
| **General** | |
| *CICSPlex SM Master Index* | SC33-1812 |
| *CICSPlex SM Concepts and Planning* | GC33-0786 |
| *CICSPlex SM User Interface Guide* | SC33-0788 |
| *CICSPlex SM View Commands Reference Summary* | SX33-6099 |
| **Administration and Management** | |
| *CICSPlex SM Administration* | SC34-5401 |
| *CICSPlex SM Operations Views Reference* | SC33-0789 |
| *CICSPlex SM Monitor Views Reference* | SC34-5402 |
| *CICSPlex SM Managing Workloads* | SC33-1807 |
| *CICSPlex SM Managing Resource Usage* | SC33-1808 |
| *CICSPlex SM Managing Business Applications* | SC33-1809 |
| **Programming** | |
| *CICSPlex SM Application Programming Guide* | SC34-5457 |
| *CICSPlex SM Application Programming Reference* | SC34-5458 |
| **Diagnosis** | |
| *CICSPlex SM Resource Tables Reference* | SC33-1220 |
| *CICSPlex SM Messages and Codes* | GC33-0790 |
| *CICSPlex SM Problem Determination* | GC33-0791 |

## Other CICS books

| | |
|---|---|
| *CICS Application Programming Primer (VS COBOL II)* | SC33-0674 |
| *CICS Application Migration Aid Guide* | SC33-0768 |
| *CICS Family: API Structure* | SC33-1007 |
| *CICS Family: Client/Server Programming* | SC33-1435 |
| *CICS Family: General Information* | GC33-0155 |
| *CICS 4.1 Sample Applications Guide* | SC33-1173 |
| *CICS/ESA 3.3 XRF Guide* | SC33-0661 |

If you have any questions about the CICS Transaction Server for OS/390 library, see *CICS Transaction Server for OS/390: Planning for Installation* which discusses both hardcopy and softcopy books and the ways that the books can be ordered.

# Summary of changes

Further changes made to this book can be seen in the section "Authorizing access to named counter pools and servers" on page 50 in "Chapter 3. CICS data set and system security" on page 37.

A new section, , has been added to "Chapter 11. Security for CICS Web support" on page 137.

## Changes for CICS Transaction Server for OS/390 Release 3

Changes for this edition are indicated by vertical bars to the left of the changes.

"Chapter 4. Verifying CICS users" on page 65 has additional information about automatic preset security for consoles, and describes the use of the TSO CONSOLE command.

"Changing RACF profiles that are in use—caution" on page 156 contains additional information about temporary storage, including the use of temporary storage long queue names.

"Chapter 7. Surrogate user security" on page 103 describes CICS business transaction services (BTS) processes and activities.

In "Chapter 10. Security for CICS-supplied transactions" on page 125 additions have been made to the Category 1 transactions.

"Chapter 11. Security for CICS Web support" on page 137 discusses security considerations for the HTML template manager PDS, and the alias transaction, in addition to the requirements for the CICS Web interface transactions. This includes a section that describes the secure sockets layer (SSL).

"Chapter 14. APPC password expiration management" on page 173 contains information about the PEM sample program previously featured in an appendix of this manual. The external security interface (ESI) is also described here.

"Chapter 17. Security for data tables" on page 213 has an additional section discussing security for coupling facility data tables.

Many additions have been made to the table, "Appendix B. Resource and command check cross reference" on page 321

### Implementing RACF security for CICSPlex SM

A new part, "Part 7. CICSPlex SM security" on page 265 has been added to explain how to implement RACF security for CICSPlex SM. This information was previously available in the *CICSPlex SM Setup* book at the previous release. It contains the following chapters

- "Chapter 22. Implementing CICSPlex SM security" on page 267 explains how to implement RACF security for CICSPlex SM
- "Chapter 23. Invoking a user-supplied external security manager" on page 303 provides information on using a SAF-compliant external security manager other than RACF.

- "Chapter 24. Writing an API security exit" on page 307 describes how to write an API security exit and describes the role of the default security routine, EYU9XESV.
- "Chapter 25. Example tasks: security" on page 313 provides examples of typical security setup tasks that you can use as a model for your own.

## Changes for CICS Transaction Server for OS/390 Release 2

Changes for this edition are indicated by vertical bars to the left of the changes.

- The CICS DB2 attachment facility provides resource definition online (RDO) support for DB2 resources as an alternative to resource control table (RCT) definitions.
- "Appendix B. Resource and command check cross reference" on page 321 includes the EXEC CICS commands, and the relevant resource classes for the attachment facility.
- For information about the XDB2 system initialization parameter, see "Resource classes for DB2ENTRYs" on page 28, "Universal access authority for undefined terminals" on page 23, and "Defining your own resource class names" on page 33.

## Changes for CICS Transaction Server for OS/390 Release 1

References to RACF 1.9 have been removed because CICS Transaction Server for OS/390 Release 3 requires RACF 2.1.

A section has been added to "Chapter 3. CICS data set and system security" on page 37, explaining the security authorization checks to be used in connection with the temporary storage data sharing facility. See "Authorizing access to temporary storage pools and servers" on page 48, and "Access to temporary storage servers" on page 49.

A description is included on security checks that can be made on a region using an SMSVSAM server. See "Authorizing access to SMSVSAM servers" on page 52.

LOGSTRM processing as a resource class has been introduced. See "Chapter 3. CICS data set and system security" on page 37.

In "Chapter 7. Surrogate user security" on page 103, a section has been added about surrogate user checking and the external CICS interface.

Several changes have been made to the EXEC CICS COMMANDS and their resource checks in "Appendix B. Resource and command check cross reference" on page 321.

## Changes for CICS/ESA 4.1

- "Chapter 1. Security facilities in CICS" on page 3 had additional introductory information on the following:
  - Non-terminal security
  - Surrogate user security
  - MRO security
  - CICS/ESA Front End Programming Interface security
  - Generating and using RACF PassTickets
- "Chapter 2. RACF facilities" on page 9 was amended as follows:

  - The signon table (SNT) was removed. This change affected "CICS segment" on page 13, "Defining XRFSOFF" on page 15, and "CICS default user" on page 15.
  - Information about TIMEOUT data was included in the section on "CICS segment" on page 13.
  - "Generating and using RACF PassTickets" on page 8 was added.
  - The ability to define each user as belonging to several groups was mentioned in "Security classification of data and users" on page 21.
  - Situations in which it is necessary to use the PERFORM SECURITY REBUILD command were indicated in "Refreshing resource profiles in main storage" on page 27.
  - Specific information about refreshing resource profiles was added in several places.
- "Chapter 3. CICS data set and system security" on page 37 was amended as follows:
  - The section on console profiles discusses the security check that can be implemented on the console.
  - CICS/ESA 4.1 did not work with RACF versions before 1.9. Any mention of earlier versions of RACF were removed from "CICS-supplied RACF dynamic parse validation routines" on page 37.
  - "SEC" on page 56 was updated to reflect the fact the SEC system initialization parameter no longer supported MIGRATE.
  - The XUSER resource class for surrogate user checking was included in Table 5 on page 59.
- "Chapter 4. Verifying CICS users" on page 65 was changed in the following ways:
  - Mention of TCAM terminals was removed from "Controlling access to CICS from specific ports of entry" on page 68.
  - Surrogate user checking was mentioned in "Surrogate job submission in a CICS environment" on page 54.
  - The way CICS obtains information about users was reflected in "Obtaining CICS-related data for a user" on page 74.
  - Information about national languages, and non-terminal transactions appeared in "National language and non-terminal transactions" on page 77.
  - Details of CSGM and CESN were moved to *CICS Supplied Transactions*.
- "Chapter 6. Resource security" on page 85 included information about transactions not attached to terminals (see "Transactions started without terminals" on page 94). chapter.
- "Chapter 7. Surrogate user security" on page 103 was an additional chapter discussing when you use surrogate user checking.
- In "Chapter 8. CICS command security" on page 109, several new resources and their related CICS commands were added to Table 12 on page 110.
- SEC=MIGRATE is no longer supported, so references to this option were removed from "Chapter 10. Security for CICS-supplied transactions" on page 125.
- In "Chapter 13. Implementing LU6.2 security" on page 153, the following changes were made:
  - A new section described attach-time security processing and addition of SNA profile support. See "SNA profiles and attach-time security" on page 163.

  - Mention of internal bind-time security was removed. (See "Defining bind-time security" on page 155).
  - The use of ATTACHSEC(VERIFY) in addition to ATTACHSEC(IDENTIFY) in checking the user identifier was included in "Specifying user security in link definitions" on page 159.
- Information about CICS-APPC password expiration management, which previously appeared in a separate manual, was included in "Chapter 14. APPC password expiration management" on page 173.
- In "Chapter 16. Implementing MRO security" on page 199, the following changes were made:
  - "Bind-time security with MRO" on page 199 was reorganized and expanded because of the introduction of an external security manager and the cross-system coupling facility (XCF).
  - Information on the external call interface was included in "Distributed program link security with MRO" on page 210.
- "Chapter 17. Security for data tables" on page 213 was added to provide information on the security checks available when using SDT.
- In "Chapter 18. Customizing security processing" on page 221, the operation of the external security manager was described in the section, "Determining the userid of the CICS region" on page 225.
- "Chapter 21. Problem determination in a CICS-RACF security environment" on page 251 was updated to reflect the levels of RACF for which PERFORM SECURITY REBUILD is still necessary. Mention of RACF releases earlier than 1.9 were also removed.
- Changes were made in various chapters to reflect the replacement of several message numbers (for example, DFHXS0100 by DFHXS1111).
- "Appendix A. National Language" on page 319, provided information on language codes that could be defined to a user in the LANGUAGE segment of RACF.
- A new "Appendix B. Resource and command check cross reference" on page 321, was also added.

# Part 1. Introduction

This part introduces you to the subject of CICS security, using RACF as the CICS external security manager. It provides an overview of the CICS security requirements, and the facilities RACF provides to satisfy those requirements. Part 1 contains the following:

- "Chapter 1. Security facilities in CICS" on page 3 introduces you to the various aspects of CICS transaction and resource security.
- "Chapter 2. RACF facilities" on page 9 describes the basic facilities that RACF provides, and that CICS relies upon for its security administration.

# Chapter 1. Security facilities in CICS

This chapter describes, from the CICS viewpoint, the following aspects of CICS transaction and resource security:



## Why CICS needs security

Today, an unprecedented number of computer system users are completely dependent on their systems, and on the data managed by those systems. There are now terminals in many different locations in most organizations, and their use is commonplace. At the same time, easy-to-use, high-level inquiry languages are available, and there is much greater familiarity with data processing methods. This means that more and more people can use computers to retrieve or modify data stored within a computer system.

The speed, flexibility, and size of modern systems make large quantities of data accessible to many terminal users. As the systems become easier to use, there is also more scope for terminal users to gain access to confidential or valuable data.

Without a corresponding growth in awareness of good data security practices, these advances can result in accidental (or deliberate) data exposure. This means that your data can be subject to:

- Unauthorized access
- Disclosure
- Modification
- Destruction

As an online transaction-processing system (often supporting many thousands of terminals), CICS clearly needs the protection of a security system to ensure that the resources to which it manages access are protected, and are secure from unauthorized access.

To provide the necessary security for your CICS regions, CICS uses the MVS system authorization facility (SAF) to route authorization requests to an external security manager (ESM), such as RACF, at appropriate points within CICS transaction processing.

## What CICS security protects

Let us take a brief look at the assets that CICS manages, and potential exposures. The assets are the application programs, the application data, and the application output. To prevent disclosure, destruction, or corruption of these assets, you must first safeguard the CICS system components themselves.

There are two distinct areas from which exposures to the CICS system can arise. The first of these is from sources external to CICS. You can use RACF data set protection as the primary means of preventing unauthorized access, from either TSO users or batch jobs, to the assets CICS manages.

The other potential area of exposure arises from CICS users. CICS provides a variety of security and control mechanisms. These can limit the activities of CICS terminal users to only those functions that any particular individual user is authorized to use.

## What CICS security does not protect

CICS itself does **not** provide facilities to protect its own assets from external access. You should restrict access to the program libraries, to the CICS regions, and to those responsible for incorporating approved application and system changes. Similarly, the data sets and databases used by CICS and by CICS applications must be accessible only by approved batch processing and operations procedures.

CICS does not protect your system from application programs that use undocumented or unsupported interfaces to bypass CICS security. You are responsible for ensuring that such programs are not installed on your system.

CICS does not protect your application source libraries. You should ensure that procedures are established and followed that prevent the introduction of unauthorized or untested application programs into your “production” application base. You should also protect the integrity of your system by exercising control over libraries that are admitted to the system, and changes to those libraries.

## Terminal user security

To secure resources from unauthorized access, CICS needs some means of uniquely identifying individual users of the system. For this purpose, first define the users to RACF by creating an entry in the RACF database, referred to as a **user profile**. To identify themselves to CICS, users sign on by specifying their RACF user identification (userid) and the associated password, or operator identification card (OIDCARD) in the CICS-supplied sign-on transaction, CESN. Alternatively, they can use an equivalent transaction developed by your own installation by issuing the EXEC CICS SIGNON command provided for this purpose.

When users enter the CESN transaction, CICS verifies userids and passwords by a call to RACF. If the terminal user signon is valid, the CICS user domain keeps track of the signed-on user. Thereafter, CICS uses the information about the user when calling RACF to make authorization checks.

See “Terminal profiles” on page 22 for information about the terminal security facilities provided by RACF. See “Chapter 4. Verifying CICS users” on page 65 for information about using terminal user security in CICS.

## Preset terminal security

For some selected terminals, and MVS consoles when used as CICS terminals, consider using CICS preset terminal based security as an alternative to terminal user security. A terminal becomes a preset security terminal when you specify the USERID operand on the terminal definition.

CICS preset terminal security allows you to associate a userid permanently with a terminal that is defined to CICS. This means that CICS implicitly “signs on” the terminal when it is being installed, instead of the terminal being signed on subsequently. Preset security is often defined for devices without keyboards, such as printers, at which users cannot sign on.

You can also use this form of security on ordinary display terminals as an alternative to terminal user security. This permits anyone with physical access to a terminal with preset security to enter the transactions that are authorized for that terminal, without the need to sign on to CICS. The terminal remains signed on as long as it is installed, and no explicit sign-off can be performed against it. If the userid associated with a display terminal with preset security authorized to use any sensitive transactions, ensure that the terminal is in a secure location to which access is restricted. For example, terminals physically located within a CICS network control center might be appropriate for preset security.

You can use preset security to assign a userid with **lower** authority than the default, for terminals in unrestricted areas.

For example, to define a terminal with preset security, use RACF and CICS (CEDA) commands as follows:

```
ADDUSER userid NAME(preset_terminal_user_name) OWNER(owner_userid or group_id)
               DFLTGRP(group_name)
CEDA DEFINE TERMINAL(cics_termid) NETNAME(vtam_termid) USERID(userid)
            TYPETERM(cics_typeterm)
```

For further information on preset security terminals in the transaction routing environment refer to “Preset-security terminals and transaction routing” on page 166 (LU6.2 security) and “Preset-security terminals and transaction routing” on page 208 (MRO security).

## Non-terminal security

You can also specify security for transactions that are not associated with terminals. These are:

- Started non-terminal transactions
- Transient data trigger-level transactions
- Program List Table (PLT) programs that run during CICS initialization

For more information about non-terminal security, see “Transactions not associated with a terminal” on page 82.

## Transaction security

CICS facilities for transaction security ensures that CICS calls RACF each time a transaction is initiated, to verify that the userids associated with that transaction are permitted access to it.

See "General resource profiles" on page 25 for information about the resource classes that RACF supports for CICS transaction security. See "Chapter 5. Transaction security" on page 79 for information about using transaction security.

## CICS resource security

You can control access to CICS resources that a transaction uses. You do this by specifying YES on the resource security parameter, RESSEC, in the CICS TRANSACTION resource definition. These CICS resources can be:

- Application programs
- DL/I program specification blocks (PSBs)
- Files—VSAM and BDAM
- Journals
- Temporary storage queues
- Transient data queues
- Transactions initiated by a CICS START command

See "Chapter 6. Resource security" on page 85 for information about using CICS resource security.

## CICS command security

You can control security for a system programming subset of the CICS application programming interface (SPI) commands. You do this by specifying YES in the command security parameter, CMDSEC, on the CICS TRANSACTION resource definition. This is known as CICS command security, and operates on all the commands that require the special CICS translator option, SP. (These can be seen in Table 11 on page 109). Command security operates in addition to any transaction or resource security you define for a transaction. For example, if a user is permitted to use a transaction called FILA, which issues an EXEC CICS INQUIRE FILE command that the user is **not** permitted to use, CICS issues a "not authorized" (NOTAUTH) condition in response to the command, and the command fails.

See "General resource profiles" on page 25 for information about the resource classes that RACF supports for CICS command security. See "Chapter 8. CICS command security" on page 109 for information about using CICS command security.

## Surrogate user security

CICS performs surrogate user security checking in a number of instances to ensure that a surrogate user is authorized to act for another user. For more information see "Chapter 7. Surrogate user security" on page 103.

Surrogate user checking can be enforced for:

- CICS default user
- Started transactions
- Preset terminal security
- PLT security
- EXCI calls

- Installation of transient data queues.

## QUERY SECURITY command

In addition to using CICS security checking for CICS-controlled resources (or as an alternative to it), you can use the EXEC CICS QUERY SECURITY command to control security access within the CICS application. This method also allows you to define security profiles to RACF for resources other than CICS resource profiles, and enables a more detailed level of security checking than is available through the standard resource classes.

See "General resource profiles" on page 25 for information about the resource classes that RACF supports for resource security checking within transactions. For more information about resource security checking, see "Chapter 6. Resource security" on page 85.

## APPC (LU6.2) session security

So far, all the discussion has been about the security CICS performs for transactions running within a single CICS region, with its own resources and terminal network. A number of CICS regions can also be connected by means of **intercommunication**; for example, **intersystem communication (ISC)** using an SNA access method, such as ACF/VTAM, to provide the necessary communication protocols. This method is normally used for communication between CICS regions residing in different host computers, but it can also connect CICS regions in the same host computer. (See the *CICS Intercommunication Guide* for more information about CICS intercommunication facilities.)

One of the ISC protocols that CICS uses is for advanced program-to-program communication (APPC), which is the CICS implementation of the LU6.2 part of the SNA architecture.

For interconnected systems, the same basic security principles apply, but the resource definition is more complex, and you have additional security requirements. CICS treats APPC sessions, connections, and partners as resources, all of which have security requirements. In addition to the transaction, resource, and command security introduced earlier, CICS provides the following security mechanisms for the APPC environment:

- Bind-time (or session) security, prevents an unauthorized connection between CICS regions.
- Link security defines the authority of the remote system to access transactions or resources to which the connection itself is not authorized.
- User security checks that a user is authorized both to attach a transaction and to access all the resources and SP-type commands that the transaction is programmed to use.

See "Chapter 13. Implementing LU6.2 security" on page 153 for more information.

## Multiregion operation (MRO) security

Another means of using intercommunication is **multiregion operation** (MRO). This is available for links between CICS regions in a single sysplex, independent of the systems network architecture (SNA) access method. See "Chapter 16. Implementing MRO security" on page 199 for information about MRO security.

## Front End Programming Interface security

The security options provided for the Front End Programming Interface are equivalent to those provided for CICS command security (see page 6). Front End Programming Interface security is not discussed in this book, but in the *CICS Front End Programming Interface User's Guide*.

## CICS Business Transaction Services

CICS Business Transaction Services (BTS) also uses security options equivalent to those provided for CICS command security (see page 6). Details of security for BTS is not discussed in this book, but in the *CICS Business Transaction Services* manual.

## Generating and using RACF PassTickets

A PassTicket is a program-generated character string that can be used in place of a password, with the following constraints:

- A specific PassTicket may be used for authentication **once**.
- The PassTicket must be used within 10 minutes of being generated.
- To ease the problem of system time differences, a specific PassTicket can be used up to 10 minutes earlier or later in a target system, compared to the generating system.

Front end programming interface (FEPI) security can generate a PassTicket for use on a target system. The PassTicket can be used anywhere a password can be used.

**Note:** The PassTicket generation and validation algorithm means that the system that creates the PassTicket and the system that validates it must both use the same level of this function. That is, if the creating system has the function applied, and the validating system does not, the PassTicket is invalid.

For more information about the system time differences, and the use of the PassTicket within the 10 minute interval, see the *OS/390 Security Server (RACF) Security Administrator's Guide*.

# Chapter 2. RACF facilities

For its security management capability, CICS relies on a number of facilities provided by RACF. Although RACF provides the basic security access and authorization facilities, it does not by itself perform any security checking.

This chapter covers:



## Overview

RACF provides the following facilities:

- The necessary functions to record information identifying individual users of system resources, and information identifying the resources that require protection. The information you define to RACF about users and resources is stored in user and resource **profiles**.
- The facilities to define which users, or groups of users, are either permitted access, or excluded from access, to the resources for which profiles have been defined. The information recording the users, or groups of users, permitted to access any particular resource is held in an **access list** within the profile that protects a resource.
- A method to process requests, issued by subsystems or jobs running in an MVS system, to authenticate the identity of users defined to RACF, and to check their access authorization to resources.
- The facilities for logging security-related events, such as users signing on and signing off, the issuing of RACF commands, and attempts to access protected resources. Successful attempts to access protected resources may be recorded by the MVS System Management Facility (SMF). If you want to record all attempts to access protected resources, whether successful or not, use RACF auditing, as described in the *OS/390 Security Server (RACF) Auditor's Guide* . The RACF auditor can run the RACF report writer to generate reports based on the SMF records.

  For information on using RACF to perform **auditing** functions (specifying auditing operands on RACF commands, and using the RACF report writer to generate reports of audited security-related activity), see the *OS/390 Security Server (RACF) Auditor's Guide*.

## RACF administration

As the security administrator for one or more CICS regions, and for the users of the CICS applications, it is your job to ensure that your installation's data is properly protected. Using RACF, you are responsible for protecting all system resources, and, in the context of this manual, CICS resources in particular.

A key feature of RACF is its hierarchical management structure. The RACF security administrator is defined at the top of the hierarchy, with authority to control security for the whole system. If you are not yourself the RACF security administrator, you must ask that person to delegate to you sufficient authority to work with RACF profiles and system-wide settings. You must also work with the RACF auditor, who can produce reports of security-relevant activity based on auditing records generated by RACF.

RACF security administrators have either the system-SPECIAL attribute, the group-SPECIAL attribute, or a combination of other authorities.

- If you have the system-SPECIAL attribute, you can issue any RACF command, and you can change any RACF profile (except for some auditing-related operands).
- If you have the group-SPECIAL attribute, your authority is limited to the scope of the RACF group for which you have the SPECIAL attribute.
- The other authorities include:
  - The CLAUTH (class authority) attribute, which allows you to define RACF profiles in specific RACF classes
  - That authority which goes with being the OWNER of existing RACF profiles, allows you to list profiles, change the access, and delete them
  - Having a group authority such as CONNECT or JOIN in a RACF group

For complete information about the authorities required to issue RACF commands, and for information on delegating authority and on the scope of a RACF group, see the *OS/390 Security Server (RACF) Auditor's Guide*.

For information on the RACF requirements for issuing RACF commands, see the descriptions of the commands in the *OS/390 Security Server (RACF) Command Language Reference*.

You can find out whether you have the system-SPECIAL or group-SPECIAL attribute by issuing the LISTUSER command from a TSO session. If you have the system-SPECIAL attribute, SPECIAL appears after the USER ATTRIBUTES phrase in the first part of the output. If you have the group-SPECIAL attribute, SPECIAL appears after the USER ATTRIBUTES phrase in the offset section that describes your connection to a RACF group. For a complete description, with an example of LISTUSER output, see the *OS/390 Security Server (RACF) General User's Guide*.

## Delegation of RACF administrative responsibility

As CICS security administrator, you perform the following tasks (if you do not have the system-SPECIAL attribute, obtain the necessary authority):

- **Define and maintain profiles in CICS-related general resource classes**. In general, you grant authority to do this by assigning a user the CLAUTH (class authority) attribute in the specified classes. For example, the RACF security administrator could issue the following command:

```
ALTUSER  your_userid  CLAUTH(TCICSTRN)
```

The above command gives access to all classes of the same POSIT number. The POSIT number is an operand of the ICHERCDE macro of the class descriptor table (CDT). For more information, see "Activating the CICS classes" on page 26.

- **Define and maintain profiles in other resource classes**. Many of the general resource classes mentioned in this book (such as APPL, APPCLU, FACILITY, OPERCMDS, SURROGAT, TERMINAL, and VTAMAPPL) affect the operation of products other than CICS. If you are not the RACF security administrator, you may need to ask that person to define profiles at your request.
- **Add RACF user profiles to the system**. In general, you grant this authority by assigning the CLAUTH (class authority) attribute for "USER" in the user's profile. For example, the RACF security administrator could issue the following command:

  ```
  ALTUSER  your_userid  CLAUTH(USER)
  ```

  Whenever you add a user to the system, assign that user a default connect group. This changes the membership of the group (by adding the user as a member of the group). Therefore, if you have JOIN group authority in a group, the group-SPECIAL attribute in a group, or are OWNER of a group, CLAUTH(USER) lets you add users to the system and connect them to groups that are within the scope of the group.
- **List RACF system-wide settings and work with all profiles related to CICS**. You grant authority to do this by setting up a RACF group, ensuring that certain CICS-related RACF profiles are in the scope of that group, and connecting a user to the group with the group-SPECIAL attribute. For example, the RACF security administrator could issue the following command:

  ```
  CONNECT  your_userid  GROUP(applicable-RACF_groupid)  SPECIAL
  ```

With the SETROPTS GENERICOWNER command in effect and with prefixing active, administrators can be assigned. You do this by creating a generic profile in each class using the prefix as a high-level qualifier. For example:

```
RDEFINE  TCICSTRN   cics_region_id.**  UACC(NONE)
         OWNER(cics_region_administrator_userid)
```

The SETROPTS GENERIC command must be used before defining generic profiles, as described in "Brief summary of RACF commands" on page 19.

For more information on delegating RACF administration, see the *OS/390 Security Server (RACF) Security Administrator's Guide* .

## RACF user profiles

RACF holds user data in the form of user profiles in the RACF database. These user profiles consist of one or more segments—a RACF segment, and others that are optional. For CICS users, the important segments are:

- The RACF segment, which holds the basic information for a RACF user profile
- The CICS segment, which holds data for each CICS user
- The LANGUAGE segment, which specifies the user's national language preference

These segments are explained briefly in the following sections.

Table 1 on page 12 summarizes where the RACF userids for different types of CICS users are obtained.

*Table 1. Types of CICS users and their userids*

| User type | Userid obtained from |
|---|---|
| Region user | The userid under which the CICS region executes. It is specified in the RACF ICHRIN03 started-procedures table, in the USER parameter of the CICS startup JOB statement, or in the STARTED class. |
| CICS default user | The userid specified on DFLTUSER in the system initialization parameters or at startup. It is used for terminal users who have not signed on. (See "CICS default user" on page 15.) |
| PLTPI user | The userid for PLTPI programs. It is specified on the PLTPIUSR system initialization parameter. The default is the region ID. |
| CICS terminal user who signs on | The userid specified by a terminal user during explicit sign-on. (See "Identifying CICS terminal users" on page 65.) |
| Preset terminal user | The userid specified on the terminal definition. (See "Preset terminal security" on page 5.) |
| ATI user | The userid operand specified within an intrapartition transient data queue definition, or EXEC CICS SET TDQUEUE ATIUSERID option. |
| Started transaction user | The userid for a started non-terminal transaction. |
| Link user | The userid used during MRO or ISC communication. (See "Link security" on page 148.) |
| Remote user | The userid for a transaction attached by the userid on a remote system. For example, by using transaction routing. |
| Surrogate user | The userid specified for a user who has the authority to start work on behalf of another user and is authorized to act for that user. See "Chapter 7. Surrogate user security" on page 103. |
| Surrogate job user | The userid used for batch jobs submitted by CICS, but not using the region userid. (See "Coding the USER parameter on the CICS JOB statement" on page 41.) |
| Operator command user | The userid specified for the user who issues operator commands from operator consoles, and is authorized to issue the MODIFY command, as described in "OPERCMDS resource class" on page 32, as well as having authority to issue the CICS transaction, as described in "Chapter 5. Transaction security" on page 79. |

## RACF segment

You identify a RACF user by an alphanumeric userid, which RACF associates with the user profile for that user. The "user" that you define to RACF need not be a person, such as a CICS terminal user. For example, in the CICS environment, a RACF userid can be associated with the procedure you use to start CICS as a started task; and a userid can be associated with a CICS terminal (for the purpose of preset security). The following list shows some of the basic segment information that RACF holds for a user:

**Keyword**
**Description**

**USERID**
The user's userid

**NAME**
The user's name

**OWNER**
The owner of the user's profile—the RACF administrator or other user authorized by the administrator, or a RACF group

**DFLTGRP**
The default group that the user belongs to

**AUTHORITY**
The user's authority in the default group

**PASSWORD**
The user's password

You define the RACF segment of a user profile using the ADDUSER command, or the RACF ISPF panels. When planning RACF segments of user profiles for CICS users, identify the groups that you want them to be in. Start by identifying RACF administrative units for the users. For example, you could consider all users who have the same manager, or all users within an order entry function, an administrative unit. RACF handles these units as groups of individual users who have similar requirements for access to CICS system resources.

For an overview of the steps required to add users to the system, see the *OS/390 Security Server (RACF) Security Administrator's Guide*.

## CICS segment

The CICS segment of the RACF user profile contains data for CICS users. For information on the order in which CICS searches for the operator information, see "Obtaining CICS-related data for a user" on page 74.

### CICS user data

The information you can specify in the CICS segment is as follows:

**OPCLASS**
CICS uses the operator classes when routing basic mapping support (BMS) messages initiated within a CICS transaction. The operator classes are numeric values in the range 1–24.

Specify operator classes for users who use CICS transactions that issue EXEC CICS ROUTE commands with the (optional) OPCLASS parameter. For automatic routing to occur, you specify the corresponding value as an operator class in the CICS segment of the user profile.

See the *CICS Application Programming Guide* for information about BMS and the use of the OPCLASS parameter for routing messages.

The default value for OPCLASS is 1. (See "When the defaults are effective" on page 14.)

**OPIDENT**
The 1- to 3-character operator identification code that you assign to each operator.

CICS stores the code in the operator's terminal entry in the CICS terminal control table (TCTTE) when the operator signs on. This operator ID is displayed in certain CICS messages and can also be used in the EXEC CICS ROUTE command for routing BMS messages. (For more information about

BMS, see the *CICS Application Programming Guide*). It is also used when using the CEDA LOCK function, as described in the *CICS Resource Definition Guide*.

The default value for OPIDENT is blank. (See "When the defaults are effective".)

**OPPRTY**
The operator priority value—a decimal number that you want CICS to use when determining the task priority for CICS transactions that the operator invokes at a CICS terminal. The priority value can be in the range 0 through 255, where 255 is the highest priority.

CICS uses the sum of operator priority, terminal priority, and transaction priority to determine the dispatching priority of a transaction.

The default value for OPPRTY is 0. (See "When the defaults are effective".)

**TIMEOUT**
The time that must elapse since the user last used the terminal before CICS "times-out" the terminal.

The time must be a decimal integer in the range 0 through 9959 (the last two digits represent a number of minutes, and must be 00 through 59. Any digits to the left of these represent hours).

To specify one hour and eight minutes you would code a value here of 0108. For example:

```
ALTUSER userid CICS(TIMEOUT(0108))
```

The value of 0 (the default) means that the terminal is **not** timed out (see "When the defaults are effective"). For a discussion of coexistence issues, see "Extending timeout values" on page 246.

**XRFSOFF**
The CICS extended recovery facility (XRF) sign-off option. You specify this to indicate whether or not you want CICS to sign off the operator following an XRF takeover.

**FORCE**
Specify FORCE if you want CICS to sign off the operator automatically in the event of an XRF takeover.

**NOFORCE**
Specify NOFORCE if you want CICS to leave an operator signed on in the event of an XRF takeover.

The default value for XRFSOFF is NOFORCE. (See "When the defaults are effective".)

## When the defaults are effective

The defaults listed are effective only when a CICS segment has been defined for that userid. You can make the CICS segment default by defining it as follows:

```
ADDUSER userid DFLTGRP(group_name) NAME(user_name)
       OWNER(group_id | userid)
       PASSWORD(password)
       CICS
```

For example, you may want to define a CICS segment in this way if you want to enforce the **system** defaults, rather than the default user attributes, or if you are setting up a test system and have not yet decided on the values you want to use.

If you omit the CICS segment completely, defaults are obtained as described in "Obtaining CICS-related data for a user" on page 74.

If you specify some of the CICS segment options, but omit others, the defaults described above apply to the omitted options.

You can remove the CICS segment as follows:

```
ALTUSER userid NOCICS
```

## Defining XRFSOFF

The XRFSOFF function is also available at the TYPETERM definition level, as described in the *CICS Resource Definition Guide*, and at the CICS system level in the form of a system initialization parameter, as described in the *CICS System Definition Guide*. (As for the CICS segment, the default value for XRFSOFF in the system initialization parameters and in the TYPETERM definition is NOFORCE.)

Note that the FORCE option in the system initialization table or the TYPETERM definition overrides NOFORCE in the CICS segment.

Table 2 shows how specifying FORCE or NOFORCE in the system initialization parameters, on the TYPETERM definition (or the terminal control table (TCT)), and in the CICS segment together determine whether a terminal remains signed on after an XRF takeover.

**As Table 2 shows, for a terminal to remain signed-on after an XRF takeover, NOFORCE must be specified in all three locations**.

*Table 2. Effects of FORCE and NOFORCE options*

| TYPETERM definition | CICS segment | System initialization parameter | |
|---|---|---|---|
| | | FORCE | NOFORCE |
| FORCE | FORCE | Signed-off | Signed-off |
| | NOFORCE | Signed-off | Signed-off |
| NOFORCE | FORCE | Signed-off | Signed-off |
| | NOFORCE | Signed-off | **Signed-on** |

**Note:** If takeover has exceeded the time specified by the XRFSTME system initialization parameter, users at terminals that have a nonzero TIMEOUT value do not remain signed-on after takeover. For example, suppose the following has been specified in a system that has XRFSOFF=NOFORCE:

```
RDEFINE USER1 CICS(XRFSOFF(NOFORCE) TIMEOUT(10))
RDEFINE USER2 CICS(XRFSOFF(NOFORCE) TIMEOUT(1))
```

If an XRF takeover occurs to a system in which XRFSTME=5 is specified in the system initialization parameters, and that takeover takes longer than five minutes, USER1 does not remain signed-on, but USER2 does.

## CICS default user

When you are using CICS with external security, CICS assigns the security attributes of the CICS **default user** to all CICS terminal users who do not sign on. CICS also assigns the operator data from the CICS segment of the default user to

signed-on users who do not have their own CICS segment data. To enable CICS to assign default security attributes and operator data, you define a CICS default userid to RACF. You then tell CICS which default user to use by specifying the DFLTUSER system initialization parameter. (See the *CICS System Definition Guide* for information about this parameter.) If you do not specify a default userid on the DFLTUSER parameter, CICS uses the name "CICSUSER."

Whether you use installation-defined operator data on your DFLTUSER parameter, or use the default, it is essential that the userid is defined to RACF and that the region userid has installed surrogate security to use the default user (see "Surrogate user security" on page 6).

CICS "signs on" the default user during system initialization. **If you specify SEC=YES as a system initialization parameter, and CICS cannot "sign on" the default userid, CICS initialization fails**.

CICS uses the security attributes of the default userid to perform all the security checks for terminal users who do not explicitly sign on. These security checks include **resource** and **command** security checking, in addition to **transaction-attach** security checking.

## LANGUAGE segment

The language segment holds information about the national language in which the user receives messages. You can specify two languages, but CICS assigns each user only one language. It assigns the primary language if it is specified and CICS supports that language. If the primary language is not specified or is not supported, CICS assigns the secondary language if it is specified and CICS supports it.

Specify the user's preferred national languages in the LANGUAGE segment of the RACF user profile, using the LANGUAGE parameter on the ADDUSER or ALTUSER command:

**LANGUAGE**
Use this parameter to specify primary and secondary languages for CICS users. CICS accepts and uses the languages you define in the segment, but ignores the RACF system-wide defaults. This is because CICS has its own system default for national languages, which you specify on the CICS system initialization parameter, NATLANG.

**PRIMARY(primary_language)**
This parameter identifies the user's primary language, overriding the system default. Depending on the national language feature you have installed, you can specify this as one of the 3-character codes in "Appendix A. National Language" on page 319.

**SECONDARY(secondary_language)**
This parameter identifies the user's secondary language, overriding the system default. You can specify this as one of the 3-character codes listed in "Appendix A. National Language" on page 319.

For more information about national language, see "National language and non-terminal transactions" on page 77.

## Creating or updating segment data for a CICS user

To create or update CICS segment data for a CICS user, specify the CICS option on the RACF ADDUSER command for a new user, or on the ALTUSER command for an existing user. For example, the following command adds a new CICS user to the RACF database with associated CICS operator data:

```
ADDUSER userid  DFLTGRP(group_name) NAME(user_name) OWNER(group_id)
        PASSWORD(password)
        CICS(OPCLASS(1,2,...,n) OPIDENT(identifier) OPPRTY(priority)
             TIMEOUT(timeout_value) XRFSOFF(NOFORCE))
        LANGUAGE(PRIMARY(primary_language))
```

The following example of the ALTUSER command adds CICS operator data to an existing user in the RACF database:

```
ALTUSER userid
        CICS(OPCLASS(1,2,...,n) OPIDENT(identifier) OPPRTY(priority)
             TIMEOUT(timeout_value) XRFSOFF(NOFORCE))
        LANGUAGE(PRIMARY(primary_language))
```

Before issuing these commands to define CICS operator data, ensure that the CICS-supplied RACF dynamic parse validation routines are installed in an APF-authorized library in the linklist. See "CICS-supplied RACF dynamic parse validation routines" on page 37 for details of these exits.

If you do not have the system-SPECIAL attribute, ask your RACF security administrator for the authority to list or update the CICS and LANGUAGE segments in the user profiles. Listing or updating these segments is done by creating profiles in the RACF FIELD class, of the form shown in "FIELD resource class" on page 31.

If you want to change the opclass but you do not want to respecify the list, you can use the ADDOPCLASS and DELOPCLASS operands. For example:

```
ALTUSER userid
        CICS(ADDOPCLASS (1,2)
             DELOPCLASS (6,7))
```

# RACF group profiles

In addition to defining individual user profiles in RACF, you can define **group profiles**.

A group profile defines a group of **users**. (This is not the same thing as a resource group profile, which defines a group of **resources** and is explained in "General resource profiles" on page 25.) A group profile can contain information about the group, such as who owns it; what subgroups it has; a list of connected users; and other information. For details of how to define and use group profiles, see the *OS/390 Security Server (RACF) Security Administrator's Guide* .

Users who are members of groups can share common access authorities to protected resources. For example, you might want to set up groups as follows:

- Users who work in the same department
- Users who work with the same sets of transactions, files, terminals, or other resources that you choose to protect with RACF
- Users who sign on to the same regions (if you have more than one CICS region)

In a CICS environment, group profiles offer a number of advantages:

- Easier control of access to resources
- The ability to assign authorities using the group-SPECIAL attribute or CONNECT group authority
- Fewer refreshes to in-storage profiles.

Aim to make your point of control the presence (or absence) of a userid within a group, not the access list of the resource profile. When someone leaves a department, simply removing the userid from the department's user group revokes all privileges. No other administration of profiles is required. Doing this keeps RACF administration to a minimum and avoids an excessive number of resource profiles.

RACF maintains in-storage copies of resource profiles, so changes to those profiles do not take effect on the system until the in-storage profiles are refreshed.

The authority to access a resource is kept in an access list that is part of the resource profile. The authority can be granted to a user or to a group. To add or remove a user from the access list, refresh the profile in main storage. For more information see "Refreshing resource profiles in main storage" on page 27.

If you connect and remove a user from a group that is already in the access list, that user acquires or loses the authority of the group without needing to refresh the profile. Any user with CONNECT group authority in that group can change the membership of the group (using the CONNECT and REMOVE commands). This avoids the need to change the access list of the affected profiles (through the use of the PERMIT command). If you do not actually change a CICS general resource profile, you need not refresh its in-storage copy. However, users may need to sign on again, if their group membership has been changed.

For other benefits obtained from creating groups, see the *OS/390 Security Server (RACF) Security Administrator's Guide* .

For example, the following command sequence creates a new group of users and moves a user from an existing group to the new group:

```
ADDGROUP  group_name2
REMOVE  user1  GROUP(group_name1)
CONNECT  user1  GROUP(group_name2)
```

Note that in an ISC or MRO environment, the interval that elapses before a **remote** userid is deleted is determined by the CICS system initialization parameter USRDELAY, which specifies how long an unused userid can remain signed on. (This can be up to 7 days.) For information about specifying USRDELAY, see the *CICS System Definition Guide*.

## Data set profiles

Using RACF facilities, you can protect data sets on direct access storage devices (DASD) and tapes. You do this by defining profiles for the data sets you want to protect. The rules for defining data set profiles to RACF are described in the *OS/390 Security Server (RACF) Security Administrator's Guide*, and the *OS/390 Security Server (RACF) Command Language Reference*. For examples, see the *OS/390 Security Server (RACF) General User's Guide*.

You define profiles to protect two RACF categories of data sets:

1. Profiles for **user data sets**, where the high-level qualifier is a RACF userid. All RACF-defined users can protect their own data sets.
2. Profiles for **group data sets**, where the high-level qualifier is a RACF group name (see "RACF group profiles" on page 17 for information about RACF groups). A RACF-defined user can RACF-protect group data sets provided the user has the necessary authority or attributes. (See the *OS/390 Security Server (RACF) Security Administrator's Guide* for details.)

**Note:** Data set profiles do not apply to CICS terminal users, but only to the CICS region userid.

### Generic data set profiles

By using generic profiles, you can reduce the number of profiles needed to protect data sets, and also reduce the required size of the RACF database. In addition, generic profiles are not volume-specific (that is, data sets protected by a generic profile can reside on any volume).

Usually, you specify generic data set profile names by specifying a generic character; for example percent (%) or asterisk (*) in the profile name. For data set profiles, RACF distinguishes between asterisk (*) and double asterisk (**) if RACF's enhanced generic naming is in effect. See the *OS/390 Security Server (RACF) Command Language Reference* for the rules governing generic profile names in the RACF DATASET class.

For example, if you have a group called CICSTS13.CICS, you can define a generic profile named 'CICSTS13.CICS.**', and any user in the access list of this profile can access, at the authorized level, data sets with the high-level qualifier CICSTS13.CICS. For example:

```
ADDSD 'CICSTS13.CICS.**' UACC(NONE) NOTIFY(admin_userid)
```

Use the SETROPTS GENERIC command before defining generic profiles, as described in "Brief summary of RACF commands".

**Note:** Examples in this book show double asterisks (**), which require that enhanced generic naming be in effect. If enhanced generic naming is not in effect, use a single asterisk (*) in place of double asterisks. (You put enhanced generic naming into effect by issuing the RACF SETROPTS EGN command. Note that SETROPTS EGN affects only data set names. Enhanced generic naming is always in effect for general resource profiles, such as TCICSTRN.)

## Brief summary of RACF commands

Much of the RACF activity dealing with protected CICS resources involves creating, changing, and deleting **general resource profiles**.

### Creating a general resource profile

To create a general resource profile, use the RDEFINE command. Generally, once you have created a profile, you then create an access list for the profile using the PERMIT command. For example:

```
RDEFINE  class_name  profile_name  UACC(NONE)
PERMIT  profile_name  CLASS(class_name)
        ID(user or group)  ACCESS(access_authority)
```

This book provides many examples of how to do this for specific CICS-related classes.

## Removing a user or group entry from an access list

To remove the entry for a user or group from an access list, issue the PERMIT command with the DELETE operand instead of the ACCESS operand:

```
PERMIT  profile_name  CLASS(class_name)
        ID(user or group)  DELETE
```

## Changing a profile

If you want to change a profile (for example, changing UACC from NONE to READ), use the RALTER command:

```
RALTER  class_name  profile_name  UACC(READ)
```

## Deleting a profile

To delete a profile, use the RDELETE command. For example:

```
RDELETE class_name  profile_name
```

## Copying from a profile

You can copy an access list from one profile to another. To do so, specify the FROM operand on the PERMIT command:

```
PERMIT  profile_name  CLASS(class_name)
        FROM(existing_profile_name) FCLASS(class_name)
```

You can copy information from one profile to another. To do so, specify the FROM operand on the RDEFINE or RALTER command:

```
RDEFINE class_name profile_name
        FROM(existing-profile_name) FCLASS(class_name)
```

**Note:** Do not plan to do this if you are using resource group profiles. RACF does not copy the members (specified with the ADDMEM operand) when copying the profile. Also, there are other ways in which the new profile might not be an exact copy of the existing profile. For example, RACF places the userid of the resource profile owner in the access list with ALTER access authority. For complete information, see the description of the FROM operand on the appropriate commands in the *OS/390 Security Server (RACF) Command Language Reference*.

## Listing profiles in a class

To list the names of profiles in a particular class, use the SEARCH command. The following command lists profiles in the TCICSTRN class:

```
SEARCH  CLASS(TCICSTRN)
```

The following command lists all profiles and their details in the GCICSTRN class:

```
SEARCH  CLASS(GCICSTRN)
RLIST GCICSTRN * ALL
```

For information on resource classes, see “General resource profiles” on page 25.

### Group-SPECIAL users

If you are a group-SPECIAL user (not system-SPECIAL), the SEARCH command might not list all the profiles that exist in a class. To get a complete list of profiles in a class, you must have at least the authority to list each profile. For further

information, see the description of RACF requirements for the SEARCH command in the *OS/390 Security Server (RACF) Command Language Reference*, and "Which profile is used to protect the resource?" on page 253.

### Activating protection for a class

To begin protecting all the resources protected by profiles in a RACF class, activate that class by issuing the SETROPTS command with CLASSACT specified:

```
SETROPTS CLASSACT(class_name)
```

### Defining a generic profile

Before you can use RDEFINE to define a generic profile (that is, one that uses an asterisk (*), double asterisk (**), ampersand (&), or percentage (%) character), first issue the command:

```
SETROPTS GENERIC(class_name)
```

### Deactivating protection for a class

Deactivating a class turns off protection without disturbing the profiles themselves. If a class is deactivated, RACF issues a "not protected" return code to CICS for **all resources** in that class. CICS treats this response as "access denied". To deactivate a RACF class, issue the SETROPTS command with NOCLASSACT specified:

```
SETROPTS NOCLASSACT(class_name)
```

### Determining active classes

To determine which RACF classes are currently active, issue the SETROPTS command with LIST specified:

```
SETROPTS LIST
```

## Security classification of data and users

RACF gives you the means to classify some or all of the resources on your system. You can use security levels, security categories, or both, to protect any CICS-related resource.

Consider classifying resources if you want to control access to them without having to specify access lists in each resource profile. If you classify a resource, only users whose user profiles are appropriately classified will be able to access that resource. For information on using security levels and security categories, see the *OS/390 Security Server (RACF) Security Administrator's Guide*. Because CICS uses the RACROUTE REQUEST=FASTAUTH function, some services such as security labels and global access checking are not available under CICS. See the *OS/390 Security Server (RACF) Security Administrator's Guide* for information on what is available with FASTAUTH.

You can also put users with the same access or logging requirements into groups. A user can belong to one or more groups, one of which is their default. The sign-on process allows the user to override the default RACF user group name. If "list of groups checking" is inactive, signing on with different group names might give a user different authorities.

# Defining port of entry profiles

***Port of entry*** is the generic term for the device at which the end user signs on. For CICS, the port of entry can be either a terminal or a console. You can use associated port of entry profiles to control whether a user can sign on at a particular device.

## Terminal profiles

This section briefly describes some aspects of terminal profiles that are of interest to CICS users. For more detailed information about defining and protecting terminals on MVS systems, particularly on the following topics, see the *OS/390 Security Server (RACF) Security Administrator's Guide*.

- Creating a profile in the TERMINAL or GTERMINL class
- Preventing the use of an undefined terminal
- Restricting specific groups of users to specific terminals
- Restricting the days or times when a terminal can be used
- Using a security label to control a terminal.

You can control user access to a terminal by defining it to RACF. (User access is determined at CICS sign-on time.) RACF supports two IBM-supplied resource class names for terminals:

**TERMINAL**
For defining a profile of an **individual** terminal.

**GTERMINL**
For defining a profile of a **group** of terminals.

**Note:** For a GTERMINL profile, RACF always uses an in-storage profile, which must be manually refreshed. Every time you create, change, or delete a GTERMINL profile, you (or the RACF security administrator) must issue a SETROPTS RACLIST(TERMINAL) REFRESH command for the change to take effect.

## Defining a profile of an individual terminal

To define terminals with NETNAMEs netid1, netid2, and netid3 in the TERMINAL resource class, use the command:

```
RDEFINE  TERMINAL (netid1, netid2, netid3) UACC(NONE)
                  NOTIFY(sys_admin_userid)
```

If the terminal IDs start with the same characters, you can create a generic profile to cover a group of terminals with the same initial characters. You must use the SETROPTS GENERIC command before defining generic profiles, as described in "Brief summary of RACF commands" on page 19. This reduces the amount of effort needed to create the access list. For example:

```
RDEFINE  TERMINAL  netid*  UACC(NONE)
         NOTIFY(sys_admin_userid)
PERMIT netid* CLASS(TERMINAL)
       ID(group1, group2,.., groupn) ACCESS(READ)
```

## Defining a profile of a group of profiles

Alternatively, you could define the same terminals in the resource group class, by including them as members of a suitable terminal group. For example:

```
RDEFINE  GTERMINL term_groupid
         ADDMEM(netid1, netid2, netid3) UACC(NONE)
         NOTIFY(sys_admin_userid)
```

To remove a terminal from a resource group profile, specify the DELMEM operand on the RALTER command. For example:

```
RALTER  GTERMINL term_groupid
        DELMEM(netid3)
```

To allow a group of users in a particular department to have access to these terminals, use the PERMIT command as follows:

```
PERMIT term_groupid CLASS(GTERMINL) ID(dept_groupid) ACCESS(READ)
```

## Profiles in the TERMINAL or GTERMINAL class

For CICS, the terminal profiles to define to RACF in the TERMINAL or GTERMINL class are used only for VTAM® terminals. The name of the profile is the value of the NETNAME that is specified in the RDO terminal definition or autoinstall. It is not possible to use TERMINAL profiles with non-VTAM terminals.

## Universal access authority for undefined terminals

RACF supports a universal access facility for undefined terminals, which you can control by means of the SETROPTS TERMINAL command (provided you have the necessary authorization). When SETROPTS TERMINAL(NONE|READ) is in effect, it affects **all** MVS terminal subsystems.

If SETROPTS TERMINAL(READ) is in effect, RACF allows any user to log on at any undefined terminal (that is, a terminal not defined in the TERMINAL or GTERMINL resource classes). If SETROPTS TERMINAL(NONE) is in effect, RACF does not allow anyone to log on at any undefined terminal.

**Note:** Before issuing the SETROPTS TERMINAL(NONE) command, define some TERMINAL or GTERMINL class profiles, with enough authorizations to ensure that at least some of the terminals can be used otherwise no one will be able to access any terminal.

### Overriding the SETROPTS TERMINAL command

You can override the SETROPTS TERMINAL command at the group level by specifying the TERMUACC or NOTERMUACC option on the ADDGROUP or ALTGROUP command. The effect of the TERMUACC parameter is to enforce the universal access option. For example, if SETROPTS TERMINAL(READ) is active, the TERMUACC option means that any users in the group can access any undefined terminal. On the other hand, if you specify NOTERMUACC for the group, the SETROPTS TERMINAL command has no effect for that group, and a user in the group needs explicit authorization to use a terminal. To enable a group with the NOTERMUACC option to access terminals, you must add group userid to the access list of the appropriate TERMINAL or GTERMINL profile.

## Console profiles

If the CONSOLE class has been activated, you can control whether:

- A user is allowed to sign on to a console.
- CICS is allowed to sign on a userid for a console defined with preset security.

Console protection is implemented in a similar method to that for protecting terminals, with the exception of the following, which were discussed in "Controlling the use of preset-security" on page 70:

1. The SETROPTS TERMINAL command does not apply to consoles
2. The TERMUACC group attribute does not apply to consoles

Before activating the CONSOLE class, check the *OS/390 MVS Planning: Operations* manual for the effects of console protection on MVS consoles.

The profile used in the console class is the console name or number. For example:

```
RDEFINE CONSOLE CICSCONS UACC(NONE)
```

The user must have READ access to the console name to sign-on at a console. The following example shows how user CICSOPR would be permitted to sign on to the console named CONCICS1:

```
RDEFINE CONSOLE CONCICS1 UACC(NONE)
PERMIT CONCICS1 CLASS(CONSOLE) ID(CICSOPR) ACCESS(READ)
```

Note that, unlike the case with TERMINAL protection, a sign-on attempt will fail if made at a console that has not been defined in the activated CONSOLE class. The access authority to an undefined console is NONE. Port-of-entry checking is not used for the default userid, link userids, or other implicit sign-ons. It is not used for attach-time sign-ons performed by ISC. It **is** used for attach-time sign-ons performed by MRO.

## Conditional access processing

RACF can give you a greater authority to access resources if that user is signed on at a particular terminal or console. This is called **conditional access** processing.

You grant conditional access to a resource by adding

```
WHEN(TERMINAL(netname))
```

or

```
WHEN(CONSOLE(console-name))
```

to the PERMIT command.

The following example allows members of the PAYROLL group to read the SALARY file wherever they are signed on. They would be able to update it only from the terminal with netname PAY001, by issuing the following commands:

```
RDEFINE FCICSFCT SALARY UACC(NONE)
PERMIT SALARY CLASS(FCICSFCT) ID(PAYROLL) ACCESS(READ)
PERMIT SALARY CLASS(FCICSFCT) ID(PAYROLL)
       (WHEN(TERMINAL(PAY001)) ACCESS(UPDATE)
```

To allow members of the operations group OPS to be able to use the CEMT transaction only from the console names MVS1MAST, issue the following command:

```
RDEFINE TCICSTRN CEMT UACC(NONE)
PERMIT CEMT CLASS(TCICSTRN) ID(OPS) WHEN(CONSOLE(MVS1MAST)) AC(READ)
```

**Notes:**

1. The CONSOLE class must be active before CONSOLE conditional access lists can be used.
2. Conditional access lists may only increase authority and not decrease it.

   For other considerations on conditional access lists see, the *OS/390 Security Server (RACF) Security Administrator's Guide*.

## General resource profiles

RACF and CICS have default names for each matching class of resource. These defaults match for the corresponding releases of CICS and RACF. These classes are described in Table 3 on page 26.

### RACF resource class names

For each resource class unique to CICS, there are two resource class names defined to RACF. The first of these is the name of the **member** class in which you define profiles whose names match the names of the resources, such as CICS transactions, programs, or DL/I PSBs. For profiles in this class, you define an access list for each individual resource name. In the following example, the RDEFINE commands define three profiles named CEMT, CEDA, and CEDB in the TCICSTRN resource class. The PERMIT commands allow one or more users or groups of users to access the CEMT transaction:

```
RDEFINE  TCICSTRN CEMT UACC(NONE)
         NOTIFY(sys_admin_userid)
RDEFINE  TCICSTRN CEDA UACC(NONE)
         NOTIFY(sys_admin_userid)
RDEFINE  TCICSTRN CEDB UACC(NONE)
         NOTIFY(sys_admin_userid)
PERMIT   CEMT CLASS(TCICSTRN) ID(group1, group2) ACCESS(READ)
PERMIT   CEDA CLASS(TCICSTRN) ID(group1, group2) ACCESS(READ)
PERMIT   CEDB CLASS(TCICSTRN) ID(group1, group2) ACCESS(READ)
```

The second class name is the RACF **resource group** class. To define a profile in a resource group class, use the RDEFINE command with the ADDMEM operand to add resources as members of the group. For example, you could define a profile named CICSTRANS in the GCICSTRN resource class, adding the CICS-supplied transactions (CEMT, CEDA, CEDB, CEDF, and so on) as members of the group. You then only need to specify an access list for the group, and not for each individual transaction, as in the following example for the CICSTRANS group profile:

```
RDEFINE  GCICSTRN CICSTRANS UACC(NONE)
         ADDMEM(CEMT, CEDA, CEDB)
         NOTIFY(sys_admin_userid)

PERMIT   CICSTRANS  CLASS(GCICSTRN) ID(group1, group2) ACCESS(READ)
```

By using the resource group profiles, you can reduce the number of profiles you need to maintain in the resource classes. Further, provided you avoid defining duplicate member names, using this method reduces the storage requirements for the RACF in-storage profiles that CICS builds during initialization.

RACF provides an in-storage checking service to avoid the I/O operations that would otherwise be needed in RACF. (It does this by means of the RACROUTE REQUEST=FASTAUTH macro.) For this purpose, CICS requests RACF to bring its resource profiles into main storage during CICS initialization.

To make administration easier, avoid defining duplicate profiles. If duplicates are encountered as RACF loads the profiles into storage, it merges the profiles according to the ICHRLX02 selection exit. If no selection exit is installed, RACF follows the default merging rules as indicated in the RLX2P data area. For more information about this, see the *OS/390 Security Server (RACF) Data Areas*.

## IBM-supplied resource class names for CICS

The IBM-supplied set of default resource names for use by CICS is held in the RACF class descriptor table (CDT). You can also use resource classes defined by your installation. For more information, see "Defining your own resource class names" on page 33.)

*Table 3. RACF default resource class names for CICS*

| Default class name | Description | Class |
|---|---|---|
| TCICSTRN<br>GCICSTRN | CICS transactions, normal attach security<br>CICS transaction groups | Member<br>Group |
| PCICSPSB<br>QCICSPSB | CICS PSBs<br>CICS PSB groups | Member<br>Group |
| ACICSPCT<br><br><br><br><br>BCICSPCT | CICS-started transactions and the following EXEC CICS commands:<br>COLLECT STATISTICS, TRANSACTION,<br>DISCARD, TRANSACTION,<br>and INQUIRE\|SET, TRANSACTION<br>Groups for the above | Member<br><br><br><br><br>Group |
| DCICSDCT<br>ECICSDCT | CICS transient data queues<br>Groups for the above | Member<br>Group |
| FCICSFCT<br>HCICSFCT | CICS files<br>CICS file groups | Member<br>Group |
| JCICSJCT<br>KCICSJCT | CICS journals<br>CICS journal groups | Member<br>Group |
| MCICSPPT<br>NCICSPPT | CICS programs<br>CICS program groups | Member<br>Group |
| SCICSTST<br>UCICSTST | CICS temporary storage queues<br>CICS temporary storage queue groups | Member<br>Group |
| CCICSCMD<br><br>VCICSCMD | EXEC CICS SYSTEM commands and<br>EXEC CICS FEPI system commands<br>EXEC CICS SYSTEM command groups and<br>EXEC CICS FEPI system command groups | Member<br><br>Group |
| **Note:** Each default class name has been allocated a group or class category according to its initial character. | | |

**Note:** There are no default resource class names for DB2ENTRY resources. You define your own resource classes for these resources. See "Resource classes for DB2ENTRYs" on page 28 for more information.

## Activating the CICS classes

To activate the CICS resource class for use in security checking by the CICS region, use the RACF SETROPTS command. As soon as the CICS resource class is defined in the active RACF class descriptor table, administrators can define general resource profiles to the class. For more information, see the descriptions of RDEFINE and PERMIT in "General resource profiles" on page 25. Note that the class must be activated before the CICS system can use the profiles that the administrators define.

The format of the SETROPTS command is SETROPTS CLASSACT(*classname*). For example:

```
SETROPTS  CLASSACT(TCICSTRN)
```

All sets of RACF general resource classes that have the same POSIT number in their CDT definitions are activated and deactivated together. Therefore, you need only specify one IBM-supplied CICS class to activate all the IBM-supplied CICS-related classes. If you define your own installation-defined classes with the same POSIT number as the IBM-supplied classes, they are activated and deactivated with the IBM-supplied classes. To provide separate controls for sets of installation-defined classes, define them with different POSIT numbers. (For more information on the POSIT number, see the *OS/390 Security Server (RACF) Macros and Interfaces* manual.)

## Refreshing resource profiles in main storage

Refresh the classes defined in RACLIST by using the TSO command:

```
SETROPTS RACLIST(xxxxxxxx) REFRESH
```

where (*xxxxxxxx*) is the RACF class to be refreshed; for example TCICSTRN. A CEMT PERFORM SECURITY REBUILD command gives a response of "NOT REQUIRED".

## Other IBM-supplied RACF resource class names affecting CICS

The following other IBM-supplied RACF resource class names affect CICS:

**APPCLU**
The resource class in which you define profiles for verifying the identity of APPC partner logical units (LU6.2) during VTAM session establishment.

**APPL** The resource class in which you define profiles for controlling terminal users' access to VTAM applications, such as CICS.

**CONSOLE**
The resource class used to define profiles for consoles.

**DIGTCERT**
The resource class contains certificate information and the certificate itself.

**FACILITY**
The resource class that includes profile definitions for controlling:
- Library lookaside (LLA) libraries
- MRO bindtime security
- Shared data tables security
- Temporary storage pool security
- Coupling facility data table pool security
- Named counter pool security
- Access to log streams in coupling facility structures
- Access to AUTHTYPE and COMAUTHTYPE userids in DB2® definitions.

**FIELD**
The resource class that includes profile definitions for listing or updating the CICS and language segments in the user profiles, and the session segments in APPCLU profiles.

**LOGSTRM**
The resource class that controls which MVS log streams CICS is authorized to use for the purposes of writing and reading journaling and logging data.

**OPERCMDS**
The resource class that controls which operator commands CICS is authorized to issue.

**PROPCNTL**
The resource class that controls userid propagation.

**PTKTDATA**
The resource class that includes PassTicket encryption keys.

**RACFVARS**
The resource class that controls RACF variables.

**RACGLIST**
The resource class that controls the optimization classes activated by RACLIST.

**STARTED**
The resource class that provides the userids for MVS started jobs.

**SUBSYSNM**
The resource class that supports authorization for a subsystem wishing to connect to SMSVSAM. For more information, see "Authorizing access to SMSVSAM servers" on page 52.

**SURROGAT**
The resource class that includes profiles for the following userids:
- preset
- default
- non-terminal
- PLTPI

It is also used for transactions started without a terminal, for controlling job submission, and for DB2 security checking to verify a user's authority to modify AUTHIDs and COMAUTHIDs.

**TERMINAL**
The resource class used to define profiles for terminals.

**VTAMAPPL**
The resource class in which you define profiles for controlling the userids that can open VTAM ACBs from non-APF authorized programs.

| See "Chapter 22. Implementing CICSPlex SM security" on page 267 for details of
| CICSPlex SM classes.

Unlike the IBM-supplied RACF resource classes provided for CICS, you cannot change the class names of these general resource classes. Two of them have CICS system initialization parameters—XAPPC for APPCLU and XUSER for SURROGAT profiles.

## Resource classes for DB2ENTRYs

CICS supports resource security checking for CICS-defined DB2ENTRY resources, for which there are no IBM-supplied RACF resource classes. For DB2ENTRYs, you define security profiles in user-defined class names, and use the XDB2 system initialization parameter to specify the class name to CICS. The syntax for the XDB2 system initialization parameter is XDB2=NO|*name*, which does not support a default class name like the other security system initialization parameters. Use the DFH$RACF sample job as an example of how to define DB2 resource class names for CICS use.

Do not use one of the CICS default resource classes in which to define DB2ENTRY profiles. CICS uses RACLIST to activate the profiles in the default resource classes according to the X*name* system initialization security parameters you specify, and XDB2 should specify a user-defined class name defined specifically for DB2ENTRY resources.

## APPCLU resource class

Before you can use RACF to control which APPC (LU6.2) logical units can establish connections with each other, you need to know the NETID and the LU identifiers of each session partner. With this information, you can use the RDEFINE command to create two profiles in the APPCLU resource class for each LU6.2 pair, defining one profile on each MVS system. For example, on the local system, use the command:

```
RDEFINE  APPCLU  netid1.luid1.luid2 UACC(NONE)
         SESSION(SESSKEY(password))
```

On the remote system, use the command:

```
RDEFINE  APPCLU  netid2.luid2.luid1 UACC(NONE)
         SESSION(SESSKEY(password))
```

In these examples:

**netid1 and netid2**
: is the network id, as specified on the NETID parameter in the VTAM startup member (ATCSTRxx) of SYS1.VTAMLST. If the VTAM in the local system is different from that in the remote system, netid1 and netid2 are different.

**luid1 and luid2**
: are the LU names of the partners. In each case, the first LU name is the local LU name and the second is the remote LU name.

**Note:** CICS does not use the CONVSEC parameter information of the RDEFINE command, although this can be specified in the session segment. The equivalent information is kept in the ATTACHSEC operand of the CONNECTION definition.

You have the following options when you specify the SESSION keyword:

- Specifying the session key (using the SESSKEY suboperand on the SESSION operand).
- Specifying the interval for which the session key will be in effect for LU-LU pairs controlled by the profile (using the INTERVAL suboperand on the SESSION operand).
- Specifying locking or unlocking the LU-LU pairs controlled by this profile using the LOCK and UNLOCK suboperands on the SESSION operand.

  You can use LOCK to prevent users using a link. If LOCK is in force, the relevant profile cannot be used, the session does not bind, and CICS issues message DFHZC4941.

Defining the session key in the profile is optional for RACF, but you must supply the key if CICS is to make use of the profiles. The session key must be the same in both systems.

You can specify either an 8-character alphanumeric session key, or a 16-digit hexadecimal session key. If the session keys at each end of the link do not match, the link cannot be established.

You can also specify an interval after which the password expires, but be aware of the impact this may have on the users at the remote end of the link. If either password expires, the link cannot be established. Depending upon the auditing of the profile records, ICH415I messages may or may not be written out. See "Defining bind-time security" on page 155. (CICS issues message DFHZC4942 to the CSNE destination when the password has expired.) Ensure that you are aware when a password interval is about to expire so that links do not fail for this reason. CICS does not display messages when the password is about to expire, but it does write records to the SMF log.

For a more detailed example of RDEFINE APPCLU, see the section on controlling VTAM LU6.2 binds in the *OS/390 Security Server (RACF) Security Administrator's Guide*. See also "Example of defining an APPCLU profile" on page 154.

See "Chapter 13. Implementing LU6.2 security" on page 153 for information about implementing LU6.2 session security.

## APPL resource class

RACF provides the APPL resource class for defining profiles of applications such as CICS. CICS passes the generic APPLID of the originating region in MRO (for example, the TOR) with the RACROUTE REQUEST=VERIFY ENVIR=CREATE macro. The APPLID is propagated across MRO sessions, but not across ISC sessions. For more information about defining CICS APPLID profiles, and VTAM generic resource, see "Authorizing access to the CICS region" on page 52. By restricting the access lists for the APPL profiles you define, you can control which terminal users (RACF user IDs) can sign on in the various CICS regions. For example:

```
RDEFINE APPL applida UACC(NONE) NOTIFY(sys_admin_userid)
PERMIT  applida CLASS(APPL) ID(group1,..,groupn)  ACCESS(READ)
```

**Note:** An APPLID represents a CICS region. See "Authorizing access to the CICS region" on page 52.

See the *OS/390 Security Server (RACF) Security Administrator's Guide* for more information about controlling access to applications.

## CONSOLE resource class

If the CONSOLE class has been activated, you can control whether a user is allowed to sign on to a console. Console protection is implemented in a method similar to that for protecting terminals, with the exception of the following, which were discussed in "Controlling the use of preset-security" on page 70:

1. The SETROPTS TERMINAL command does not apply to consoles.
2. The TERMUACC group attribute does not apply to consoles.

Before activating the CONSOLE class, check the *OS/390 MVS Planning: Operations* manual for the effects of console protection on MVS consoles.

The profile used in the console class is the console name or number. For example:

```
RDEFINE CONSOLE CICSCONS UACC(NONE)
```

The user must have READ access to the console name to sign-on at a console. The following example shows how user CICSOPR would be permitted to sign on to the console named CONCICS1:

```
RDEFINE CONSOLE CONCICS1 UACC(NONE)
PERMIT CONCICS1 CLASS(CONSOLE) ID(CICSOPR) ACCESS(READ)
```

Note that, unlike the case with TERMINAL protection, a sign-on attempt will fail if made at a console that has not been defined in the activated CONSOLE class. The access authority to undefined consoles is NONE.

## FACILITY resource class

If you are using the library lookaside (LLA) facility of MVS, you can control a program's ability to use the LLACOPY macro. You authorize CICS jobs to use this macro by giving each CICS job UPDATE authority to the CSVLLA data set resource in the FACILITY class for each LLA-controlled data set used by that job. For example:

```
RDEFINE FACILITY CSVLLA.lldataset UACC(NONE)  NOTIFY
PERMIT  CSVLLA.lladataset CLASS(FACILITY) ID(....)  ACCESS(UPDATE)
```

The FACILITY class is also used for MRO bind-time security. For more information about this, see "Chapter 16. Implementing MRO security" on page 199, and "Bind security" on page 215, which discusses MRO bind-time security in connection with shared data tables.

It is also used in the definition of access to log streams in MVS coupling facility structures. See "Authorizing access to MVS log streams" on page 44 for more information.

Additionally, the FACILITY resource class is used by the CICS authorized cross memory (AXM) server environment to validate access to the CICS data sharing servers for the following resources:

- Temporary storage pools (DFHXQ.*poolname*)
- Coupling facility data table pools (DFHCF.*poolname*)
- Named counter pools (DFHNC.*poolname*)

It is also used by the XES component of MVS to validate server access to the coupling facility structure (IXLSTR.*structure_name*) for each of the above pools; for example, IXLSTR.DFHCFLS_*poolname*). For more information about how to define FACILITY resource class profiles for these resources and structures, see "Authorizing access to temporary storage pools and servers" on page 48, "Security for coupling facility data tables" on page 216, and "Authorizing access to named counter pools and servers" on page 50.

See "Chapter 22. Implementing CICSPlex SM security" on page 267 for details about CICSPlex SM usage of the FACILITY resource class. Access to the named counter pool and to the named counter server is controlled in the same way as the other two servers.

The server region userid must have access to IXLSTR.structure in order to be able to connect to the coupling facility structure. The structure name is of the form (DFHNCLS_*poolname*). The server region userid must have CONTROL access to the resource (DFHNC.*poolname*). The client region (CICS region or batch job region) userid must have UPDATE access to the resource (DFHNC.*poolname*).

## FIELD resource class

Resources in the FIELD class control access to certain fields in the RACF database. By creating profiles in the RACF FIELD class, in the following form, you can permit listing or updating of the CICS or LANGUAGE segments in the user profiles, and of appropriate fields in partner-LU profiles.

```
USER.CICS.OPIDENT
USER.CICS.OPCLASSN
USER.CICS.OPPRTY
USER.CICS.TIMEOUT
```

```
USER.CICS.XRFSOFF
USER.LANGUAGE.USERNL1
USER.LANGUAGE.USERNL2
APPCLU.SESSION.SESSKEY
APPCLU.SESSION.KEYINTVL
APPCLU.SESSION.SLSFLAGS
```

Alternatively, you can set up a generic profile `USER.CICS.**`, to control access to all fields in the CICS segment. Before defining generic profiles use the SETROPTS GENERIC command, as described in "Brief summary of RACF commands" on page 19.

You need READ access to list these profiles, and UPDATE access to change them. For further guidance, see the section on field level access checking in the *OS/390 Security Server (RACF) Security Administrator's Guide*.

## LOGSTRM resource class

Before a CICS region can write to (and create, if necessary) the MVS log streams that it uses for its system log and general logs, it must have the appropriate authority. The LOGSTRM general resource class contains the log stream profiles for which the CICS region requires access authority.

The generic profile in the following example covers all log streams referenced by the CICS region identified by its region userid and applid:

```
RDEFINE LOGSTRM region_userid.applid.* UACC(NONE)
```

## OPERCMDS resource class

This resource class controls which console users are allowed to issue MODIFY commands directed to particular CICS regions. For more information, see the *OS/390 Security Server (RACF) Security Administrator's Guide*. The OPERCMDS resource class specifies which operator commands CICS is authorized to issue; for example, commands in the command list table (CLT), and MODIFY network commands.

## PROPCNTL resource class

The PROPCNTL resource class is described in "Controlling userid propagation" on page 54.

## PTKTDATA resource class

The PTKTDATA resource class holds the encryption key used for generating and validating PassTickets.

A profile is added for each APPLID that receives sign-ons with PassTickets. The format of the command to add profiles is:

```
RDEFINE PTKTDATA applid
        SSIGNON(KEYMASKED(password-key))
        KEYENCRYPTED(password-key))
```

## RACFVARS resource class

The RACFVARS resource class contains profile names which start with an ampersand (&). They act as RACF variables that can be specified in profile names in other RACF general resource classes.

## RACGLIST resource class

Contains the resolved copies of profiles globally activated by RACLIST.

### STARTED resource class

The STARTED resource class allows profiles to be defined in this class for each job, or group of jobs, that needs to run under a unique userid.

### SURROGAT resource class

The SURROGAT resource class is used for CICS use of surrogate user validation and for JES job submission. See "Surrogate job submission in a CICS environment" on page 54, "Using the SURROGAT resource class" on page 71, and "Chapter 7. Surrogate user security" on page 103.

### TERMINAL resource class

The TERMINAL resource class is used to authorize the ability to signon at terminals. It is fully described in "Preset terminal security" on page 69.

### VTAMAPPL resource class

The VTAMAPPL resource class controls which userids running non-APF-authorized programs can OPEN the VTAM ACB associated with the CICS address space (which runs as a VTAM application). You can use this resource class to prevent any user from impersonating a CICS region by opening a VTAM ACB with the APPLID of a CICS region.

For specific information, see "Controlling the opening of a CICS region's VTAM ACB" on page 53.

## Defining your own resource class names

You can define your own resource classes so that you have a unique resource class name for each CICS region.

### Benefits

Defining your own resource class names can have the following benefits.

**Controlling access from other regions:** You can prevent users running in one CICS region from accessing the resources of other CICS regions that have different class names specified. (You can also do this by using prefixing; see the description of the SECPRFX parameter in "Defining security-related system initialization parameters" on page 56.)

**Group administrator for each region:** For each CICS region with installation-defined classes, you can authorize a different group administrator to create profiles to be used by that region.

To get this benefit, define the installation-defined classes with a POSIT number other than 5 (the POSIT number of the IBM-supplied CICS classes). Then give the group administrator the CLAUTH (class authority) for at least one of those classes.

Use the SETROPTS GENERIC command before defining generic profiles, as described in "Brief summary of RACF commands" on page 19.

With prefixing active, you can also assign different administrators without fear of conflict. To do this, create a generic profile in each class, using the prefix as a high-level qualifier. For example:

```
RDEFINE  TCICSTRN   cics_region_id.**  UACC(NONE)
         OWNER(cics_region_administrator_userid)
```

The administrator specified as the OWNER of each such profile can create and maintain more specific profiles. The other administrators cannot do so.

**Note:** If you are running CICS with XRF, think of the active CICS and its alternate as one CICS system as far as RACF is concerned, and define the same resource class names to both the active and alternate CICS region.

## Setting up installation-defined classes

To set up installation-defined classes, work with your RACF system programmer to add new class descriptors to the installation-defined part (module ICHRRCDE) of the RACF class descriptor table (CDT). For an example of how to add installation-defined classes to the CDT, see "Chapter 18. Customizing security processing" on page 221.

All installation-defined classes defined in the CDT must also be defined in the MVS router table. This is because the MVS router checks any class used in a router request to determine if it actually exists. If it does not, no request is sent to RACF. To define classes to the MVS router, add them to ICHRFR01, the user-modifiable portion of the MVS router table, as described in the *OS/390 Security Server External Security Interface (RACROUTE) Macro Reference*. Also see "Specifying user-defined resources to RACF" on page 226.

When setting up installation defined classes, we recommend that you copy the IBM-supplied defaults from the CDT, an example of which is in the *OS/390 Security Server (RACF) Macros and Interfaces* manual. You will then need to change the name, group or member name, POSIT number, and ID. See the description of the ICHERCDE macro in the *OS/390 Security Server (RACF) Macros and Interfaces* manual for details of valid values for these operands. See the same manual for information about creating installation-defined resource classes. For an example of how to add resource classes, see the IBM-supplied sample, DFH$RACF, which is in CICSTS13.CICS.SDFHSAMP.

However, if you are using Long Temporary storage Queue Names with an OS/390 Release 5, or an earlier release of RACF, you must use an installation-defined temporary storage resource class (SxxxxTST) with a modified MAXLENGTH capable of handling the combined length of the security prefix, For CICS TSQnames and separator (up to a maximum of 25 characters).

For CICS resources, the first character of the resource class name is predefined by CICS, consistent with the default resource class name. You can define the second through eighth characters of the resource class name, but for ease of administration it is recommended that you specify the same characters for both the member and group class. The seven characters specified for the member class are the part of the resource class name you define to CICS in the various X*name* parameters, except for the following:

- XDB2, which has no CICS-defined prefix letter, so any defined class name of 1- to 8-characters can be specified. It is recommended that you use a specific class or classes dedicated to these resources.
- XAPPC and XUSER, which have no "name" option, and are either YES or NO to say whether security is activie or not.

You should avoid using the letters "CICS" in the second through fifth characters in any class name you define. RACF requires that at least one of the characters in the classname should be a national or numeric character.

# Part 2. Implementing RACF protection for a single-region CICS

This part discusses how to implement security on a single-region CICS, regardless of where the task needs to be performed—either in the CICS environment or in the RACF environment. Where necessary, it refers you to other manuals in the CICS and RACF libraries for more detailed information about resource and security-related definitions.

- **"Chapter 3. CICS data set and system security" on page 37** deals with protecting the MVS data sets that CICS requires—the program load libraries and the CICS system data sets (such as the local and global catalogs, journal, auxiliary temporary storage, and transient data intrapartition data sets).
- **"Chapter 4. Verifying CICS users" on page 65** deals with all aspects of sign-on security, including the part played by the CICS segment.
- **"Chapter 5. Transaction security" on page 79** describes the security checks that CICS performs to verify that a user entering a transaction at a CICS terminal is authorized to use the transaction. This is known as **transaction-attach security**. It also explains the part played by the CICS **default userid**.
- **"Chapter 6. Resource security" on page 85** describes the RESSEC and CMDSEC attributes on resource definitions. It explains the purposes of the RACF user, group, profile, and resource class definitions, and gives examples illustrating how CICS and RACF together control access to resources.
- **"Chapter 7. Surrogate user security" on page 103** describes the surrogate user checking activity that CICS can perform. It describes the RACF definitions needed, and gives some examples using the RACF surrogate user facility.
- **"Chapter 8. CICS command security" on page 109** describes CICS command security for the system programming commands. You can use these commands either through the CEMT master terminal transaction, or through the CICS API. This chapter also discusses the CMDSEC attribute on resource definitions.
- **"Chapter 9. Security checking using the QUERY SECURITY command" on page 117** describes security checking by the user application using the EXEC CICS QUERY SECURITY command, which enables an application program to request from RACF the level of access a user has to a particular resource. The application program can determine what action to take based on the CICS-value data area (CVDA) values that CICS returns.
- **"Chapter 10. Security for CICS-supplied transactions" on page 125** describes how to protect the CICS-supplied transactions, both those that are for CICS internal use only (and cannot be invoked directly from a CICS terminal), and those provided explicitly for users at CICS terminals.

# Chapter 3. CICS data set and system security

This chapter describes how to protect the MVS data sets that CICS requires. It discusses the following:



## CICS installation requirements for RACF

You can control access to the resources used by your CICS region (or regions) by using RACF facilities. The CICS libraries supplied on the distribution volume include the CICS modules you need to support external security management.

### CICS-supplied RACF dynamic parse validation routines

To define CICS terminal operator data, use the CICS-supplied RACF dynamic parse validation routines. Install these routines in SYS1.CICSTS13.CICS.SDFHLINK, which should be made an APF-authorized library in your MVS linklist. (For more information, see the *CICS Transaction Server for OS/390 Installation Guide*.)

The routines are as follows:

**DFHSNNFY**
CICS segment update notification

**DFHSNPTO**
CICS segment TIMEOUT print formatting

**DFHSNVCL**
CICS segment OPCLASS keyword validation

**DFHSNVID**
CICS segment OPIDENT keyword validation

**DFHSNVPR**
CICS segment OPPRTY keyword validation

**DFHSNVTO**
CICS segment TIMEOUT keyword validation

## Using RACF support in a multi-MVS environment

If you are operating a multi-MVS environment with shared DASD, which is probably the case if you are running CICS with XRF, you are likely to want the active and alternate CICS systems to have access to the same terminal user characteristics. You can enable this by having the active and alternate CICS systems share the same RACF database.

## Setting options on the MVS program properties table

For performance reasons, consider making your CICS regions nonswappable, by specifying the NOSWAP option in the PPT statement of the SCHEDxx member of the SYS1.PARMLIB library. If your installation has an entry for the DFHSIP program in the MVS program properties table (PPT), ensure that the NOPASS option is **not** set for DFHSIP in the PPT statement of the SCHEDxx member of the SYS1.PARMLIB library. Setting the NOPASS option would bypass password and RACF authorization checking on data sets accessed by the CICS region. For more information about specifying CICS MVS PPT options, see the *CICS Transaction Server for OS/390 Installation Guide*.

## Protecting CICS load libraries

Although, in general, CICS runs in unauthorized state, the CICS initialization program, DFHSIP, needs to run in authorized state for part of its execution. For this reason, the version of the DFHSIP module supplied on the distribution tape is link-edited with the "authorized" attribute (using the linkage-editor SETCODE AC(1) control statement), and is installed in CICSTS13.CICS.SDFHAUTH. This library must be defined to the operating system as APF-authorized.

To prevent unauthorized or accidental modification of CICSTS13.CICS.SDFHAUTH, make this library RACF-protected. Without such protection, the integrity and security of your MVS system are at risk. To control the unauthorized start-up of a CICS system using DFHSIP, also consider implementing the following:

- If DFHSIP is in a library that has been placed in the MVS link list, protect DFHSIP with a profile in the PROGRAM resource class. Give READ access to this profile only to those users who are allowed to execute CICS.
- If DFHSIP has been placed in the link pack area (LPA), it cannot be protected by the PROGRAM resource class. Instead, control the start-up of CICS by controlling the loading of any suffixed DFHSIT load module. Ensure that no DFHSIT load module is included in the LPA, then control the loading of DFHSIT by creating a generic 'DFHSIT*' profile in the PROGRAM resource class. Give READ access to this profile only to those users who are allowed to execute CICS.

Also give RACF protection to SYS1.CICSTS13.CICS.SDFHLINK and to SYS1.CICSTS13.CICS.SDFHLPA; and the other libraries (including CICSTS13.CICS.SDFHLOAD) that make up the STEPLIB and DFHRPL library concatenations.

See "Authorizing access to CICS data sets" on page 45 for more information about protecting CICS data sets and creating suitable data set security profiles.

**Note:** The source statements of your application programs are sensitive; consider having RACF protect the data sets containing them.

## Specifying the CICS region userid

When you start a CICS region (either as a job or as a started task) in an MVS environment that has RACF installed, the job or task is associated with a userid, referred to as the **CICS region userid**. The authority associated with this userid determines which RACF-protected resources the CICS region can access.

Each CICS region, for either production or test use, should be subject to normal RACF data set protection based on the region userid under which the CICS region executes. You specify the region userid under which CICS executes in one of three ways:

As a started task:

- In the RACF started procedures table, ICHRIN03, when you start CICS as a started task using the MVS START command. (See "Authorizing CICS procedures to run under RACF".) However, do not assign the "trusted" or "privileged" attributes to CICS entries in the started procedures table. For more information, see the description of associating MVS started procedures with userids in the *OS/390 Security Server (RACF) System Programmer's Guide*.

As a started job:

- In a STARTED general resource class profile, on the user parameter of the STDATA segment.

As a job:

- On the USER parameter of the JOB statement when you start CICS as a JOB.

**To ensure the authorizations for different CICS regions, are properly differentiated, run each with a unique region userid**. For example, the userid under which you run the production CICS regions to process payroll and personnel applications should be the only CICS userid authorized to access production payroll and personnel data sets.

If you are using intercommunication, it is particularly important to use unique userids, unless you want to bypass link security checking by using equivalent systems. For more information, see "Link security with LU6.2" on page 157, "Link security with LU6.1" on page 193, or "Link security with MRO" on page 202, depending on the environment you are using.

## Authorizing CICS procedures to run under RACF

You can invoke your CICS startup procedure to start CICS as a started task or as a started job. RACF provides the ICHRIN03 procedure table for started tasks, and the STARTED general resource class for started jobs. Both options are discussed here:

### Using the ICHRIN03 table for started tasks

If you run CICS as a started task, associate the cataloged procedure name with a suitably authorized RACF user through the RACF table, ICHRIN03. RACF supplies a default ICHRIN03 table, which you can modify. See the *OS/390 Security Server (RACF) System Programmer's Guide* for more information about this table, and how you can add the default entry for the cataloged procedure name for starting CICS.

If your ICHRIN03 table contains the default entry, you need not update the table; but define a RACF user with the same name as the cataloged procedure.

If your ICHRIN03 table does not contain the default entry (or you choose not to set the default entry), update the table with an entry that contains the cataloged procedure name and its associated RACF user. This RACF user need not have the same name as the cataloged procedure.

Whether your ICHRIN03 table contains the default entry or a specific entry you have defined, ensure that the RACF user identified through ICHRIN03 has the necessary access authority to the data sets in the cataloged procedure.

For example, if you associate a cataloged procedure called DFHCICS with the RACF userid CICSR, the userid CICSR needs to have access to the CICS resources accessed by the task started by DFHCICS.

## Using STARTED profiles for started jobs

Using a single procedure to start all your CICS regions as started tasks limits you to a single CICS region userid, as defined in the RACF started task table, ICHRIN03.

Using the started job security support provided by RACF 2.1 removes this constraint, and allows you to use separate userids for each started job, even though they are all started from the same procedure. Alternatively, you can use generic profiles for groups of CICS regions that are to share the same userid—for example, for all regions of the same type, such as terminal-owning regions.

The support for started jobs is provided by the RACF STARTED general resource class, and its associated STDATA segment. You define profiles in this class for each job, or group of jobs, that needs to run under a unique userid.

Ensure that the userids specified in STDATA segments are defined to RACF. Also ensure that the userids are properly authorized to the data set profiles of the CICS regions that run under them.

**Example of a generic profile for multiple AORs:** The following example shows how to define a generic profile for jobs that are to be started using a procedure called CICSTASK. In this example the job names begin with the letters CICSDA for a group of CICS application-owning regions (AORs):

```
RDEFINE  STARTED  (CICSTASK.CICSDA*)  STDATA( USER=(CICSDA##) )
```

When you issue the START command to start CICSTASK with a job name of, say, CICSDAA1, MVS passes the procedure name (CICSTASK), and the job name (CICSDAA1) in order to obtain the userid under which this CICS application-owning region is to run. In the example shown above, the CICS region userid is defined as CICSDA##, for all regions started under the generic profile CICSTASK.CICSDA*.

**Example of a unique profile for each region::** The following example shows how to define a unique profile for jobs that are to be started using a procedure called CICSTASK, and where each started job is to run under a unique CICS region userid:

```
RDEFINE  STARTED  (CICSTASK.CICSDAA2)  STDATA( USER=(CICSDAA2) )
```

When you issue the START command to start CICSTASK with the job name CICSDAA2, MVS passes the procedure name (CICSTASK) and the job name

(CICSDAA2) to obtain the userid under which this CICS application-owning region is to run. In the example shown above, the CICS region userid is defined as CICSDAA2, the same as the APPLID.

## Defining user profiles for CICS region userids

Before bringing up a CICS region, ensure that the required userids are defined - the CICS region userid and the CICS default userid. If you are suitably authorized, you can define a RACF user profile for a CICS region by means of the ADDUSER command. For example, to define CICSR as a userid for a CICS region, enter the following RACF command from TSO:

```
ADDUSER CICSR  NAME(user-name) DFLTGRP(cics_region_group)
```

In this example, DFLTGRP has been specified, so the initial password is the DFLTGRP name. If you do not specify DFLTGRP, the password is set by default to the name of the group to which the person issuing the ADDUSER command belongs. Alternatively, you can specify a password explicitly on the PASSWORD parameter of the ADDUSER command. See "Coding the USER parameter on the CICS JOB statement" for details about changing new userid passwords.

**Do not assign the OPERATIONS attribute to CICS region userids**. Doing so would allow the CICS region to access RACF-protected data sets for which no specific authorization has been performed. CICS region userids do not need the OPERATIONS attribute if the appropriate CONNECT or PERMIT commands have been issued. These commands authorize the CICS region userid for each CICS region to access only the specific data sets required by that region.

### Coding the USER parameter on the CICS JOB statement

If you start CICS from a job, include the parameters USER= and PASSWORD= on the JOB statement. For example:

```
//CICSA   JOB ... ,USER=CICSR,PASSWORD=password
```

When you define a new user to RACF, the password is automatically flagged as expired. For this reason, the first time you start CICS under a new userid, change the PASSWORD parameter on the JOB statement. For example:

```
PASSWORD=(oldpassword,newpassword)
```

If you want to avoid specifying the password on the JOB statement, you can allow a surrogate user to submit the CICS job. A surrogate user is a RACF-defined user who is authorized to submit jobs on behalf of another user (the original user), without having to specify the original user's password. Jobs submitted by a surrogate user execute with the identity of the original user. See "Surrogate job submission in a CICS environment" on page 54 for more information. The region userid must also have surrogate authority to use the default user; see "Chapter 7. Surrogate user security" on page 103.

### Authorities required for CICS region userids

The CICS control program runs under the CICS region userid. Therefore, this userid needs access to all the resources that CICS itself needs to use. There are two types of these resources:

1. Resources external to CICS, such as disk files, the spool system, and the VTAM network.
2. Resources internal to CICS, such as system transactions and auxiliary userids.

**Authorizing external resources:** Like a batch job, each CICS region must be able to access many external resources. The authority for CICS to access these resources

is obtained from the CICS region userid. It doesn't matter which signed-on user requests CICS to perform the actions that access these resources. The external services are aware only that CICS is requesting them, under the region userid's authority.

Give access to these resources:

- The MVS system logger

  CICS needs authority to use log streams defined in the MVS logger. See "Authorizing access to MVS log streams" on page 44.
- External disk data sets used by CICS

  CICS needs authority to open all the disk data sets that it uses. See "Authorizing access to CICS data sets" on page 45.
- External disk data sets used by application programs

  CICS needs authority to open all the disk data sets that your own application programs need. See "Authorizing access to user data sets" on page 48.
- Temporary storage servers

  CICS needs authority to access temporary storage servers if any TS queues are defined as shared. See "Access to temporary storage servers" on page 49.
- SMSVSAM servers

  CICS needs authority to access the SMSVSAM server if you are using VSAM record-level sharing (RLS). See "Authorizing access to SMSVSAM servers" on page 52.
- VTAM applications

  Consider carefully for each program whether you will allow it to become a VTAM application. If you do this, CICS needs authority to open its VTAM ACB. See "Controlling the opening of a CICS region's VTAM ACB" on page 53.
- Jobs submitted to the internal reader

  If any of your applications submit JCL to the JES internal reader, you should prevent CICS from allowing them to be submitted without the USERID parameter. See "Controlling userid propagation" on page 54.

  However, you should not usually require your applications to provide a PASSWORD parameter on submitted jobs. So you **should** allow CICS to be a surrogate user of all the possible userids that may be submitted. See "Surrogate job submission in a CICS environment" on page 54.
- System spool data sets

  CICS needs authority to access data sets in the JES spool system. See "JES spool protection in a CICS environment" on page 55.

**Authorizing internal resources:** There are several internal functions in which CICS behaves like an application program, but is actually performing housekeeping functions that are not directly for any end user. The associated transactions execute under control of the CICS region userid, and because they access CICS internal resources, you must give the CICS region userid authority to access them. These are:

- CICS system transactions

  CICS needs authority to attach all the internal housekeeping transactions that it uses. See "Category 1 transactions" on page 126.
- Auxiliary userids

  If CICS surrogate user checking is specified with the XUSER system initialization parameter (the default), CICS needs authority to use certain additional userids. These are:

  - The default userid

    See “CICS default user” on page 103.
  - The userid used for post-initialization processing (PLTPIUSR)

    See “Post-initialization processing” on page 103.
  - The userid used for transient data trigger transactions

    See “Transient data trigger-level transactions” on page 105.
- Resources used by PLTPI programs

  If the PLTPIUSR system initialization parameter is omitted, the CICS region userid is used for all PLTPI programs. In this case, give the CICS region userid access to all the CICS resources that these programs use. See “PLT programs” on page 83.

## Defining the default CICS userid to RACF

For each CICS region for which you specify SEC=YES, define a RACF user profile whose userid matches the value of the system initialization parameter, DFLTUSER. For example, if you specify DFLTUSER=NOTSIGND, define a RACF user profile named NOTSIGND.

If you do not specify a value for the DFLTUSER parameter, the CICS-supplied default userid is CICSUSER—define a RACF user profile named CICSUSER.

Define a different default CICS userid for each CICS region if any of the following considerations applies:

- The default CICS userid requires different security attributes (such as membership in RACF groups).
- The default CICS userid requires different operator data (CICS segment of the RACF user profile).
- The default CICS userid requires a different default language (LANGUAGE segment of the RACF user profile).

To define a CICS default user with the system initialization parameter default name (CICSUSER), use the ADDUSER command with the CICS operand, as follows:

```
ADDUSER CICSUSER  DFLTGRP(group_id) NAME(user_name)
        OWNER(userid or group)
        PASSWORD(password)
        CICS(OPCLASS(1,2,...,n) OPIDENT(identifier) OPPRTY(priority)
             TIMEOUT(timeout_value)  XRFSOFF(xrf_sign-off_option))
```

The security administrator should always define the password for default userids and started tasks, instead of allowing it to default.

Each CICS region should use its own default user, as an aid to debugging. Set up a RACF default user group to keep the definitions similar.

If you have specified the system initialization parameter XUSER=YES (the default), authorize the CICS region userid to be a surrogate user of the default userid. For example:

```
PERMIT CICSUSER.DFHINSTL CLASS(SURROGAT) ID(cics_region_userid)
```

During startup, CICS “signs on” the default userid. If the default user sign-on fails (because, for example, the userid is not defined to RACF), CICS issues message DFHXS1104 and terminates CICS initialization.

When CICS successfully signs on a valid RACF userid as the default user, it establishes the terminal user data for the default user from one of the following sources:

- The CICS segment of the default user's RACF user profile
- Built-in CICS system default values

See "Obtaining CICS-related data for a user" on page 74 for details of the sign-on process for obtaining CICS terminal operator data.

CICS assigns the security attributes of the default userid to all CICS terminals before any terminal user begins to sign on. The security attributes and terminal user data of the default user also apply to any terminals at which users do not sign on (using either the CICS-supplied CESN transaction or a user-written equivalent), unless the security has been explicitly preset by specifying a value for the USERID option in the terminal definition.

CICS also assigns the security attributes of the default userid to any "trigger level transactions" that are initiated for transient data queues without a USERID parameter.

Ensure the default userid gives at least the minimum authorities that ought to be granted to any other terminal user. In particular:

- Give the default user access to the region's APPLID. See "Authorizing access to the CICS region" on page 52.
- Give the default user access to the CICS-supplied transactions that are intended to be used by everybody. See the definitions in "Identifying CICS terminal users" on page 65, especially those transactions that are recommended for inclusion in the ALLUSER example group of transactions.

## Authorizing access to MVS log streams

Ensure that you authorize the CICS region userid to write to (and create if necessary) the log streams that are used for its system log and general logs. You do this by granting the appropriate access authorization to log stream profiles in the LOGSTRM general resource class.

The level of authorization required depends on whether log streams are always explicitly defined to the MVS system logger:

- If CICS is expected to create log streams dynamically, give CICS **ALTER** authority to the relevant log stream profiles, and **UPDATE** authority to the relevant coupling facility structures.
- If all the log streams to which CICS writes are already defined to MVS, give CICS only **UPDATE** authority to the log stream profiles.
- Permit **READ** access to those users who need to read the CICS log streams.

For example, the generic profile in the following example could be defined to cover all the log streams referenced by the CICS region and identified by its region userid and applid:

```
RDEFINE LOGSTRM region_userid.** UACC(NONE)
```

If, however, you have multiple CICS systems sharing the same region userid, but with differing security requirements, include the applid in the generic profile, as follows:

```
RDEFINE LOGSTRM region_userid.applid.* UACC(NONE)
```

The following example allows the CICS region userid under which CICS is running to write journal and log records to log streams in the named coupling facility structure:

```
PERMIT IXLSTR.structurename CLASS(FACILITY) ACCESS(UPDATE)
       ID(region_userid)
```

The following examples give access to three categories of user:

```
PERMIT region_userid.applid.* CLASS(LOGSTRM) ACCESS(ALTER)
       ID(region_userid)
PERMIT region_userid.applid.* CLASS(LOGSTRM) ACCESS(READ)
       ID(authorized_browsers)
PERMIT region_userid.applid.* CLASS(LOGSTRM) ACCESS(UPDATE)
       ID(archive_userid)
```

In these examples, `region_userid` is the CICS region userid under which CICS is running, either as a started task or batch job. The identifier `archive_userid` is the userid under which an application program runs to purge old data from CICS logs when the data is no longer needed. The identifier `authorized_browsers` refers to the userids of users allowed to read log streams, but not purge data.

If several CICS regions share the same CICS region userid, you can make profiles more generic by specifying * for the *applid* qualifier.

The number of profiles you define depends on the naming conventions of the logs, and to what extent you can use generic profiling.

## Authorizing access to CICS data sets

When you have defined a region userid for your CICS job (or started task), permit that userid to access the CICS system data sets with the necessary authorization.

When authorizing access to CICS system data sets, choose appropriately from the following levels of access: READ, UPDATE, and CONTROL. Also define data set profiles with UACC(NONE) to ensure that only CICS region userids can access those data sets. For information about the CICS region userid, see "Specifying the CICS region userid" on page 39.

For CICS load libraries, only permit READ access.

The following four data sets require CONTROL access.

- The temporary storage data set
- The transient data intrapartition data set
- The CAVM control data set (XRF)
- The CAVM message data set (XRF)

Permit UPDATE access for all the remaining CICS data sets.

Therefore, for CICS system data sets you need at least three generic profiles to restrict access to the appropriate level. See Table 4 on page 46.

*Table 4. Summary of generic data set profiles*

| Required access level | Type of CICS data sets protected |
|---|---|
| READ | Load libraries |
| UPDATE | Auxiliary trace; transaction dump; system definition; global catalog; local catalog; and restart |
| CONTROL | Temporary storage; intrapartition transient data; XRF message; and XRF control |

If you use generic naming of the data set profiles, you can considerably reduce the number of profiles you need for your CICS regions. This policy is illustrated in the examples shown in Figure 1 for a number of sample CICS regions.

You can issue the RACF commands shown in the examples from a TSO session, or execute the commands using the TSO terminal monitor program, IKJEFT01, in a batch job as illustrated in Figure 1. Alternatively, you can use the RACF-supplied ISPF panels. Any of these methods enables you to create the necessary profiles and authorize each CICS region userid to access the data sets as appropriate for the corresponding CICS region.

```
//RACFDEF  JOB 'accounting information',
//             CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
//DEFINE   EXEC PGM=IKJEFT01
//SYSPRINT DD SYSOUT=A
//SYSTSPRT DD SYSOUT=A
//SYSUDUMP DD SYSOUT=A
//SYSTSIN  DD *
ADDSD  'CICSTS13.CICS.SDFHLOAD'  NOTIFY(cics_sys_admin_id) UACC(NONE)
PERMIT 'CICSTS13.CICS.SDFHLOAD' ID(cics_id1,...,cics_group1,..,cics_groupn)
                          ACCESS(READ)
ADDSD  'CICSTS13.CICS.SDFHAUTH'  NOTIFY(cics_sys_admin_id) UACC(NONE)
PERMIT 'CICSTS13.CICS.SDFHAUTH' ID(cics_id1,...,cics_group1,..,cics_groupn)
                          ACCESS(READ)
ADDSD  'CICSTS13.CICS.DFHJPDS'  NOTIFY(cics_sys_admin_id) UACC(NONE)
PERMIT 'CICSTS13.CICS.DFHJPDS' ID(cics_id1,...,cics_group1,..,cics_groupn)
                         ACCESS(READ)
ADDSD  'CICSTS13.CICS.applid.**'  NOTIFY(cics_sys_admin_id) UACC(NONE)
PERMIT 'CICSTS13.CICS.applid.**'  ID(applid_userid) ACCESS(UPDATE)
ADDSD  'CICSTS13.CICS.applid.DFHXR*'  NOTIFY(cics_sys_admin_id) UACC(NONE)
PERMIT 'CICSTS13.CICS.applid.DFHXR*'  ID(applid_userid) ACCESS(CONTROL)
ADDSD  'CICSTS13.CICS.applid.DFHINTRA' NOTIFY(cics_sys_admin_id) UACC(NONE)
PERMIT 'CICSTS13.CICS.applid.DFHINTRA'  ID(applid_userid) ACCESS(CONTROL)
ADDSD  'CICSTS13.CICS.applid.DFHTEMP' NOTIFY(cics_sys_admin_id) UACC(NONE)
PERMIT 'CICSTS13.CICS.applid.DFHTEMP'  ID(applid_userid) ACCESS(CONTROL)
ADDSD  'CICSTS13.CICS.DFHCSD'          NOTIFY(cics_sys_admin_id) UACC(NONE)
PERMIT 'CICSTS13.CICS.DFHCSD' ID(cics_group1,..,cics_groupn) ACCESS(UPDATE)
/*
//
```

*Figure 1. Example of commands to authorize access to CICS data sets*

**Note:** Data sets that need to be accessed in the same way by all CICS regions (for example, with READ or UPDATE access) should be protected by profiles that do **not** include an APPLID. For example, define the partitioned data sets that contain the CICS load modules with profiles that give all CICS region groups (or userids) READ access.

You could also consider protecting all these data sets with one generic profile called 'CICSTS13.CICS.**'. However, you must strictly control who has read access to CICSTS13.CICS.SDFHAUTH, because it contains APF-authorized programs, and the profile protecting this data set **must** be defined with UACC(NONE). In Figure 1 all of the partitioned data sets are defined with UACC(NONE) and have an explicit access list.

Although CICS modules exist in libraries SYS1.CICSTS13.CICS.SDFHLPA and SYS1.CICSTS13.CICS.SDFHLINK, no CICS region userid requires access to these libraries.

By establishing a naming convention for the data sets belonging to each region,and one generic profile for each CICS region, with the CICS VTAM APPLID as one of the data set qualifiers, you can ensure that only one CICS region has access to the data sets. In the examples shown in Figure 1, all the names have a high-level qualifier of CICSTS13.CICS, but your installation will have its own naming conventions for you to follow.

CICS needs UPDATE access to all the data sets covered by these profiles. The CICS DDNAMEs for the data sets in this category are as follows:

**DFHGCD**
: Global catalog data set

**DFHLCD**
: Local catalog data set

**DFHAUXT**
: Auxiliary trace data set, A extent

**DFHBUXT**
: Auxiliary trace data set, B extent

**DFHDMPA**
: Transaction dump data set, A extent

**DFHDMPB**
: Transaction dump data set, B extent

**Note:** The auxiliary trace data set, the transaction dump data set, and the MVS dump data set may contain sensitive information. Protect them from unauthorized access.

CICS needs CONTROL access for the transient data intrapartition, temporary storage, and CICS availability manager (CAVM) data sets.

The CICS DDNAMEs for the data sets in this category are as follows:

**DFHINTRA**
: Transient data intrapartition data set

**DFHTEMP**
: Temporary storage data set

**DFHXRCTL**
: XRF control data set

**DFHXRMSG**
: XRF message data set

The CICS system definition data set (CSD) is protected by a discrete profile to which all CICS groups have access. This assumes that all the CICS regions are sharing a common CSD. If your CICS regions do not share a common CSD and each region has its own CSD, or if groups of regions share a CSD, define discrete or generic data set profiles as appropriate.

## Authorizing access with the MVS library lookaside (LLA) facility

If any of the load-module libraries in the DFHRPL concatenation is controlled by the MVS library lookaside (LLA) facility, authorize the CICS region's userid in *one* of the following ways:

- It must have UPDATE authority to the data set that contains the LLA module.
- It must have UPDATE authority in the FACILITY class to the resource CSVLLA. *datasetname*, where *datasetname* is the name of the library that contains the LLA module.

## Authorizing access to user data sets

When you have defined the RACF userids for your CICS regions and given them access to the CICS system data sets, permit the userids to access the CICS **application** data sets with the necessary authority. The following RACF commands permit the userid specified on the ID parameter to access some CICS user application data sets, with READ authority for the first two data sets, and UPDATE authority for the last two:

```
PERMIT 'CICSTS13.CICS.appl1.dataset1'  ID(user or group)  ACCESS(READ)
PERMIT 'CICSTS13.CICS.appl1.dataset2'  ID(user or group)  ACCESS(READ)
PERMIT 'CICSTS13.CICS.appl2.dataset3'  ID(user or group)  ACCESS(UPDATE)
PERMIT 'CICSTS13.CICS.appl2.dataset4'  ID(user or group)  ACCESS(UPDATE)
```

**ACCESS(CONTROL) for VSAM entry-sequenced data sets (ESDS)**

CICS file control uses control interval processing when opening a VSAM ESDS (non-RLS mode only). This means that you must specify ACCESS(CONTROL) for all such data sets, otherwise the OPEN command fails with message DFHFC0966.

**ACCESS(ALTER) for VSAM data sets when using BWO**

In order to use backup while open (BWO) to back up VSAM data sets that are currently in use and are defined as BACKUPTYPE(DYNAMIC), or BWO(TYPECICS) in the integrated catalog facility (ICF) catalog, give the CICS region userid RACF ALTER authority to the data set or to the ICF catalog in which that data set is defined. If you do not, the OPEN command fails with message DFHFC5803. See the *CICS Recovery and Restart Guide* for guidance on using BWO.

# Authorizing access to temporary storage pools and servers

You can control access by:

- Temporary storage (TS) servers to TS pools (see "Access to temporary storage pools")
- CICS regions to the TS servers (see "Access to temporary storage servers" on page 49).

## Access to temporary storage pools

You can control access by temporary storage (TS) servers to the TS pools in the coupling facility. Each TS server can be started as a job or started task. The name

of the TS queue pool for a TS server is specified at server startup. For each TS pool there can be only one TS server running on each MVS image in the sysplex.

Two security checks are made against the TS server's userid—that is, the userid under which the job or started task is running. To ensure the server passes these checks, do the following:

- Authorize the TS server region to connect to the coupling facility list structure used for its own TS pool. This requires that the TS server userid has ALTER authority to a coupling facility resource management (CFRM) RACF profile called IXLSTR.*structure_name* in the FACILITY general resource class.

  For example, if the userid of the server is DFHXQTS1, and the list structure is called DFHXQLS_TSPRODQS, the following RACF commands define the profile and provide the required access:

  ```
  RDEFINE FACILITY IXLSTR.DFHXQLS_TSPRODQS UACC(NONE)
  PERMIT IXLSTR.DFHXQLS_TSPRODQS CLASS(FACILITY) ID(DFHXQTS1) ACCESS(ALTER)
  ```

To reduce security administration, use the same TS server userid to start each TS server that supports the same TS pool.

- Give the TS server's userid CONTROL access to the CICS RACF profile called DFHXQ.*poolname* in the FACILITY general resource class. This authorizes the TS server to act as a server for the named TS pool.

  For example, if the userid of the server is DFHXQTS1, and the pool name is TSPRODQS, the following RACF commands define the profile and provide the required access:

  ```
  RDEFINE FACILITY DFHXQ.TSPRODQS UACC(NONE)
  PERMIT DFHXQ.TSPRODQS CLASS(FACILITY) ID(DFHXQTS1) ACCESS(CONTROL)
  ```

  See "System authorization facility (SAF) responses to the TS server" for information about the responses to the TS server.

## Access to temporary storage servers

You can control access by CICS regions to the TS servers. A security check is made against the CICS region userid to verify that the region is authorized to use the services of a TS server. This check is made each time that a CICS region connects to a TS server.

Give each CICS region userid that connects to a TS server UPDATE access to the CICS RACF profile called DFHXQ.*poolname* in the FACILITY general resource class. This authorizes the CICS region to use the services of the TS server for the named TS pool.

For example, if the userid of a CICS region is CICSDAA1, and the pool name is TSPRODQS, the following RACF commands define the profile and provide the required access:

```
RDEFINE FACILITY DFHXQ.TSPRODQS UACC(NONE)
PERMIT DFHXQ.TSPRODQS CLASS(FACILITY) ID(CICSDAA1) ACCESS(UPDATE)
```

When a CICS region has connected to a TS pool, it can write, read, and delete TS queues without any further security checks being performed by the server. However, the CICS application-owning regions issuing TS API requests can use the existing mechanisms for TS resource security checking.

### System authorization facility (SAF) responses to the TS server

If the security profile for a TS pool cannot be retrieved, SAF neither grants nor refuses the access request. In this situation:

Access to the TS pool, either by a CICS region or by the TS server itself, is rejected if:

- A security manager is installed, but is either temporarily inactive or inoperative for the duration of the MVS image. This is a fail-safe action, on the grounds that, if the security manager is active, it might retrieve a profile that does not permit access to the TS pool.

Access to the TS pool, either by a CICS region or by the TS server itself, is accepted if:

- There is no security manager installed, or
- There is an active security manager, but the FACILITY class is inactive, or there is no profile in the FACILITY class. The access request is allowed in this case because there is no evidence that you want to control access to the TS server.

Access is permitted to any TS server without a specific DFHXQ.*poolname* profile, or an applicable generic profile. No messages are issued to indicate this. To avoid any potential security exposures, you can use generic profiles to protect all, or specific groups of, TS servers. For example, specifying:

```
RDEFINE FACILITY (DFHXQ.*) UACC(NONE)
```

ensures that access is allowed only to TS servers with a more specific profile to which a TS server or CICS region is authorized.

## Authorizing access to named counter pools and servers

You can control access by:

- Coupling facility data table (named counter) servers to named counter pools (see "Access to named counter pools")
- CICS regions to the named counter servers (see "Access to named counter servers" on page 51).

### Access to named counter pools

You can control access by named counter servers to the named counter pools in the coupling facility. Each named counter server can be started as a job or started task. The name of the named counter pool for a named counter server is specified at server startup. For each named counter pool there can be only one server running on each MVS image in the sysplex.

Two security checks are made against the named counter server's userid—that is, the userid under which the job or started task is running. To ensure the server passes these checks, do the following:

- Authorize the named counter server region to connect to the coupling facility list structure used for its own named counter pool. This requires that the named counter server userid has ALTER authority to a coupling facility resource management (CFRM) RACF profile called IXLSTR.*structure_name* in the FACILITY general resource class.

  For example, if the userid of the server is DFHNCSV1, and the list structure is called DFHNCLS_DFHNC001, the following RACF commands define the profile and provide the required access:

  ```
  RDEFINE FACILITY IXLSTR.DFHNCLS_DFHNC001 UACC(NONE)
  PERMIT IXLSTR.DFHNCLS_DFHNC001 CLASS(FACILITY) ID(DFHNCSV1) ACCESS(ALTER)
  ```

To reduce security administration, use the same named counter server userid to start each named counter server that supports the same named counter pool.

- Give the named counter server's userid CONTROL access to the CICS RACF profile called DFHNC.*poolname* in the FACILITY general resource class. This authorizes the named counter server to act as a server for the named counter pool.

  For example, if the userid of the server is DFHNCSV1, and the pool name is DFHNC001, the following RACF commands define the profile and provide the required access:

  ```
  RDEFINE FACILITY DFHNC.DFHNC001 UACC(NONE)
  PERMIT DFHNC.DFHNC001 CLASS(FACILITY) ID(DFHNCSV1) ACCESS(CONTROL)
  ```

  See "System authorization facility (SAF) responses to the named counter server" for information about the responses to the CFDT server.

## Access to named counter servers

You can control access by CICS regions to the named counter servers. A security check is made against the CICS region userid to verify that the region is authorized to use the services of a named counter server. This check is made each time that a CICS region connects to a named counter server.

Give each CICS region userid that connects to a named counter server UPDATE access to the CICS RACF profile called DFHCF.*poolname* in the FACILITY general resource class. This authorizes the CICS region to use the services of the named counter server for the named named counter pool.

For example, if the userid of a CICS region is CICSDAA1, and the pool name is DFHNC001, the following RACF commands define the profile and provide the required access:

```
RDEFINE FACILITY DFHNC.DFHNC001 UACC(NONE)
PERMIT DFHNC.DFHNC001 CLASS(FACILITY) ID(CICSDAA1) ACCESS(UPDATE)
```

When a CICS region has connected to a named counter pool, it can define, update, delete, get, rewind, and query named counters without any further security checks being performed by the server.

**Note:** Unlike shared temporary storage pools and coupling facility data table pools, named counters can also be accessed by batch application regions. Batch jobs are subject to the same security mechanisms as a CICS region.

### System authorization facility (SAF) responses to the named counter server

If the security profile for a named counter pool cannot be retrieved, SAF neither grants nor refuses the access request. In this situation:

Access to the named counter pool, either by a CICS region or by the named counter server itself, is rejected if:

- A security manager is installed, but is either temporarily inactive or inoperative for the duration of the MVS image. This is a fail-safe action, on the grounds that, if the security manager is active, it might retrieve a profile that does not permit access to the named counter pool.

Access to the named counter pool, either by a CICS region or by the named counter server itself, is accepted if:

- There is no security manager installed, or

- There is an active security manager, but the FACILITY class is inactive, or there is no profile in the FACILITY class. The access request is allowed in this case because there is no evidence that you want to control access to the named counter server.

Access is permitted to any named counter server without a specific DFHCF.*poolname* profile, or an applicable generic profile. No messages are issued to indicate this. To avoid any potential security exposures, you can use generic profiles to protect all, or specific groups of, named counter servers. For example, specifying:

```
RDEFINE FACILITY (DFHNC.*) UACC(NONE)
```

ensures that access is allowed only to named counter servers with a more specific profile to which a named counter server or CICS region is authorized.

## Authorizing access to SMSVSAM servers

SMSVSAM is a data-sharing subsystem running on its own address space to provide the RLS support required by CICS.

For CICS regions using VSAM record-level sharing (RLS), access to SMSVSAM servers is controlled by RACF security checks. The security check is made against the CICS region userid to verify that the region is authorized to register with an SMSVSAM server.

The general resource class, SUBSYSNM, supports authorizations for subsystems that want to connect to SMSVSAM. The SUBSYSNM profile name is the name by which a given subsystem is known to VSAM. CICS uses its applid as its subsystem name; define a profile for the CICS applid in the SUBSYSNM resource to enable CICS to register the control ACB.

When CICS attempts to register the control ACB during initialization, SMSVSAM calls RACF to check that the CICS region userid is authorized to the CICS profile in the SUBSYSNM class. If the CICS region userid does not have READ authority, the open request fails.

For example, if the applid of a CICS AOR is CICSDAA1, and the CICS region userid (shared by a number of AORs) is CICSDA##, define and authorize the profile as follows:

```
RDEFINE SUBSYSNM  CICSDAA1  UACC(NONE) NOTIFY(userid)
PERMIT  CICSDAA1  CLASS(SUBSYSNM)  ID(CICSDA##)  ACCESS(READ)
```

## Authorizing access to the CICS region

You can restrict access by terminal users, or other CICS regions, to specific CICS regions by defining CICS APPLID profiles in the RACF APPL class. To authorize CICS terminal-owning regions to access a VTAM generic resource, you must define a VTAMAPPL profile with the generic resource name as the VTAMAPPL profile name. Authorize each CICS terminal-owning region with READ access to the VTAMAPPL profile. For these purposes, the APPLID of a CICS region is the VTAM generic resource if the GRNAME system initialization parameter is specified, or the XRF generic APPLID if XRF=YES is specified. Otherwise, it is the specific APPLID named in the APPLID system initialization parameter. If you define a profile in the APPL class for a CICS APPLID, or a generic profile that applies to one or more CICS APPLIDs with UACC(NONE), all terminal users trying to sign on to a CICS

region must have explicit access to the profile that applies to that region's APPLID, either as an individual profile, or as a member of a group. For example:

```
RDEFINE APPL cics_region_applid UACC(NONE)  NOTIFY(sys_admin_userid)
```

For MRO only, the APPLID is propagated from the terminal-owning region (TOR) to the other region that the user accesses. Therefore, you can force users to sign on through a TOR, by denying users access to any APPLID except that of the TOR.

Use the RACF PERMIT command to add authorized users to the access list of CICS APPL profiles. For example:

```
PERMIT cics_region_applid CLASS(APPL) ID(group1,...,groupn) ACCESS(READ)
```

permits all users defined in the listed groups to sign on to `cics_region_applid`.

The APPL class must be active for this protection to be in effect:

```
SETROPTS CLASSACT(APPL)
```

Also, for performance reasons, consider activating profiles in the APPL class using RACLIST.

```
SETROPTS RACLIST(APPL)
```

If the APPL class is already active, refresh the in-storage APPL profiles with the SETROPTS command:

```
SETROPTS RACLIST(APPL) REFRESH
```

**Notes:**

1. CICS always passes the APPLID to RACF when requesting RACF to perform user sign-on checks, and there is no mechanism within CICS to prevent this.
2. RACF treats undefined CICS APPLIDs as UACC(READ).
3. If the APPL class is active, and a profile exists for a CICS region in the APPL class, ensure that authorized remote CICS regions can sign on to a CICS region protected in this way.

## Controlling the opening of a CICS region's VTAM ACB

You can control which users among those who are running non-APF-authorized programs can OPEN the VTAM ACB associated with a CICS address space (CICS region). This ensures that only authorized CICS regions can present themselves as VTAM applications providing services with this APPLID, thus preventing unauthorized users impersonating real CICS regions. (Note that the CICS region userid needs the OPEN access, not the issuer of the SET VTAM OPEN command.)

For each APPLID, create a VTAMAPPL profile, and give the CICS region userid READ access. For example:

```
RDEFINE VTAMAPPL applid UACC(NONE)  NOTIFY(userid)
PERMIT applid CLASS(VTAMAPPL) ID(cics_region_userid) ACCESS(READ)
```

The correct CICS APPLID to specify in the VTAMAPPL class is the specific APPLID, as specified in the CICS system initialization parameters. If you are using XRF (that is, if CICS is started with XRF=YES in effect), define two VTAMAPPL profiles—one each for both the active and alternate CICS region's **specific** APPLID (the second operand on the CICS APPLID startup option).

**Note:** If your alternate is on another MVS image, ensure that the RACF database is shared, or define the VTAMAPPL profiles in the other system's RACF database.

The VTAMAPPL class must be activated using RACLIST for this protection to be in effect:

```
SETROPTS CLASSACT(VTAMAPPL) RACLIST(VTAMAPPL)
```

If the VTAMAPPL class is already active, refresh the in-storage VTAMAPPL profiles with the SETROPTS command:

```
SETROPTS RACLIST(VTAMAPPL) REFRESH
```

## Controlling userid propagation

Jobs submitted from CICS to the JES internal reader without the USER operand being specified on the JOB statement run under the CICS region's userid. These jobs have the access authorities of the CICS region itself, and so could potentially expose other data sets in the MVS system.

You (or the RACF security administrator) can prevent the CICS userid from being propagated to these batch jobs by defining a profile in the PROPCNTL class where the profile name is the CICS region's userid. For example, if the CICS region runs under userid CICS1, define a PROPCNTL profile named CICS1:

```
RDEFINE  PROPCNTL  CICS1
```

The PROPCNTL class must be activated using RACLIST for this protection to be in effect:

```
SETROPTS CLASSACT(PROPCNTL) RACLIST(PROPCNTL)
```

If the PROPCNTL class is already active, refresh the in-storage PROPCNTL profiles with the SETROPTS command:

```
SETROPTS RACLIST(PROPCNTL) REFRESH
```

You (or the RACF security administrator) must issue the SETROPTS command to refresh these profiles. Issuing the CICS PERFORM SECURITY REBUILD command does not affect the PROPCNTL class.

## Surrogate job submission in a CICS environment

Batch jobs submitted by CICS can be allowed to run with a USER parameter other than the CICS region's userid, but without specifying the corresponding PASSWORD. This is called surrogate job submission. These jobs have the access authorities of the USER parameter actually specified on the JOB statement. If the PASSWORD parameter is specified on the JOB statement, surrogate processing does not occur.

You (or the RACF security administrator) can allow this by defining a profile in the SURROGAT class. For example, if the CICS region's userid is CICS1, and the job is to run for userid JOE, define a SURROGAT profile named JOE.SUBMIT:

```
RDEFINE  SURROGAT  JOE.SUBMIT  UACC(NONE)
         NOTIFY(JOE)
```

Further, you must permit the CICS region's userid to act as the surrogate to the profile just defined:

```
PERMIT JOE.SUBMIT CLASS(SURROGAT) ID(CICS1) ACCESS(READ)
```

The SURROGAT class must be activated using RACLIST for this protection to be in effect:

```
SETROPTS CLASSACT(SURROGAT) RACLIST(SURROGAT)
```

### Attention

Any CICS user, whether signed on or not, is able to submit jobs that use the SURROGAT userid, if the CICS userid has authority for SURROGAT. If your installation is using transient data queues to submit jobs, you can control who is allowed to write to the transient data queue that goes to the internal reader. However, if your installation is using EXEC CICS SPOOLOPEN to submit jobs, you cannot control who can submit jobs (without writing an API global user exit program to screen the commands). CICS spool commands do no CICS resource or command checking.

You can use an EXEC CICS ASSIGN USERID command to find the userid of the user who triggered the application code. Application programmers can then provide code that edits a USER operand onto the JOB card destined for the internal reader.

For a complete description of surrogate job submission support, see the *OS/390 Security Server (RACF) Security Administrator's Guide*.

## Authorizing the CICS region userid as a surrogate user

When CICS performs surrogate user checking, the CICS region userid must be authorized as a surrogate. Grant authorization for the CICS region userid acting as a surrogate user for the following:

- The CICS default user
- The userid used for post-initialization processing (PLTPIUSR)
- All userids used for transient data trigger level transactions
- All userids specified on the AUTHID or COMAUTHID parameters of a DB2 resource definition.

For more information about surrogate user checking, see "Chapter 7. Surrogate user security" on page 103.

## JES spool protection in a CICS environment

Your installation can protect JES spool data sets with profiles in the JESSPOOL class. Spool files created by the SPOOLOPEN commands have the userid of the CICS region in their security tokens, not the userid of the person who issued the SPOOLOPEN command. Thus, the userid qualifier in the related JESSPOOL profiles is the CICS region's userid.

When using the SPOOLOPEN INPUT command, CICS checks that the first four characters of the APPLID correspond to the external writer name of the spool file. This checking is independent of any RACF checking that may also be done.

You should define ALTER access to the appropriate PROFILE in the JESSPOOL class when the JESSPOOL class is active on a SPOOLOPEN INPUT command, to prevent a NOTFND condition being returned.

## Defining security-related system initialization parameters

There are several system initialization parameters that CICS provides for specifying your security requirements at the system level. These parameters are coded in the CICS system initialization table (SIT) or as system initialization overrides. For full details of system initialization parameters, see the *CICS System Definition Guide*.

### SEC

You use the SEC system initialization parameter to specify the level of resource security management you want for your CICS region. There are two options:

**YES**
This means that the CICS external security interface will be initialized, and control of CICS security is determined by the other security-related SIT options:

| | |
|---|---|
| SECPRFX | XRFSTME |
| DFLTUSER | XCMD |
| ESMEXITS | XDB2 |
| SNSCOPE | XDCT |
| PSBCHK | XFCT |
| CMDSEC | XJCT |
| RESSEC | XPCT |
| PLTPIUSR | XRCT |
| PLTPISEC | XPPT |
| XAPPC | XPSB |
| XUSER | XTRAN |
| XRFSOFF | XTST |

**NO**
This means that there is no security checking of whether users are allowed to access CICS (and non-CICS) resources from this region, and sign-on cannot take place.

**Note:** Even if you have specified SEC=NO, with MRO bind-time security, the CICS region userid is sent to the secondary system, and bind-time checking is carried out in the secondary system. See "Bind-time security with MRO" on page 199 for more information.

### SECPRFX

This parameter is effective only if you also specify SEC=YES. You use the SECPRFX system initialization parameter to specify whether you want CICS to prefix the resource names that it passes to RACF for authorization. The prefix that CICS uses is the RACF userid under which the CICS region is running.

Prefixing is useful mainly when you have more than one CICS region. It enables you to prevent users on one CICS region from accessing the resources of a different CICS region that has a different prefix. For example, you might have one CICS region with the prefix CICSPROD and another with prefix CICSTEST. Users of the CICSTEST system would be able to use profiles that included the CICSTEST prefix, and users of the CICSPROD system would be able to use profiles that included the CICSPROD prefix. Users of both systems would be able to use resources protected by profiles that included CICS.

There are two options on the SECPRFX parameter:

**NO**
CICS does not prefix the resource names in authorization requests that it passes to RACF from this CICS region.

**YES**
CICS prefixes the resource names with its RACF userid when passing authorization requests to RACF. The prefix corresponds to the CICS region userid.

To change these values employ an ICHRTX00 SAF preprocessing exit. For more information, see "Determining the userid of the CICS region" on page 225. For example, if a CICS job specifies USER=CICSREG on the JOB statement, and SECPRFX=YES is specified, you can define and allow access to the CICS master terminal transaction (CEMT) in the TCICSTRN resource class as follows:

```
RDEFINE  TCICSTRN  CICSREG.CEMT
         UACC(NONE)   NOTIFY(sys_admin_userid)
PERMIT  CICSREG.CEMT  CLASS(TCICSTRN)
        ID(groupid1,...,groupidn) ACCESS(READ)
```

You can also use a resource group profile in the GCICSTRN resource class. If you do, specify the prefix on the ADDMEM operand. The following shows CICSREG specified in a profile named CICSTRANS:

```
RDEFINE  GCICSTRN  CICSTRANS
         ADDMEM(CICSREG.CEMT)
         UACC(NONE)   NOTIFY(sys_admin_userid)
PERMIT  CICSTRANS  CLASS(GCICSTRN)
        ID(groupid1,...,groupidn) ACCESS(READ)
```

**Note:** If you protect a resource with a resource group profile, avoid protecting the same resource with another profile. If the profiles are different (for example, if they have different access lists), RACF merges the profiles for use during authorization checking. Not only can the merging have a performance impact, but it can be difficult to determine exactly which access authority applies to a particular user. (For more information, see the *OS/390 Security Server (RACF) Security Administrator's Guide*.)

## CMDSEC

Code CMDSEC to specify whether or not you want CICS to honor the CMDSEC option specified on a transaction's resource definition. CMDSEC specified with the option ASIS means that CICS obeys the CMDSEC option. CMDSEC specified with the option ALWAYS means that CICS ignores the CMDSEC option, and always performs the command check. For more information about these options, see the *CICS System Definition Guide*.

## DFLTUSER

Specify a value for DFLTUSER to identify to CICS the name you have defined to RACF as the default userid. If you omit this parameter, the name defaults to CICSUSER. See "Defining the default CICS userid to RACF" on page 43.

## ESMEXITS

Use ESMEXITS to specify whether you want CICS to pass installation data for use by the RACF installation exits. For more information on ESMEXITS, see "Chapter 18. Customizing security processing" on page 221.

## PLTPISEC

Code PLTPISEC to specify whether or not you want CICS to perform command security or resource security checking for PLT programs that run during CICS initialization.

## PLTPIUSR

Code PLTPIUSR to specify the userid that CICS is to use for security checking for PLT programs that run during CICS initialization.

## PSBCHK

Code PSBCHK to specify that you want CICS to perform PSB authorization checks for remote terminal users who use transaction routing to initiate a transaction in this CICS region (to access an attached IMS™ system). The default PSBCHK=NO specifies that CICS is to check the remote link but not the remote user. The remote user is checked by specifying PSBCHK=YES.

## RESSEC

Code this to specify whether or not you want CICS to honor the RESSEC option specified on a transaction's resource definition. RESSEC specified with the option ASIS means that CICS obeys the RESSEC option. RESSEC specified with the option ALWAYS means that CICS ignores the RESSEC option, and always performs the resource check. For more information about these options, see the *CICS System Definition Guide*.

## SNSCOPE

SNSCOPE—the sign-on SCOPE—applies to all userids signing on by explicit sign-on request; for example, the EXEC CICS SIGNON command or the CESN transaction. Use it to specify whether or not a userid can have more than one CICS session active at the same time.

The sign-on SCOPE is enforced with the MVS ENQ macro. The SNSCOPE values correspond to the STEP, SYSTEM, and SYSTEMS levels of ENQ scoping. This means that only those CICS systems that specify exactly the same value for SNSCOPE can check the scope of each other.

SNSCOPE affects only users signing on at local terminals, or signing on after using the CRTE transaction to connect to another system. For more information about using SNSCOPE, and the restrictions involved, see the *CICS System Definition Guide*.

# CICS resource class system initialization parameters

You specify at the system level (with the SEC=YES parameter) that you want CICS to use RACF to authorize access to CICS resources. You also specify at the system level which particular CICS resources you want CICS to check by means of the X*name* system initialization parameters. The full list of the CICS resource classes is shown in Table 5 on page 59, each with corresponding X*name* system initialization parameter.

*Table 5. System initialization parameters for the CICS resource classes*

| System initialization parameter | Resource |
|---|---|
| XAPPC={**NO** \| YES} | APPC partner-LU verification |
| XCMD={**YES** \| name \| NO} | EXEC CICS system commands EXEC CICS FEPI system commands |
| XDB2={**NO** \| name} | CICS DB2 resources |
| XDCT={**YES** \| name \| NO} | Transient data destinations |
| XFCT={**YES** \| name \| NO} | Files |
| XJCT={**YES** \| name \| NO} | Journals and logs |
| XPCT={**YES** \| name \| NO} | Started transactions and EXEC CICS commands: COLLECT STATISTICS TRANSACTION DISCARD TRANSACTION INQUIRE TRANSACTION and SET TRANSACTION |
| XPPT={**YES** \| name \| NO} | Programs |
| XPSB={**YES** \| name \| NO} | DL/I program specification blocks (PSBs) |
| XTRAN={**YES** \| name \| NO} | Attached transactions |
| XTST={**YES** \| name \| NO} | Temporary storage entries |
| XUSER={**YES** \| NO} | Surrogate user checking DB2 AUTHTYPE checking |

**Notes:**

1. The parameters are effective only with SEC=YES.
2. None of the parameters can be entered as a console override.

If you specify YES for any X*name* system initialization parameter, CICS uses the default class name for that parameter. (See "IBM-supplied resource class names for CICS" on page 26.)

As an example, the effect of specifying SEC=YES with three of the resource class parameters specified as X*name*=YES is illustrated in the following tref refid='rules'.

*Table 6. Specifying external security with default resource classes*

| System initialization parameter | Effect |
|---|---|
| SEC=YES | CICS initializes external security interface. |
| XTRAN=YES | CICS uses the TCICSTRN and GCICSTRN resource class profiles for transaction-attach security checking. |
| XFCT=YES | CICS uses the FCICSFCT and HCICSFCT resource class profiles for file access security checking. |
| XPSB=YES | CICS uses the PCICSPSB and QCICSPSB resource class profiles for PSB access security checking. |

As a second example, the effect of specifying SEC=YES with the same three associated resource class parameters specified as X*name*=*username* is shown in Table 7.

*Table 7. Specifying external security for user-defined resource classes*

| System initialization parameter | Effect |
|---|---|
| SEC=YES | CICS uses full RACF security support. |

*Table 7. Specifying external security for user-defined resource classes (continued)*

| System initialization parameter | Effect |
|---|---|
| XTRAN=$usrtrn | CICS uses the T$usrtrn and G$usrtrn user-defined resource class profiles for transaction-attach security checking. |
| XFCT=$usrfct | CICS uses the F$usrfct and H$usrfct user-defined resource class profiles for file access security checking. |
| XPSB=$usrpsb | CICS uses the P$usrpsb and Q$usrpsb user-defined resource class profiles for PSB access security checking. |

When CICS is being initialized, it requests RACF to bring resource profiles into main storage to match all the resource classes that you specify on system initialization parameters. Note that (except for XAPPC and XDB2) X*name*=YES is the default in the system initialization parameters, and CICS will use the default classnames, for example, GCICSTRN. Supply RACF profiles for all those resources for which you do not specify X*name*=NO explicitly. If CICS requests RACF to load a general resource class that does not exist or is not correctly defined, CICS issues a message indicating that external security initialization has failed, and terminates CICS initialization.

For guidance on the syntax of external security system initialization parameters, see the *CICS System Definition Guide*.

The way you define the individual transaction definitions in the CSD determines whether you want to use RACF security for the resources and commands used with transactions. See "Chapter 4. Verifying CICS users" on page 65 and "Chapter 5. Transaction security" on page 79 for information about specifying resource and command security for transactions.

### XAPPC and XUSER

The syntax of the XAPPC and XUSER system initialization parameters is slightly different from that of the other X*name* parameters. You can only specify YES or NO.

XAPPC=YES indicates that you want session security for APPC sessions. If XAPPC=YES is specified and the APPCLU class is not activated in RACF, CICS fails to initialize. For more information on what happens in these circumstances, see "CICS initialization failures related to security" on page 258.

XAPPC enables RACF LU6.2 bind-time (also known as APPC) security. For more information, see "Bind-time security with LU6.2" on page 153.

For more information on the APPCLU class, see "APPCLU resource class" on page 29.

XUSER activates surrogate user security, and AUTHTYPE checking for DB2. For more information, see "Chapter 7. Surrogate user security" on page 103. If XUSER=YES is specified and the SURROGAT class is not activated in RACF, CICS fails to initialize.

## Using IBM-supplied classes without prefixing

To set up external security for transactions, files, and PSBs, using IBM-supplied resource classes and without prefixing, take the steps described in this section.

Before you define a profile, activate the relevant classes, using the SETROPTS CLASSACT and SETROPTS GENERIC commands, as described in "Brief summary of RACF commands" on page 19.

**To ensure the least interruption to actual business processes, work in a test region first**.

1. Plan and create RACF profiles in the relevant classes:

```
RDEFINE  TCICSTRN  transaction-name  UACC(NONE)  NOTIFY(userid)
RDEFINE  FCICSFCT  file-name         UACC(NONE)  NOTIFY(userid)
RDEFINE  PCICSPSB  PSB-name          UACC(NONE)  NOTIFY(userid)
```

2. Permit appropriate users or groups (preferably groups) to have access to the profiles:

```
PERMIT  transaction-name  CLASS(TCICSTRN)  ACCESS(READ)
        ID(userid or groupid)
PERMIT  file-name         CLASS(FCICSFCT)  ACCESS(READ)
        ID(userid or groupid)
PERMIT  PSB-name          CLASS(PCICSPSB)  ACCESS(READ)
        ID(userid or groupid)
```

3. Specify the following CICS system initialization parameters:

```
SEC=YES          XTRAN=YES        XCMD=NO
SECPRFX=NO       XFCT=YES         XDB2=NO
                 XPSB=YES         XDCT=NO
                                  XJCT=NO
                                  XPCT=NO
                                  XPPT=NO
                                  XTST=NO
                                  XUSER=NO
                                  XAPPC=NO
```

4. Start the CICS region in which you will be using external security.
5. If you add, change, or delete RACF profiles in the related classes, refresh the in-storage profiles. (For more information, see "Refreshing resource profiles in main storage" on page 27.)

## Using IBM-supplied classes with prefixing

To set up external security for transactions, files, and PSBs, using IBM-supplied resource classes with prefixing, take the steps described in this section.

Before you define a profile, you must activate the relevant classes, using the SETROPTS CLASSACT and SETROPTS GENERIC commands, as described in "Brief summary of RACF commands" on page 19.

**To ensure the least interruption to actual business processes, work in a test region first**.

**Note:** The following examples assume that the CICS region userid is CICS1.

1. Plan and create RACF profiles in the relevant classes:

```
RDEFINE  TCICSTRN  CICS1.transaction-name  UACC(NONE)  NOTIFY(userid)
RDEFINE  FCICSFCT  CICS1.file-name         UACC(NONE)  NOTIFY(userid)
RDEFINE  PCICSPSB  CICS1.PSB-name          UACC(NONE)  NOTIFY(userid)
```

2. Permit appropriate users or groups (preferably groups) to have access to the profiles:

```
PERMIT  CICS1.transaction-name  CLASS(TCICSTRN)  ACCESS(READ)
        ID(userid or groupid)
PERMIT  CICS1.file-name         CLASS(FCICSFCT)  ACCESS(READ)
        ID(userid or groupid)
PERMIT  CICS1.PSB-name          CLASS(PCICSPSB)  ACCESS(READ)
        ID(userid or groupid)
```

3. Specify the following system initialization parameters:

```
SEC=YES             XTRAN=YES        XCMD=NO
SECPRFX=YES         XFCT=YES         XDB2=NO
                    XPSB=YES         XDCT=NO
                                     XJCT=NO
                                     XPCT=NO
                                     XPPT=NO
                                     XTST=NO
                                     XUSER=NO
                                     XAPPC=NO
```

4. Start the CICS region in which you will be using external security.
5. If you add, change, or delete RACF profiles in the related classes, refresh the in-storage profiles. (For more information, see "Refreshing resource profiles in main storage" on page 27.)

## Using installation-defined classes without prefixing

To set up external security for transactions, files, and PSBs in installation-defined classes, without prefixing, take the steps described in this section. For an example of how to define installation-defined classes (T$USRTRN and G$USRTRN) for the XTRAN parameter, see the IBM-supplied sample, DFH$RACF, in CICSTS13.CICS.SDFHSAMP. See also "Specifying user-defined resources to RACF" on page 226.

Before you define a profile, activate the relevant classes, using the SETROPTS CLASSACT and SETROPTS GENERIC commands, as described in "Brief summary of RACF commands" on page 19.

**To ensure the least interruption to actual business processes, work in a test region first**.

1. Set up the following installation-defined classes:

   T$USRTRN like TCICSTRN, and G$USRTRN like GCICSTRN

   F$USRFCT like FCICSFCT, and H$USRFCT like HCICSFCT

   P$USRPSB like PCICSPSB, and Q$USRPSB like QCICSPSB

For specific information on setting up installation-defined classes, see the *OS/390 Security Server (RACF) System Programmer's Guide*.

1. Plan and create RACF profiles in the relevant classes:

```
RDEFINE  T$USRTRN  transaction-name  UACC(NONE)  NOTIFY(userid)
RDEFINE  F$USRFCT  file-name         UACC(NONE)  NOTIFY(userid)
RDEFINE  P$USRPSB  PSB-name          UACC(NONE)  NOTIFY(userid)
```

2. Permit appropriate users or groups (preferably groups) to have access to the profiles:

```
PERMIT  transaction-name  CLASS(T$USRTRN)  ACCESS(READ)
        ID(userid or groupid)
PERMIT  file-name         CLASS(F$USRFCT)  ACCESS(READ)
        ID(userid or groupid)
PERMIT  PSB-name          CLASS(P$USRPSB)  ACCESS(READ)
        ID(userid or groupid)
```

3. Specify the following system initialization parameters:

```
SEC=YES           XTRAN=$USRTRN         XCMD=NO
SECPRFX=NO        XFCT=$USRFCT          XDB2=NO
                  XPSB=$USRPSB          XDCT=NO
                                        XJCT=NO
                                        XPCT=NO
                                        XPPT=NO
```

```
XTST=NO
XUSER=NO
XAPPC=NO
```

4. Start the CICS region in which you will be using external security.
5. If you add, change, or delete RACF profiles in the related classes, refresh the in-storage profiles. (For more information, see "Refreshing resource profiles in main storage" on page 27.)

# Chapter 4. Verifying CICS users

This chapter covers all aspects of CICS sign-on security, including the use of the RACF CICS segment. It discusses the following:



## Identifying CICS terminal users

If you are running CICS with RACF security checking, you control users' access to CICS resources through levels of authorization you define in RACF-managed resource profiles. You define these authorizations for specific users by adding individual RACF userids (or RACF group IDs) to the resource access lists; or, for unsigned-on users, by adding the default CICS userid to selected resource access lists.

All CICS terminal-user data is defined in the RACF CICS segment. See "Obtaining CICS-related data for a user" on page 74 for more information about CICS terminal-user data, and how CICS obtains it.

## Sign-on process

When users log-on to CICS through VTAM (or TCAM DCB), but do not sign on, they can use only those transactions that the CICS default user is permitted to use. As these are likely to be strictly limited, users must sign on to obtain authorization to run the transactions that they are permitted to use.

### Explicit sign-on

Users can explicitly sign on either by using the CICS-supplied transaction, CESN, which can be defined as the "good morning" transaction on the GMTRAN system initialization parameter; or by using an installation-provided sign on transaction which uses the EXEC CICS SIGNON command. OIDCARD users can use CESN to sign on if the card reader supports the DFHOPID identifier (AID). If it does not, use your own installation-provided sign-on transaction. For information about CESN, see the *CICS Supplied Transactions* manual. For programming information about EXEC CICS SIGNON, see the *CICS Application Programming Reference* manual. When a user signs on to CICS, the sign-on process involves the following **phases**:

**Scoping**
: After the sign-on panel is completed and sent, CICS verifies that the entered userid does not match a userid already signed on within the scope of the SNSCOPE definition for the CICS system.

**Identification**
CICS calls RACF with the supplied userid to confirm that a profile has been defined for the user.

**Verification**
CICS passes information to RACF to verify that the user is genuine. For RACF this is either a password or an OIDCARD or both. If the password entered has expired, CICS prompts the user for a new password. When the new password conforms to the RACF password formatting rules for an installation, the new password and the date-of-change are recorded in the RACF user profile.

Immediately following the request to RACF for userid and password verification, CICS clears the internal password field. This minimizes the possibility of the password being revealed in any dump of the CICS address space that may be taken.

You may also voluntarily change your password by entering a new value.

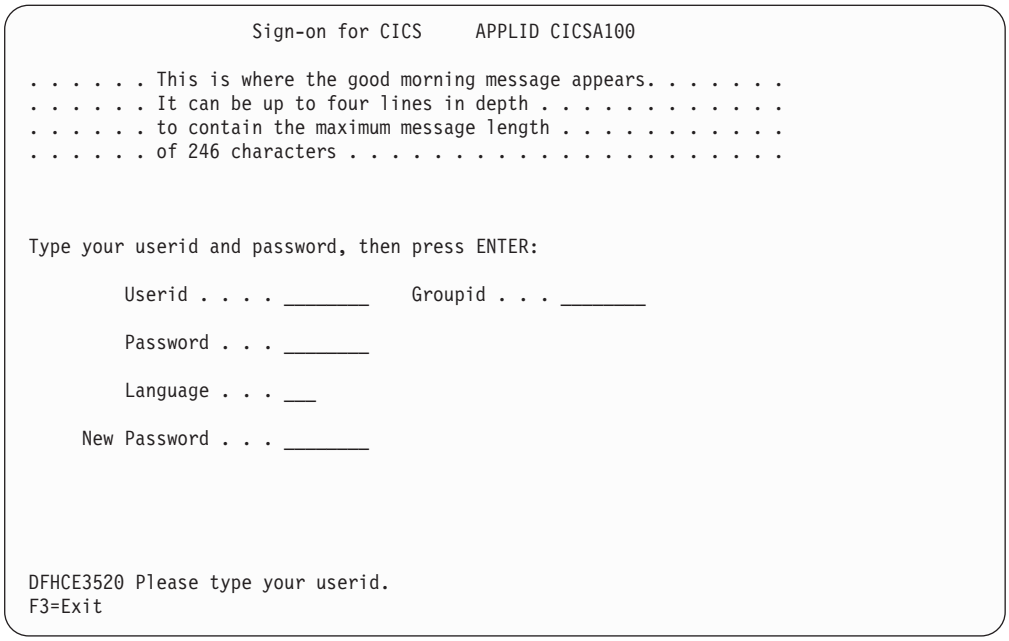



*Figure 2. The CICS sign-on panel*

**Authorization**
RACF performs checks on the application name and the port of entry to verify that the user is allowed to use the CICS system. In the application name check, RACF determines whether the user is authorized to access the application named in the APPLID or GRNAME system initialization parameter. RACF does this by checking the access list of the CICS application profile defined in the RACF APPL resource class. (See "Other IBM-supplied RACF resource class names affecting CICS" on page 27 for information about how to define profiles in the APPL resource class.)

With the port of entry check, RACF verifies that the user is authorized to sign on using that port of entry. The use of defined terminals can be restricted to

certain times of the day, and to certain days of the week. See "Controlling access to CICS from specific ports of entry" on page 68.

These checks restrict the user to signing on only to those CICS regions for which they are authorized, and only from terminals they are authorized to use.

Explicit sign-on, reached through CESN or EXEC CICS SIGNON, is performed by the user at the port of entry.

*Table 8. Explicit and implicit signons*

| Phase | Explicit | Implicit |
|---|---|---|
| Scoping | Yes | No |
| Identification | Yes | Yes |
| Verification | Yes | No except with ATTACHSEC(IDENTIFY) |
| Authorization | Yes | Yes |

**User attributes**
CICS obtains CICS user attributes from the CICS and LANGUAGE segments of the RACF database.

## Sign-off process

The sign-off process dissociates a user from a terminal where the user had been previously signed on. The user can explicitly sign off using the CESF transaction or an installation-provided transaction that uses the EXEC CICS SIGNOFF command. If the attributes of the signed-on user include a non-zero value for TIMEOUT, an implicit sign-off occurs if this interval expires after a transaction terminates at this terminal.

When the time-out period expires, if the default GNTRAN=NO is specified, CICS performs an immediate signoff. If GNTRAN specifies a transaction-id to be scheduled and that transaction performs a signoff, the action CICS takes depends on the SIGNOFF option specified in the terminal's TYPETERM resource definition.

An exceptional case is that the goodnight transaction is not used for the user of a CRTE session. A surrogate user whose time expires is signed off, losing the security capabilities the terminal previously had. Message DFHSN1200 is sent to the CSCS log, and indicates what has happened.

For more information about the use of system initialization parameter GNTRAN, see "Goodnight transaction" on page 236. The possible signoff options and the associated actions are as follows:

**SIGNOFF(YES)**
CICS signs off the operator from CICS, but the terminal remains connected.

**SIGNOFF(LOGOFF)**
CICS signs off the operator from CICS **and** logs off the terminal from VTAM.

In addition, if the terminal is autoinstalled, the delay period specified by the AILDELAY operand in the system initialization parameters commences, and if the delay period expires before the terminal attempts to log on again, CICS deletes the terminal entry (TCTTE) from the TCT. For information about CICS autoinstall, see the *CICS Resource Definition Guide*.

**SIGNOFF(NO)**
CICS leaves the user signed on and the terminal remains logged on, effectively overriding the time-out period.

## Explicit sign-off

Explicit sign-off removes the user's scoping. The user must be explicitly signed on before signing off with CESF or EXEC CICS SIGNOFF. The user is returned to the default level of security.

**Note:** CESN will not sign the user off until a valid attempt has been made to use the panel, even if the sign-on attempt subsequently fails. It is not recommended that CESN be used for the Goodnight transaction.

## Implicit sign-on and implicit sign-off

Implicit sign-on means that all other userids added to the system by CICS are considered to be implicitly signed on without a password. A user is implicitly signed off if the transaction suffers a TERMERR condition while attempting to send data to its principal facility. However, the user is not subject to USRDELAY but is signed off immediately. If SNSCOPE is in use, the scope will be released at the time of sign off. If the transaction handles the ABEND, it continues running as a non-terminal task with the authority of the starting user.

# Controlling access to CICS from specific ports of entry

During sign-on processing, CICS issues a request to RACF to verify the user's password, and to check whether the user is allowed to access that terminal. This check is also performed for the userid specified for preset security terminal definitions. Autoinstalled consoles that are using automatic sign-on are treated as though they have a preset security definition (see "Preset terminal security" on page 69). If the terminal is not defined to RACF, RACF responds to CICS according to the system-wide RACF option specified by the SETROPTS command. The options are as follows:

**TERMINAL(READ)**
With this option in force, terminal users can sign on at any terminal covered by a profile to which they have been permitted access, or at any terminal not defined as protected by RACF.

**TERMINAL(NONE)**
With this option in force, terminal users can sign on at only those terminals with specific terminal profiles defined to RACF, and which they are authorized to use.

**Note:** The TERMINAL class does not control access from MVS consoles. These are controlled by the CONSOLE resource class. See"Console profiles" on page 23.

You can override the system-wide terminal options at the RACF group level by means of the group terminal options, TERMUACC or NOTERMUACC.

See "Universal access authority for undefined terminals" on page 23 for more information about the SETROPTS command for terminals, and about the TERMUACC|NOTERMUACC option on groups.

## Auditing sign-on and sign-off activity

RACF can log all sign-on and sign-off activity to SMF, including any invalid or unsuccessful sign-on attempts. You can only properly interpret the logging of unsuccessful sign-on attempts by also recording successful sign-ons. For example, if a user makes one or two unsuccessful attempts followed immediately by a successful sign-on, the unsuccessful sign-ons can be interpreted as being caused by keying errors at the terminal. However, several unsuccessful attempts for a variety of userids occurring within a short space of time, and without any subsequent successful sign-on activity being recorded, may well be cause for a security concern that warrants investigation.

Recording the successful sign-on and sign-off activities establishes an audit trail of the access to particular systems by the terminal user population. This may also be useful for systems capacity planning, and generally constitutes a very modest portion of the information recorded to SMF.

CICS uses its CSCS transient data destination for security messages. Messages of interest to the security administrator for the CICS region are directed to this destination. In some instances, when security-related messages are directed to terminal users, corresponding messages are written to the CSCS transient data destination. In the case of the DFHCE3544 and DFHCE3545 messages that are sent to terminal users, for example, the corresponding messages DFHSN1118 and DFHSN1119 are sent to CSCS. The DFHSNxxxx messages include reason codes that indicate the precise nature of the invalid sign-on attempt.

## Preset terminal security

For some selected terminals, and MVS consoles when used as CICS terminals, consider using CICS preset terminal security as an alternative to terminal user security. A terminal becomes a preset security terminal when you specify the userid operand on the terminal definition.

There are two types of preset security for consoles:

1. Normal preset security (the same as preset security for other terminals)
2. Automatic preset security

### Normal preset security

CICS preset terminal security allows you to associate a userid permanently with a terminal, or console, that is defined to CICS. This means that CICS implicitly signs on the device when it is being installed, instead of a subsequent sign-on of that terminal by a user. Typically, you define preset security for devices without keyboards, such as printers, at which users cannot sign on.

You can also use the normal preset security on ordinary display terminals as an alternative to terminal user security. This permits anyone with physical access to a terminal with preset security to enter the transactions that are authorized for that terminal. The terminal remains signed on as long as it is installed, and no explicit sign-off can be performed against it. If the userid associated with a display terminal with preset-security has been authorized to use any sensitive transactions, ensure that the terminal is in a secure location to which access is restricted. Preset-security might be appropriate, for example, for the terminals physically located within a CICS network control center.

You can use preset-security to assign a userid with **lower** authority than the default, for terminals in unrestricted areas.

For example, to define a terminal with preset-security, use RACF and CICS (CEDA) commands as follows:

```
ADDUSER userid NAME(preset_terminal_user_name) OWNER(owner_userid or group_id)
               DFLTGRP(group_name)
CEDA DEFINE TERMINAL(cics_termid) NETNAME(vtam_termid) USERID(userid)
            TYPETERM(cics_typeterm)
```

For further information on preset-security terminals in the transaction routing environment, refer to "Preset-security terminals and transaction routing" on page 166 (LU6.2 security) and "Preset-security terminals and transaction routing" on page 208 (MRO security).

## Automatic preset security for consoles

Automatic preset security applies only to console definitions. CICS automatic preset security allows you to associate the userid, which MVS has already verified through RACF, with the CICS definition for the console. Instead of specifying an actual userid on the TERMINAL definition, you specify a special value (*FIRST or *EVERY), to indicate that CICS is to use the userid passed by MVS on the MODIFY command. This means that CICS implicitly signs on the console when it is being installed, and optionally on each input message, instead of a subsequent sign-on of that console by a user. Particularly in the context of autoinstalled consoles, this allows you to gain the advantage of preset security without having to define the userid/console relationship in the CICS terminal definition. Thus, console users do not have to sign-on with passwords in the clear to each CICS region.

You can use this automatic form of preset security on predefined consoles, autoinstalled consoles, and consoles installed by EXEC CICS CREATE commands.

For example, to define a console with automatic preset-security, which is checked, and altered (if necessary) on every MODIFY, use CICS (CEDA) commands as follows:

```
CEDA DEFINE TERMINAL(cics_termid)
            CONSNAME(consolename) USERID(*EVERY)
            TYPETERM(cics_typeterm)
```

To define a console with automatic preset-security that is defined on the first valid MODIFY command only, use CICS (CEDA) commands as follows:

```
CEDA DEFINE TERMINAL(cics_termid)
            CONSNAME(consolename) USERID(*FIRST)
            TYPETERM(cics_typeterm)
```

## Controlling the use of preset-security

When a preset-security terminal is installed, the specified userid is implicitly signed on at the terminal. Ensure that only a trusted person is allowed to define and install terminals with preset security, because the userid specified on the terminal may have access to CICS resources not available to the installer. Automatic preset security for consoles does not carry the same risks because the console user is associated with their true identity (verified by RACF). For this reason, no checking is carried out when a console device is defined to CICS with either USERID(*EVERY) or USERID(*FIRST).

Surrogate user checking ensures that a user is authorized to act for another user. Surrogate user checking can be enforced when a user installs a terminal that is preset for a different userid, and is specified by the RACF SURROGAT resource class. The CICS *userid*.DFHINSTL resource can be defined in the SURROGAT resource class for authorization to install terminals that are preset for that specific userid.

When a terminal is installed with a preset userid, the surrogate user is the userid performing the installation. See "Chapter 7. Surrogate user security" on page 103 for more information.

The CEDA command checks the authority of the user to install preset terminals. Consider, therefore, whether to restrict the following functions with a view to controlling who can define and install terminals with preset security:

- The CEDA transaction
- The SURROGAT resource class
- The XUSER system initialization parameter
- Batch access to the CSD using the DFHCSDUP utility
- The LOCK command for locking CSD definitions

**Note:** When CICS installs a GRPLIST that contains preset terminal definitions, no checking is done at initialization time. However, you can still ensure that you control who can define and install terminals and sessions with preset security by using the CEDA LOCK command to control the contents of GRPLIST groups.

## Restricting use of the CEDA transaction

If the CEDA transaction is enabled on your production CICS regions, restrict its use to authorized users. This gives you control over who can define resources, such as terminals, to CICS. See "Chapter 10. Security for CICS-supplied transactions" on page 125 for information about protecting CICS-supplied transactions.

## Using the SURROGAT resource class

Also ensure that you restrict who can install terminals with preset security, so that even when such terminals are defined in the CSD, only authorized users can install them on CICS. (This authority is additional to the authority needed to run CEDA.) The user must already have authority to run the CEDA transaction.

To define a surrogate profile and authorize a user to install a terminal definition with preset security, use the following commands:

```
RDEFINE userid1.DFHINSTL SURROGAT UACC(NONE)
PERMIT userid1.DFHINSTL CLASS(SURROGAT) ID(userid2) ACCESS(READ)
```

This permits *userid2* to install a terminal preset with *userid1*

## Defining the XUSER system initialization parameter

To ensure that CICS can perform surrogate user security checks on the use of the CEDA INSTALL command for terminals with preset security, define the XUSER system initialization parameter. See "CICS resource class system initialization parameters" on page 58 for information about defining the XUSER parameter.

## Restricting batch access to the CSD

You can also use the CSD batch utility program, DFHCSDUP, to define resources in the CSD. So that only authorized users are allowed to update the production CSDs,

you should restrict the access list on the CSD data set profile to the CICS region userids and other authorized users only. The INSTALL command is not available in DFHCSDUP.

### Using the LOCK command

CICS also installs resource definitions in the CSD during an initial or cold start, from the list of groups defined on the GRPLIST system initialization parameter. To control the addition of resource groups to the CICS startup group list, you should use the CEDA or DFHCSDUP LOCK command to lock the list. This protects the group list from unauthorized additions. Also, lock all the groups that are specified in this list.

**Note:** The OPIDENT of the signed-on user is used as the key for the LOCK and UNLOCK commands. For information about LOCK and UNLOCK, see the *CICS Resource Definition Guide*.

## Other preset security considerations

If you intend to use preset security, consider these additional topics:

- Autoinstall models
- Sessions with preset security
- Terminals defined in the TCT

### Autoinstall models

If you are using autoinstall models with preset security, CICS makes the same surrogate authorization check as for ordinary terminals when the model is installed. It does not check surrogate authorization when the autoinstall model is used to perform autoinstall for a device. Also, CICS does not make a surrogate authorization check when installing models defined with automatic preset security for consoles.

If an autoinstall model with a preset userid becomes invalid (for example, if the userid is revoked), any attempt to install a terminal with the model fails.

### Sessions

A session becomes governed by preset security if you specify the userid operand on the session definition. The same checking is performed if you install preset security sessions.

### Terminals defined in the terminal control table

For terminals defined in the terminal control table (TCT) (for example, TCAM DCB terminals), the userid is also defined in the TCT, and, when CICS initializes, it signs on these terminals. If the sign-on fails (for example, if the userid is revoked), the terminal is put out of service. If the userid later becomes valid (for example, if it is resumed), setting the terminal in service results in a successful sign-on. CICS does not perform a surrogate user check for these terminals.

# Using an MVS system console as a CICS terminal

If you intend to use an MVS system console as a CICS terminal, you may need authorization to use the MVS MODIFY command. This is done using the OPERCMDS resource class, and is described in “OPERCMDS resource class” on page 32.

We recommend that you specify automatic preset security on the console’s CICS terminal definition, so that the console user obtains the correct level of authority without explicitly performing a CICS signon (which exposes the password).

If preset security is not defined, console users must sign on to get authority different from the default user. In this case, the password can generally be seen on the console and system log. However, if CICS has been defined as an MVS subsystem in a JES2 system, you can use the HIDEPASSWORD=YES option of the DFHSSIxx member in SYS1.PARMLIB, which enables CICS to intercept the command and overwrite the password with asterisks. For details about defining CICS as an MVS subsystem, see the *CICS Transaction Server for OS/390 Installation Guide*.

The format of the CESN command, when entered from a console, is as follows:

```
MODIFY jobname,CESN [USERID=userid][,PS=password]
       [,NEWPS=newpassword][,GROUPID=groupid]
       [,LANGUAGE=language-code]
```

If any of the data entered on the CESN command is invalid, or if the password is missing or expired, CICS prompts the user to enter the missing or invalid data by issuing a system message that requires a response (a WTOR message). Provide a response using the REPLY command. When CICS prompts for a password, it uses a security routing code to ensure that the response is not recorded on the console or in the system hardcopy log. To terminate the sign-on process, a REPLY command with a null operand is required. That is, enter:

```
REPLY nn,
```

with nothing after the comma, where `nn` is the number of the message corresponding to the reply.

You can authorize TSO users to use the TSO CONSOLE command. (For information on this command, see the *OS/390 TSO/E System Programming Command Reference*, SC28-1972.) These users must be defined to CICS as consoles, using the CONSNAME option of the DEFINE TERMINAL command, or be supported by autoinstall for consoles, as described in the *CICS Resource Definition Guide*.

When the password parameter is omitted from the CESN command, RACF can produce a security violation message, ICH408I. CESN cannot distinguish a user defined with OIDCARD, NOPASSWORD from a user defined with a PASSWORD who intentionally omits the password. To establish whether to prompt for a PASSWORD or to reject the signon (a user defined with OIDCARD cannot sign on at a console), the signon must be attempted. If the signon fails, message ICH408I is produced, and CICS interprets the return code from RACF to determine whether the PASSWORD or OIDCARD authenticator is required.

These users can sign on using CESN, or you may prefer to use preset security (the normal preset security for CICS terminals, or automatic preset security for consoles). When the TSO user uses the CONSOLE command, that user's userid, by default, becomes a console name. (But it can be changed to be any name using the CONSNAME(*name*) option on the TSO CONSOLE command). This console name can then be used as a CICS terminal if there is a corresponding TERMINAL definition (or one can be autoinstalled) with the CONSNAME option in CICS. If another name has been specified, that name is the one CICS uses to communicate with the console. For example, it is possible for one TSO user to use a name that is the same as another TSO user's ID.

Furthermore, if the CONSOLE command is used to allow TSO operators to sign on to CICS with the CESN transaction, their passwords may be exposed on the TSO screen and in the MVS system log. These potential exposures can be removed by

defining the terminal as having preset security. We recommend that you use automated preset security for the following reasons:

- It means that TSO users do not have to sign on, which may expose their ID and password on the log.
- It means that you do not have to define a relationship, in a CICS definition, between a console name and a user, which may change frequently or become invalid.
- It allows you to define one autoinstall model which covers the majority of your console definitions and gives each user the correct level of preset security.

To define automatic preset security, specify USERID(*EVERY) to ensure that the correct user ID is signed on for every command, or USERID(*FIRST) to sign on the console using the userid that first issues a MVS MODIFY command to CICS, and retain this for subsequent commands.

- Choose USERID(*FIRST) if use of a console is restricted to one or more users who have similar security characteristics to CICS using RACF, and you don't use the user ID as an identifier in applications.
- Use USERID(*EVERY) if you need to ensure that each input request is tested to be sure that the console user has the correct security level. You should be aware that checking the user ID imposes an overhead on MODIFY, and changing the preset userid imposes another overhead which is equivalent to the console user signing on using CESN.

# Obtaining CICS-related data for a user

CICS obtains CICS-related data from one of the following sources:

- The CICS and LANGUAGE segments of the RACF profile
- Built-in CICS system default values.

This section explains how the data is obtained, for the default user and terminal users signing on.

## Obtaining CICS-related data for the default user

When implicitly signing on the CICS default user during initialization, CICS obtains attributes in the following way:

1. CICS calls RACF to request user data for the CICS default user from the CICS segment and the LANGUAGE segment. If the CICS segment **or** the LANGUAGE segment data is present for the default userid, RACF returns this data to CICS. See "CICS segment" on page 13 for details of the information that you can define in the CICS segment. See "LANGUAGE segment" on page 16 for details of the LANGUAGE segment.
2. If RACF does not return the CICS segment or LANGUAGE segment data for the default userid, CICS assigns the following built-in system default values:

   **National language**
   Obtained from the first operand on the NATLANG system initialization parameter. This defaults to US English if not specified.

   **Operator class**
   One (OPCLASS=1)

   **Operator identification**
   Blank (OPIDENT=' ')

   **Operator priority**
   Zero (OPPRTY=0)

**Timeout**
Zero (TIMEOUT=0)

**XRF signoff**
Signoff not forced (XRFSOFF=NOFORCE)

## Obtaining CICS-related data at signon

When handling an explicit sign-on for a CICS terminal user, CICS obtains the terminal user attributes in the following way:

1. CICS calls RACF to request data about the CICS terminal user from the CICS segment and the LANGUAGE segment. If the CICS segment **or** the LANGUAGE segment data is present for the terminal user, RACF returns this data to CICS. See "CICS segment" on page 13 for details of the information that you can define in the CICS segment. See "LANGUAGE segment" on page 16 for details of the LANGUAGE segment.
2. If RACF does not return the CICS segment or LANGUAGE segment data for the user, CICS uses the user attributes of the CICS default user, defined during system initialization. (See "Obtaining CICS-related data for the default user" on page 74.)

CICS obtains the national language attribute in the following order:

1. The LANGUAGE option on the CICS-supplied CESN transaction, or the LANGUAGECODE or NATLANG option of the EXEC CICS SIGNON command, if supported by CICS. A **supported** national language is a **valid** national language that has been specified in the NATLANG system initialization parameter and has the corresponding message definitions. See the *CICS System Definition Guide* for more information about defining this parameter.
2. The PRIMARY *primary-language* parameter in the LANGUAGE segment of the user's RACF profile, if supported by CICS.
3. The SECONDARY *secondary-language* parameter in the LANGUAGE segment of the user's RACF profile, if supported by CICS.
4. The NATLANG parameter in the CSD definition of the user's terminal.
5. The language established for the default user as described on page 74.

See "Appendix A. National Language" on page 319 for a list of valid national languages.

**Note:** CICS ignores the RACF default national language defined by the command:

```
SETROPTS LANGUAGE(PRIMARY(...) SECONDARY(...))
```

### Defining terminal users and user groups to RACF

You should plan to define your CICS terminal users in groups. For this purpose, try to place the users of CICS systems in groups for ease of administration. For example, you might consider that all users who have the same manager, or all users within an order entry function, are an administrative unit. You can define such users to RACF as **groups** of individual users who have similar access requirements to CICS system resources. See the *OS/390 Security Server (RACF) Security Administrator's Guide* for more information about:

- Access control and flexibility of operation for the system administrator
- Use of the group-SPECIAL attribute and its scope of control
- Reducing the need to refresh in-storage profiles

When you define a group, and then define users as members of that group, all the users in the group can access the resources to which the group has been given access.

The group structure selected depends on your own installation's requirements. Use the RACF command ADDGROUP to create a new group:

```
ADDGROUP groupname OWNER(userid)
```

Use the ADDUSER command to add new users to the group, defining the group name as the user's default group:

```
ADDUSER userid NAME(username) DFLTGRP(group_id)
        CICS(OPCLASS(1,2,..,n) OPIDENT(abc) OPPRTY(255) TIMEOUT(minutes)
             XRFSOFF(NOFORCE) LANGUAGE(PRIMARY(language))
```

You can make a terminal user a member of more than one group by using the CONNECT command to add the user to a group other than that user's default group:

```
CONNECT userid GROUP(groupname)
```

Use the ALTUSER command to change a user's default group, as follows:

```
ALTUSER userid DFLTGRP(groupname)
```

Use the ALTUSER command to add CICS data for an existing userid. See "CICS segment" on page 13 for details of the CICS optional data.

See the *OS/390 Security Server (RACF) Command Language Reference*for the full syntax of these commands.

## Example of defining terminal users and user groups to RACF

Assume there is a customer service department that:

- Takes orders
- Answers enquiries about those orders
- Establishes new customers

Consider creating the following customer service group:

```
ADDGROUP custserv OWNER(grpmangr)
```

In this example, *grpmangr* is the RACF userid of the person in charge of the customer service department system.

The person represented by *grpmangr*, or the RACF security administrator, can then create additional groups within the group CUSTSERV, as follows:

```
ADDGROUP ORDERS OWNER(SUP1) SUPGROUP(CUSTSERV)
ADDGROUP ORDINQ OWNER(SUP2) SUPGROUP(CUSTSERV)
ADDGROUP NEWCUST OWNER(SUP3) SUPGROUP(CUSTSERV)
```

The group owners, the person represented by *grpmangr* or the RACF security administrator can then define users within the groups. For example, the person represented by SUP1 could define users of the group ORDERS, as follows:

```
ADDUSER AARCHER NAME('ANNE ARCHER') DFLTGRP(ORDERS)
ADDUSER JBRACER NAME('JOHN BRACER') DFLTGRP(ORDERS) PASSWORD(XPRDTD)
           CICS(OPCLASS(1) OPIDENT(JBR) OPPRTY(0) TIMEOUT(15) XRFSOFF(FORCE))
           LANGUAGE(PRIMARY(ENU))
```

**Notes:**

1. The password of the user Anne Archer defaults to ORDERS, but the password of the user John Bracer is initially set as XPRDTD.
2. The user John Bracer is defined with a CICS segment and with a LANGUAGE segment.

## National language and non-terminal transactions

When a user specifies a national language during sign-on, the sign-on option overrides the language specified in the user's RACF CICS segment. The language thus specified is set for the that the user is signed on at the terminal. Any transaction invoked by the signed-on user runs with the national language specified on the sign-on.

However, if a transaction uses the EXEC CICS START command to start another transaction, the national language attribute for the started transaction is derived as follows:

1. If the USERID parameter is specified on the START command, the national language is taken from the RACF CICS segment of the specified userid.
2. If the user is signed on at a terminal with a preset national language specified on the terminal definition, this preset national language is assigned to the started transaction.
3. If there is no userid on the START command, and no preset national language on the terminal, the started transaction inherits the national language specified in the RACF CICS segment of the signed-on user (not the national language used in the sign-on).

If the national language of the original terminal is required, the terminal's national language can be inquired about before the EXEC CICS START command is issued. The information can then be passed as data in the START command for the use of the transaction that has been started.

# Chapter 5. Transaction security

CICS can apply two levels of security to a transaction. The first is security checking on the transaction itself, sometimes referred to as **attach-time**, or **transaction-attach security**. This chapter discusses transaction-attach security—the security checks that CICS performs to verify that a terminal user is authorized for the transaction to be run at the user's terminal.

Transaction-attach security applies to transactions that a user enters directly at a terminal, and also to transactions started from another CICS transaction.

The other level of security you can use for CICS transactions applies to the resources used by the transactions: files, databases, PSBs, and CICS commands. For more information, see "Chapter 6. Resource security" on page 85.

This chapter discusses transaction-attach security under the following main headings:



## CICS parameters controlling transaction-attach security

You control CICS transaction-attach security checking through CICS system initialization parameters. These are:

**SEC** Specify SEC=YES if you want to use RACF services to control access to any CICS resources—in particular, CICS transactions. (For more information, see "SEC" on page 56.)

**SECPRFX**
Specify SECPRFX=YES if your transaction profiles are defined to RACF with a prefix that corresponds to the userid of the CICS region. (For more information, see "SECPRFX" on page 56.)

**XTRAN**
Specify XTRAN=YES or XTRAN=*resource_class_name* if you want CICS to control who can initiate transactions. If you specify YES, CICS uses profiles defined in the RACF default resource classes TCICSTRN and GCICSTRN. (See "IBM-supplied resource class names for CICS" on page 26 for details of these resource classes.)

If you specify XTRAN=NO, CICS does not perform any authorization check on users initiating transactions.

Note that the default is YES. Therefore if you specify SEC=YES and omit the XTRAN parameter, transaction-attach security is in effect, using the default resource class names.

There are no CICS parameters that allow you to control transaction-attach security at the individual transaction level. When you specify SEC=YES and XTRAN=YES (or XTRAN=resource_class_name), CICS issues an authorization request for every transaction. It does this whether the transaction is started from a terminal, by using an EXEC CICS START command, or triggered from the transient data queue, either

with or without the termid operand. CICS performs this security check even if no user has signed on. Users who do not sign on can use only those transactions that are authorized to the default user.

Figure 3 shows the main elements of CICS transaction security.

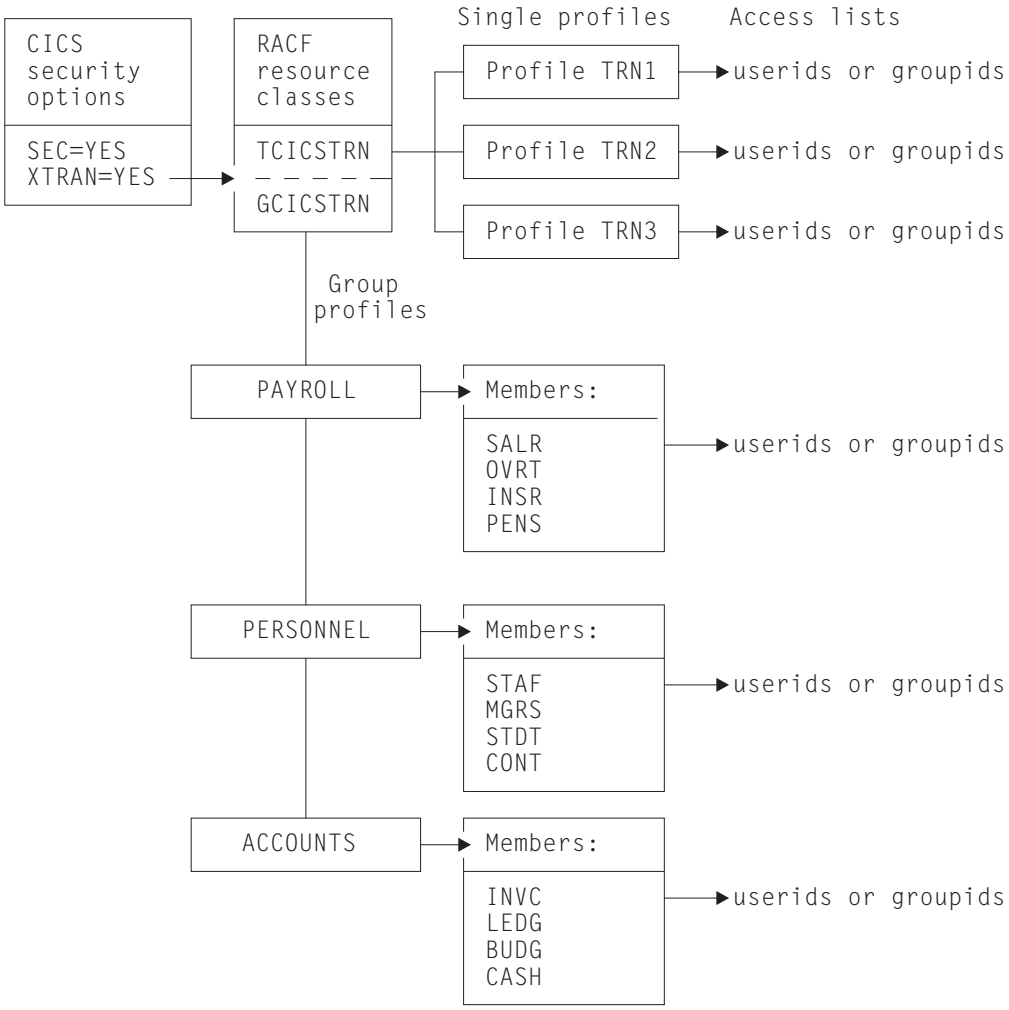



*Figure 3. Illustration of the main elements of CICS transaction security*

## Transaction-attach processing when SEC=YES and XTRAN=YES

Every time a transaction is initiated at a CICS terminal, CICS issues an authorization request to determine whether the user associated with the terminal is authorized for that transaction. CICS and RACF process the authorization request using the currently active transaction profiles in the RACF class identified by the XTRAN SIT parameter. (For more information, see "Refreshing resource profiles in main storage" on page 27.)

## Defining transaction profiles to RACF

For those CICS regions running with transaction security checking, define transaction profiles for all transactions that need to be protected from unauthorized access. You can define these profiles either in the default transaction resource classes, or in installation-defined classes that you have added to the RACF class descriptor table. (See "IBM-supplied resource class names for CICS" on page 26 for information about the transaction resource classes.)

### Some recommendations

The following recommendations are intended to reduce the amount of work involved:

- Define transactions in the resource group class, GCICSTRN. This minimizes the amount of effort needed to define and maintain transaction profiles and their associated access lists, and also keeps down the size of in-storage profiles. However, note that using resource groups only reduces the amount of storage required if you avoid defining duplicate member names.
- Add users to the access list in groups rather than as individual users, and define access as READ.
- Use generic profiles or member names wherever possible.

For example, the following RDEFINE and PERMIT commands illustrate some payroll transactions, with access given to members of the payroll department:

```
RDEFINE  GCICSTRN  salarytrans
         NOTIFY(pay_manager)
         UACC(NONE)  ADDMEM(Pay1, Pay2, Pay3,..., Payn)
PERMIT  salarytrans CLASS(GCICSTRN)
        ID(paydept_group_userid) ACCESS(READ)
```

In this example, you could instead define the members generically, such as `P*` or `Pay*`.

However, before you define a generic profile you must issue the command:

```
SETROPTS GENERIC(TCICSTRN)
```

You cannot specify the GCICSTRN class, because you cannot group classes with the SETROPTS GENERIC command.

If you have transactions that anyone can use, you can avoid maintaining access lists for them by defining RACF transaction profiles for them with UACC(READ). For example:

```
RDEFINE TCICSTRN tranid UACC(READ)
```

If you want to avoid defining any of your transactions to RACF, you can specify universal access as follows:

```
RDEFINE TCICSTRN ** UACC(READ)
```

You then need to define to RACF only those transactions that require more restrictive security.

**Note:** If you use a profile like that described above, define new profiles to RACF before installing new CICS resources.

### Using conditional access lists for transaction profiles

You can add another element of security by making the access list conditional upon the user being signed on at a particular terminal or console.

For example, if the earlier payroll examples are defined as generic transactions in the TCICSTRN class, you could define conditional access as follows:

```
RDEFINE  TCICSTRN  PAY*
         NOTIFY(pay_manager)  UACC(NONE)
PERMIT   pay*  CLASS(TCICSTRN)  ID(userid) ACCESS(READ)
               WHEN(TERMINAL(termid))
               WHEN(CONSOLE(*))
```

**Notes:**

1. The TERMINAL or CONSOLE class must be active for this support to take effect.
2. WHEN(TERMINAL(*termid*)) applies only to explicitly signed-on users, and only in the region where the user is explicitly signed on, and in regions connected to it by MRO links only.
3. CICS uses only the console and terminal ports of entry.

### CEBT transaction

The CEBT transaction (the master terminal transaction used to control the alternate CICS system in an XRF environment) is not subject to transaction security checking. This means that any user is authorized to use CEBT. CEBT can only be issued from the operating system console, using the MODIFY command. You can use the OPERCMDS resource class to control who is allowed to use the MODIFY command. (For more information, see "OPERCMDS resource class" on page 32.)

## Authorization failures and error messages

If a terminal user tries to initiate an unauthorized transaction, CICS issues a security violation message (DFHAC2033) to the terminal. CICS then sends a corresponding message (DFHAC2003) to the CSMT transient data destination, and a DFHXS1111 message to CSCS. RACF issues an ICH408I message to the CICS region's job log and to the security console (the console defined for routing code 9 messages). For a description of the ICH408I message, see the *OS/390 Security Server (RACF) Messages and Codes* manual.

For more information on resolving authorization problems, see "Chapter 21. Problem determination in a CICS-RACF security environment" on page 251.

If auditing (such as that requested by the AUDIT operand) is requested for this access, RACF writes an SMF type 80 log record. Your RACF auditor can use the RACF report writer to generate reports based on these records. For more information, see the *OS/390 Security Server (RACF) Auditor's Guide*.

## Transactions not associated with a terminal

For all resource security checking, CICS needs a userid in order to check the user's authority to access the resource. CICS can protect resources against unauthorized use if those resources are used in transactions that are not associated with a terminal. In addition to transactions started by an EXEC CICS START command without a terminal identifier specified, there are two other types:

- Transactions started without a terminal when the trigger level is reached for an intrapartition transient data queue

- Programs executed from the second phase of the program list table (PLT) during CICS startup

## Triggered transactions

The CEDA transaction, the DFHDCT macro, the CEDA DEFINE TDQUEUE, the EXEC CICS CREATE, and the ATIUSERID option of the EXEC CICS SET command establish security for non-terminal transactions started by a transient data trigger level. The user issuing the SET, INSTALL, or CREATE command must have surrogate authority for the userid specified on the ATIUSERID option. The user to be associated with the triggered transaction is specified on the USERID attribute of the transient data queue definition.

## PLT programs

If PLT programs are to be executed during CICS startup, CICS performs a surrogate user security check for the region userid. See "Defining user profiles for CICS region userids" on page 41. This check determines whether the CICS job is authorized to be the surrogate of the userid specified on the PLTPIUSR parameter. The PLTPIUSR and PLTPISEC system initialization parameters specify security options for PLT programs that are run from the third stage of CICS startup (which is the second phase of the PLTP initialization).

If the PLTPIUSR parameter is not specified, the PLT programs are run under the CICS region userid when the PLTPISEC=NONE option is defined. No surrogate check is required for this. If your PLT programs issue START commands, the jobstep userid has surrogate authority to start them when no userid is coded. Note that the starter always has surrogate authority to itself. When the started transaction starts up, another check is made to see if the userid has authority to attach the transaction and access this transaction in the TCICSTRN class. Rather than giving the jobstep access to additional resources, you can use the PLTPIUSR and PLTPISEC parameters.

During shutdown, CICS runs PLT programs under the authority of the userid for the transaction that requested the shutdown. The values of the RESSEC and CMDSEC options for that transaction are also applied to the PLT programs. If RESSEC(YES) and CMDSEC(YES) are specified on the definition of the transaction issuing the EXEC CICS PERFORM SHUTDOWN command, security checking is done at the first stage of shutdown.

# Chapter 6. Resource security

This chapter describes the facilities provided by CICS and RACF for controlling access to resources protected by RACF general resource security classes. They are discussed in the following sections:

- "General resource security checking by CICS and RACF"
- "Security for general resource types" on page 89
- "Security checking of transactions running under CEDF" on page 99
- "Defining generic profiles for resources" on page 100

"Chapter 5. Transaction security" on page 79 described how to control access to CICS transactions, using CICS transaction-attach security. This chapter describes how you can implement a further level of security, by controlling access to the resources used by the CICS transactions. The implication of this is that a user who is authorized to invoke a particular CICS transaction may not be authorized to access files, PSBs, or other general resources used within the transaction. Unlike transaction-attach security, which cannot be turned off for individual transactions, you can control resource security checking at the individual transaction level.

Resources defined to CICS to support application programming languages are also subject to security checking if resource or command security checking is specified. For example, if a PL/I program abends, it may attempt to write diagnostic information to the CPLI transient data queue. If resource checking is active, and the user is not authorized to write to the CPLI transient data queue, the program will terminate with an APLI abend.

You control who can access the general resources used by CICS transactions, by:

- Specifying SEC=YES as a system initialization parameter
- Specifying RESSEC=ALWAYS as a system initialization parameter
- Specifying RESSEC(YES) in the transaction resource definition
- Specifying the types of resource you want to protect by defining CICS system initialization parameters for the RACF general resource classes
- Defining the CICS resources to RACF in resource class profiles, with appropriate access lists

## General resource security checking by CICS and RACF

CICS uses RACF to protect the general resources that you can access through a CICS application program. Each resource is described briefly in Table 9 on page 86, with the associated CICS system initialization parameter that you use to specify the RACF class name.

Note that no authorization processing is done for BMS commands.

*Table 9. General resource checking by CICS*

| CICS parameter | General resource protected | Further information |
|---|---|---|
| XAPPC | Partner logical units (LU6.2). This resource is included in this list for completeness, but is not discussed in this chapter. | "Chapter 13. Implementing LU6.2 security" on page 153. |
| XCMD | The subset of CICS application programming commands that are subject to command security checking. This resource is included in this list for completeness, but is not discussed in this chapter. EXEC CICS FEPI system commands are also controlled by this parameter. | "Chapter 8. CICS command security" on page 109. |
| XDB2 | DB2 resource classes for DB2ENTRY, are specified to CICS on the XDB2 system initialization parameter | "Resource classes for DB2ENTRYs" on page 28. |
| XDCT | CICS extrapartition and intrapartition transient data destinations, also known as queues. Define profiles in the destination class to control who is allowed to access CICS transient data queues. | "Transient data" on page 89. |
| XFCT | CICS file-control-managed VSAM and BDAM files. Define profiles in the file class to control who is allowed to access CICS VSAM and BDAM files. | "Files" on page 91. |
| XJCT | CICS system log and general logs. Define profiles in the journal class to control who is allowed to access CICS journals on CICS log streams. | "Journals and log streams" on page 92. |
| XPCT | CICS started transactions and EXEC CICS commands: COLLECT STATISTICS TRANSACTION, DISCARD TRANSACTION, INQUIRE TRANSACTION, INQUIRE REQID, SET TRANSACTION, and CANCEL. Define profiles in the started-transactions class to control who is allowed access to started CICS transactions. | "Started and XPCT-checked transactions" on page 93. |
| XPPT | CICS application programs. Define profiles in the program class to control who is allowed to access CICS application programs that a CICS application invokes by means of a LINK, XCTL, or LOAD command. | "Application programs" on page 96. |
| XPSB | DL/I program specification blocks (PSBs). Define profiles in the program specification block class to control who is allowed to access the DL/I PSBs used in CICS application programs. | "Program specification blocks" on page 98. |
| XTRAN | CICS transactions. This resource is included in this list for completeness, but is not discussed in this chapter. | "Chapter 5. Transaction security" on page 79. |
| XTST | CICS temporary storage destinations. Define profiles in the temporary storage class to control who is allowed to access CICS temporary storage queues. | "Temporary storage" on page 97. |
| XUSER | Surrogate user security. This resource is included in this list for completeness, but is not discussed in this chapter. | "Chapter 7. Surrogate user security" on page 103. |

## RESSEC transaction resource security parameter

Specifying RESSEC(YES) in the definition of a transaction, together with the appropriate resource classes defined in the system initialization parameters, introduces another layer of security checking in addition to the transaction-attach security described in "Transaction-attach processing when SEC=YES and XTRAN=YES" on page 80.

For most simple (or single-function) transactions, this extra layer of security is not necessary. For example, if the transaction is designed to enable the terminal user to

update a personnel file and nothing else, it should be sufficient to authorize access to the transaction without controlling access to the file also. However, if you have a complex transaction that offers users a choice of functions, or you are unsure about all the options available within a transaction, you may want to add the extra layer of security to restrict access to the data as well as to the transaction. Before implementing resource security checking, take into account the extra overhead that resource security checking involves, and only implement it if you believe the extra cost is worthwhile.

If you specify RESSEC(YES) on a transaction definition, CICS calls RACF for each CICS command that applies to a resource for which you have requested security, using an X*name* resource class parameter. This is shown in Figure 4, in which the execution of transaction TRN1 results in seven RACF calls.

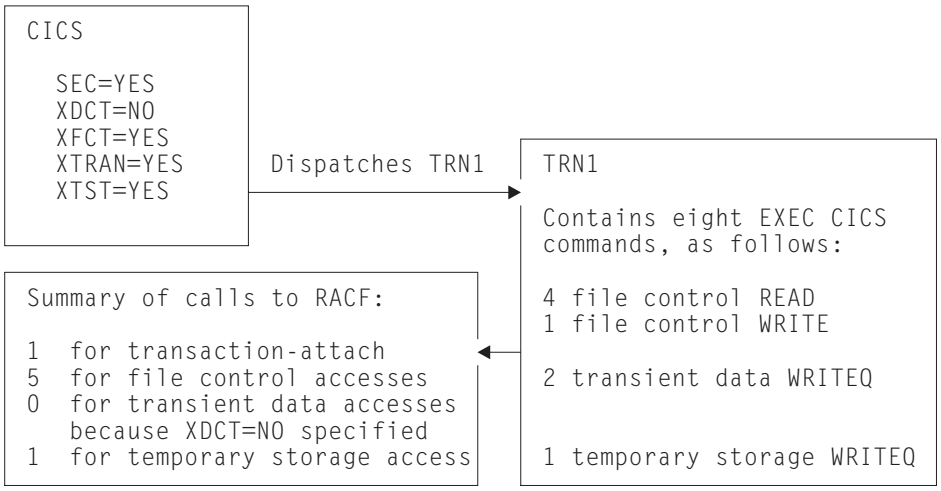



*Figure 4. Multiple calls to RACF with resource security checking*

## The RESSEC system initialization parameter

You can force the effect of RESSEC=YES for all CICS transactions by specifying the RESSEC=ALWAYS system initialization parameter. In general, this is not recommended, for the following reasons:

- For most simple transactions, just controlling access to the transaction is enough to control everything that the transaction can do.
- Invoking a resource check for every CICS resource consumes extra overhead that reduces the performance of all your transactions.
- Some CICS-supplied transactions may access resources of which you are unaware. It is your responsibility to ensure that users of these transactions are given enough authority to allow the transactions to continue to work.

## Authorization failures

If a terminal user is not authorized to access the resource specified on a CICS command, CICS returns the NOTAUTH condition to the application program. CICS indicates this authorization failure by setting the EIBRESP field of the EXEC interface block (DFHEIBLK) to a value of 70 (and X'46' in byte 0 of the EIBRCODE field). Design your CICS applications to handle security violations by passing control to an appropriate routine. They can do this in either of the following ways:

- Test the EIBRESP condition by adding the RESP option to each command that may receive a NOTAUTH condition. For example (in COBOL):

```
EXEC CICS FILE('FILEA')
   INTO(REC) RIDFLD(KEY)
        RESP(COMMAND-RESPONSE)
END-EXEC.


EVALUATE COMMAND-RESPONSE
   WHEN DFHRESP(NORMAL)
        CONTINUE
   WHEN DFHRESP(NOTAUTH)
    PERFORM SECURITY-ERROR
END-EVALUATE.
```

- Code an EXEC CICS HANDLE CONDITION NOTAUTH(*label*) command, where *label* is the name of the security violation routine.

If an application does not cater for security violations, CICS abends the transaction with an AEY7 abend code.

## Logging RACF audit messages to SMF

Except when processing certain security commands (see "Chapter 9. Security checking using the QUERY SECURITY command" on page 117), CICS issues security authorization requests with the logging option. This means that RACF writes SMF type 80 log records to SMF. Which events are logged depends on the auditing in effect. For example, events requested by the AUDIT or GLOBALAUDIT operand in the resource profile, or by the SETROPTS AUDIT or SETROPTS LOGOPTIONS command, can be logged.

In addition to the SMF TYPE 80 log record, RACF issues an ICH408I message to consoles designated to receive messages for route code 9.

For more information on auditing, including how to use the RACF report writer to review SMF type 80 log records, see the *OS/390 Security Server (RACF) Auditor's Guide*.

### Use of the WARNING option

The RACF WARNING option, if used on RACF profiles, is honored by CICS. The WARNING option allows users access to resources that otherwise would be denied. RACF logs to SMF those accesses that would have failed had WARNING not been in effect.

The selective use of WARNING can be particularly useful during the initial implementation of resource security for an application, as a means of checking for errors or omissions in the RACF security definitions. When WARNING results in an SMF type 80 record being recorded, you should verify whether the user should be added to the access list for the resource, and modify the RACF profiles accordingly. You should strictly limit the time during which resources are accessed with the warning option in force, and keep logging to a minimum during the warning period.

**Note:** Specify the NOTIFY option, if you want to be notified at once when access is denied to a user.

## Security for general resource types

This section discusses some of the resource types for which security can be implemented. This includes:



### Transient data

To implement security for transient data destinations (queues), do the following:

1. Specify RESSEC(YES) in the CSD resource definition of the appropriate transactions.
2. Define profiles to RACF in the DCICSDCT or ECICSDCT resource classes (or their equivalent if you have user-defined resource class names), with access lists as appropriate. Transient data queue names are a maximum of 4 characters in length, such as CSMT, CPLI, L86O, L86P, and so on.

   For example, use the following commands to define queues in the DCICSDCT class, and to authorize users to both read from and write to these queues:

   ```
   RDEFINE  DCICSDCT  (qid1, qid2, ..., qidn) UACC(NONE)
                      NOTIFY(sys_admin_userid)
   PERMIT qid1 CLASS(DCICSDCT) ID(group1, group2) ACCESS(UPDATE)
   PERMIT qid2 CLASS(DCICSDCT) ID(group1, group2) ACCESS(UPDATE)
   ```

   To define transient data queues as members of a profile in the CICS transient data resource group class, with an appropriate access list, use the following commands:

   ```
   RDEFINE  ECICSDCT  (queue_groupname) UACC(NONE)
                      ADDMEM(qida, qidb, ..., qidz) NOTIFY(sys_admin_userid)
   PERMIT queue_groupname CLASS(ECICSDCT) ID(group_userid) ACCESS(UPDATE)
   ```
3. Specify SEC=YES as a CICS system initialization parameter (and SECPRFX=YES if you define profiles with the CICS region userid as a prefix).
4. Specify XDCT=YES for the default resource class names of DCICSDCT and ECICSDCT (or XDCT=class_name for user-defined resource class names).

#### Defining profiles for transient data queues

When you are defining profile names to RACF to control access to transient data queues, define profiles only for queues that are defined to CICS as follows:

**TYPE=INTRA**
: For an intrapartition transient data queue held on the CICS intrapartition (VSAM) data set, DFHINTRA. When the destination facility is a file, you can specify a USERID. See "Considerations for triggered transactions" on page 90 for more information about intrapartition TD queues in this category, and "Transient data trigger-level transactions" on page 105 for more information about the USERID specification.

**TYPE=EXTRA**
: For an extrapartition transient data queue on a sequential data set.

If you define an indirect queue, CICS directs this to another queue, which can be extrapartition, intrapartition, or remote. The redirection can even be to another indirect queue. See the *CICS Resource Definition Guide* for more information about defining CICS transient data queues.

If you are running CICS with security checking for transient data queues, CICS issues a call to RACF for each command that specifies a queue name. However, the resource name that CICS passes to RACF is the queue name of the final queue, which is not necessarily the name of the queue specified on the command.

For example, if an EXEC CICS command specifies queue QID2, which is defined as indirect to QID1, CICS calls RACF for an authorization check on QID1, not QID2. This is illustrated as follows:

```
TDQ definition: DEFINE TDQUEUE(QID1)
                       TYPE(EXTRA)
                       TYPEFILE(OUTPUT)
                       RECORDSIZE(132)
                       BLOCKSIZE(136)
                       RECORDFORMAT(VARIABLE)
                       BLOCKFORMAT(UNBLOCKED)
                       DDNAME(CICSMSGS)
                       GROUP(DFHDCTG)

                DEFINE TDQUEUE(QID2)
                       TYPE(INDIRECT)
                       INDIRECTNAME(QID1)
                       GROUP(DFHDCTG)

 CICS transaction:   EXEC CICS WRITEQ TD
                          QUEUE(QID2)
                          FROM(data_area)
                          LENGTH(length)
 CICS calls RACF:    Does the terminal user of the CICS transaction
                     have UPDATE authorization for QID1?
```

## Access authorization levels

You can read an item from a transient data queue only once, because whenever you read from a transient data queue, CICS deletes the entry (by performing a "destructive read"). Therefore, if you specify security with SEC=YES as a system initialization parameter, CICS requires a minimum authorization level of UPDATE for all TD commands (DELETEQ, WRITEQ, and READQ).

## CICS-required destination control table entries

CICS itself uses a number of queues. These queues are defined in the sample group, DFHDCTG. If you want to protect access to these definitions from user application programs, define them to RACF with UACC(NONE) and without an access list. In the sample table, most of the queue names are indirect, pointing to the final queues: CPLI, CSSL, or CCSO. Therefore, if you use the definitions as supplied, you need define to RACF only the queue names CPLI, CSSL, and CCSO, as follows:

```
RDEFINE  ECICSDCT  CICSQUEUES UACC(NONE)
                   ADDMEM(CPLI, CSSL, CCSO)
                   NOTIFY(sys_admin_userid)
```

## Considerations for triggered transactions

For intrapartition TD queues with a trigger level greater than zero, CICS derives the userid associated with the triggered transaction from the following sources:

- The USERID parameter specified on the intrapartition transient data resource definition (DESTFAC=FILE).

- The userid associated with the terminal (for queues that have been defined with a destination facility of terminal) (DESTFAC=TERMINAL). This can be the CICS default userid if no user is signed on at the terminal.
- The link userid on the connection definition (for queues that have been defined with a destination facility of system) (DESTFAC=SYSTEM).

## Files

CICS application programs process files, which, to CICS, are logical views of physical VSAM or BDAM data sets. You identify a file to CICS by an 8-character file name, and you can define many files to CICS that refer to the same physical data set, which is separately identified by a 44-character data set name (DSNAME). For example, you can define file resource definitions called FILEA, FILEB, and FILEC, all of which refer to one physical VSAM data set, but with each file definition specifying different attributes.

CICS transactions access the data in physical data sets using the CICS file control name. Therefore, you control access to CICS-managed files by defining profiles in the RACF general resource classes for CICS files, not in the RACF data set class. You define the profiles using the CICS 8-character file name to identify the resource. (RACF data set authorization based on the 44-character data set name is used only during OPEN processing, to determine whether the CICS region userid is authorized to access the data set for which the OPEN has been requested. This does not depend on the userid running the transaction that caused the OPEN to be performed.)

To implement security for files managed by CICS file control:

1. Specify RESSEC(YES) in the CSD resource definition of the transactions that access the files.
2. Define profiles to RACF in the FCICSFCT or HCICSFCT resource classes (or their equivalent if you have user-defined resource class names), using the CICS file names to identify the profiles. For example, use the following commands to define files in the FCICSFCT class, and authorize users to read from or write to the files:

   ```
   RDEFINE  FCICSFCT  (file1, file2, .., filen) UACC(NONE)
                    NOTIFY(sys_admin_userid)
   PERMIT file1 CLASS(FCICSFCT) ID(group1, group2) ACCESS(UPDATE)
   PERMIT file2 CLASS(FCICSFCT) ID(group1, group2) ACCESS(READ)
   ```

   To define files as members of a profile in the CICS file resource group class, with an appropriate access list, use the following commands:

   ```
   RDEFINE  HCICSFCT  (file_groupname) UACC(NONE)
                     ADDMEM(filea, fileb, .., filez) NOTIFY(sys_admin_userid)
   PERMIT file_groupname CLASS(HCICSFCT) ID(group_userid) ACCESS(UPDATE)
   ```
3. Specify SEC=YES as a CICS system initialization parameter (and SECPRFX=YES if you define profiles with the CICS region userid as a prefix).
4. Specify XFCT=YES for the default resource class names of FCICSFCT and HCICSFCT (or XFCT=class_name for user-defined resource class names).

Note that RDO transactions do not use file commands to access the CSD, and are not, therefore, subject to these mechanisms.

### Access authorization levels

If you specify security with SEC=YES as a system initialization parameter, CICS requires a level of authorization appropriate to the file access intended: a minimum of READ for read intent, and a minimum of UPDATE for update or delete intent.

## Journals and log streams

The CICS log manager provides facilities to write to and read from:

- The CICS system log
- The CICS general logs, which comprise user journals, forward recovery logs, and autojournals

The system log is used only for recovery purposes—for example, during dynamic transaction backout, or during emergency restart. Do not use it for any other purpose. Do not, therefore, write to it from a user application using the EXEC CICS WRITE JOURNALNAME command.

CICS uses journal identifier **DFHLOG** for its primary system log. Do not permit user transactions to write to this. You can prevent them doing so by using the following command to define the system log in the JCICSJCT class, without any access list:

```
RDEFINE  JCICSJCT  DFHLOG  UACC(NONE) NOTIFY(sys_admin_userid)
```

In addition to the automatic journaling and forward recovery logging that CICS performs for user transactions (depending on the options in the file resource definitions), user applications can also write user journal records using the EXEC CICS WRITE JOURNALNAME command.

Users needing to write journal records must have authority to write to the JOURNALNAME (as defined in JCICSJCT). CICS calls RACF to perform a security check only for attempts to access a user journal by a CICS API command, and not for the journaling it performs in response to journaling options in the file resource definition. The CICS API does not provide a READ command for reading journals from a CICS transaction. For this reason, with proper exercise of control over the installation of applications on your CICS systems, you might consider it unnecessary to add RACF protection for journals that cannot be read from within CICS.

If you decide to implement security for CICS journals:

1. Specify RESSEC(YES) in the CSD resource definition of the transactions that write to journals.
2. Define profiles to RACF in the JCICSJCT or KCICSJCT resource classes (or their equivalent if you have user-defined resource class names) using the CICS journal name to identify the profiles.

   To define journals as members of a profile in the journal resource group class, with an appropriate access list, use the following commands:

   ```
   RDEFINE  KCICSJCT  userjnls UACC(NONE)
                      ADDMEM(JRNL001, JRNL002, ....)
                      NOTIFY(sys_admin_userid)
   PERMIT   userjnls  CLASS(KCICSJCT) ID(group_userid) ACCESS(UPDATE)
   ```
3. Specify SEC=YES as a CICS system initialization parameter (and SECPRFX=YES if you define profiles with the CICS region userid as a prefix).
4. Specify XJCT=YES for the default resource class names of JCICSJCT and KCICSJCT (or XJCT=class_name for user-defined resource class names).

### Access authorization levels

If you specify security with SEC=YES as a system initialization parameter, CICS requires a minimum authorization of UPDATE for journal access.

## Started and XPCT-checked transactions

A CICS transaction initiated by a terminal user can start other transactions by means of an EXEC CICS START command. Transactions started in this way are known as **started transactions**, and you can use CICS RACF security to control who can start other transactions using the START command.

Started transactions are defined in the ACICSPCT and BCICSPCT resource class profiles. These profiles also control access to transactions specified in certain other EXEC CICS commands, if the transaction issuing the command is defined with RESSEC(YES). The commands affected are:

- COLLECT STATISTICS TRANSACTION
- DISCARD TRANSACTION
- INQUIRE TRANSACTION
- SET TRANSACTION
- INQUIRE REQID
- CANCEL

When a transaction issues an EXEC CICS START TRANSID(*tranid*) command, CICS calls RACF to check that the user of the transaction issuing the command is authorized for the started transaction.

To implement security for started transactions and for transactions checked against the XPCT class:

1. Specify RESSEC(YES) in the CSD resource definition of the transactions that issue START commands.
2. Define profiles to RACF in the ACICSPCT or BCICSPCT resource classes (or their equivalent if you have user-defined resource class names) using the name of the started transaction to identify the profiles.

   For example, use the following commands to define a transaction in the ACICSPCT class, and to authorize one user only:

   ```
   RDEFINE  ACICSPCT (tran1, tran2, ..., trann) UACC(NONE)
                    NOTIFY(sys_admin_userid)
   PERMIT  tran1 CLASS(ACICSPCT)  ID(userid) ACCESS(READ)
   PERMIT  tran2 CLASS(ACICSPCT)  ID(userid) ACCESS(READ)
   ```

   To define started transactions as members of a profile in the started transaction resource group class, with an appropriate access list, use the following commands:

   ```
   RDEFINE  BCICSPCT  started_trans  UACC(NONE)
                      ADDMEM(trana, tranb, ..., tranx)
                      NOTIFY(sys_admin_userid)
   PERMIT started_trans  CLASS(BCICSPCT) ID(group_userid) ACCESS(READ)
   ```
3. Specify SEC=YES as a CICS system initialization parameter (and SECPRFX=YES if you define profiles with the CICS region userid as a prefix).
4. Specify XPCT=YES for the default resource class names of ACICSPCT and BCICSPCT (or XPCT=class_name for user-defined resource class names).

### Transactions started at terminals

The EXEC CICS START command enables a CICS application program to start another transaction associated with a terminal other than the one from which the start command is issued. For example, the following command issued in CICS transaction `tranid1`, invoked at `termid1`, starts another transaction called `tranid2` at `termid2`:

```
EXEC CICS START
     TRANSID(tranid2)
     AT HOURS('18') MINUTES('50')
     TERMID(termid2)
```

When a TERMID is specified for the started transaction, CICS performs a transaction-attach security check, using the classes TCICSTRN and GCICSTRN, on the userid associated with the terminal (`termid2` in this example). You must therefore ensure that the userid associated with the terminal (`termid2`) is authorized to invoke the transaction. This userid is that of the signed-on user, or the CICS default userid if no user is signed on. If `termid2` is **not** authorized, message DFHAC2033 is issued to the user of `termid2`. The user of the terminal that issued the START command gets a "normal" response. If the started transaction is defined with RESSEC(YES), also ensure that the userid associated with the terminal (`termid2` in this example) is suitably authorized to access protected resources.

**Starting tasks at terminals defined with preset security:** Typically, started transactions associated with a terminal are printing tasks, where the specified terminal is a printer. In this case, to associate a specific userid with the terminal, you define the terminal with preset security. See "Preset terminal security" on page 5 for more information.

## Transactions started without terminals

The EXEC CICS START command enables a CICS application program to start another transaction that is not associated with any terminal. When no TERMID is specified for the started transaction, the userid associated with the new transaction depends on whether you also specify the USERID option.

**Userid of a non-terminal started transaction::** The USERID option of the EXEC CICS START command (or the terminal user if no TERMID or USERID is included in the START command) determines the userid for a non-terminal started transaction. Without the USERID option, the non-terminal started transaction has the same userid as the transaction that executed the EXEC CICS START command. If the USERID option is specified on the EXEC CICS START command, the specified userid is used instead.

When an EXEC CICS START command is executed without the TERMID option, CICS performs a surrogate user check to ensure that the transaction is authorized for the userid to be used by the non-terminal started transaction. For information about the link authorization of surrogate users, see "Link security" on page 148. For information about EDF authorization of surrogate users, see "Conditional access processing" on page 24.

**Access to resources by a non-terminal started transaction:** If the USERID option is not specified on an EXEC CICS START command, the non-terminal started transaction does not always inherit all of the security of the transaction that executed the command. Also, it does not inherit resource access determined by link security, or resource access determined by a userid for EDF when used in dual-screen mode. This means:

- If a transaction-routed transaction executes an EXEC CICS START command, or if an EXEC CICS START command is function shipped, the non-terminal started transaction is not subject to link security.
- If EDF is used in dual-screen mode for a transaction that issues an EXEC CICS START command, the non-terminal started transaction is not subject to resource access determined by the userid of the EDF terminal.

If you want the started transaction to have exactly the same security capabilities as the starting transaction, omit the USERID option. Without the USERID option, resource access by the non-terminal started transaction is determined by the sign-on parameters of the terminal transaction. These include the RACF group and the port of entry at which the terminal user signed on; that is, the terminal or console used to sign on, as shown in the following example:

A terminal user signs on using the CESN transaction at a terminal with netname NETNAMEX. For RACF, therefore, the port of entry is NETNAMEX. At the CESN screen the terminal user enters userid USERID1, and groupid GROUPID2. The terminal user then runs a terminal transaction which executes an EXEC CICS START command without the TERMID option or the USERID option specified. The non-terminal started transaction has resource access determined by userid USERID1, groupid GROUPID2, and port of entry NETNAMEX.

If a non-terminal transaction is denied access to a resource by RACF, the error message produced can include the terminal sign-on parameters, userid, and groupid. It can also include a port of entry. The userid, groupid, and port of entry can be those inherited from the terminal transaction that started the non-terminal transaction.

If the USERID option is specified on an EXEC CICS START command, the non-terminal started transaction has access to resources determined by the userid specified on the USERID option.

We recommend that you do not specify the current userid of a terminal transaction on the USERID option. The non-terminal started transaction may not have the same resource access as the terminal transaction. The following examples show how the non-terminal started transaction can have different resource access:

*Example 1*:

RACF conditional access lists can be used by specifying WHEN(TERMINAL( ... )) or WHEN(CONSOLE( ... )) on the RACF PERMIT command to allow a terminal transaction access to certain resources because the specified port of entry is in use. See "Conditional access processing" on page 24.

If an EXEC CICS START TRANSID USERID command is executed by a terminal transaction specifying the same userid that the terminal user entered when signing on with CESN, the started transaction has access to resources determined by the specified userid, but not to the resources determined by the port of entry.

The started transaction is not subject to the conditional access list effective for the terminal transaction that executed the EXEC CICS START USERID command.

*Example 2*:

Using RACF you can grant (or deny) group access to a RACF protected resource.

A terminal user can enter a groupid and a userid when signing on with CESN. When the terminal user runs a terminal transaction, the groupid can determine resource access.

If an EXEC CICS START TRANSID USERID command is executed by a terminal transaction specifying the same userid as that entered by the terminal user when signing on with CESN, the started transaction has access to resources determined by the specified userid. Resource access is not determined by the groupid that the

terminal user entered when signing on with CESN. Resource access for the non-terminal started transaction can be determined by the default groupid for the specified userid.

The started non-terminal transaction is not subject to the group access effective for the terminal transaction that executed the EXEC CICS START USERID command.

### Access authorization levels

CICS requires a minimum authorization of READ for started transactions.

## Application programs

You control access to the initial program specified in the transaction resource definition by authorizing the user to initiate the transaction (transaction-attach security). However, CICS application programs can invoke other programs by means of the LINK, LOAD, and XCTL commands. Also, the load status of programs can be altered by the CICS RELEASE, ENABLE, and DISABLE commands. Note, however, that there is no separate security check on the RELEASE of programs loaded for task lifetime. This is done on the corresponding LOAD.

You control access to programs invoked using these commands by defining profiles in the CICS application program classes, and which you define to CICS on the XPPT system initialization parameter.

To control which users can invoke or change the load status of other programs:

1. Specify RESSEC(YES) in the CSD resource definition of the transactions that use the above commands.
2. Define profiles to RACF in the MCICSPPT or NCICSPPT resource classes (or their equivalent if you have user-defined resource class names) using the name of the program invoked on the LINK, LOAD, or XCTL command to identify the profiles.

   For example, use the following commands to define a program in the MCICSPPT class, and to authorize one user only:

   ```
   RDEFINE  MCICSPPT  (prog1, prog2, ..., progn)  UACC(NONE)
                    NOTIFY(sys_admin_userid)
   PERMIT  prog1 CLASS(MCICSPPT)  ID(userid) ACCESS(READ)
   PERMIT  prog2 CLASS(MCICSPPT)  ID(userid) ACCESS(READ)
   ```

   To define programs as members of a profile in the application program resource group class, with an appropriate access list, use the following commands:

   ```
   RDEFINE  NCICSPPT  cics_programs  UACC(NONE)
                      ADDMEM(proga, progb, ..., progx)
                      NOTIFY(sys_admin_userid)
   PERMIT cics_programs CLASS(NCICSPPT) ID(group_userid) ACCESS(READ)
   ```
3. Specify SEC=YES as a CICS system initialization parameter (and SECPRFX=YES if you define profiles with the CICS region userid as a prefix).
4. Specify XPPT=YES as a CICS system initialization parameter for the default resource class names of MCICSPPT and NCICSPPT (or XPPT=class_name for user-defined resource class names).

**Exception for distributed program link (DPL) commands**

If CICS finds that a program referenced on an EXEC CICS LINK command is a remote program, it does not perform the security check in the region in which the link command is issued. The security check is performed only in the CICS region in which the linked-to program finally executes.

For example, if CICSA function ships a DPL command to CICSB, where the program then executes, CICSB issues the security check. If the DPL request is function shipped again to CICSC for execution, it is CICSC that issues the security check.

### Access authorization levels

CICS requires a minimum authorization of READ for programs.

## Temporary storage

Unlike the other resources for which you specify RESSEC(YES), temporary storage queues, for which you require RACF protection, also require the security attribute in a suitable TSMODEL resource definition. You specify TSMODEL definitions in the CSD. See the *CICS Resource Definition Guide* for information about TSMODEL resource definitions.

### Implementing security for temporary storage queues

To implement security for temporary storage queues:

1. Specify RESSEC(YES) in the CSD resource definition of the appropriate transactions.
2. Specify the security attribute on suitable TSMODEL resource definitions in the CSD. CICS does not perform any security checks on temporary storage queues that specify SECURITY=NO on the matching TSMODEL definition.
3. Define profiles to RACF in the SCICSTST or UCICSTST resource classes (or their equivalent if you have user-defined resource class names), with access lists as appropriate. For example, use the following commands to define queues in the SCICSTST class, and to authorize users to both read from and write to these queues:

   ```
   RDEFINE  SCICSTST  (tsqueue1, tsqueue2, ..., tsqueuen) UACC(NONE)
                      NOTIFY(sys_admin_userid)
   PERMIT tsqueue1 CLASS(SCICSTST) ID(group1, group2) ACCESS(UPDATE)
   PERMIT tsqueue2 CLASS(SCICSTST) ID(group1, group2) ACCESS(UPDATE)
   ```

   To define temporary storage queues as members of a profile in the CICS temporary storage resource group class, with an appropriate access list, use the following commands:

   ```
   RDEFINE  UCICSTST  tsqueue_group  UACC(NONE)
                      ADDMEM(tsqueuea, tsqueueb, ..., tsqueuex)
                      NOTIFY(sys_admin_userid)
   PERMIT tsqueue_group CLASS(UCICSTST) ID(group_userid) ACCESS(UPDATE)
   ```

   For more information about defining temporary storage profiles, see "Other temporary storage security considerations" on page 98.
4. Specify SEC=YES as a CICS system initialization parameter (and SECPRFX=YES if you define profiles with the CICS region userid as a prefix).

5. Specify XTST=YES as a CICS system initialization parameter for the default resource class names of SCICSTST and UCICSTST (or XTST=class_name for user-defined resource class names).

**Note:** CICS continues to support the DFHTST TYPE=SECURITY macro for defining temporary storage security. However, you are recommended to migrate your temporary storage tables (TSTs) to the CSD as TSMODEL definitions.

### Long temporary storage queue names

On OS/390 Release 6 the SCICSTST and UCICSTST general resource classes support profile names that can be up to 25 characters. This allows you to use 16 character queue names for your temporary storage queue names in combination with a security prefix of up to 8 characters and a separator. For earlier releases of OS/390 the SCICSTST and UCICSTST general resource classes support profiles of up to 17 characters. If you intend to use long temporary storage queue names with security prefixing on these releases, you must use an installation-defined resource class with a MAXLENGTH sufficiently increased to support the length of the security prefix, queue name and separator (up to a maximim of 25 characters).

### Other temporary storage security considerations

You can define the queue names on the PREFIX attribute of the TSMODEL resource definition as follows:

- By specifying a fully identified name that exactly matches the queue name specified on a READQ TS or WRITEQ TS command. This can be from 1 to 16 alphanumeric characters.
- By specifying a generic name, or prefix, that corresponds to the leading alphanumeric characters of a set of queue names.

  It follows that a prefix can only be from 1 to 15 characters, because if you specify the full 16 characters for a queue name, it must be the name of a specific temporary storage queue.

When a CICS application issues a temporary storage command (for example, DELETEQ TS, READQ TS, or WRITEQ TS) and temporary storage security is in effect, CICS searches the TST for a DATAID that corresponds to the leading characters of the queue name.

Note that if you include a temporary storage queue with hexadecimal characters in a temporary storage queue name, unpredictable results may occur. Also, if a temporary storage queue name contains an imbedded blank, RACF truncates the resource name to that blank.

### Access authorization levels

If you specify security with SEC=YES as a system initialization parameter, CICS requires a level of authorization appropriate to the temporary storage queue access intended, for example, a minimum of READ for READQ TS, and a minimum of UPDATE for DELETEQ TS and WRITEQ TS.

## Program specification blocks

DL/I program specification blocks (PSBs) are IMS control blocks that describe databases and logical message destinations used by an application program. PSBs consist of one or more program communication blocks (PCBs), which describe an application program's interface to an IMS database.

To implement security for PSBs scheduled in CICS applications:

1. Define profiles to RACF in the PCICSPSB or QCICSPSB resource classes (or their equivalent if you have user-defined resource class names), with access lists as appropriate. The resource profile names you define to RACF must correspond to the names of PSBs specified in CICS PSB schedule commands. For example, use the following commands to define PSBs in the PCICSPSB class, and to authorize users to access these queues:

   ```
   RDEFINE  PCICSPSB  (psbname1, psbname2, ..., psbnamen) UACC(NONE)
                      NOTIFY(sys_admin_userid)
   PERMIT psbname1 CLASS(PCICSPSB) ID(group1, group2) ACCESS(READ)
   PERMIT psbname2 CLASS(PCICSPSB) ID(group1, group2) ACCESS(READ)
   ```

   To define PSBs as members of a profile in the CICS PSB resource group class, with an appropriate access list, use the following commands:

   ```
   RDEFINE  QCICSPSB  psbname_group UACC(NONE)
                      ADDMEM(psbnamea, psbnameb, ..., psbnamex)
                        NOTIFY(sys_admin_userid)
   PERMIT psbname_group CLASS(QCICSPSB) ID(group_userid) ACCESS(UPDATE)
   ```

2. Specify SEC=YES as a CICS system initialization parameter (and SECPRFX=YES if you define profiles with the CICS region userid as a prefix).
3. Specify XPSB=YES as a CICS system initialization parameter for the default resource class names of PCICSPSB and QCICSPSB (or XPSB=class_name for user-defined resource class names).
4. Specify PSBCHK=YES if you want full security for PSBs that are accessed in transaction-routed transactions. This applies to both types of DL/I interface (remote and DBCTL). If you specify PSBCHK=NO, the authority of the remote user is **not used** in transaction-routed transactions.

**Note:** CICS requires a minimum authorization of READ for PSBs.

If you are using DBCTL, read the chapter on security in the *CICS IMS Database Control Guide* for information on defining security in a CICS-DBCTL environment.

## Security checking of transactions running under CEDF

When a transaction is run under the CEDF transaction, CICS determines the security processing for the target transaction from the logical OR of RESSEC in the resource definitions for the target transaction and the CEDF transaction.

Table 10 shows the security checking performed for the transaction XSUB for different settings of RESSEC.

*Table 10. Security checking of transactions running under CEDF*

| CEDF | XSUB | Security checking |
|---|---|---|
| RESSEC(YES) | RESSEC(YES) | Any access to CICS resources causes a security check. |
| RESSEC(YES) | RESSEC(NO) | Any access to CICS resources causes a security check. (Logical OR results in RESSEC on.) |
| RESSEC(NO) | RESSEC(YES) | Any access to CICS resources causes a security check. (Logical OR results in RESSEC on.) |
| RESSEC(NO) | RESSEC(NO) | Access to CICS resources does not cause a security check. (Logical OR results in RESSEC off.) |

To achieve the expected security processing for a transaction when it runs under CEDF, ensure that RESSEC for the CEDF transaction definition is set to NO. The IBM-supplied definition of CEDF in the DFHEDF group specifies RESSEC(YES). Definitions in the IBM-supplied groups cannot be modified, so to change the definition, copy it to another group.

When the CEBR and CECI are invoked from within EDF they are transaction-attach checked. The CMDSEC and RESSEC definitions are forced when CEBR or CECI are invoked in this environment, regardless of what is coded in their transaction definitions

When CEDF is used in **two-terminal mode**, it is entered at a different terminal from the transaction being tested. The authorities of the user executing the CEDF transaction are taken into account, as well as those of the user executing the transaction being tested. For each resource accessed by the tested transaction, both users must have access authority, otherwise a NOTAUTH condition is raised. This applies to all resource checks:

- Transaction attach
- CICS resource
- CICS command
- Non-CICS resources accessed through the QUERY SECURITY command
- Surrogate user

## Defining generic profiles for resources

If you control access to CICS transactions by means of transaction-attach security, there is probably only a very small subset of other resource types for which you need a further level of RACF protection. For example, there may be just a few programs in the CICS application program resource class that are particularly sensitive, and a much larger number that constitute no significant risk. In this case, you could protect the few by defining specific RACF profiles for only those programs that are sensitive. You ensure that everyone can access the remaining, nonsensitive, programs by defining a completely generic resource profile, as follows:

```
RDEFINE  MCICSPPT  *  UACC(READ) ...
```

This profile applies to any authorization request for programs not covered by one of the specific profiles. RACF processing logic is such that the most specific profile for any given resource name is always used.

Note that to determine whether a profile is generic, you need only check if 'G' appears after the name of the profile when it is listed with RLIST or SEARCH. For example:

SEARCH CLASS(TCICSTRN)

may give the following output:

```
C*
CED% (G)
** (G)
```

The above output shows that both CED% and ** are generic profiles. The C* profile is not generic because it is not followed by (G). This could have occurred if the C* profile was created before generic profiles had been enabled with a SETROPTS command. The C* profile can be deleted and redefined as a proper generic profile as follows:

```
SETROPTS NOGENERIC(TCICSTRN)
SETROPTS NOGENCMD(TCICSTRN)
RDEL TCICSTRN C*
SETROPTS GENERIC(TCICSTRN)
RDEFINE TCICSTRN C* UACC(NONE)
```

## Access to all or access to none?

If RACF can find neither a specific nor generic profile, it returns a "no profile found" condition. CICS treats this return code exactly the same as the "user not authorized" return code, and returns the NOTAUTH condition to the CICS application program. If RACF cannot find the APPL class, it returns a "READ access intent" condition.

You can either use the completely generic profile to permit access to any resources not otherwise covered by more specific profiles, or, to prevent any access, use the UACC(READ|UPDATE) or UACC(NONE) options. For example,

```
RDEFINE  DCICSDCT  *  UACC(NONE)
```

prevents access to any transient data queue not covered by any of the other profiles defined to RACF, and results in RACF writing an SMF record.

On the other hand, you can define files as "public" by the following command:

```
RDEFINE  FCICSFCT  *  UACC(READ)
```

If you are using generic profiles, ensure that generic profile checking has been activated for the CICS RACF resource classes (both the IBM-supplied classes and any installation-defined classes added to the RACF class descriptor table) by issuing a SETROPTS GENERIC(*classname*) command for any one of the CICS classes having the same POSIT value. This ensures generic checking for all other CICS classes with the same POSIT value. If you change a generic profile, you must issue a SETROPTS GENERIC(*classname*) REFRESH command. For more information about POSIT values and defining generic classes, see the *OS/390 Security Server (RACF) System Programmer's Guide*.

# Chapter 7. Surrogate user security

This chapter is in two main sections:

- "Where surrogate user checking applies"
- "RACF definitions for surrogate user checking" on page 107

## Where surrogate user checking applies

CICS performs surrogate user security checking in a number of situations, using the surrogate user facility of an external security manager (ESM) such as RACF. A surrogate user is one who has the authority to start work on behalf of another user. A surrogate user is authorized to act for that user without knowing that other user's password. To enable surrogate user checking, XUSER=YES must be specified as a system initialization parameter.

If surrogate user checking is in force, it applies to:

- The CICS default user
- PLT post-initialization processing
- Preset terminal security
- Started transactions
- The userid associated with a CICS business transaction services (BTS) process or activity that is started by a RUN command
- The userid associated with a transient data destination
- The userid supplied as a parameter on an EXCI call
- The userid supplied on the AUTHID and COMAUTHID parameters of DB2CONN and DB2ENTRY resource definitions.

### CICS default user

CICS performs a surrogate user security check against its own userid (the CICS region userid) to ensure that it is properly authorized as a surrogate of the default userid specified on the DFLTUSER system initialization parameter.

### Post-initialization processing

If you specify a program list table on a PLTPI system initialization parameter, CICS checks that the region userid is authorized as a surrogate user of the userid specified in the PLTPIUSR system initialization parameter.

The PLTPIUSR system initialization parameter specifies the userid that CICS is to use for PLT programs that run during CICS initialization. All PLT programs run under the authority of the specified userid, which must be authorized to all the resources referenced by the programs.

The scope of PLT security checking is defined by the PLTPISEC parameter. This specifies whether command security checks and resource security checks are to apply to PLTPI programs.

If you do not specify the PLTPIUSR parameter, CICS runs PLTPI programs under the authority of the CICS region userid, in which case CICS does not perform a surrogate user check. However, the CICS region userid must then be authorized to all the resources referenced by the PLT programs. Furthermore, the CICS region

userid is associated with any transactions started by PLT programs, and therefore must be authorized to run such transactions.

## Preset terminal security

When you install a terminal that is defined with a preset security userid, CICS checks that the userid performing the install is authorized as a surrogate user of the preset userid. This is discussed in "Controlling the use of preset-security" on page 70.

## Started transactions

CICS performs surrogate user checks when you use the EXEC CICS START command to start a transaction that is not associated with a terminal.

In the following, the userid under which the transaction issuing the START command runs is called the *starting-userid*, and the userid under which the started transaction runs is called the *started-userid*:

- If the TERMID option is specified on the START command, surrogate user checking does not apply. The *started-userid* is inherited from the terminal at which the transaction runs.
- If the USERID option is specified on the START command, the *started-userid* is set to that specified userid.
- If neither TERMID nor USERID is specified on the START command, the *started-userid* is set to be the same as the *starting-userid*.

CICS requires that all the userids associated with the transaction issuing the START are surrogates of the *started-userid*. CICS also assumes that any userid is always a surrogate of itself. So userids that are the same as *started-userid* are regarded as surrogates already, and the external security manager is not called for them.

A transaction can be associated with userids that are different from *starting-userid* when it is using CICS intercommunication, and when it is using EDF in the two-terminal mode.

### Intercommunication and started transactions

If an EXEC CICS START command (without TERMID) is function shipped or is executed from a transaction-routed transaction, the command can be subject to link security. If link security is in effect, CICS also performs a surrogate user check to verify that the userid for link security is authorized as a surrogate user to the userid for the started transaction. The surrogate check is done at this stage even if the USERID is omitted (if the *started-userid* is different from the link userid). For more information see "Link security" on page 148.

### EDF in dual-screen mode and started transactions

If an EXEC CICS START command (without TERMID) is executed under control of EDF in dual-screen mode, CICS also performs a surrogate user check, to verify that the userid for the EDF terminal is authorized as a surrogate user of the userid for the started transaction. This check is done even if USERID is omitted, if the *started-userid* is different from the EDF userid.

Surrogate user checking can be subject to link security. If EDF is in use in dual-screen mode, the security of the user executing EDF is also checked. If a NOTAUTH condition occurs with an EXEC CICS START command, this can be because of link security or because of EDF user security.

## BTS processes and activities

When a CICS business transaction services (BTS) process or activity is activated by an EXEC CICS RUN command, it may run under a different userid from that of the transaction that issues the RUN. (BTS is described in the *CICS Business Transaction Services*.)

The application programmer can specify under whose authority a process or activity is to run, when it is activated by a RUN command, by coding the USERID option of the DEFINE PROCESS or DEFINE ACTIVITY command. If the USERID option is omitted, the value defaults to the userid of the transaction that issues the DEFINE command.

If the USERID option is specified, CICS performs (at define time) a surrogate security check to verify that the userid of the transaction that issued the DEFINE command is authorized to use the userid specified by USERID.

## Transient data trigger-level transactions

When a transient data queue is defined by a DFHDCT macro with a non-terminal trigger-level transaction and a USERID parameter, CICS checks that its own userid (the CICS region userid) is authorized as a surrogate user of the userid specified on the trigger-level transaction, during the installation of the transient data resource definition. When such a transient data queue is defined by RDO, the user installing the definition is checked. Likewise, when such a transient data queue is created with the EXEC CICS CREATE command, the user executing the command is checked.

The userid for a transient data trigger-level transaction that is not associated with a terminal can be specified on the transient data definition or on the EXEC CICS SET TDQUEUE system programming command.

### Intrapartition transient data resources.

CICS uses the userid specified on transient data queue definition for security checking in any trigger-level transactions that are not associated with a terminal. Code the USERID operand with the userid that you want CICS to use for security checking for the trigger-level transaction specified on the TRANSID operand. USERID is valid only when the destination facility is a file.

The trigger-level transaction runs under the authority of the specified userid, which must be authorized to all the resources used by the transaction.

If you omit the userid from a qualifying trigger-level entry, CICS uses the default userid specified on the DFLTUSER system initialization parameter. Ensure that the userid of any CICS region in which the transient data queue definition is installed is defined as a surrogate of all the userids specified in the DCT. This is because, during a cold start, CICS performs a surrogate user security check for the CICS region userid against all the userids specified in transient data queue definitions that are being installed. If the surrogate security check fails, the transient data queue definition is not installed.

### EXEC CICS SET TDQUEUE ATIUSERID

The system programming command, EXEC CICS SET TDQUEUE ATIUSERID, specifies the userid for a transient data trigger-level transaction that is not associated with a terminal. The destination facility must be a file.

CICS performs a surrogate user security check against the userid of the transaction that issues the EXEC CICS SET TDQUEUE command, to verify that the transaction userid is authorized as a surrogate user of the userid specified on the ATIUSERID parameter.

## Userid passed as parameter on EXCI calls

A surrogate user check is performed to verify that the batch region's userid is authorized to issue DPL calls for another user (that is, is authorized as a surrogate of the userid specified on the DPL_request call).

If you want your external CICS interface (EXCI) client jobs to be subject to surrogate user checking, specify SURROGCHK=YES in the EXCI options table, DFHXCOPT. If you specify SURROGCHK=YES, authorize the batch region's userid as a surrogate of the userid specified on all DPL_request calls. This means the batch region's userid must have READ access to a profile named "userid.DFHEXCI" in the SURROGAT general resource class (where "userid" is the userid specified on the DPL call). For example, the following commands define a surrogate profile for a DPL userid, and grant READ access to the EXCI batch region:

```
RDEFINE  SURROGAT dpl_userid.DFHEXCI UACC(NONE) OWNER(DPL_userid)
PERMIT   userid.DFHEXCI CLASS(SURROGAT) ID(batch_region_userid)
                   ACCESS(READ)
```

If surrogate user checking is enabled (SURROGCHK=YES), but no userid is specified on the DPL call, no surrogate user check is performed, because the userid on the DPL call defaults to the batch region's userid.

If you do not want surrogate user security checking, specify SURROGCHK=NO in the DFHXCOPT options table.

Surrogate user checking is useful when the batch region's userid is the same as the CICS server region userid, in which case the link security check is bypassed. In this case, a surrogate user check is recommended, because the USERID specified on the DPL call is not an authenticated userid (no password is passed).

If the batch region's userid and the CICS region userid are different, link security checking is enforced. With link security, an unauthenticated userid passed on a DPL call cannot acquire more authority than that allowed by the link security check. It can acquire only the same, or less, authority than allowed by the link security check.

## The userid on DB2 AUTHID and COMAUTHID parameters

When you install a DB2 resource definition that specifies an AUTHID, SIGNID or COMAUTHID, or try to modify one of these parameters, CICS checks that the userid performing the operation is authorized as a surrogate user of AUTHID, COMAUTHID or SIGNID. This also applies to the CICS region userid during group list install on a CICS cold or initial start.

For more information about these parameters, see the *CICS Resource Definition Guide*.

**Note:** The XUSER system initialization parameter is also used to control access to the AUTHTYPE and COMAUTHTYPE paremeters, but the security control

for these parameters is managed through the facility general resource class. See the *CICS DB2 Guide* for more information.

## RACF definitions for surrogate user checking

To enable CICS surrogate user checking:

- Define the appropriate SURROGAT class profiles for CICS in the RACF database.
- Authorize CICS surrogate users to the appropriate SURROGAT profiles.

There are two forms of surrogate class profile names that you can define for CICS surrogate user checking. The names of these SURROGAT class profiles must conform to the following naming conventions:

*userid*.**DFHSTART**
: *userid* represents one of the following:
  - The userid under which a started transaction is to run
  - The userid associated with a CICS business transaction services (BTS) process or activity that is started by a RUN command

*userid*.**DFHINSTL**
: *userid* represents one of the following:
  - The PLT userid specified on the PLTPIUSR system initialization parameter
  - The userid associated with a trigger-level transaction
  - The CICS default userid specified on the DFLTUSER system initialization parameter
  - The userid specified for preset terminal security
  - The userid specified on the AUTHID or COMAUTHID parameter of a DB2 resource definition.

There is also a form of surrogate class profile that you can define for external CICS interface (EXCI) security checking:

*userid*.**DFHEXCI**
: *userid* represents the user specified on the DPL call in the client batch region.

  To authorize a surrogate to this EXCI profile, grant the EXCI batch region's userid READ access.

  Note that surrogate security checks in an EXCI batch region are independent of security definitions in the target CICS region. If SURROGCHK is specified in the EXCI options table (DFHXCOPT), surrogate security checks are performed in the EXCI client program's address space regardless of the CICS security settings.

To authorize a surrogate user to one of these profiles, you must grant READ access.

You do not need to define a user as that user's own surrogate. CICS bypasses the surrogate check in this case.

The *OS/390 Security Server (RACF) Security Administrator's Guide* gives more information about defining surrogate resource classes. Refer to it if you need to use RACF facilities such as generic resource classes or RACFVARS profiles to help with making many RACF definitions.

## Examples of RACF definitions for surrogate user checking

You define surrogate users to RACF by:

- Defining a *userid.resource_name* profile in the SURROGAT general resource class for each user requiring a surrogate user to act on their behalf. For this purpose you use the RACF RDEFINE SURROGAT command.
- Authorizing each userid that is to act as a surrogate for a user defined in a SURROGAT class profile. For this purpose you use the RACF PERMIT command.

### PLT security

For PLT security checking, the CICS region userid must be authorized as a surrogate of the PLT userid defined on the PLTPIUSR system initialization parameter. This means granting the CICS region userid access to a SURROGAT resource class profile owned by the PLT userid, as shown in the following example, where the CICS region userid is CICSHT01, and the PLT security userid is PLTUSER:

```
RDEFINE  SURROGAT  PLTUSER.DFHINSTL  UACC(NONE) OWNER(PLTUSER)
PERMIT   PLTUSER.DFHINSTL  CLASS(SURROGAT) ID(CICSHT01)  ACCESS(READ)
```

In addition to enabling PLT security by defining SURROGAT profiles, ensure that when PLT security is active (through the use of the PLTPISEC system initialization parameter) you also add the PLT userid to the access lists of all the resources accessed by PLT programs. For example, if you specify PLTPISEC=RESSEC, ensure that the PLT userid is authorized to all the CICS resources for which security is active.

**Started transactions:** For started transactions, CICS can require as many as three levels of surrogate user. (See "Started transactions" on page 104 for details of the different surrogate users that can be required for a START command.)

For started transaction security at the first level, the userid of the transaction that issues the START command must be authorized as a surrogate for the userid specified on the START command.

For example, a transaction running under USERID2 issues:

```
EXEC CICS START TRANSID('TBAK') USERID('USERID1').
```

USERID2 must be defined to RACF as a surrogate of USERID1 (with READ authority). This is illustrated in the following RACF commands:

```
RDEFINE  SURROGAT  USERID1.DFHSTART  UACC(NONE)  OWNER(USERID1)
PERMIT   USERID1.DFHSTART  CLASS(SURROGAT) ID(USERID2)  ACCESS(READ)
```

For more information about surrogate security, see "Querying a user's surrogate authority" on page 122.

# Chapter 8. CICS command security

CICS command security applies to System Programming (SP)-type commands; that is, commands that require the special CICS translator option, SP. Security checking is performed for these commands when they are issued from a CICS application program, and for the equivalent commands that you can issue with the CEMT master terminal transaction. Table 11 shows the commands that are subject to command security checking.

This chapter discusses security for these commands as follows:



CICS/ESA Front End Programming Interface security uses the same mechanism for authorization as the SP-type commands, using the FEPIRESOURCE resource name. Front End Programming Interface security is not discussed in this book. See the *CICS Front End Programming Interface User's Guide* for details.

*Table 11. Access required for system programming commands*

| Command name | Access required |
|---|---|
| COLLECT INQUIRE | READ |
| DISABLE ENABLE EXTRACT PERFORM RESYNC SET | UPDATE |
| CREATE DISCARD | ALTER |

**Note:** To determine who is allowed to use the (SP) option on the CICS translator, you can use RACF to control who is allowed to load the DFHEITBS table at translation time. For a description of RACF program control, see the *OS/390 Security Server (RACF) Security Administrator's Guide*. DFHEITBS is the language definition table that defines the SP-type commands, and is loaded only on demand.

You can issue an EXEC CICS link to DFHEDAP to install resources. CREATE implies a CEDA INSTALL for which you must have ALTER access to the required resource.

## CICS resources subject to command security checking

For transaction and resource security checking, you identify the resources to RACF using the identifiers you have assigned to them, such as file names, queue names, transaction names, and so on. However, in the case of command security, the resource identifiers are all predefined by CICS, and you use these predefined names when defining resource profiles to RACF. The full list of resource identifiers that are subject to command security checking, together with the associated commands, is shown in Table 12 on page 110. Note that most of these commands are common to both the CEMT and EXEC CICS interfaces; where they are unique to one or the other they are prefaced with **CEMT**, or **EXEC CICS**, as appropriate.

*Table 12. CICS resources subject to command security checking*

| Resource name (see note 1) | Related CICS command(s) |
|---|---|
| AUTINSTMODEL | INQUIRE \| DISCARD AUTINSTMODEL |
| AUTOINSTALL | INQUIRE \| SET AUTOINSTALL |
| CFDTPOOL | INQUIRE CFDTPOOL |
| CONNECTION | INQUIRE \| SET \| CREATE \| DISCARD CONNECTION |
| DB2CONN | INQUIRE \| SET \| CREATE \| DISCARD DB2CONN |
| DB2ENTRY | INQUIRE \| SET \| CREATE \| DISCARD DB2ENTRY |
| DB2TRAN | INQUIRE \| SET \| CREATE \| DISCARD DB2TRAN |
| DELETSHIPPED | INQUIRE \| SET \| PERFORM DELETSHIPPED |
| DOCTEMPLATE | INQUIRE \| SET |
| DSNAME | INQUIRE \| SET DSNAME |
| DUMP | PERFORM DUMP<br>**CEMT** PERFORM SNAP |
| DUMPDS | INQUIRE \| SET DUMPDS |
| ENQMODEL | INQUIRE \| CREATE \| SET |
| EXITPROGRAM | **EXEC CICS** ENABLE PROGRAM<br>**EXEC CICS** DISABLE PROGRAM<br>**EXEC CICS** EXTRACT EXIT<br>**EXEC CICS** RESYNC ENTRYNAME |
| FEPIRESOURCE | Certain EXEC CICS FEPI commands (see note 3) |
| FILE | INQUIRE \| SET \| CREATE \| DISCARD FILE |
| IRBATCH | **CEMT** INQUIRE IRBATCH |
| IRC | INQUIRE \| SET IRC |
| JOURNALMODEL | **EXEC CICS** INQUIRE \| SET \| CREATE \| DISCARD JOURNALMODEL<br>**CEMT** INQUIRE \| SET JMODEL |
| JOURNALNAME | INQUIRE \| SET JOURNALNAME |
| LINE | **CEMT** INQUIRE \| SET LINE |
| LSRPOOL | CREATE LSRPOOL |
| MAPSET | CREATE \| DISCARD MAPSET |
| MODENAME | INQUIRE \| SET MODENAME |
| MONITOR | INQUIRE \| SET MONITOR |
| PARTITIONSET | CREATE \| DISCARD PARTITIONSET |
| PARTNER | INQUIRE \| CREATE \| DISCARD PARTNER |
| PROCESSTYPE | **CEMT** DEFINE PROCESSTYPE<br>**EXEC CICS** CREATE PROCESSTYPE<br>**EXEC CICS** DISCARD PROCESSTYPE<br>**CEMT** INQUIRE PROCESSTYPE<br>**CEMT** SET PROCESSTYPE |
| PROFILE | INQUIRE \| CREATE \| DISCARD PROFILE |
| PROGRAM | INQUIRE \| SET \| CREATE \| DISCARD PROGRAM |
| RECONNECT | **CEMT** PERFORM RECONNECT |
| REQID | **EXEC CICS** INQUIRE \| SET REQID |

*Table 12. CICS resources subject to command security checking (continued)*

| Resource name (see note 1) | Related CICS command(s) |
|---|---|
| RESETTIME | PERFORM RESETTIME (see note 4) |
| REQUESTMODEL | INQUIRE \| SET |
| RRMS | INQUIRE RRMS |
| SECURITY | PERFORM SECURITY REBUILD |
| SESSIONS | CREATE \| DISCARD SESSIONS |
| SHUTDOWN | PERFORM SHUTDOWN (see note 2) |
| STATISTICS | INQUIRE \| SET STATISTICS<br>**EXEC CICS** COLLECT STATISTICS, and<br>PERFORM STATISTICS RECORD |
| STORAGE | INQUIRE STORAGE |
| STREAMNAME | INQUIRE \| SET STREAMNAME |
| SYSDUMPCODE | INQUIRE \| SET SYSDUMPCODE (see note 4) |
| SYSTEM | INQUIRE \| SET SYSTEM |
| TASK | INQUIRE \| SET TASK and TASK LIST |
| TCLASS | INQUIRE \| SET \| DISCARD TCLASS and<br>INQUIRE \| SET \| CREATE \| DISCARD TRANCLASS |
| TCPIP | INQUIRE \| SET |
| TCPIPSERVICE | INQUIRE \| SET \| CREATE \| DISCARD |
| TDQUEUE | INQUIRE \| SET \| CREATE \| DISCARD TDQUEUE |
| TERMINAL | INQUIRE \| SET \| CREATE \| DISCARD TERMINAL and<br>INQUIRE \| SET NETNAME |
| TRACEDEST | **EXEC CICS** INQUIRE \| SET TRACEDEST |
| TRACEFLAG | **EXEC CICS** INQUIRE \| SET TRACEFLAG |
| TRACETYPE | **EXEC CICS** INQUIRE \| SET TRACETYPE |
| TRANDUMPCODE | INQUIRE \| SET TRANDUMPCODE (see note 4) |
| TRANSACTION | INQUIRE \| SET \| DISCARD \| CREATE TRANSACTION |
| TSMODEL | CREATE \| INQUIRE \| SET \| DISCARD |
| TSPOOL | INQUIRE |
| TSQUEUE | **EXEC CICS** INQUIRE TSQUEUE |
| TSQNAME | INQUIRE \| SET |
| TYPETERM | CREATE \| DISCARD TYPETERM |
| UOW | INQUIRE \| SET UOW |
| UOWDSNFAIL | INQUIRE UOWDSNFAIL |
| UOWENQ | INQUIRE UOWENQ |
| UOWLINK | INQUIRE UOWLINK<br>**EXEC CICS** SET UOWLINK |
| VTAM | INQUIRE \| SET VTAM |
| WEB | INQUIRE \| SET |

**Notes:**

1. If you are using prefixing, the CICS region userid must be prefixed to the command resource name.
2. Be particularly cautious when authorizing access to these and any other CICS commands that include a SHUTDOWN option.
3. For more information about FEPI security, see the *CICS Front End Programming Interface User's Guide*.
4. See "Resource names for CEMT" on page 115.

If you are running CICS with command security, define resource profiles to RACF, with access lists as appropriate, using the resource names in Table 12 on page 110 as the profile names. Alternatively, you can create resource group profiles in the VCICSCMD class.

In the following example, the RDEFINE command defines a profile named CMDSAMP. The commands protected by this profile are specified on the ADDMEM operand. The PERMIT command allows a group of users to issue the commands for INQUIRE:

```
RDEFINE  VCICSCMD CMDSAMP UACC(NONE)
                  NOTIFY(sys_admin_userid)
                  ADDMEM(AUTINSTMODEL, AUTOINSTALL, CONNECTION,
                         DSNAME, TRANSACTION, TRANDUMPCODE, VTAM)
PERMIT CMDSAMP CLASS(VCICSCMD) ID(operator_group) ACCESS(READ)
```

The second example defines a profile called CMDSAMP1 with the same commands in the ADDMEM operand, as in the previous example. The PERMIT command allows a group of users to issue PERFORM, SET, and DISCARD against these commands:

```
RDEFINE  VCICSCMD CMDSAMP1 UACC(NONE)
                  NOTIFY(sys_admin_userid)
                  ADDMEM(AUTINSTMODEL, AUTOINSTALL, CONNECTION,
                         DSNAME, TRANSACTION, TRANDUMPCODE, VTAM)
PERMIT CMDSAMP1 CLASS(VCICSCMD) ID(op_group_2) ACCESS(UPDATE)
```

If you are running CICS with SEC=YES, users require the access levels shown in Table 12 on page 110.

## Parameters for specifying command security

In addition to the SEC and SECPRFX system initialization parameters, which are described in "SEC" on page 56, and "SECPRFX" on page 56, CICS provides the XCMD system initialization parameter and the CMDSEC resource definition option to enable you to specify that you want command security.

### XCMD system initialization parameter

The XCMD security parameter is a CICS system initialization parameter. You can specify whether you want command security active in the CICS region, and optionally specify the RACF resource class name in which you have defined the command security profiles.

If you are using the IBM-supplied RACF resource class names for CICS command profiles (CCICSCMD and VCICSCMD), specify XCMD=YES. CICS then requests RACF to build the in-storage profiles from these default resource classes.

If you are using installation-defined resource class names for CICS command profiles, specify XCMD=*user_class*, and CICS requests RACF to build the in-storage profiles from your own installation-defined resource classes.

If you do not want command security in a CICS region, specify XCMD=NO.

## The CMDSEC system initialization parameter

You can force the effect of CMDSEC=YES for all CICS transactions by specifying the CMDSEC=ALWAYS system initialization parameter. In general, this is not recommended, for the following reasons:

- For most simple transactions, just controlling access to the transaction is enough to control everything that the transaction can do.
- Invoking a command check for every CICS command consumes extra overhead that reduces the performance of all your transactions.

The CMDSEC option is recommended for installations that need total control of the SP-type commands.

## The CMDSEC transaction definition parameter

As described earlier in this section, the XCMD parameter enables command security to be active. You specify which transactions you want command security to apply to by using the CMDSEC option on the transaction resource definition, as follows:

**CMDSEC(NO)**
You do not want command security checking the transaction.

**CMDSEC(YES)**
You want command security checking on the SP™ commands in Table 11 on page 109.

For each of these commands issued in a user application or by the CICS-supplied transactions CEMT and CECI, CICS calls RACF to check that the terminal operator who initiated the transaction has authority to use the command for the specified resource.

# Security checking of transactions running under CEDF

When a transaction runs under the CEDF transaction, CICS determines the security processing for the target transaction from the logical OR of the CMDSEC settings in the resource definitions for the target transaction and the CEDF transaction.

Table 13 shows the security checking performed for the transaction XSUB for different settings of CMDSEC.

*Table 13. Security checking for transactions running under CEDF*

| CEDF | XSUB | Security checking |
|---|---|---|
| CMDSEC(YES) | CMDSEC(YES) | Any access to CICS commands causes a security check. |
| CMDSEC(YES) | CMDSEC(NO) | Any access to CICS commands causes a security check. (Logical OR results in CMDSEC on.) |
| CMDSEC(NO) | CMDSEC(YES) | Any access to CICS commands causes a security check. (Logical OR results in CMDSEC on.) |
| CMDSEC(NO) | CMDSEC(NO) | Access to CICS commands does not cause a security check. (Logical OR results in CMDSEC off.) |

To achieve the expected security processing for a transaction when it runs under CEDF, ensure that CMDSEC for the CEDF transaction definition is set to NO. The IBM-supplied definition of CEDF in the DFHEDF group specifies CMDSEC(YES). Definitions in the IBM-supplied groups cannot be modified, so to change the definitions, copy them to another group.

When CEBR or CECI is invoked from within EDF it is transaction-attach checked. In the same environment the CMDSEC and RESSEC definitions are forced regardless of what is coded in their transaction definitions.

When CEDF is used in **two-terminal mode** (the CEDF is entered at a different terminal from the transaction being tested), the authorities of the user executing the CEDF transaction are taken into account, as well as those of the user executing the transaction being tested. For each resource accessed by the tested transaction, both users must have access authority, otherwise a NOTAUTH condition is raised. This applies to all resource checks:

- Transaction-attach
- CICS resource
- CICS command
- Non-CICS resources accessed through the QUERY SECURITY command
- Surrogate user

**Note:** When an EXEC CICS SIGNON, EXEC CICS VERIFY PASSWORD, or EXEC CICS CHANGE PASSWORD command is issued by a transaction running under CEDF, the password (and new password, where applicable) is blanked out.

## CEMT considerations

In general, the resources that the CICS-supplied CEMT master terminal transaction operates on are the same as the equivalent SP-type commands shown in Table 11 on page 109 of the CICS API. If, in addition to normal transaction-attach security, you are using command security, you must ensure that authorized users of CEMT are also authorized for the CICS commands, as appropriate. If a user is authorized to initiate the CEMT transaction, but is not authorized for the resources on which the SP commands in Table 11 on page 109 depend, CICS returns a NOTAUTH

condition. To allow your system programmers to use the CEMT command in a command security environment, give them UPDATE access to the group profile that protects commands on which you want them to issue the PERFORM, SET, and DISCARD commands. UPDATE authority should be given to users specifying XPPT=YES and XCMD=YES when they issue a CEMT SET PROG(xxx) command. and you should provide READ access to the group profile that protects the commands on which you want them to issue only INQUIRE and COLLECT commands.

```
PERMIT profile_name CLASS(VCICSCMD) ID(user or group) ACCESS(READ)
PERMIT profile_name CLASS(VCICSCMD) ID(user or group) ACCESS(UPDATE)
```

### Resource names for CEMT

In general, the resource names of the CEMT commands correspond to the resource names of the equivalent CICS API command. However, there are some exceptions, and in all these cases it is the API resource name that you use to define the security profile to RACF.

- The CEMT system dump option is spelled differently from the EXEC CICS equivalent. CEMT INQUIRE|SET SYDUMPCODE corresponds to EXEC CICS INQUIRE|SET SYSDUMPCODE.
- The CEMT transaction dump option is spelled differently from the EXEC CICS equivalent. CEMT INQUIRE|SET TRDUMPCODE corresponds to EXEC CICS INQUIRE|SET TRANDUMPCODE.
- The CEMT PERFORM RESET option corresponds to the EXEC CICS PERFORM RESETTIME command.
- The AUXTRACE, INTTRACE, and GTFTRACE options of the CEMT INQUIRE and SET commands all correspond to the TRACEDEST option of the API.

To use the CEMT INQUIRE|SET NETNAME command, you need access to the resource TERMINAL, not NETNAME.

## Authorization failures

If you are running with CICS command security, CICS returns the NOTAUTH condition (RESP value 70) to your application, which is the same condition as for a resource security failure. (CICS also issues message DFHXS1111 to the CICS security transient data destination CSCS.) To test for this value in your application, we recommend you code DFHRESP(NOTAUTH) rather than explicitly coding a value. To distinguish between a command security failure and a resource security failure, check the RESP2 value. For a command security failure, CICS returns a value of 100 in RESP2. For a resource security failure, a value of 101 is returned in RESP2.

For background information on using RESP and RESP2, see the *CICS Application Programming Guide*; for programming information, see the *CICS Application Programming Reference* and the *CICS System Programming Reference* manuals.

# Chapter 9. Security checking using the QUERY SECURITY command

This chapter describes security checking by the user application using the EXEC CICS QUERY SECURITY command. The following topics are included:



An application can use the EXEC CICS QUERY SECURITY to request from RACF the level of access a user has to a particular resource. The user in this context is the user invoking the transaction that contains the QUERY SECURITY command.

Issuing the QUERY SECURITY command does not actually grant or deny access to a resource (by issuing a NOTAUTH condition), but instead enables the application program to determine what action to take based on the CICS-value data area (CVDA) values that CICS returns. (For programming information on CVDAs, see the *CICS Application Programming Reference* manual.)

**Note:** QUERY SECURITY is **not** affected by the RESSEC and CMDSEC keywords on the transaction definition.

There are two distinct forms of the QUERY SECURITY command, depending on the options chosen.

- QUERY SECURITY RESTYPE
- QUERY SECURITY RESCLASS

(For programming information on the QUERY SECURITY command, see the *CICS Application Programming Reference* manual.)

## How the QUERY SECURITY mechanism works

How the QUERY SECURITY mechanism works depends on:

- Whether SEC=YES or SEC=NO is specified in the system initialization parameters
- Whether SECPRFX=YES or SECPRFX=NO is specified in the system initialization parameters
- Which resource classes are active
- Whether the transaction issuing the request is subject to transaction routing, and if so:
  - Which ATTACHSEC parameter was specified on the connection definition
  - For RESTYPE('PSB') only, whether the PSBCHK system initialization parameter is specified as YES or NO

### SEC system initialization parameter

Table 14 assumes that the relevant resource class is active; for example, that XFCT=YES is specified when issuing QUERY SECURITY RESTYPE('FILE').

*Table 14. The effect of the SEC parameter on QUERY SECURITY commands*

| SEC | RACF Access | Query Security | | | |
|---|---|---|---|---|---|
| | | Read | Update | Control | Alter |
| YES | NONE<br>READ<br>UPDATE<br>CONTROL<br>ALTER | notreadable<br>readable<br>readable<br>readable<br>readable | notupdatable<br>notupdatable<br>updatable<br>updatable<br>updatable | notctrlable<br>notctrlable<br>notctrlable<br>ctrlable<br>ctrlable | notalterable<br>notalterable<br>notalterable<br>notalterable<br>alterable |
| NO | n/a | readable | updatable | ctrlable | alterable |

## SECPRFX system initialization parameter

If SECPRFX=YES is specified, CICS prefixes the resource with the CICS region userid. For example, issuing:

```
QUERY SECURITY RESTYPE('FILE') RESID('PAYFILE')
```

calls RACF to check the terminal user's access to *cics_region_userid.PAYFILE* if SECPRFX=YES is specified. If SECPRFX=NO is specified, *PAYFILE* is checked.

## Resource class system initialization parameters

Table 14 shows how the QUERY SECURITY RESTYPE command works if the system initialization parameter for the relevant resource class (for example, XFCT) system initialization parameter is active. If, however, the relevant X*name* parameter is **not** active (for example, if XFCT=NO has been specified), the resource is READABLE, UPDATABLE, CTRLABLE and ALTERABLE.

## Transaction routing

When the QUERY SECURITY command is issued from a transaction that has been routed to a remote system, CICS checks the link user's access to the specified resource, and the terminal user's access to the resource, if appropriate. For more information, see "Link security with LU6.2" on page 157, "Link security with LU6.1" on page 193, environment you are using. or "Link security with MRO" on page 202 according to the environment you are using.

In order to perform a check against the terminal user as well as the link user when transaction routing a QUERY SECURITY RESTYPE('PSB') RESID(*psb_name*), the following conditions must both be satisfied:

- ATTACHSEC on the connection definition must not be LOCAL (that is, it can be IDENTIFY, PERSISTENT, MIXIDPE, or VERIFY).
- PSBCHK=YES must be specified as a system initialization parameter in the remote system.

# QUERY SECURITY RESTYPE

Use the QUERY SECURITY RESTYPE command to query access levels to CICS resources (including DB2 resource definitions) contained in the classes activated at initialization by RACLIST. The response to the QUERY SECURITY command indicates the result of a resource check on this resource. If the resource is not defined to RACF, CICS does not grant access and the response is NOTREADABLE. Note that responses returned for category 3 transactions may not reflect that there is no attach time (TRANSATTACH) checking performed on category 3 transactions. Ensure the length of the resource name passed to RACF with a RESTYPE request is the actual maximum length for that resource type.

## RESTYPE values

RESTYPE is a resource type that corresponds to one of the X*name* system initialization parameters, and can take any of the values shown in Table 15.

*Table 15. QUERY SECURITY RESTYPE values*

| **RESTYPE value** | **Xname parameter** |
|---|---|
| DB2ENTRY | XDB2 |
| FILE | XFCT |
| JOURNALNAME | XJCT |
| PROGRAM | XPPT |
| PSB | XPSB |
| SPCOMMAND | XCMD |
| TDQUEUE | XDCT |
| TRANSACTION | XPCT |
| TRANSATTACH | XTRAN |
| TSQUEUE | XTST |

## RESID values

In all cases (except for the SPCOMMAND resource type), the resource identifiers (RESID values) are defined by your installation.

When defining RESID values, be aware of the effects of using blanks (X'40') in resource identifiers. For example, in:

```
QUERY SECURITY RESTYPE('PSB') RESID('A B')
```

the blank delimits the RESID and causes RACF to use a resource name of A.

For SPCOMMAND, the identifiers are predetermined by CICS. Table 16 lists the possible RESID values for SPCOMMAND.

*Table 16. RESID values for RESTYPE(SPCOMMAND)*

| | | |
|---|---|---|
| AUTINSTMODEL | AUTOINSTALL | CFDTPOOL |
| CONNECTION | DB2CONN | DB2ENTRY |
| DB2TRAN | DLIDATABASE | DOCTEMPLATE |
| DSNAME | DUMP | DUMPDS |
| ENQUEUE | EXITPROGRAM | FEPIRESOURCE |
| FILE | IRC | JOURNALMODEL |
| JOURNALNAME | MODENAME | MONITOR |
| PARTNER | PROCESS | PROFILE |
| PROGRAM | REQID | REQUEST |
| RESETTIME | RRMS | SECURITY |
| SHUTDOWN | STATISTICS | STORAGE |
| SYSDUMPCODE | SYSTEM | TASK |
| TCLASS | TCPIP | TCPIPSERVICE |
| TDQUEUE | TERMINAL | TRACEDEST |
| TRACEDEST | TRACEFLAG | TRACETYPE |
| TRANDUMPCODE | TRANSACTION | TSQUEUE |
| TSMODEL | TSPOOL | UOW |
| UOWDSNFAIL | UOWENQ | UOWLINK |
| VOLUME | VTAM | WEB |

QUERY SECURITY RESTYPE enables an application program to request from RACF the level of access a terminal user has to the specified resource for the environment in which the transaction is running.

Before calling RACF, CICS checks that the resource is installed. If the resource does not exist, CICS does not call RACF and returns the NOTFND condition. However, note that this check is **not** made for PSBs.

When the RESTYPE is TRANSATTACH and the transaction specified on the RESID parameter is unknown in the local region, a NOTFND condition is returned. However, if dynamic transaction routing is being used, there is no need for the transaction to be installed in the terminal-owning region. The transaction specified on the DTRTRAN system initialization parameter is attached if an unknown transaction identifier is entered.

Application programmers should be aware that the NOTFND condition does not necessarily indicate that a terminal user will be unable to enter a transaction identifier, because the transaction may be routed dynamically.

## Examples of values returned by QUERY SECURITY RESTYPE

This section gives a number of examples of the values returned by QUERY SECURITY RESTYPE, depending on what has been specified in the system initialization parameters.

### SEC=NO

When SEC=NO is specified, issuing:

```
QUERY SECURITY RESTYPE('FILE') RESID('PAYFILE') ALTER(alter_cvda)
```

returns:

```
alter_cvda = DFHVALUE(ALTERABLE)
```

because SEC=NO means that no security checking is done for the entire CICS region.

### SEC=YES and XFCT=NO

When SEC=YES and XFCT=NO are specified, issuing:

```
QUERY SECURITY RESTYPE('FILE') RESID('PAYFILE') ALTER(alter_cvda)
```

returns:

```
alter_cvda = DFHVALUE(ALTERABLE)
```

because XFCT=NO means that no security checking is done for files.

### SEC=YES, XDCT=YES, and SECPRFX=NO

When SEC=YES, XDCT=YES, and SECPRFX=NO are specified, issuing:

```
QUERY SECURITY RESTYPE('TDQUEUE') RESID('TDQ1') READ(read_cvda)
```

returns:

```
read_cvda = DFHVALUE(READABLE)
```

if the user has READ (or higher) access to 'TDQ1' in the DCICSDCT class or the ECICSDCT group class.

### SEC=YES, XTRAN=YES, and SECPRFX=YES

When SEC=YES, XTRAN=YES, and SECPRFX=YES are specified, issuing:

```
QUERY SECURITY RESTYPE('TRANSATTACH') RESID('TRN1') READ(read_cvda)
```

returns:

```
read_cvda = DFHVALUE(NOTREADABLE)
```

if the user **does not** have READ (or higher) access to `cics_region_userid.TRN1` in the TCICSTRN class or GCICSTRN group class.

### SEC=YES, XCMD=$USRCMD, and SECPRFX=NO

When SEC=YES, XCMD=$USRCMD, and SECPRFX=NO are specified, issuing:

```
QUERY SECURITY RESTYPE('SPCOMMAND') RESID('VTAM') UPDATE(updt_cvda)
```

returns:

```
updt_cvda = DFHVALUE(UPDATABLE)
```

if the user has UPDATE access (or higher) to 'VTAM' in the C$USRCMD or V$USRCMD class.

## QUERY SECURITY RESCLASS

Use the QUERY SECURITY RESCLASS command when you want to query access levels for non-CICS resources. RESCLASS is the name of a valid RACF general resource class, such as TERMINAL, FACILITY, or a similar installation-defined resource class. See "Other IBM-supplied RACF resource class names affecting CICS" on page 27. The class name identified by RESCLASS is treated literally, with no translation.

**Note:** The RACF classes DATASET, GROUP, and USER do not appear in the class descriptor table (CDT), which means that you cannot query against these classes.

Prefixing, as specified in the SECPRFX system initialization parameter, does not apply to QUERY SECURITY RESCLASS. That is, CICS does **not** prefix the RESID with the CICS-region userid before calling RACF.

If SEC=NO is specified in the system initialization parameters, QUERY SECURITY RESCLASS always returns READABLE, UPDATABLE, CTRLABLE and ALTERABLE.

For QUERY SECURITY RESCLASS, both the RESID **and** the RESIDLENGTH option must be specified. The maximum length of a resource (RESID) within a RACF class is specified in the class descriptor table (CDT). When defining RESID values, you should be aware of the effects of including blanks (X'40') in RESIDs. For example, in:

```
QUERY SECURITY RESCLASS('MYCLASS') RESID('MY PROFILE') RESIDLENGTH(10)
```

the presence of a blank causes an INVREQ condition. This is because RACF does not allow blanks to be embedded in a profile name.

**Note:** To determine access to CICS resources you should normally use RESTYPE, when the resource class is determined by the *Xname* system initialization parameter. However, if, for special reasons, you want to inquire about specific CICS resource classes, you should note that the class name must be

the member class, and **not** the group class; that is, CCICSCMD, and not VCICSCMD. The profiles in the grouping class are checked automatically if the member class has been activated by RACLIST. For example, if SEC=YES, and XCMD=YES are specified, both CCICSCMD and VCICSCMD are activated by RACLIST in the CICS region, which means that QUERY SECURITY RESCLASS('CCICSCMD') checks profiles in both CCICSCMD and VCICSCMD.

CICS can RACLIST groups only if the relevant X*name* classes are active (for example, XCMD=YES or XCMD=$USRCMD).

You can also use the RESCLASS option for querying access to DB2ENTRY resources defined in a user-defined resource class, which you specify to CICS on the XDB2 system initialization parameter. The rules about activating classes by means of the RACLIST command also apply to DB2ENTRY resource classes named on the XDB2 system initialization parameter. See "Resource classes for DB2ENTRYs" on page 28 for more information about user-defined DB2ENTRY resource clases.

Issuing QUERY SECURITY RESCLASS('TERMINAL') checks profiles in both TERMINAL and GTERMINL (the terminal grouping class) only if the TERMINAL class has been activated by RACLIST at the system level by the command:

```
SETROPTS RACLIST(TERMINAL)
```

For **non-CICS** resource classes, you can issue the SETROPTS RACLIST(classname) command to perform a global RACLIST. See "Specifying user-defined resources to RACF" on page 226 for details.

## Querying a user's surrogate authority

To query a user's surrogate authority, you can use the QUERY SECURITY command with the RESCLASS('SURROGAT') option. You also need to specify the RESID and RESIDLENGTH options. The RESID value you should provide is described is **not** in "RACF definitions for surrogate user checking" on page 107. However, this command is **not** controlled by the XUSER system initialization parameter, so you might obtain an unexpected response of NOTREADABLE if XUSER=NO has been specified. For example, to check whether the current user is allowed to start a transaction with a new userid of NEWUSER, when XUSER=YES is specified, issue the command:

```
QUERY SECURITY RESCLASS('SURROGAT') RESID('NEWUSER.DFHSTART')
RESIDLENGTH(16)  READ(read cvda)
```

## Logging for QUERY SECURITY RESTYPE and RESCLASS

You can control logging on the QUERY SECURITY command by specifying one of the following options:

- LOG
- NOLOG
- LOGMESSAGE(*cvda*), where *cvda* value is 54 for LOG, or 55 for NOLOG

The default is LOG.

If logging is in effect, and the terminal user does not have the requested access to the specified resource, message DFHXS1111 is issued to the CICS security transient

data destination CSCS. Where relevant, RACF message ICH408I is also issued. SMF records may also be recorded, depending on the auditing and logging options that have been specified for that resource. For more information, see the *OS/390 Security Server (RACF) Auditor's Guide*.

For programming information about CVDAs, refer to the *CICS System Programming Reference* manual.

## Uses for QUERY SECURITY RESTYPE and RESCLASS

You can use the two forms of the QUERY SECURITY command in a number of different ways to customize resource security checking within an application. This section gives a number of examples of doing so.

### Changing the level of security checking

You can use QUERY SECURITY to perform a different level of security checking from that which CICS would perform for application programs that specify RESSEC(YES) or CMDSEC(YES).

For example, suppose a transaction has RESSEC(YES) and contains a number of EXEC CICS READ FILE commands and a number of EXEC CICS WRITE FILE commands. For each command, CICS performs a security check to ensure that the terminal user has access to the relevant file, even though the same file may be being accessed each time. An alternative to this is to switch off security checking at the transaction level by specifying RESSEC(NO) on the transaction definition and then, when the application starts, execute a command such as:

```
EXEC CICS QUERY SECURITY RESTYPE('FILE') RESID(file_name) UPDATE(cvda)
```

This command allows the transaction to continue without any further calls to RACF.

**Note:** Switching resource security checking off, using RESSEC(NO), means that **all** resource checks—not just of files as in the above example—are bypassed.

### Checking which transactions to offer a user

You can use the QUERY SECURITY command to check whether a user is authorized to use a particular transaction **before** displaying the transaction code as part of an introductory menu. When you use the command for this purpose, you will probably want to avoid logging the checks for users who are not allowed to use certain transactions. To do this, use the NOLOG option.

### Example of use of QUERY SECURITY RESCLASS

Normal CICS resource security checking for files operates at the file level only. You can use QUERY SECURITY to enable your application to control access to data at the **record** or **field** level.

To do this, define resource names (which represent records or fields within particular files) with the appropriate access authorizations for the records or fields you want to control. You could define these resources in an installation-defined RACF general resource class and then use the QUERY SECURITY RESCLASS command to check a terminal user's access to a specific field within a file before displaying or updating the field. (The application logic would determine which field.) For example:

```
QUERY SECURITY RESCLASS('$FILEREC') RESID('PAYFILE.SALARY')
RESIDLENGTH(14) READ(read_cvda) NOLOG
```

where '$FILEREC' is an installation-defined RACF general resource class. For more information, see “Designing applications to use the user-defined resources” on page 228.

# Chapter 10. Security for CICS-supplied transactions

This chapter discusses security for CICS-supplied transactions, and contains a number of recommendations to ensure that your CICS regions are adequately protected. Where applicable, it describes the recommended security specifications that you will need for the CICS-supplied transactions defined in the group list DFHLIST, and stored in the CICS system definition data set (CSD). These recommendations cover all CICS-supplied transactions—those that are intended for use from a user terminal or console, and those that are for CICS internal use only. (For information about the CICS-supplied groups of resource definitions, and the DFHLIST group list, see the *CICS Resource Definition Guide*.)

By default, all CICS transactions are subject to RACF protection (with the exception of category 3 transactions—see "JES spool protection in a CICS environment" on page 55), unless you run your CICS regions with transaction security switched off. You can do this either by:

- Specifying the system initialization parameter SEC=NO, which switches off all security checking, or
- Specifying the system initialization parameter XTRAN=NO, which switches off transaction-attach security checking only.

There is no parameter on the transaction resource definition that allows you to run with transaction security on some transactions but not others. If you are running with transaction security (SEC=YES and XTRAN=YES), CICS issues a security check for each transaction attach, other than a transaction within category 3, to establish whether the user is permitted to run that transaction.

CICS—supplied transactions CDBN and CSXM are not subject to security checking, and are exempt from security categorization. Any security definitions for these transactions are redundant.

## Categories of CICS-supplied transactions

For the purposes of this description, we divide the RACF profile definitions for your CICS-supplied transactions into three categories. Each transaction is identified within a category that describes its use within CICS. Each category specifies the recommended security specifications you need, in terms of both the CICS transaction definitions and the corresponding RACF profiles.

The three categories contain all the required CICS transactions, which are generated in their designated groups when you initialize your CICS system definition data set (CSD). The CSD does not include the CICS sample transactions (those that are in groups starting with DFH$). Sample applications should not require RACF protection, because you are unlikely to install them on a CICS production system.

| See "Chapter 22. Implementing CICSPlex SM security" on page 267 for details of
| CICSPlex SM-related transactions.

## Category 1 transactions

Category 1 transactions are never associated with a terminal—that is, they are for CICS internal use only, and should not be invoked from a user terminal. CICS checks that the region userid has the authority to attach these transactions.

However, if the region userid is not authorized to access all of the category 1 transactions, CICS issues message DFHXS1113 and fails to initialize. For category 1 transactions, specify the following:

**To CICS**
RESSEC(NO) and CMDSEC(NO) on the transaction resource definition.

**To RACF**
UACC(NONE) and AUDIT(FAILURES) in the corresponding transaction profiles. AUDIT(FAILURES) is the default and need not be specified. The access list should contain only userids (or groups containing userids) that can be specified as CICS region userids.

For example:

```
RDEFINE GCICSTRN CICSCAT1 UACC(NONE)
         ADDMEM(CSPQ CDBD . . . . . . . . FCRD TSDQ)
         NOTIFY(security_admin_userid)
         OWNER(userid or groupid)
PERMIT CICSCAT1 CLASS(GCICSTRN) ID(cat1grp1,...,cat1grpz) ACCESS(READ)
```

By defining these transactions to RACF with UACC(NONE), and an access list, you prevent any terminal user initiating these transactions (accidentally or otherwise). It is important that you do this, because permitting the initiation of these transactions at a terminal has unpredictable results. The sample CLIST DFH$CAT1 has been provided to help you define the category 1 profiles to RACF. The sample CLIST can be seen in library *CICSTS13.CICS.SDFHSAMP.* Table 17 lists the category 1 transactions.

*Table 17. Category 1 transactions*

| CSD group | Transaction | Program invoked | Description |
|---|---|---|---|
| DFHBMS | CSPQ | DFHTPQ | Performs terminal page cleanup (BMS) |
| DFHDBCTL | CDBD | DFHDBDI | Provides DBCTL disable function |
| | CDBO | DFHDBCT | Provides DBCTL control function |
| DFHDB2 | CEX2 | DFHD2EX2 | Provides CICS DB2 protected thread purge mechanism and other CICS DB2 services. |
| | CDBQ | DFHD2CM2 | CICS DB2 attachment facility shutdown quiesce transaction |
| | CDBF | DFHD2CM3 | CICS DB2 attachment facility shutdown force transaction |
| DFHDLI | CSGX | DFHDLG | Processes DL/I global commands |
| | CSSX | DFHDLS | DL/I status condition processor |

*Table 17. Category 1 transactions (continued)*

| CSD group | Transaction | Program invoked | Description |
|---|---|---|---|
| DFHFEPI | CSZI | DFHSZRMP | Implements Front End Programming Interface |
| DFHIIOP | CIOD | DFHIIOPA | IIOP ORB function |
| | CIOF | DFHIIOPA | CORBA GenericFactory |
| | CIOR | DFHIIOP | IIOP receiver program |
| DFHLGQC | CSQC | DFHLGQC | Quiesces CICS |
| DFHOPCLS | CSFU | DFHFCU | Opens user file-control managed files |
| DFHRMI | CRSY | DFHRMSY | Resynchronizes resource manager |
| DFHSIGN | CESC | DFHCESC | Processes time-out and sign-off for idle terminals |
| DFHSPI | CATA | DFHZATA | Defines autoinstall automatic terminal |
| | CATD | DFHZATD | Deletes autoinstall terminal |
| | CDTS | DFHZATS | Provides remote single delete transaction |
| | CITS | DFHZATS | Provides remote autoinstall transaction |
| | CMTS | DFHZATS | Remote mass delete transaction |
| | CFTS | DFHZATS | Provides remote mass flag transaction |
| | CRMD | DFHZATMD | Provides remote mass delete transaction |
| | CRMF | DFHZATMF | Provides remote mass flag transaction |
| DFHRSPLG | CLSG | DFHZRLG | Logs responses for protected message support |
| DFHSTAND | CSTE | DFHTACP | Processes terminal abnormal conditions |
| | CXCU | DFHZXCU | Performs XRF tracking catch-up |
| | CXRE | DFHZXRE | Reconnects terminals following XRF takeover |
| DFHVTAM | CSNE | DFHZNAC | Provides VTAM node error recovery |
| DFHWEB | CWBG | DFHWGBG | CICS Web support cleanup transaction |
| | CWXN | DFHWBXN | CICS Web support attach transaction |

*Table 17. Category 1 transactions (continued)*

| CSD group | Transaction | Program invoked | Description |
|---|---|---|---|
| N/A | CSHQ | DFHSHSY | Scheduler services domain long running task |
| | CPLT | DFHSIPLT | Initializes PLT processing |
| | CSKP | DFHRMXN3 | Writes system log activity keypoint |
| | CSSY | DFHAPATT | Provides entry point attach |
| | CGRP | DFHZCGRP | Provides VTAM persistent sessions transaction |
| | COVR | DFHZCOVR | Provides open VTAM retry transaction |
| | CSTP | DFHZCSTP | Provides terminal control transaction |
| | CSOL | N/A | CICS socket listener transaction |
| | CSHA | N/A | CICS BTS scheduler services handle-abend transaction |
| | CFCL | DFHFCDL | File control CFDT load |
| | CFOR | DFHFCOR | File control offsite recovery |
| | CFQR | DFHFCQT | File control RLS quiesce receive |
| | CFQS | DFHFCQT | File control RLS quiesce send |
| | CFSL | DFHDTLX | File control SDT load |
| | CSFR | DFHFCRD | File control RLS cleanup |
| | CTSD | DFHTSDQ | TS delete recoverable queue |

## Category 2 transactions

Category 2 transactions either are initiated by the terminal user, or are associated with a terminal. Restrict authorizations to initiate these transactions to userids belonging to specific RACF groups.

For the CICS resource definitions, the IBM-supplied transactions are defined with the recommended RESSEC and CMDSEC options. In particular, CECI, CEDF, CEMT, and CEST are all supplied with RESSEC(YES) and CMDSEC(YES). The mirror transactions are defined with RESSEC(YES). If you need to change any of these definitions, you can do so by copying them to another group. We recommend that you do **not** change the supplied definitions of any other transactions.

For most category 2 transactions, you are recommended to specify the following to RACF:

- UACC(NONE) and AUDIT(FAILURES) in the transaction profile.

  AUDIT(FAILURES) is the default, and need not be specified.
- Access list as appropriate.

It is unlikely that you will want to give all users access to all of the transactions in this category; consider defining them in several subcategories. In the examples that follow, the category 2 transactions are further subdivided into a number of groups. Please note that these are only examples. You can choose to group CICS transactions in the ways that best suit your installation's needs.

- SYSADM, containing: CBRC, CDBT, CEDA, CEMT, and CETR
- DEVELOPER, containing: CEBR, CECI, CECS, CEDB, and CEDF
- INQUIRE, containing: CDBI and CEDC
- OPERATOR, containing: CEOT, CEST, CMSG, and CWTO
- INTERCOM, containing: CEHP, CEHS, CPMI, CRTE, CSMI, CSM1, CSM2, CSM3, CSM5, and CVMI
- WEBUSER, containing: CWBA

  If function shipping is being used, the mirror transactions must be available to remote users in a function shipping environment. When a database or file resides on another CICS region, CICS function ships the request to access the data, and this request runs under one of the CICS-supplied mirror transactions. This means that:

  – The terminal user running the application must be authorized to use the mirror transaction. (See "Chapter 5. Transaction security" on page 79.)
  – The terminal user must also be authorized to use the data that the mirror transaction accesses. (See "Chapter 6. Resource security" on page 85.) The mirror transactions are supplied with RESSEC(YES) defined; so, even if the user's transaction specifies RESSEC(NO), the mirror transaction fails if the user is not authorized to access the data.

    If you do not use resource security checking, change the mirror transaction definitions to specify RESSEC(NO). Because the mirror transactions are an IBM-protected resource, first copy these definitions into your own groups and then change them.

  ALLUSER, containing CMAC and CSGM—the CICS "messages and codes" and "good morning" transactions, respectively. We recommend you define CMAC and CSGM (or, if your installation does not use CSGM, whatever transaction is defined as GMTRAN) as UACC(READ) in a group of their own, because all users need access to them. If your installation uses CSGM as its "good morning" transaction, users who are not authorized to use CSGM will receive message DFHAC2002 when they attempt to use CICS. Also include your "goodnight" transaction in this group, if you defined one with the GNTRAN system initialization parameter

The sample CLIST DFH$CAT2 (in library *CICSTS13.CICS.SDFHSAMP*) can help you define the category 2 profiles to RACF. If you want to use this example setup, review this CLIST and make the changes necessary for your installation before running it. If you want to use a different setup, you can adapt this CLIST, or provide your own.

Figure 5 on page 130 shows how to use RDEFINE and PERMIT commands to define the example groups for category 2 transactions.

```
RDEFINE GCICSTRN SYSADM UACC(NONE)
         ADDMEM(CDBC,CDBT,CBRC,CEMT,CETR,CEDA)
         NOTIFY(security_admin_userid)
         OWNER(userid or groupid)
PERMIT   SYSADM  CLASS(GCICSTRN) ID(sysgrp1,..,sysgrpz) ACCESS(READ)
RDEFINE GCICSTRN DEVELOPER UACC(NONE)
         ADDMEM(CEDF,CEBR,CECI,CECS,CEDB)
         NOTIFY(security_admin_userid)
         OWNER(userid or groupid)
PERMIT   DEVELOPER  CLASS(GCICSTRN) ID(devgrp1,..,devgrpz) ACCESS(READ)
RDEFINE GCICSTRN INQUIRE UACC(NONE)
         ADDMEM(CDBI,CEDC)
         NOTIFY(security_admin_userid)
         OWNER(userid or groupid)
PERMIT   INQUIRE  CLASS(GCICSTRN) ID(inqgrp1,..,inqgrpz) ACCESS(READ)
RDEFINE GCICSTRN OPERATOR UACC(NONE)
         ADDMEM(CWTO,CRTE,CMSG,CEST,CEOT)
         NOTIFY(security_admin_userid)
         OWNER(userid or groupid)
PERMIT   OPERATOR  CLASS(GCICSTRN) ID(opsgrp1,..,opsgrpz) ACCESS(READ)
RDEFINE GCICSTRN INTERCOM UACC(NONE)
         ADDMEM(CEHP,CEHS,CPMI,CSMI,CSM1,CSM2,CSM3,CSM5,CVMI)
         NOTIFY(security_admin_userid)
         OWNER(userid or groupid)
PERMIT   INTERCOM  CLASS(GCICSTRN) ID(intrgrp1,..,intrgrpz) ACCESS(READ)
RDEFINE GCICSTRN ALLUSER UACC(READ)
         ADDMEM(CMAC,CRTX,CSGM)
         NOTIFY(security_admin_userid)
         OWNER(userid or groupid)
PERMIT WEBUSER CLASS (GCICSTRN) ID(webgrp1,..,
webgrpz) ACCESS(READ)
RDEFINE GCICSTRN WEBUSER UACC(NON)
ADDMEM(CWBA, )
         NOTIFY(security_admin_userid)
         OWNER(userid or groupid)
```

*Figure 5. Example of defining groups for category 2 transactions*

**Notes:**

1. With RESSEC(YES) and CMDSEC(YES) defined for these transactions, you must ensure that the user groups authorized to use the transactions are also authorized to access the CICS resources and commands that the transactions use.
2. If you protect a resource with a resource group profile, you should avoid protecting the same resource with another profile. If the profiles are different (for example, if they have different access lists), RACF merges the profiles for use during authorization checking. Not only can the merging have a performance impact, but it can be difficult to determine exactly which access authority applies to a particular user. (See the *OS/390 Security Server (RACF) Security Administrator's Guide* for further information.)

Table 18 lists the category 2 transactions.

*Table 18. Category 2 transactions*

| CSD group | Transaction | Program invoked | Description |
|---|---|---|---|
| DFHCMAC | CMAC | DFHCMAC | Displays CICS messages online |
| DFHCONS | CWTO | DFHCWTO | Writes to console operator |

*Table 18. Category 2 transactions (continued)*

| CSD group | Transaction | Program invoked | Description |
|---|---|---|---|
| DFHDBCTL | CDBC | DFHDBME | DBCTL interface menu transaction |
| | CDBI | DFHDBIQ | DBCTL interface inquiry transaction |
| | CDBM | DFHDBMP | DBCTL operator transaction. A maintenance function that enables storage of IMS commands. The transaction uses BMS and runs on a subset of devices supported by BMS. |
| | CDBT | DFHDBDSC | DBCTL interface disconnection transaction |
| DFHDB2 | DSNC | DFHD2CM1 | DB2 attachment facility transaction |
| DFHDBDSC | CDBT | DBCTL | Provides disconnection transaction |
| DFHEDF | CEDF | DFHEDFP | Provides execution diagnostic facility |
| | CEBR | DFHEDFBR | Browse temporary storage |
| DFHFE | CSFE | DFHFEP | Tests field engineering terminal |
| DFHINDT | CIND | DFHINDT | Provides the in-doubt test tool |
| DFHINTER | CECI | DFHECIP | CICS command interpreter |
| | CECS | DFHECSP | Checks CICS command syntax |

*Table 18. Category 2 transactions (continued)*

| CSD group | Transaction | Program invoked | Description |
|---|---|---|---|
| DFHISC | CDFS | DFHDFST | Dynamic starts with interval |
| | CEHP | DFHCHS | Provides CICS OS/2® remote server mirror |
| | CEHS | DFHCHS | Provides CICS/VM™ remote server mirror |
| | CPMI | DFHMIRS | Provides CICS OS/2™ LU6.2 mirror |
| | CRTE | DFHRTE | Provides start transaction routing session |
| | CRTX | N/A | Provides default dynamic routing transaction |
| | CSHR | DFHMIRS | Provides scheduler services remote routing |
| | CSMI | DFHMIRS | Provides ISC mirror transaction |
| | CSM1 | DFHMIRS | Provides ISC SYSMSG model |
| | CSM2 | DFHMIRS | Provides ISC scheduler model |
| | CSM3 | DFHMIRS | Provides ISC queue model |
| | CSM5 | DFHMIRS | Provides ISC DL/I model |
| | CVMI | DFHMIRS | Provides LU6.2 synclevel 1 mirror |
| DFHMSWIT | CMSG | DFHMSP | Provides message switching |
| DFHOPER | CBAM | DFHECBAM | Browses business transaction services (BTS) objects |
| | CEMT | DFHEMTP | Processes master terminal command |
| | CEOT | DFHEOTP | Inquires on user's own terminal status |
| | CEST | DFHESTP | Processes supervisor terminal command |
| | CETR | DFHCETRA | Provides inquire and set trace options |
| DFHSDAP | CESD | DFHCESD | Provides shutdown assist transaction |

*Table 18. Category 2 transactions (continued)*

| CSD group | Transaction | Program invoked | Description |
|---|---|---|---|
| DFHSPI | CEDA | DFHEDAP | Provides resource definition online—full |
| | CEDB | DFHEDAP | Provides resource definition online—restricted |
| | CEDC | DFHEDAP | Views resource definition online |
| DFHVTAM | CSGM | DFHGMM | Provides CICS good morning message |
| DFHWEB | CWBA | DFHWBA | CICS web support alias transaction |

## Category 3 transactions

Category 3 transactions are either initiated by the terminal user or associated with a terminal. **All CICS terminal users**, whether they are signed on or not, require access to transactions in this category. For this reason, category 3 transactions are exempt from any security check, and CICS permits any terminal user to initiate these transactions.

For category 3 transactions you are recommended to specify RESSEC(NO) and CMDSEC(NO) on the CICS transaction resource definition. These transactions should be defined to RACF, but this definition does not affect actual task attach-time processing. It is used only for QUERY SECURITY purposes.

Table 19 lists the category 3 transactions.

*Table 19. Category 3 transactions*

| CSD group | Transaction | Program invoked | Description |
|---|---|---|---|
| DFHHARDC | CSPP | DFHP3270 | Provides 3270 print support |
| DFHBMS | CSPG | DFHTPR | Provides BMS terminal paging |
| | CSPS | DFHTPS | Schedules BMS paging transaction |

*Table 19. Category 3 transactions (continued)*

| CSD group | Transaction | Program invoked | Description |
|---|---|---|---|
| DFHISC | CLS1 | DFHZLS1 | Provides ISC LU services model |
| | CLS2 | DFHLUP | Provides ISC LU services model |
| | CLS3 | DFHCLS3 | ISC LU services model |
| | CLS4 | DFHCLS4 | Manages password expiry |
| | CMPX | DFHMXP | Ships ISC local queuing |
| | CQPI | DFHCLS5 | Inbound connection quiesce and architected transaction |
| | CQPO | DFHCLS5 | Outbound connection quiesce and architected transaction |
| | CRSR | DFHCRS | Provides ISC remote scheduler |
| | CSSF | DFHRTC | Cancels CRTE transaction routing session |
| | CXRT | DFHCRT | Provides Transaction routing relay |
| DFHISCT | CLQ2 | DFHLUP | Outbound resynchronization for APPC and MRO |
| | CLR2 | DFHLUP | Inbound resynchronization for MRO |
| | CLR1 | DFHZLS1 | Inbound CNOS for APPC and MRO |
| DFHRSEND | CSRS | DFHZRSP | Synchronizes 3614 message |
| DFHSIGN | CESN | DFHSNP | Signs on terminal user |
| | CESF | DFHSNP | Signs off terminal user |
| | CEGN | DFHCEGN | Schedules goodnight transaction |
| DFHSPI | CATR | DFHZATR | Deletes autoinstall restart terminal |
| DFHSTAND | CQRY | DFHQRY | Provides ATI query support |
| | CSAC | DFHACP | Processes program abnormal condition |

*Table 19. Category 3 transactions (continued)*

| CSD group | Transaction | Program invoked | Description |
|---|---|---|---|
| DFHVTAMP | CSCY | DFHCPY | Provides 3270 screen print |
| | CSPK | DFHPRK | Provides 3270 screen print support |
| | CSRK | DFHRKB | Provides 3270 screen print—release keyboard |

# Chapter 11. Security for CICS Web support

This chapter discusses the following:

- Security considerations for the HTML template manager PDS, and CICS Web support transactions.
- The operation of the sample security analyzer, converter, and sign-on program.

## Security for the HTML template manager PDS

If your CICS programs use the partitioned data set facilities of the HTML template manager, the CICS region user ID must have READ authority for the data set described in the DOCTEMPLATE PDS definition.

## Security for CICS Web support transactions

You can specify security requirements for each of the transactions that compose CICS Web support. In the following explanations:

- Authority to attach means that the associated user must be given READ authority to the named transaction in the resource class specified by the XTRAN system initialization parameter.
- Authority to start means that the associated user must be given READ authority to the named transaction in the resource class specified by the XPCT system initialization parameter.
- Authority to specify a user ID means that the associated user must be given READ authority to the `userid.DFHSTART` profile in the SURROGAT resource class, if the XUSER system initialization parameter is specified as YES.
- Authority to read a CICS file means that the associated user must be given READ authority to the named file in the resource class specified by the XFCT system initialization parameter.
- Authority to update a CICS file means that the associated user must be given UPDATE authority to the named file in the resource class specified by the XFCT system initialization parameter.
- Authority to use a program means that the associated user must be given READ authority to the named program in the resource class specified by the XPPT system initialization parameter.

.

### Security for the alias

The alias transaction executes as a non-terminal CICS transaction. Its name is user-specified. If you use the default analyzer, DFHWBADX, the transaction name is the second “index level” in the absolute path specified by the client, and is usually CWBA. APAR PQ36169 30/3/00

The alias transaction executes under the user ID specified in **wbra_userid**, if it is specified by the analyzer, otherwise it executes under the same user ID as the server controller. If running with the secure socket layer (CLIENTAUTH). If running with the option SSL(CLIENTAUTH) on the TCPIPSERVICE definition, **wbra_userid** may contain a user ID on input to the analyzer. If the AUTHENTICATE option on the TCPIPSERVICE is specified as anything other than NO, **wbra_userid** will always contain a user ID on input to the analyzer.

If you define your own alias transactions, however, this user ID must have the following authorities:

- The authority to attach the alias transaction
- The authority to access any CICS resources used by the alias transaction, if it is defined with the RESSEC(YES) option
- The authority to access any CICS system programming commands used by the alias transaction, if it is defined with the CMDSEC(YES) option

## Sample programs for security

Two sets of sample programs are provided:

- The security sample programs, described in "The security sample programs":
  - The security analyzer, DFH$WBSA
  - The security converter, DFH$WBSC
  - The sign-on program, DFH$WBSN
- The basic authentication sample programs, described in "The basic authentication sample programs" on page 139:
  - The basic authentication analyzer, DFH$WBAU
  - The basic authentication converter, DFH$WBSB

The CICS resource definitions for these programs are in group DFH$WBSN.

## The security sample programs

The security sample programs use the state management sample program, DFH$WBST.

A typical sequence of interactions between a user and CICS Web support might be as follows:

1. The end user sends an HTTP request in which the URL has no query string.
2. The security analyzer checks the URL for a converter name, alias name, program name, and query string. As there is no query string, it sets its outputs so that the converter is the security converter sample program DFH$WBSC, while the alias and CICS program are the ones requested in the URL. The user token output is zeros.
3. The **Decode** function of the security converter, finding a zero user token, calls the Create function of the state management sample program to assign a token. It saves the token in its user token output. It uses the Store function of the state management program to save the original URL. It sets the CICS program name to DFH$WBSN, the security sign-on sample program.
4. The sign-on program builds an HTML form asking for a user ID and a password. The form specifies an HTML ACTION that generates a URL. The generated URL causes the sign-on program to be invoked again, but with the state management token as its query string.
5. The **Encode** function of the security converter builds the HTTP response.
6. The user gets the form, fills in the user ID and the password, and sends it back.
7. The security analyzer finds a query string. It uses the Retrieve function of the state management program to validate the token. As the token is not yet associated with a valid user ID, it sets its outputs so that the converter name is the security converter. The state management token is passed as the user token.

8. The sign-on program extracts the user ID and password from the form, and uses EXEC CICS VERIFY PASSWORD to validate the user ID. It uses the Store function of the state management program to associate the validated user ID with the token.
9. The **Encode** function of the security converter builds the HTTP response, and adds a redirection (HTTP response 302) to it, incorporating the original URL.
10. The web browser receives the redirected URL, and sends a request for the original program with the token that identifies the validated user ID.
11. The security analyzer finds that the query string is a valid user token associated with a user ID, so the original request proceeds.

Once the user token has been established as the key to the authenticated user ID, it is the responsibility of the CICS program, or the converter that builds the HTTP response, to ensure that any URLs that are generated to continue the conversation with the client contain the conversation token as query string. This ensures that subsequent programs in the conversation execute under the specified user ID. Since the CICS program is already running with the correct conversation token as its query string, it can extract its value by using the environment variable program to obtain the value of the query string. If necessary, the correct value for the conversation token can be substituted into HTML templates by using the symbol

```
&QUERY_STRING;
```

provided that the environment variable string has first been loaded into the symbol table in the template manager's page environment.

## The basic authentication sample programs

APAR PQ 36169 MJO 16/05/00

**Note:** You do not need to use sample programs to implement HTTP basic authentication. You can specify AUTHENTICATE(BASIC) or AUTHENTICATE(AUTOMATIC) on the TCPIPSERVICE definition instead.

The basic authentication sample programs use HTTP basic authentication. On the first reference by a Web browser to a CICS region (identified by its application ID), the browser will prompt the user for a user ID and password. The user ID and password supplied at the prompt will be sent to the CICS region for every request. CICS will validate the user ID and password for each request. There is no user prompt for the second or later requests.

The user ID and password are encoded, but not encrypted, for transmission.

To use the security analyzer sample program, you must specify its name as the Analyzer Program name in panel DFHWB02 when you enable the interface.

The basic authentication analyzer searches the incoming HTTP headers for an Authorization header with a **basic** operand. If it finds one, it decodes the BASE64-encoded user ID and uses it as the alias user ID. It always schedules DFH$WBAU as the converter.

The basic authentication converter searches the incoming HTTP headers for an Authorization header. It decodes the user ID and the password. It uses VERIFY PASSWORD to validate the password. If the user ID and password combination is invalid, or if the Authorization header is absent, an HTTP 401 response is returned

to the Web browser, and the user is prompted for a password. If the user ID and password combination is correct, the application continues, and runs under the specified user ID.

## Using the secure sockets layer

The secure sockets layer (SSL) is an architecture for allowing Internet servers and clients to authenticate each other and to encrypt the data flowing between them. When a server uses SSL it must hold three pieces of data: a private key, a public key, and a server certificate. Before you can use SSL with CICS you need to create a key database, which contains these three items of data. The database is created as a file in the hierarchical file system (HFS) of OS/390 Unix System Services by using the `gskkyman` utility program.

The `gskkyman` program runs under the OpenEdition® shell of TSO/E, which is entered by using the OMVS command. For further information on `gskkyman`, enter the following command to display help information, or see the *Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* manual, (SC24 5877):

```
gskkyman -h
```

When you create the key database with `gskkyman`, you will be prompted for a password that will be used to protect the database. You will need to specify the password whenever you access the database. Before using the key database in CICS, you should use `gskkyman` to create a **stashed password file**, which will allow CICS to access the database without specifying the password.

When you add a server certificate to the key database, you can give the certificate a name, or **certificate label**. You can also choose to make one of the certificates the **default certificate** for that key database.

### Establishing an SSL service

When you have created the key database using `gskkyman` you can establish SSL services in CICS as follows:

- Use KEYFILE to specify the fully qualified HFS filename of the key database created by the **gskkyman** utility program. When you specify this parameter, the CICS region userid must be authorized to read the specified HFS file. Be aware that the database name is case sensitive. In order to access the key database created by `gskkyman` a password is required. `gskkyman` option 11 provides the facility to create a 'stashed password file', which contains an encrypted version of the required password. CICS reads and decrypts this stashed password to gain access to the key database. The following is an example:

  ```
  KEYFILE=/u/cicsssl/keys/key.kdb
  ```

  In this case, `gskkyman` option 11 will create the stashed password file as

  ```
  /u/cicsssl/keys/key.sth
  ```

  This name is derived from the key database filename.

  If you code the password on the KEYFILE parameter of your system initialization table or as an override, this will have to be changed to use a 'stashed password file' instead. Coding a password on the KEYFILE parameter is no longer supported.

  **Note:** The maximum length of the KEYFILE parameter is 47 characters.

- Use the CEDA transaction to define a TCPIPSERVICE to accept the SSL protocol requests. Choose a TCP/IP port number upon which you will provide the SSL service, and specify this in the PORTNUMBER field of the TCPIPSERVICE definition. You must specify SSL(YES) or SSL(CLIENTAUTH) to provide secure socket layer services. The TCPIPSERVICE uses one of the certificates in the key database as its server certificate. You can select a particular certificate within the key database by specifying the certificate label (which you assigned in `gskkyman`) in the CERTIFICATE field of the TCPIPSERVICE definition. If you omit the CERTIFICATE field from the TCPIPSERVICE definition, CICS uses the certificate in the database that you nominated as the default certificate.
- Activate the TCPIPSERVICE by specifying STATUS(OPEN) and installing the definition, or by installing the definition and later using CEMT SET TCPIPSERVICE OPEN.

When the TCPIPSERVICE is activated, clients connecting to the specified port number must use the SSL protocol to communicate with CICS. They do this by specifying `https` as the protocol in the URL used to access the service. For example, if PORTNUMBER(8081) and SSL(YES) are specified, clients might access your service with a URL of the form:

```
https://www.yourservice.com:8081/cics/cwba/DFH0WBCA
```

## Using client authentication

When SSL(CLIENTAUTH) is specified in a TCPIPSERVICE definition, the service uses a feature of secure sockets layer known as client authentication. This protocol then requires that the client, as well as the server, must have a certificate. The client certificate is received by CICS during the SSL negotiation. The received certificate can be used in two ways:

- It can be used to determine the userid under which the CICS transaction can be executed.
- It can provide information about the client that can be extracted by the EXEC CICS EXTRACT CERTIFICATE command. For more information about this command, see the *CICS Application Programming Reference*

Simply specifying SSL(CLIENTAUTH) does not force the client to supply a certificate. Your client's browser program may not support client certificates, or the client may not have installed a certificate. In these cases, the SSL connection can still be established without the client certificate unless you also specify AUTHENTICATE(CERTIFICATE) or AUTHENTICATE(AUTOREGISTER) in the TCPIPSERVICE definition.

## Determing the userid for a transaction

The client certificate can only be used to determine the userid for the CICS transaction if the certificate has been previously associated with a RACF userid, as described in"Associating a RACF userid with a certificate". If such an association exists, CICS obtains the userid from RACF and passes it to the Analyzer program in field **wbra_userid** of the analyzer parameter list. If the Analyzer does not alter this field, this userid is used when the CICS Web Interface attaches the alias transaction. However, if the Analyzer ignores the value input in **wbra_userid**, and chooses to output a different userid, then the client certificate userid is not used. The application program can still discover what userid is associated with the client certificate by executing the EXEC CICS EXTRACT CERTIFICATE USERID command.

## Associating a RACF userid with a certificate

You can associate a certificate with a RACF userid in two ways: either by having your clients register their certificates online through their browser program, or by

using the RACDCERT command under TSO. You enable clients to register their certificates themselves by specifying AUTHENTICATE(AUTOREGISTER) on the TCPIPSERVICE definition. Users connecting to CICS through such a TCPIPSERVICE must have a client certificate. If that certificate is already registered to a userid, that userid is used, otherwise the client is prompted for a userid and password with HTTP basic authentication. If the client then enters a valid userid and password, that userid is registered to the certificate, and the client will not be prompted for a password again. If you do not wish to allow your clients to register their own certificates, you must register them with RACDCERT command. (This command does not execute under the OpenEdition shell.) Before executing RACDCERT, you must download the certificate that you wish to process into an MVS sequential file with RECFM=VB that is accessible from TSO. The syntax of RACDCERT is:

```
RACDCERT ADD('datasetname') TRUST [ ID(userid) ]
```

where *datasetname* is the name of the dataset containing the client certificate, and *userid* is the userid that is to be associated with the certificate. If the optional ID(userid) parameter is omitted, the certificate is associated with the user issuing the RACDCERT command.

You can add certificate information for your own userid if you have READ access to the 'IRR.DIGTCERT.ADD' profile in the FACILITY class. You can add certificate information for other userids if you have UPDATE access to the 'IRR.DIGTCERT.ADD' profile in the FACILITY class. If you have RACF SPECIAL authority, you can execute RACDCERT ADD for any userid.

For further information on the RACDCERT command, including the format of data allowed in the downloaded certificate dataset, see the *OS/390 Security Server (RACF) Command Language Reference* APAR PQ 36169 MJO 16/05/00

## Marking a certificate untrusted

If a certificate has been registered in the RACF database but you do not want it to be used by clients, you can mark it as UNTRUSTED using the RACDCERT command. To do this, first issue:

```
RACDCERT ID(userid) LIST
```

to find the label associated with the certificate you wish to change, and then issue:

```
RACDCERT ID (userid) ALTER(LABEL(label)) NOTRUST
```

to mark the certificate as untrusted. Clients are then prevented from establishing CLIENTAUTH connections with this certificate.

However, the NOTRUST option is not fully supported by CICS when using OS/390 Version 2 Release 7, so if you want to make a client certificate unusable on that release you should use RACDCERT to delete the current registration, and then register the certificate to a 'revoked' userid. That is, use

```
RACDCERT ID(userid) LIST
```

to find the label associated with the certificate you wish to disable, then issue:

```
RACDCERT ID (userid) DELETE(LABEL(label)) NOTRUST
```

to delete the registration with 'userid', and, finally, issue

```
RACDCERT ID (revoked-userid) ADD('datasetname') TRUST
```

to register the certificate to a revoked userid, where 'datasetname' is the name of a dataset containing a copy of the client certificate that you wish to disable, and 'revoked-userid' is a userid that is permanently revoked. Be sure that you specify TRUST, otherwise CICS will not recognize the registration to the revoked userid.

# Part 3. Intercommunication security

This part discusses how to plan and implement security in an intersystem communication (ISC) environment, using LU6.2 or LU6.1, or in a multiregion operation (MRO) environment. This part contains the following chapters:

- "Chapter 12. Overview of intercommunication security" on page 147, which introduces the concepts of bind-time, link, user, transaction, and resource security in an intercommunication environment
- "Chapter 13. Implementing LU6.2 security" on page 153, covering bind-time, link, user, transaction, resource, and command security; plus transaction routing, and function shipping
- "Chapter 14. APPC password expiration management" on page 173, which contains information on evaluating and using APPC password expiration management
- "Chapter 15. Implementing LU6.1 security" on page 193, covering link, transaction, resource, and command security; plus function shipping
- "Chapter 16. Implementing MRO security" on page 199, covering bind-time, link, user, transaction, resource, and command security; plus transaction routing and function shipping
- "Chapter 17. Security for data tables" on page 213, covering provision made for security of CICS shared data tables; plus logon security checks, and connection security checking for bind security and file security.

# Chapter 12. Overview of intercommunication security

This chapter gives an overview of how security works when CICS systems are interconnected or connected to other compatible systems.

It is organized under the following main topics:

- Introduction
- "Planning for intercommunication security"
- "Summary of intercommunication security levels" on page 149
- "Implementing intercommunication security" on page 150.

## Introduction

In a single CICS system, you use security to make sure that terminal users can access only those parts of the system they need to work with. For interconnected systems, the same basic principles apply, but now you also include definitions for connections, sessions, and partners. You also need to allow for the fact that users of one CICS system can initiate transactions and access resources in another CICS system.

This chapter assumes that you are already familiar with setting up security for a single CICS system, as described in "Part 1. Introduction" on page 1 and "Part 2. Implementing RACF protection for a single-region CICS" on page 35.

In particular, you should understand the following concepts:

- User signon. (See "Sign-on process" on page 65.)
- How the relationship between user security and transaction security determines which transactions a particular user is allowed to invoke. (See "Chapter 4. Verifying CICS users" on page 65 and "Chapter 5. Transaction security" on page 79.)
- How resource security determines which other resources a user is allowed to access. (See "Chapter 6. Resource security" on page 85.)

An interconnected group of CICS systems differs from a single CICS system in that you may have to define a user profile or group profile more than once. (See "RACF user profiles" on page 11 and "RACF group profiles" on page 17 for information on defining these profiles.) That is, you may have to define these profiles in each CICS system that is using a separate RACF database, and in which a user is likely to want to attach a transaction or access a resource. When planning these profiles, you must consider all cases in which a user could initiate function shipping, transaction routing, asynchronous processing, distributed program link, distributed transaction processing, or external call interface (EXCI). (For descriptions of these methods of intercommunication, see the *CICS Intercommunication Guide* and the *CICS Distributed Transaction Programming Guide*.)

## Planning for intercommunication security

Intercommunication security in a CICS system is concerned with incoming requests for access to CICS resources, rather than with requests that are sent to other systems.

The security problem with incoming requests occurs when a particular user at a particular remote system is trying to access resources of your CICS system. Is this access authorized, or should it be rejected?

The following sections describe the points in the processing of an incoming request at which you can apply security checks.

## Bind-time security

The first requirement is for a session to be established between the two systems. This does not, of course, happen on every request; a session, once established, is usually long-lived. Also, the connection request that establishes the session can, depending on the circumstances, be issued either by the remote system or by your CICS system. However, the establishment of a session presents the first potential security exposure for your system.

Your security concern is to prevent unauthorized remote systems from being connected to your CICS system; that is, to ensure that the remote system is really the system that it claims to be. This level of security is called **bind-time security** (also known as **systems network architecture (SNA) session security**). It can be applied when a request to establish a session is received from, or sent to, a remote system.

**Note:** We use the term **bind** to refer both to the **SNA BIND** command that is used to establish SNA sessions between systems, and to the **CICS connection request** that is used to establish MRO sessions for CICS interregion communication.

You can specify bind-time security for LU6.2 and multiregion operation (MRO) links, but **not** for LU6.1 links. For information on defining bind-time security in your system, see either "Bind-time security with LU6.2" on page 153 or "Bind-time security with MRO" on page 199, depending on the environment you are using.

## Link security

Each link between systems is given an authority defined by a userid.

It is important to note that users cannot access any transactions or resources over a link that is itself unauthorized to access them. This means that each user's authorization is a subset of the link's authority as a whole.

To limit the remote system's access to your transactions and resources, you use **link security**. Link security is concerned with the single user profile that you assign to the remote system as a whole. Like user security in a single-system environment, link security governs:

- **Transaction security**. This controls the link's authority to attach specific transactions.
- **Resource security**. This controls the link's authority to access specific resources. This applies for transactions, executing on any of the sessions from the remote system, that have RESSEC(YES) specified in their transaction definition.
- **Command security**. This controls the link's authority for the commands that the attached transaction issues. This applies for transactions, executing on any of the sessions from the remote system, that have CMDSEC(YES) specified in their transaction definition.
- **Surrogate user security**. This controls the link's authority to START transactions with a new userid, and to install resources with an associated userid.

For more information, see "Transaction, resource, command, and surrogate user security".

### Link security with MRO

See the section "Link security with MRO" on page 202, in "Chapter 16. Implementing MRO security" on page 199.

### Link security with LU6.2

See the section "Link security with LU6.2" on page 157, in "Chapter 13. Implementing LU6.2 security" on page 153.

### Link security for LU6.1

See the section "Link security with LU6.1" on page 193, in "Chapter 15. Implementing LU6.1 security" on page 193.

## User security

In addition to the security profile that you set up for the link, you may want to further restrict each remote user's access to the transactions, commands, and resources in your system. This is done by specifying the appropriate ATTACHSEC parameters in the CONNECTION definition. This **user security**, like link security, distinguishes between transaction, resource, command, and surrogate security. User security can never **increase** a user's authority above that of the link. For more information, see "Transaction, resource, command, and surrogate user security".

For information on defining user security in your system, see either "User security with LU6.2" on page 158 or "User security with MRO" on page 203, depending on the environment you are using.

**You cannot specify user security for LU6.1 links. For LU6.1, the user security is taken to be the same as the link security**.

## Transaction, resource, command, and surrogate user security

The last step in defining security for your system is to make sure that the access parameters match the profiles you have defined for your transactions, resources, commands, and surrogate users for the link and the individual remote users. For information on defining these levels of security in a single-system environment, see "Chapter 5. Transaction security" on page 79, "Chapter 6. Resource security" on page 85, and "Chapter 8. CICS command security" on page 109.

**Resources and commands are unsecured unless you explicitly request security protection in your transaction definitions**.

For information on defining transaction and resource security in your system, see one of the following, depending on the environment you are using:

- "Transaction, resource, and command security with LU6.2" on page 164
- "Transaction, resource, and command security with LU6.1" on page 194
- "Transaction, resource, and command security with MRO" on page 206

# Summary of intercommunication security levels

Figure 6 on page 151 shows bind-time, transaction, resource, and command security, and how CICS enforces these levels of security under the LU6.2, MRO, and LU6.1 protocols. It also shows how the two levels of authorization (user and link) are involved at the three security levels.

For guidance on choosing between these environments, see the *CICS Intercommunication Guide*.

**Note:** Remember to define profiles for your resources and users to RACF, as described for single systems in "Chapter 2. RACF facilities" on page 9.

## Implementing intercommunication security

Security in the intercommunication environment is implemented through resource definition and RACF profiles. The following chapters tell you how to define your intersystem links, according to the environment you are using:

- "Chapter 13. Implementing LU6.2 security" on page 153
- "Chapter 15. Implementing LU6.1 security" on page 193
- "Chapter 16. Implementing MRO security" on page 199

Figure 6 on page 151 shows a summary of intercommunication security.

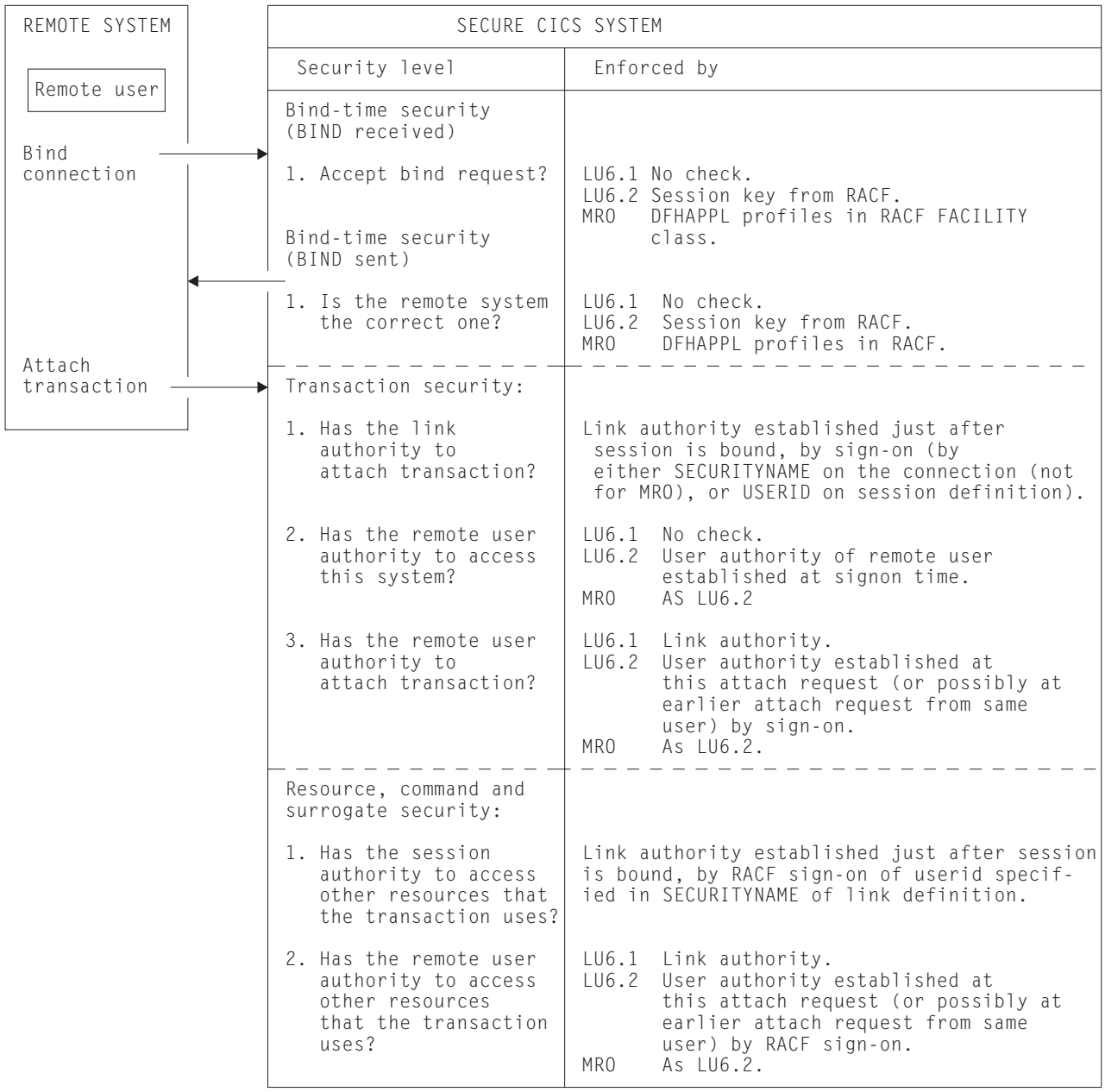



*Figure 6. Summary of intersystem and interregion security*

# Chapter 13. Implementing LU6.2 security

This chapter tells you how to implement security for LU6.2. It is organized under the following topics:



## Bind-time security with LU6.2

A security check can be applied when a request to establish an APPC session is received from, or sent to, a remote system; that is, when the session is bound. This is called **bind-time security** (or, in SNA terms, **session security**), and is part of the CICS implementation of the LU6.2 architecture. Its purpose is to prevent an unauthorized system from binding a session to one of your CICS systems.

Bind-time security is optional in the LU6.2 architecture. Of course, do not specify bind-time security if the remote system does not support it. SNA defines how session security is to be applied, and CICS conforms to this architecture. If you want to connect to a CICS system other than CICS Transaction Server for OS/390, CICS/ESA, or CICS/MVS, make sure the other system is also compatible with this architecture.

When you define an LU6.2 connection to a remote system, you assume that all inbound bind requests originate in that remote system, and that all outbound bind requests are routed to the same system. However, where there is a possibility that a transmission line might be switched or broken into, guard against unauthorized session binds by specifying session security at both ends of the connection.

For a bind request to succeed, both ends must hold the same **session key**, which is defined to RACF. When a session is bound, the action CICS takes depends on:

- How you specified the SEC and XAPPC parameters in your system initialization table (SIT)
- How you specified the BINDSECURITY option on the CONNECTION resource definition in the CSD
- Whether you have defined an APPCLU security profile for the link.

If you have SEC=YES and XAPPC=YES in your SIT, and BINDSECURITY(YES) in your CSD connection definition, and BINDSECURITY(YES) is also specified for the partner system, a bind security validation will be attempted.

If you have BINDSECURITY(NO), then the SIT specification is immaterial.

Table 20 summarizes what happens.

*Table 20. APPC bind-time security—relationship to resource definition*

| SEC value | XAPPC value | BINDSECURITY value | RACF APPCLU profile | Resulting CICS action |
|---|---|---|---|---|
| YES | YES | YES | Defined (See note 1) | CICS extracts the APPCLU profile from RACF at bind-time to verify the remote system. |
| YES | YES | YES | Not defined | CICS is unable to extract the APPCLU profile from RACF and therefore rejects the bind. |
| Any value | Any value | NO | Any value | CICS does not perform any bind validation. Bind not rejected for security reasons. |
| YES | NO | YES | Any value | CICS is unable to validate the bind, which is rejected. |
| NO | Any value | YES | Any value | CICS is unable to validate the bind, which is rejected. |
| NO | Any value | Any value | Any value | CICS is unable to validate the bind, and rejects it. |

**Notes:**

1. If the RACF APPCLU profile is defined, but the session segment is locked or expired, and no value is specified for SESSKEY, the bind request is always rejected.
2. The table shows the response when the partner has specified BINDSECURITY(YES).

## Example of defining an APPCLU profile

You can define an APPCLU profile as follows:

```
RDEFINE  APPCLU  netid.luid1.luid2 UACC(NONE)
         SESSION(SESSKEY(session-key)) AUDIT(ALL(READ))
         NOTIFY(CICS RACF Administrator)
```

In this example:

**netid** is the network ID, as specified on the NETID parameter in the VTAM startup member (ATCSTRxx) of SYS1.VTAMLST.

**luid1** is the APPLID of the system on which the CONNECTION definition is installed.

**luid2** is the NETNAME, as specified in the CONNECTION definition.

**SESSKEY**
is the 16-hexadecimal-digit or 8-character password that matches the session key of the remote system. Enclose hexadecimal digits in quotes; for example, `SESSKEY(X'0123456789ABCDEF')`.

The AUDIT and NOTIFY keywords are discussed in "Auditing bind-time security" on page 155.

You can also use the following options on the SESSION keyword if required:

- INTERVAL, which you can use to specify the number of days for which the SESSKEY is to remain valid
- LOCK, which you can use to stop the link being acquired by new sessions.

There are other parameters on the APPCLU profile, but they are not used by CICS.

## Defining bind-time security

You define bind-time security in the CONNECTION definition, although you must also choose the appropriate system initialization parameters. Figure 7 shows how to define APPC external session security, for which you need to specify the BINDSECURITY option.

```
CEDA DEFINE CONNECTION(name)
  GROUP(groupname)
  ACCESSMETHOD(VTAM)
  SECURITYNAME(name)
  PROTOCOL(APPC)
  NETNAME(name)
  BINDSECURITY(YES)
```

*Figure 7. Bind-time security*

**Note:** For APPC terminals defined as a TERMINAL-TYPETERM pair, the BINDSECURITY operand is on the TERMINAL definition.

## Auditing bind-time security

If security is active (SEC=YES is specified in the system initialization parameters), CICS performs bind security auditing. The following conditions are considered bind failures, and cause RACF to write an SMF record, and to issue a message:

- Session key does not match partner's.
- Session segment is locked.
- Session segment has expired.
- Session key is null.
- Session segment does not exist.
- Session segment retrieval was unsuccessful.
- Session bind was unsuccessful.

The following conditions are considered bind successes, and cause RACF to write an SMF record, but **not** to issue a message:

- Session was successfully bound.
- Session key will expire in less than six days.

An SMF record is written if either of the following is true:

- The profile's audit option is set (AUDIT(ALL(READ))).
- SETROPTS LOGOPTIONS(ALWAYS(APPCLU)) is set.

Two things happen when an SMF audit record is written:

- Message ICH700051 is sent to the userid specified in the profile's notify option. It is suggested that you specify the TSO userid of a RACF administrator who is responsible for the APPCLU class.
- The security console (any MVS console with a routing code of 9) receives message ICH415I, which contains text similar to message ICH70005I.

These audit records can be extracted from SMF and listed using the following sample RACF Report Writer control statements:

```
//RACFRW EXEC PGM=IKJEFT01
//SORTWKxx DD your sort files
//SYSPRINT DD SYSOUT=*
//SYSTPRT DD SYSOUT=*
//RSMFIN DD DSN=your smf dumped data, DISP=SHR
//SYSTIN DD *

RACFRW TITLE('Bind Security Reports') GENSUM
SELECT PROCESS
EVENT APPCLU
LIST SORT(DATE,TIME)
END

//
```

The RACF Report Writer is described in the *OS/390 Security Server (RACF) Auditor's Guide*.

## Changing RACF profiles that are in use—caution

Take care when changing RACF profiles for APPC connections that are in use. CICS recognizes the change in the profile after a SETROPTS RACLIST(APPCLU) REFRESH command is issued. Bind-time security processing occurs when each session in a connection is acquired. Not all the sessions in a connection are acquired and the APPC profile becomes invalid, then an attempt to establish any of the unacquired sessions causes a bind security failure. This can cause transactions that attempt to allocate one of these unused sessions to be suspended indefinitely.

### Reasons for invalid profiles

An APPC profile can become invalid for a number of reasons; for example:

- The session key expires
- The session key changes and a SETROPS REFRESH takes place in one system without the corresponding change and refresh occurring in the other system
- The profile is locked while REFRESH takes place.

Sessions that are already acquired still continue to function normally if bind security fails in another session. If you are using expiring session keys, then the connection can still be used after the expiry date, if any of the sessions on the connection were acquired before the date of expiration, and have remained acquired. Hence, you see the effect of an expiring session key only when the connection (or session) is acquired.

**Note:** No warning messages are produced stating that the session key is about to expire. However, an SMF record can be written when a key is used that will expire shortly. Therefore, you can use the RACF Report Writer regularly to find out which keys need maintenance. Otherwise, if expiring session keys are used, you must remember when the keys are due to expire. You must also take appropriate action to minimize any disruption that may occur because the connection is unavailable because of an expired session key. For example, you should plan for the changing of the session keys, for security rebuilds (for both CICS systems) and for the possibility of having to reacquire the connection.

You can avoid the problem of APPC profiles becoming invalid while the connection is in use by specifying AUTOCONNECT(YES) or

AUTOCONNECT(ALL) on the SESSIONS definition. This causes all sessions to be established (acquired) when the connection is acquired.

### Removal of internal LU6.2 bind time security

The BINDPASSWORD in a CSD CONNECTION definition is not used for LU6.2 bind time security validation. Instead, you should create RACF APPCLU profiles, and specify XAPPC=YES on the SIT to maintain validated links.

## Link security with LU6.2

Link security further restricts the resources a user can access, depending on the remote system from which they are accessed. The practical effect of link security is to prevent a remote user from attaching a transaction or accessing a resource for which the link userid has no authority.

Link security can be associated with a connection or with a session, depending on whether you want to control the link security for each group of sessions separately:

- To define link security for a connection as a whole, specify the SECURITYNAME parameter in the CONNECTION definition.
- To define link security for individual groups of sessions within a connection, specify the USERID in the SESSIONS definition as a user identifier.

Each link between systems is given an authority defined by a link userid. A link userid for LU6.2 is a userid defined on your session's definition for this connection. If not defined, the link userid is the SECURITYNAME userid specified on the connection definition. If there is no SECURITYNAME, the link userid is the default userid.

You can never transaction route or function ship to CICS without having at least one security check, but the security checks are minimized if the two regions involved are ***equivalent systems***. This term means the same thing for LU6.1, LU6.2, and MRO. If the link userid matches the local region userid, you have equivalent systems.

If you do have equivalent systems, only one security check is made. This will either be against the default user (for ATTACHSEC=LOCAL) or against the userid that is in the received FMH-5 attach request (ATTACHSEC=non-LOCAL).

If you do not have equivalent systems for ATTACHSEC=LOCAL, resource checks are done only against the link userid. For ATTACHSEC=non-LOCAL there are always two resource checks. One is against the link userid, and the second is against the userid received from the remote user in the attach request.

If a failure occurs in establishing link security, the link is given the security of the local region's default user. This would happen, for example, if the preset session userid had been revoked.

If a value is present on the USERID parameter of the SESSIONS definition, the value overrides any value specified on the SECURITYNAME parameter.

## User security with LU6.2

User security causes a second check to be made against a user signed onto a terminal, in addition to the link security described in "Link security with LU6.2" on page 157. After reading the following descriptions, consider whether you want the extra level of security checking that user security provides.

You can specify the following levels of user security using the ATTACHSEC parameter of the CONNECTION definition:

- *LOCAL*, which you specify if you do **not** want to make a further check on users by requiring a user identifier or password to be sent. Choose LOCAL if you do not want user security because you consider that the authority of the link is sufficient for your system. See "Specifying user security in link definitions" on page 159 for information on doing so.
- *Non-LOCAL*, which you specify if you *do* want to make a further check on users by requiring a user identifier, or a user identifier and a password, to be sent. Non-LOCAL includes the following types of checking:
  - IDENTIFY

    A user identifier must be sent, but no password is requested
  - VERIFY

    A password must also always be sent
  - PERSISTENT VERIFICATION

    Password is sent on the first attach request for a user
  - MIXIDPE

    Either identify or persistent verification

**Note:** "Non-LOCAL user security verification" further describes these types of user checking checking. See "Specifying user security in link definitions" on page 159 for information on specifying them.

### Non-LOCAL user security verification

In a CICS-to-CICS system connection, where you have a terminal-owning region (TOR), an application-owning region (AOR), and a data-owning region (DOR), the terminal operator signs on to the TOR, attaches a transaction in the AOR, and accesses resources in the DOR. If all three systems implement non-local user security, then the same operator is registered as a user in each of them. The usual procedure is for the operator to sign on to the TOR with a password. CICS assumes that the password is valid for the entire systems complex, and that it does not need to be passed on to the AOR and the DOR for further verification. All that is needed is for the AOR and the DOR to IDENTIFY the user, who is then signed on without a password. Therefore, the password is not sent with the attach request to the AOR. This is considered to be more secure, because the password is not passed on a network.

Specify IDENTIFY when you know that CICS can trust the remote system to verify its users (by some sort of sign-on mechanism) before letting them use the link. Use IDENTIFY if you want user security for CICS-to-CICS communication (CICS does not support password flows on CICS-to-CICS connection) which includes the following:

- CICS/MVS
- CICS Transaction Server for OS/390
- CICS for MVS/ESA™
- AIX®

This excludes CICS Transaction Server for OS/2.

If the front end does not have a security manager—for example, if it is a programmable workstation (PWS)—it is often not possible to VERIFY the user by means of a user identifier and password before the attach request reaches the AOR. The AOR must then receive both user identifier and password from the front end so that it can verify the user itself by a sign-on with password.

Specify VERIFY if you have reasons for wanting your own system to verify the remote system's users even if they have already been checked by the remote system itself, or if the remote system does not have a security manager and therefore cannot verify its own users. VERIFY must be used if the request comes from CICS for OS/2, which does not support PERSISTENT.

If programmable workstations make repeated requests to attach transactions in the AOR, performance suffers because of many verifications. The LU6.2 architecture, which defines these security procedures, allows persistent verification to reduce the software overhead. Using this protocol, the first attach request contains a user identifier and a password, but once the user has signed on, only the user identifier is needed for all the attach requests that follow.

Specify PERSISTENT to reduce the verification overhead if remote users repeatedly send attach requests. However, the remote system must be able to cooperate in the management of persistent verification by keeping a list of users who are currently signed on.

Some remote APPC systems have mixed sign-on requirements that vary from conversation to conversation (for example, CPI communications conversations). In this case, CICS must accept incoming identify or persistent requests.

To decide which of these types of user verification to use, you need to know how far the remote system is capable of managing its own security and, if it cannot, to what extent it must depend on the CICS system you are defining.

- Do you need to know the user identifier? If not, use LOCAL.
- Can the remote system verify its own users? If so, use IDENTIFY. If not, can it send a user identifier and a password with the attach request? If so, use VERIFY for PWS-to-CICS communication.
- Does the remote system support persistent verification by keeping track of its user identifiers and passwords? If you are using PWS-to-CICS communication, you may want to specify PERSISTENT, or MIXIDPE if you are using both CICS-to-CICS and PWS-to-CICS.

You specify these levels of checking for each connection using the ATTACHSEC operand of the CONNECTION definition, as described in "Specifying user security in link definitions".

## Specifying user security in link definitions

The level of user security you require for a remote system is specified in the ATTACHSEC operand in the CONNECTION definition, as shown in Figure 8 on page 160.

This topic describes how CICS interprets the parameters of the ATTACHSEC operand of the CONNECTION definition. However, special rules apply for CICS transaction routing, as described in "Transaction routing security with LU6.2" on page 166

page 166. Figure 8 shows an example of defining ATTACHSEC using CEDA.

```
CEDA DEFINE CONNECTION(name)
  GROUP(groupname)

  ATTACHSEC(LOCAL|IDENTIFY|VERIFY|PERSISTENT|MIXIDPE)
```

*Figure 8. Defining sign-on level for user security*

**Note:** For APPC terminals defined as a TERMINAL-TYPETERM pair, the ATTACHSEC operand is on the TERMINAL definition.

The ATTACHSEC operand specifies the sign-on requirements for incoming transaction attach requests. It has no effect on attach requests that are issued by your system to a remote system; these are dealt with in the remote system.

When an APPC session is bound, each side tells the other the level of attach security user verification that will be performed on its incoming requests. There is no negotiation on this.

## Meanings of ATTACHSEC operand

The following are the possible operands of ATTACHSEC:

**LOCAL**
specifies that a user identifier is not to be supplied by the remote system. If one is received, the attach fails. CICS makes the user security profile equivalent to the link security profile. You do not need to specify RACF profiles for the remote users. LOCAL is the default value.

**IDENTIFY**
specifies that a user identifier is expected on every attach request. All remote users of a system must be identified to RACF.

If an attach request with both a user identifier and a password is received on a link with ATTACHSEC(IDENTIFY), CICS does not reject the attach request. CICS handles the attach request as if the connection was defined with ATTACHSEC(VERIFY).

If a null (X'00') user identifier or an unknown user identifier is received, CICS rejects the attach request.

If the connection is to a release of CICS/ESA earlier than version 3.2.1, see "Attach-time security and the USEDFLTUSER option" on page 238.

**VERIFY**
specifies that, in addition to a user identifier, a user password is required for verification against the local RACF database. All remote users of a system must be identified to RACF.

The rules that apply to the checking of the user identifier for ATTACHSEC(IDENTIFY) also apply for ATTACHSEC(VERIFY). If a valid user identifier is received but the password verification fails, CICS rejects the attach request.

All CICS systems except CICS OS/2 and CICS for Windows NT can verify the security attributes of their users with an external security manager. CICS OS/2 does not have an external security manager and so is regarded as an insecure system. CICS OS/2 only supports (LOCAL(VERIFY). If CICS for OS/2 is the terminal-owning region (TOR) connected to CICS Transaction Server for OS/390, use the ATTACHSEC=VERIFY option in the LU6.2 connection definition on the CICS Transaction Server for OS/390 application owning region (AOR). The appropriate adjustments should also be made to the Communications Manager on the CICS OS/2 system so that the password and userid of the user signing on to CICS OS/2 are sent. (See the *CICS OS/2 Intercommunication Guide*, SC33-0826, for details of the Communications Manager changes that need to be made.) CICS Transaction Server for OS/390 is then able to VERIFY the user by performing a signon with password. If the communicating system is CICS for AIX, ATTACHSEC=IDENTIFY should be used.

**Note:** Products other than CICS can connect to a CICS Transaction Server for OS/390 AOR via an LU6.2 link. They then use the SNA LU6.2 FMH-5 ATTACH mechanism to start a transaction on the CICS AOR. Where this mechanism is being used from an insecure system, the ATTACHSEC=VERIFY option should be used on the connection definition to protect the transaction on the AOR. (See "SNA profiles and attach-time security" on page 163. For more information about ATTACHSEC and USEDFLTUSER, see "Attach-time security and the USEDFLTUSER option" on page 238.

**PERSISTENT**
specifies that a user identifier and a user password are required with the first attach request for a new user, but all following attach requests for the same user need supply only a user identifier. (All remote users of a system must be identified to RACF.) The first attach signs on the user, even if the attach request is later unsuccessful because the user is not authorized to attach the transaction.

**Note:** PERSISTENT cannot be used for CICS-to-CICS communication.

**MIXIDPE**
specifies that the sign-on level for the remote user is determined by parameters sent with the attach request. The possibilities are: PERSISTENT or IDENTIFY.

## Sign-on status

With the ATTACHSEC parameters IDENTIFY, MIXIDPE, PERSISTENT, and VERIFY, the remote user remains signed on after the conversation associated with the first attach request is complete. CICS then accepts attach requests from the same user without a new sign-on until either of the following occurs:

- The period specified in the system initialization parameter USRDELAY elapses after completion of the last transaction associated with the attach request for this user.

  When you are running remote transactions over ISC and IRC links USRDELAY defines the time for which entries can remain signed onto the remote CICS region. For information on specifying USRDELAY, see the *CICS System Definition Guide*. See the *CICS Performance Guide* for information on tuning.
- The CICS system is terminated.

If you alter the RACF profile of a signed-on remote user (for example, by revoking the user), CICS continues to use the authorization established at the first attach request until the entry is removed from the sign-on list.

### Password checking

If you are using ATTACHSEC(PERSISTENT) (or ATTACHSEC(MIXIDPE) being treated as ATTACHSEC(PERSISTENT)), CICS maintains a table for each remote system called the **persistent verification (PV) signed-on-from list**. This is a list of users whose passwords have been checked and who do not require a further password check as long as the entry remains in the list. Entries remain in the list until:

- The period specified in the system initialization parameter PVDELAY elapses after the user's sign-on entry was last used.

  PVDELAY defines how long entries can remain in the PV signed-on-from list for the remote system, which means that their passwords do not need to be revalidated for each attach request. For information on specifying a value for PVDELAY, see the *CICS System Definition Guide*. See the *CICS Performance Guide* for information on tuning.
- The connection with this system is terminated because: CICS is restarted, the connection is lost, or CICS receives an invalid attach request from the user.

When persistent verification is in operation for a remote user, and that user is removed from the PV signed-on-from list, CICS informs the remote system by issuing a sign-off request for the user to remove the entry from the PV signed-on-to list in the remote system.

If you specify ATTACHSEC(VERIFY), the remote user's password is checked for *every attach request*. This is to ensure that the user has authority to access this system, to verify that this password is correct, and to establish security authorities for the user.

## Information about remote users

Information about the user can be transmitted with the attach request from the remote system. This means that you can protect your resources not only on the basis of which remote system is making the request, but also on the basis of which user at the remote system is making the request.

This topic describes some of the concepts associated with remote-user security, and how CICS sends and receives user information.

You have to define your users to RACF. If a remote user is not defined to RACF, any attach requests from that remote user are rejected.

CICS remote-user security for LU6.2 links implements the LU6.2 architecture. The LU6.2 architecture allows user identifiers, user passwords, and user profiles to be transmitted with requests to attach a transaction.

User profiles can be transmitted instead of, or in addition to, user identifiers. The profile name, if supplied, is treated as the groupid.

If the user has been added to the front-end system with a group ID explicitly specified; for example in EXEC CICS SIGNON or by filling in the GROUPID parameter on the CESN panel, this will be propagated by CICS in outbound attach FMHs for LU6.2 links where ATTACHSEC(IDENTIFY) has been specified in the CONNECTION definition. If the group ID has been allowed to default at the time

the user was originally added to the front-end system, the profile field will not be included in the outbound FMH5. If the group ID is passed to the back-end system, the group ID will be used as part of ADD_USER processing on the back-end. That is, the user ID must be defined as a member of the group passed in the ESM on the back-end for the ADD_USER to be successful.

It is advisable to use the PLTPIUSR system initialization parameter if there is a possibility that a task started by PLTPI processing will access remote resources. This avoids problems in the remote region where the user ID is not in the same group as the user in the local region. This is because the PLTPI user in the local region is not added with an explicit groupid, and as a result the groupid is not propagated to the remote region.

CICS sends userids on ATTACHSEC(IDENTIFY) conversations. Table 21 shows how CICS decides which userid to send.

*Table 21. Attach-time user identifiers—LU6.2*

| **Characteristics of the local task** | **User identifier sent by CICS to the remote system** |
|---|---|
| Task with associated terminal—user identifier | Terminal user identifier |
| Task with associated terminal—no user signed on and no USERID specified in the terminal definition | Default user identifier for the local system |
| Task with no associated terminal or USERID started by interval control START command (if using function shipping or distributed transaction processing (DTP)) | User identifier for the task that issued the START command |
| Task started with USERID option | User identifier specified on the START command |
| CICS internal system task | CICS region userid |
| Task with no associated terminal started by transient data trigger | User identifier specified on the DCT that defines the queue |
| Task with associated terminal started by transient data trigger | Terminal user identifier |
| Task started from PLTPI | PLTPIUSR |

Signing on the remote user has two purposes:

- To ensure that the remote user is allowed to access the CICS system
- If the sign-on is successful, to establish the authority for the remote user

CICS signs off the remote user under the circumstances described in "Sign-on status" on page 161.

## SNA profiles and attach-time security

Implementation of the LU6.2 attach-time security in CICSTS13.CICS. conforms strictly to the architecture. In particular, note the following:

- The introduction of SNA profile support and the conformance to SNA attach-time security processing may cause migration problems.
- Profile support means that badly coded profiles sent in an attach FMH-5 cause certain attach requests to be rejected.

- The checks to prevent problems in the access security subfields of an FMH-5 are:
  - Check for unrecognized subfield
  - Check for invalid length subfield
  - Check for multiple subfields of the same type
- The full 10-character userid and password are accepted. Any trailing blanks ((X'40') are removed before being passed to the security manager, which either rejects the attach request, or converts the userid and password into 8- character form before proceeding.
- Attach requests are rejected if they do not contain security parameters in an FMH-5 unless the USEDFLTUSER parameter has been coded on the connection.
- Attach requests are rejected if they have a blank, or zero-length, user ID parameter in the attach FMH-5. See "Attach-time security and the USEDFLTUSER option" for a description of the exception where zero-length user IDs can be accepted for ATTACHSEC(VERIFY) and ATTACHSEC(IDENTIFY).
- Valid SNA profiles received are treated as the ESM groupid with which the userid in the FMH-5 will be associated after the userid in the FMH-5 is signed on.
- When a SNA profile is received and the connection had ATTACHSEC=PERSISTENT, it is validated to conform to the architecture. It is not used to further qualify users in the signed-on-from list. This also applies to persistent signed-on flows received on a connection that has ATTACHSEC=MIXIDPE specified.

## Attach-time security and the USEDFLTUSER option

In releases earlier than CICS/ESA 4.1, a user who was not signed on would not have an associated userid. In CICS Transaction Server for OS/390, coding USEDFLTUSER on the connection indicates that the default user can be used. The following types of incoming attach FMH-5 are accepted by CICS Transaction Server for OS/390 only if the USEDFLTUSER option is coded on the connection:

- An FMH-5 with an ATTACHSEC of IDENTITY not containing a userid subfield, for example, from a CICS for VSE/ESA™ system.
- An FMH-5 with an ATTACHSEC of VERIFY containing userid and password subfields which have zero-length; for example, from certain non-EBCDIC based systems.
- An FMH-5 with an ATTACHSEC of VERIFY containing an access security information field (ASIF) length field of zero.
- An FMH-5 received on a connection defined with ATTACHSEC(IDENTIFY) containing a user ID ASIS which specifies a zero-length user ID.

If the user does not specify the USEDFLTUSER option in these exceptions, the expected protocol violation occurs, a message is generated, and the attach fails.

## Transaction, resource, and command security with LU6.2

As in a single-system environment, users must be authorized to:

- Attach a transaction (**transaction security**)
- Access all the resources that the transaction is programmed to use. These levels are called **resource security**, **surrogate user security**, and **command security**

## Transaction security

As in a single-system environment, the security requirements of a transaction are specified when the transaction is defined, as described in "Chapter 5. Transaction security" on page 79.

In an LU6.2 environment, two basic security requirements must be met before a transaction can be initiated:

- The link must have sufficient authority to initiate the transaction.
- If anything other than ATTACHSEC(LOCAL) has been specified, user security is in force. The user who is making the request must therefore have sufficient authority to access the system and to initiate the transaction.

**Note:** Transaction security also applies to the mirror transactions. See "Function shipping security with LU6.2" on page 167.

## Resource and command security

Resource and command security in an intercommunication environment are handled in much the same way as in a single-system environment.

Resource and command security checking are performed only if the installed TRANSACTION definition specifies that they are required; for example, on the CEDA DEFINE TRANSACTION command, as shown in Figure 9.

```
CEDA DEFINE TRANSACTION
  .
  RESSEC(YES)
  CMDSEC(YES)
  .
```

*Figure 9. Specifying resource and command security for transactions*

If a TRANSACTION definition specifies resource security checking, using RESSEC(YES), both the link and the user must also have sufficient authority for the resources that the attached transaction accesses.

If a TRANSACTION definition specifies command security checking, using CMDSEC(YES), both the link and the user must also have sufficient authority for the SP commands shown in Table 11 on page 109 that the attached transaction issues.

For further guidance on specifying resource and command security, see "Chapter 6. Resource security" on page 85 and "Chapter 8. CICS command security" on page 109.

### NOTAUTH exceptional condition

If a transaction tries to access a resource, but fails the resource security checks, the NOTAUTH condition occurs.

When the transaction is the CICS mirror transaction, the NOTAUTH condition is returned to the requesting transaction, where it can be handled in the usual way.

## Transaction routing security with LU6.2

In transaction routing, the authority of a user to access a transaction can be tested in both the TOR and the AOR.

In the TOR, a test is made to ensure that the user has authority to access the transaction defined as remote, just as if it were a local transaction. This test determines whether the user is allowed to run the relay program.

The terminal through which the transaction is invoked must be defined on the remote system (or defined as "shippable" in the local system), and the terminal operator needs RACF authority if the remote system is protected. The way in which the terminal on the remote system is defined affects the way in which user security is applied:

- If the definition of the remote terminal does not specify the USERID parameter:
  - For links defined with ATTACHSEC(IDENTIFY), the transaction security and resource security of the user are established when the remote user is signed on. The userid under which the user is signed on, whether explicitly or implicitly (in the DFLTUSER system initialization parameter), has this security capability assigned in the remote system.
  - For links defined with ATTACHSEC(LOCAL), transaction security, command security, and resource security are limited by the authority of the link.

In both cases, tests against the link security are made as described in "Link security with LU6.2" on page 157.

**Note:** During transaction routing, the 3-character operator identifier from the TOR is transferred to the surrogate terminal entry in the AOR. If the surrogate terminal was shipped in, this identifier is not used for security purposes, but it may be referred to in messages.

When transaction routing PSB requests, the following conditions must both be satisfied:

- ATTACHSEC on the connection definition must not be LOCAL (that is, it can be IDENTIFY, PERSISTENT, MIXIDPE, or VERIFY).
- PSBCHK=YES must be specified as a system initialization parameter in the remote system.

## Preset-security terminals and transaction routing

A terminal has preset-security if a value is specified on the USERID parameter of the terminal definition. When considering the security aspects of transaction routing from a preset-security terminal, you must remember that preset-security is an attribute of the terminal rather than of the user who initiated the transaction routing request.

During transaction routing, a surrogate terminal is created in the AOR to represent the terminal at which the transaction routing request was issued. Whether the surrogate terminal has preset security or not depends upon a number of factors:

- If a remote terminal definition exists in the AOR for the terminal at the TOR, and specifies the USERID parameter, the surrogate terminal is preset with this userid. If the USERID parameter is not specified in the remote terminal definition, the surrogate terminal does not have preset security.
- If a remote terminal definition does not exist in the AOR, the preset security characteristics of the surrogate terminal are determined from the terminal

definition shipped from the TOR. If the shipped terminal definition has preset security, the surrogate also has preset security, unless the connection to the AOR is defined with ATTACHSEC=LOCAL, in which case any preset security information shipped to the AOR is ignored.

### CICS routing transaction, CRTE

You can use the CICS routing transaction, CRTE, with LU6.2 to run transactions that reside on a connected remote system, instead of defining these transactions as remote in the local system. CRTE is particularly useful for infrequently used transactions, or for transactions such as CEMT that reside on all systems.

Security checking done in the AOR for transactions executed under CRTE does not depend on what is specified by ATTACHSEC, or on the userid signed on in the TOR. Instead, security checking depends on whether the user signs on while using CRTE:

- If the user does **not** sign on, the surrogate terminal created is associated with the AOR default user. When a transaction is run, the security checks are carried out against this default user. A check is also done against the link userid to see whether the routing application itself has authority to access the resource.
- When a user **does** sign on to the AOR, using the CESN transaction while running CRTE, the surrogate already created then points to the userid of the signed-on user. For transactions attempting to access resources, security checking is done against the signed-on user's userid in the surrogate and the link userid.

For more information on CRTE, see the *CICS Supplied Transactions* manual and the *CICS Intercommunication Guide*.

## Function shipping security with LU6.2

When CICS receives a function-shipped request, the transaction that is invoked is the **mirror transaction**. The CICS-supplied definitions of the mirror transactions all specify resource, but not command, security checking. This means that you are prevented from accessing the remote resources if either the link or your userid profile on the other system does not have the necessary authority.

If the CICS-supplied definitions of the mirror transactions are not what your security strategy needs, you can change them by copying the definitions in group DFHISC into your own group, changing them and then reinstalling them. For more information, see "Category 2 transactions" on page 128.

If you include a remote resource in your resource definitions, you can arrange for security checking to be done locally, just as if the resource were a local one. Also, the system that owns the resource can be made to apply an independent check, if it is able to receive the user identifier. You can therefore choose to apply security restrictions on both sides, on either side, or not at all.

**Note:** Be aware that if you specify the SYSID option on a function-shipped request, security checking is done in the remote system but is **bypassed in the local system**. Figure 10 on page 168 summarizes what happens.

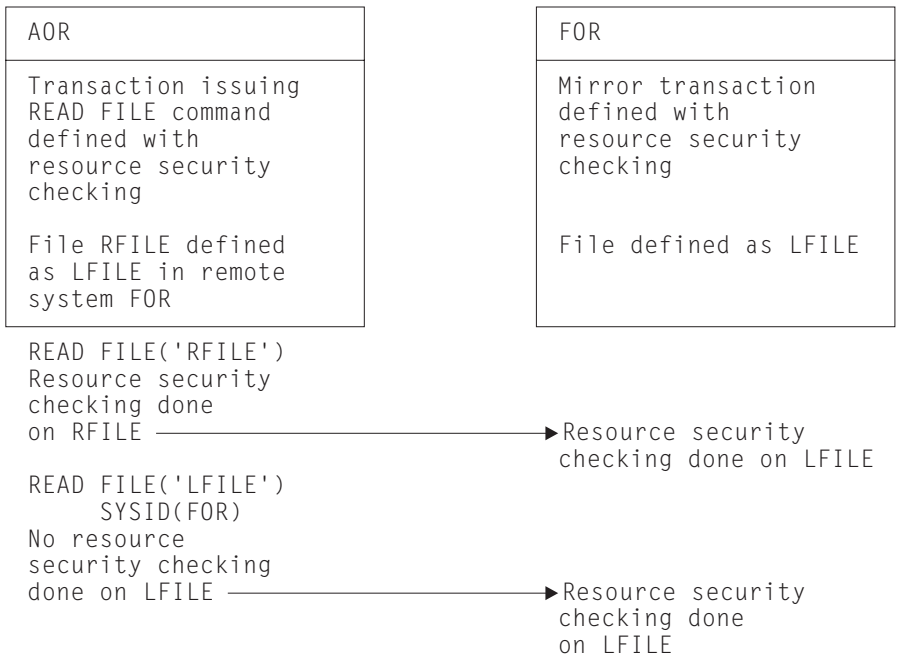



*Figure 10. Security checking done with and without SYSID*

For programming information on specifying the SYSID option, see the *CICS Application Programming Reference* manual.

## Distributed program link security with LU6.2

The CICS distributed program link (DPL) facility enables a CICS program (the client program) to call another CICS program (the server program) in a remote CICS region. DPL is used when the SYSID option on the EXEC CICS LINK PROGRAM command, or the REMOTESYSTEM option of the program resource definition, specifies a remote CICS region.

When the SYSID option on the EXEC CICS LINK command specifies a remote CICS system, the client region does not perform any resource security checking, but leaves the resource check to be performed in the server region.

The server program in the remote region is executed by a mirror transaction, in a similar way to other function-shipped CICS requests. However, the transaction name associated with the mirror depends on how the EXEC CICS LINK PROGRAM is invoked in the client region. You must be aware of the transaction name because normal attach security applies to the mirror transaction:

- If the TRANSID option is specified on the DPL command, the specified transaction name is used for the mirror.
- If the TRANSID option is omitted from the DPL command, but the TRANSID option is used in the program resource definition in the client region, the name for the mirror is taken from the program's TRANSID specification.

Otherwise, a default name for the mirror transaction is used, and this depends on the origin and LU6.2 sync level of the conversation:

- If the client program is executing in a CICS OS/2 system, the transaction name for the mirror is **CPMI**.

- If synclevel 1 is being used, the default transaction name for the mirror is **CVMI**. This transaction name is used:
  - If the SYNCONRETURN option is specified on the DPL command in the client region
  - If the LU6.2 CONNECTION definition specifies SINGLESESS(YES)
  - If the connection is by means of an LU6.2 terminal; that is, a terminal whose resource definition uses a TYPETERM with a specification of DEVICE(APPC)
- If sync level 2 is being used, the default transaction name is **CSMI**. This transaction name is used when none of the other previous conditions is met.

Authorize your users to access the transaction name that the mirror runs under. The userids to be authorized depend on whether LOCAL or non-LOCAL attach security is being used, and are described in "Security checking done in AOR with LU6.2". If the mirror transaction is defined with RESSEC(YES) in the server region, these userids must also be authorized to access the server program that is being linked to by the mirror. If the server program accesses any CICS resources, the same userids must be authorized to access them. If the server program invokes any SP-type commands, and the mirror transaction is defined with CMDSEC(YES) in the server region, the same userids must be authorized to access the commands.

If the mirror transaction cannot be attached because of security reasons, the NOTAUTH condition is not raised, but the TERMERR condition is returned to the issuing application in the client region. If the mirror transaction is successfully attached, but it is not authorized to link to the distributed program in the server region, the NOTAUTH condition is raised. The NOTAUTH condition is also raised if the server program fails to access any CICS resources for security reasons.

The server program is restricted to a DPL-subset of the CICS API commands when running in a server region. The commands that are not supported include some that return security-related information. For programming information about which commands are restricted, see the *CICS Application Programming Reference*. For information about surrogate user checking on DPL calls, see "Userid passed as parameter on EXCI calls" on page 106. For further information about DPL, refer to the *CICS Intercommunication Guide*.

## Security checking done in AOR with LU6.2

This section summarizes how security checking is done in the AOR depending on how SECURITYNAME is specified in the AOR and TOR.

The link userid referred to in Table 22 on page 170 and Table 23 on page 171 is the one specified in the SECURITYNAME on the CONNECTION definition, or the USERID on the SESSIONS definition.

If a USERID is specified on the SESSIONS definition, and a link check is done, the userid used is the one on the SESSIONS definition.

If no userid is specified in SECURITYNAME, then the default userid of the AOR is used instead. However, if the SECURITYNAME userid is the same as the region userid for the AOR, then the link is deemed to have the same security as the AOR, and **link security is omitted altogether**. The effect of omitted link security depends on whether LOCAL or non-LOCAL attach security is specified for the link:

- For LOCAL attach security, the security specified in the USERID on the SESSIONS definition is used. If this too is omitted, then the default userid for the AOR is used.

- For non-LOCAL attach security, the security specified in the USERID on the sessions definition is **not** used. Only the userid received from the TOR is used to determine security.

**Note:** Neither the region userid for the TOR, nor the SECURITYNAME in the TOR's CONNECTION definition for the AOR, is relevant to security checking in the AOR.

Table 22 shows how checking is done when ATTACHSEC(LOCAL) is specified.

*Table 22. LU6.2 and ATTACHSEC(LOCAL)*

| Region userid for AOR | SECURITYNAME in connection definition | USERID in SESSION definition | Checking in AOR |
|---|---|---|---|
| USERIDA | Not specified | Not specified | Check against AOR DFLTUSER |
| USERIDA | Not specified | USERIDA | Check against AOR DFLUTSER |
| USERIDA | Not specified | USERIDB | Check against USERIDB |
| USERIDA | USERIDA | Not specified | Check against AOR DFLTUSER |
| USERIDA | USERIDB | Not specified | Check against USERIDB |
| USERIDA | USERIDA | USERIDA | Check against AOR DFLTUSER |
| USERIDA | USERIDA | USERIDB | Check against USERIDB |
| USERIDA | USERIDB | USERIDA | Check against DFLTUSER |
| USERIDA | USERIDB | USERIDB | Check against USERIDB |
| USERIDA | USERIDB | USERIDC | Check against USERIDC |

Table 23 on page 171 shows how checking is done when the ATTACHSEC parameter IDENTIFY (or PERSISTENT, or MIXIDPE) has been specified.

*Table 23. LU6.2 and ATTACHSEC(IDENTIFY|PERSISTENT|MIXIDPE)*

| Region userid for AOR | SECURITYNAME in connection definition | USERID in SESSION definition | Checking in AOR |
|---|---|---|---|
| USERIDA | Not specified | Not specified | Transmitted userid and AOR DFLTUSER |
| USERIDA | Not specified | USERIDA | Transmitted userid only |
| USERIDA | Not specified | USERIDB | Transmitted userid and USERIDB |
| USERIDA | USERIDA | Not specified | Transmitted userid only |
| USERIDA | USERIDA | USERIDA | Transmitted userid only |
| USERIDA | USERIDA | USERIDB | Transmitted userid and USERIDB |
| USERIDA | USERIDB | Not specified | Transmitted userid and USERIDB |
| USERIDA | USERIDB | USERIDC | Transmitted userid and USERIDC |

## Summary of resource definition options for LU6.2 security

The following is a summary of the resource definition options you need to define for LU6.2 security:

- On the CONNECTION definition:
  - ATTACHSEC, with any one of the following options:
    IDENTIFY
    LOCAL
    MIXIDPE
    PERSISTENT
    VERIFY
  - BINDPASSWORD
  - BINDSECURITY
  - SECURITYNAME
- On the SESSIONS definition:
  - USERID

For guidance on specifying CONNECTION and SESSION definitions, see the *CICS Resource Definition Guide*.

# Chapter 14. APPC password expiration management

This chapter contains information on advanced program-to-program communications (APPC) password expiration management (PEM).

To use PEM you should understand APPC conversation-level security. To code the requester sign-on transaction, you also need to have basic APPC programming skills.

To find out what APPC PEM offers, read "Introduction to APPC password expiration management". System programmers responsible for coding the **PEM client** (requester) should also read "APPC PEM processing" on page 176, which explains the requirements of the PEM client and CICS PEM server.

**Note:** In this chapter the word 'sign-on' is used in the sense defined in the APPC architecture, which is different from the meaning used elsewhere in this book.

This chapter includes the following topics:


## Introduction to APPC password expiration management

This section introduces, and describes the benefits of, APPC password expiration management.

You may find it useful to copy and modify an example program. For your guidance sample programs are now shipped in library *CICSTS13.CICS.SDFHSAMP.* Their names are:

1. DFH$$.SNPW — PEM sample program for Windows NT
2. DFH$$.SNP2 — PEM sample program for OS/2

For examples of PEM requester and CICS PEM server user data produced by a program, see:


### What APPC PEM does

APPC PEM with CICS provides receive support for an APPC architected sign-on transaction that verifies user ID, password pairs, and processes requests for a password change by:

- Identifying a user and authenticating that user's identification

- Notifying specific users during the authentication process that their passwords have expired
- Letting users change their passwords when (or before) the passwords expire
- Telling users how long their current passwords will remain valid
- Providing information about unauthorized attempts to access the system using a particular user identifier

### Benefits of APPC PEM

APPC PEM has the following benefits:

- It enables users to update passwords on APPC links.

  This can be particularly useful in the case of expired passwords. On APPC links that do **not** support APPC PEM, when users' passwords expire on remote systems, they are unable to update them from their own systems. The only alternative on a non-APPC PEM system is to log on directly to the remote system using a non-APPC link, such as an LU2 3270-emulation session, to update the password.
- It provides APPC users with additional information regarding their sign-on status; for example, the date and time at which they last signed on.
- It informs users whether their userid is revoked, or the password has expired, when they provide the correct password or PassTicket.

## What you require to use APPC PEM

To use APPC PEM, you need a **PEM client** (requester) and a **PEM server** linked by an APPC session. An external security manager, such as RACF, or an equivalent ESM, must also be available to the PEM server. Figure 11 shows a sample configuration.

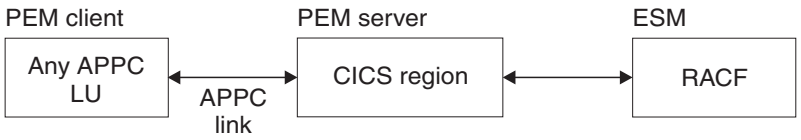



*Figure 11. Sample APPC PEM configuration*

The PEM client can be any APPC logical unit (LU) or node that is capable of initiating a conversation with the architected sign-on transaction. However, the benefits of using APPC PEM are increased when using an LU or node that does not have its own ESM; for example, a programmable workstation. APPC PEM enables users of such LUs or nodes to manage their password values within the ESM used by CICS.

The PEM server can be any APPC LU that supports APPC PEM. This chapter assumes that the PEM server is the one provided by CICS Transaction Server for OS/390 Release 3. It is referred to in the rest of this book as the CICS PEM server.

## External security interface

Password expiration management has been enhanced to include the External Security Interface (ESI). The ESI is not part of CICS Transaction server for OS/390 1.3, but it allows a non-CICS application to invoke services provided by advanced APPC PEM. ESI provides additional functions which make it easier for a non-CICS application to change and verify a password.

The 2 functions provided by ESI are:

- **CICS_VerifyPassWord** which allows a client application to verify that a password matches the password recorded by RACF, or an equivalent external security manager, for a specified user ID.
- **CICS_ChangePassWord** which allows a client application to change the password recorded by RACF for a specified user ID.

These functions allow a non-CICS application program to act as a PEM requestor without the application programmer having to manage an APPC conversation which implies knowledge of the formats for PEM requests and replies, and of the interface to the local SNA server.

For more information about the ESI password management functions, see the *Client/Server Application Programming* manual.

## Roles of PEM client and CICS PEM server

CICS Transaction Server for OS/390 Release 3 does not send passwords on APPC conversations. This means that it can **attach**, but not **initiate** the sign-on transaction, and must always act as the PEM server. Therefore, in your configuration always include an LU that is capable of initiating the sign-on transaction to act as the PEM client.

The PEM client collects sign-on information and sends it to the CICS PEM server via a sign-on transaction program. The sign-on transaction program is a SNA service transaction program, as described in *SNA LU 6.2 Peer Protocols* manual.

Note that PEM signon is not to be confused with a CICS signon. In CICS, PEM signon allows the APPC LU to verify and manage user IDs and passwords. Following verification or updating, the user ID or password is intended to be included in the ASIS part of the FMH5 attach header. When this FMH5 is sent into CICS through the APPC link, provided ATTACHSEC if non-local, the user ID will be signed on to CICS. Therefore, a PEM signon does not result in the ESM last—connect, last-access information being updated. For more information, see "APPC password expiry management" on page 238.

The CICS PEM server then processes the sign-on request, updates the user's password (if necessary), and returns a reply to the PEM client containing responses and other data, such as a password expiry and information about unauthorized attempts to sign on. The PEM client then processes the data, as appropriate.

## An example of signing on with APPC PEM

Figure 12 on page 176 shows an example sign-on for APPC PEM.

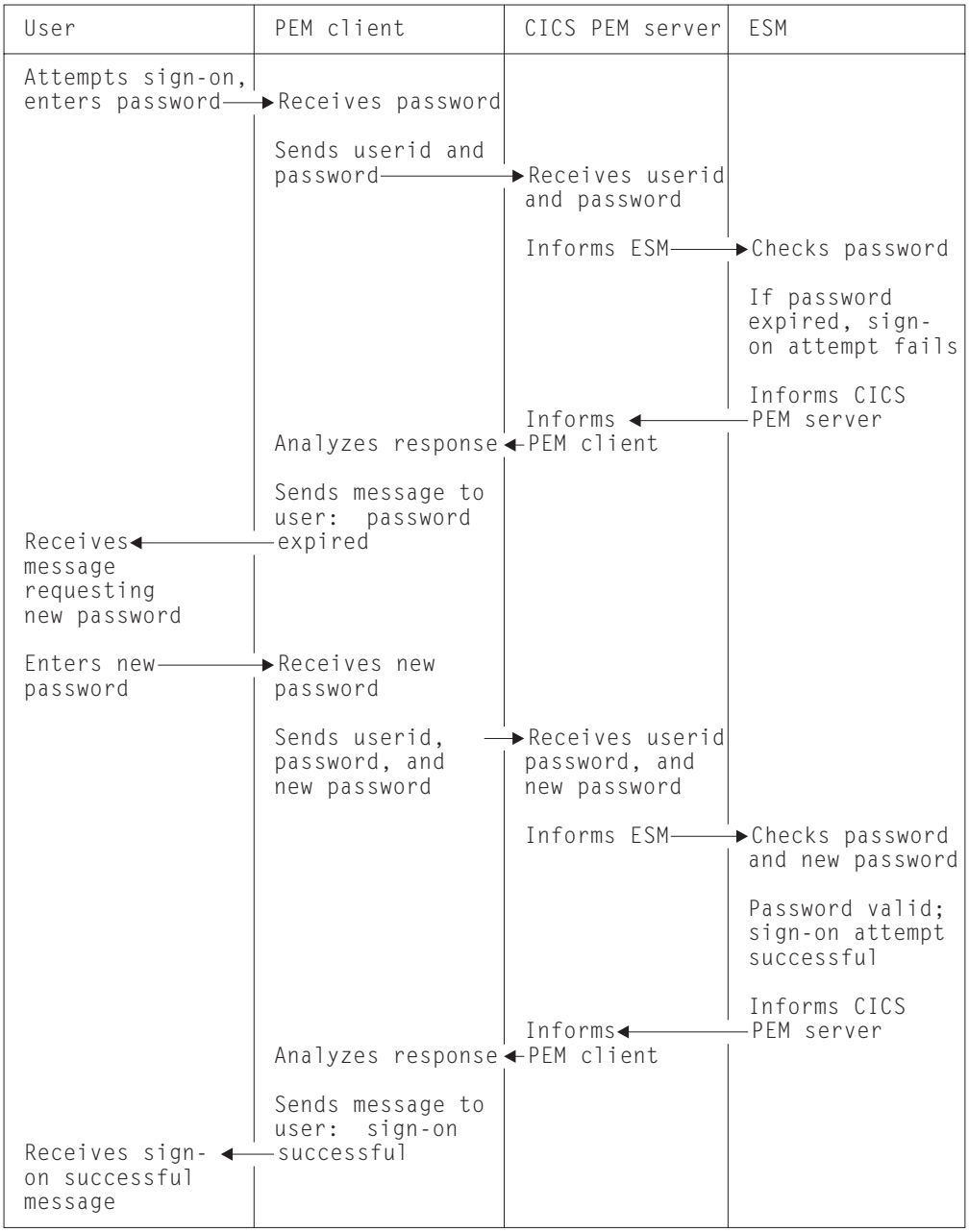



*Figure 12. Example of signing on with APPC PEM*

## APPC PEM processing

In order to code the sign-on transaction program for the PEM client to send the sign-on details to the CICS PEM server, you need to know the following:

- What happens on each side of the link—see "Overview of APPC PEM processing" on page 177.
- How to code the PEM client—see "Setting up the PEM client" on page 181, "Format of user data" on page 182, and "Examples of PEM client and CICS PEM server user data" on page 187.

  See an example of a PEM client sign-on transaction program in the examples shipped in library *CICSTS13.CICS.SDFHSAMP.*, described in "Introduction to APPC password expiration management" on page 173.

- The data the CICS PEM server requires from the PEM client—see "Sign-on input data sent by PEM client" on page 183
- The data the CICS PEM server sends in response to the PEM client—see "Sign-on output data returned by CICS PEM server" on page 184.

## Overview of APPC PEM processing

CICS provides the PEM server, the receive side of APPC PEM as a CICS transaction that is started when an ATTACH for the sign-on transaction program is received from the PEM client.

CICS retrieves the sign-on data associated with the request, calls the ESM to perform a sign-on, and retrieves sign-on details for the userid. If the sign-on data includes a new password value, CICS includes this value when it calls the ESM to request a sign-on.

If PV is being used, and sign-on completes correctly, the user is added to the PV "signed-on-*from* list" in CICS, and the PV "signed-on-*to* list" in the PEM client. The "signed-on-"lists keep track of verified user IDs.

The CICS PEM server builds a reply and returns it to the PEM client, after which the CICS PEM server transaction terminates normally.

### PEM client processing

The PEM client sign-on transaction program:

1. Obtains sign-on information, for example by:
   - Displaying a message to the user requesting sign-on information; that is, userid, password, and, if required, new password; or
   - Accessing sign on information from a user who has already been authenticated locally.
2. Sends the sign-on information to the CICS PEM server via an APPC conversation.
3. Receives replies from the CICS PEM server on the same APPC conversation.
4. If PV is being used (either ATTACHSEC=PERSISTENT or ATTACHSEC=MIXIDPE is specified on the CONNECTION definition), and sign-on is successful, adds the user's name to the PV signed-on-to list.
5. Processes the reply information from the CICS PEM server; for example, by:
   - Displaying the information to the user
   - Processing the data and saving it in a file to which only the user has access.

### CICS PEM server processing

The CICS PEM server performs the following processing:

1. Accepts the userid and password, with optional new password, from the sign-on PEM client.
2. Tries to validate the user with its ESM.

   If the userid and password are valid and the password has not expired, the CICS PEM server extracts the following information from its ESM:
   - Date and time of the last successful sign-on
   - Data and time the password will expire (calculated by data extracted from the ESM by the CICS PEM server itself)
   - Number of unsuccessful sign-on attempts since the last successful sign-on.

3. Returns a response to the PEM client (described in Table 25 on page 184, and illustrated in both Figure 18 on page 187 and Figure 20 on page 190), indicating whether the sign-on was succeeded or failed, and the reason for any failure:

```
Status           = (X'00') OK
Date-Time        = Current date and time
Last-Date-Time   = Date and time of previous successful sign-on
Expiry-Date-Time = Date and time password will expire
Revoke-Count     = Number of unsuccessful sign-on attempts made with
                   this userid since the previous successful sign-on
```

   **Note:** The ESM increments the revoke count whenever it processes an invalid sign-on attempt. The sign-on request may originate from a non-CICS system (for example, a TSO user).

   If sign-on is unsuccessful, CICS returns to the PEM client a sign-on completion status value (as described in Table 27 on page 186) and, if appropriate, a formatting error value (as described in Table 28 on page 186).

4. If PV is being used (either ATTACHSEC=PERSISTENT or ATTACHSEC=MIXIDPE is specified on the CONNECTION definition), and sign-on is successful, adds the user's name to the PV signed-on-from list.

## Expected flows between PEM client and CICS PEM server

Figure 13 on page 179 through Figure 16 on page 181 show expected flows for successful and unsuccessful sign-on attempts with and without PV. These examples do not include information on setting up the connection. For more information on doing this, see the *CICS Intercommunication Guide*.

**Note:** CICS support for the PEM client sign-on transaction assumes that the request for sign-on (or sign-on and change password) is for a single user. Batching of sign-on requests for different userids within a single sign-on transaction is not supported. If multiple sign on requests are passed in the input data, the CICS PEM server processes only the first one.

### Successful sign-on—non-PV connection

Figure 13 shows the expected flows between the PEM client and CICS PEM server during a successful sign-on when PV is not being used.

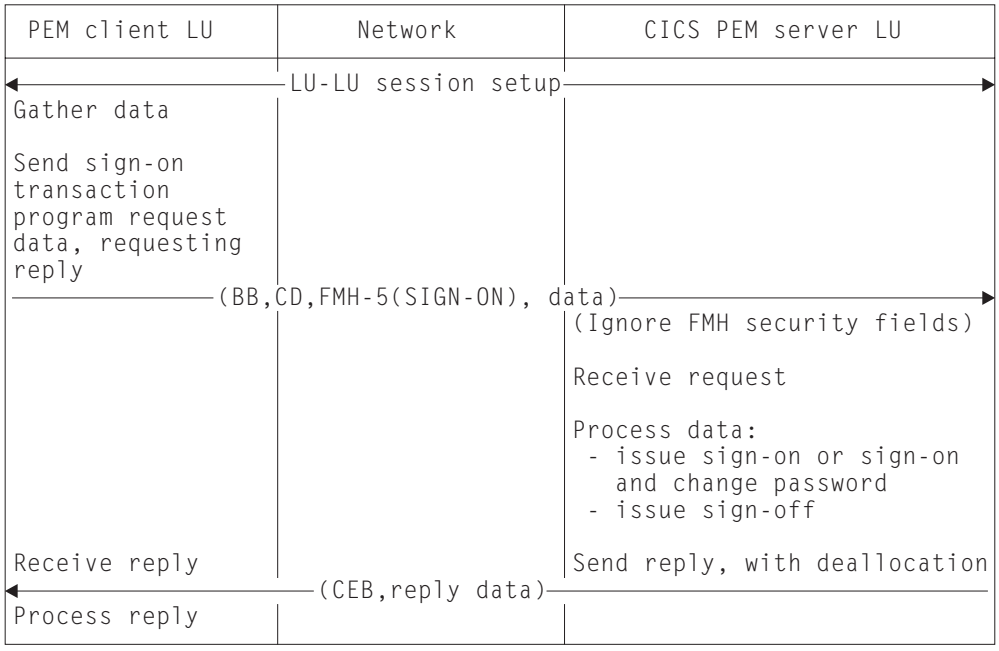



*Figure 13. Successful sign-on—non-PV connection*

**Note:** All security fields in the FMH-5 (userid, password and UP, AV, PV1 and PV2 bits) are ignored by the CICS PEM server when it attaches the sign-on transaction.

## Unsuccessful sign-on—non-PV connection

Figure 14 shows the expected flows for an unsuccessful sign-on between a PEM client and CICS PEM server when PV is not being used.

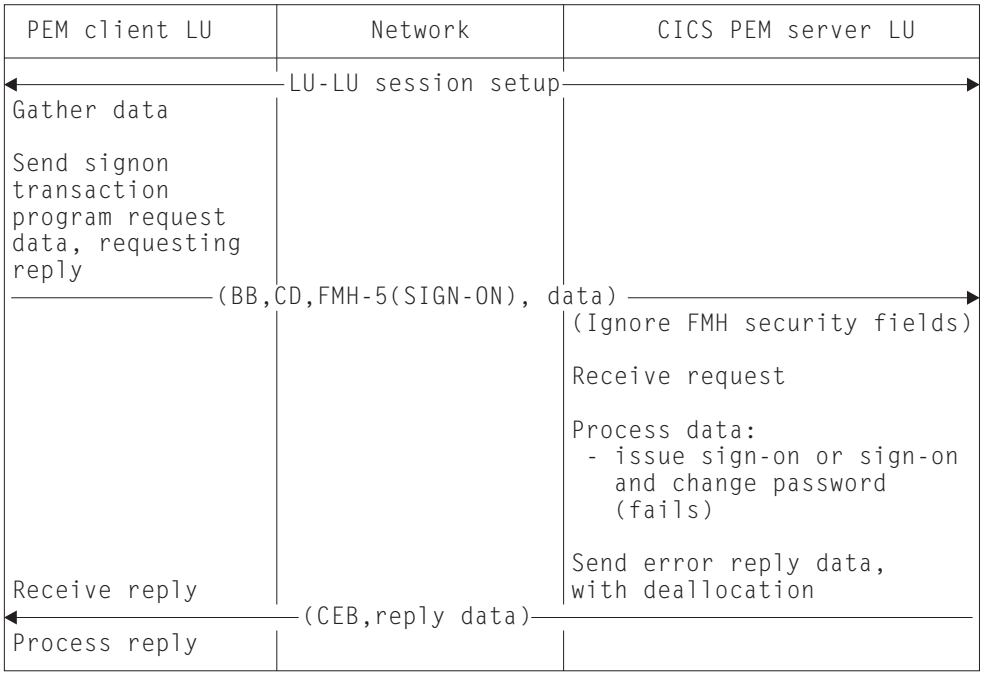



*Figure 14. Unsuccessful sign-on—non-PV connection*

**Note:** The CICS PEM server schedules sign-off against the PEM client if a sign-on request for a userid fails.

## Successful sign-on—PV connection

Figure 15 shows the expected flows between the PEM client and CICS PEM server during a successful sign-on on a PV connection.

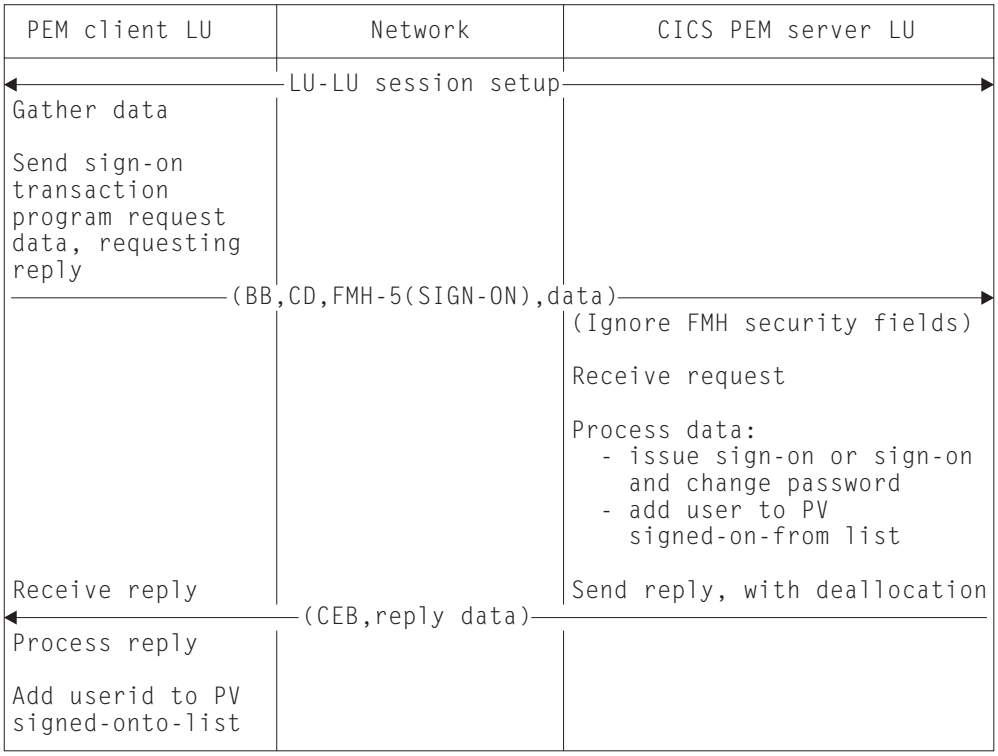



*Figure 15. Successful sign-on—PV connection*

**Notes:**

1. All security fields in the FMH-5 (userid, password and UP, AV, PV1 and PV2 bits) are ignored by the CICS PEM server when it attaches the sign-on transaction.
2. The CICS PEM server adds the userid to its PV signed-on-from list only if the sign-on and change password request is successful and either ATTACHSEC=MIXIDPE or ATTACHSEC=PERSISTENT is specified in the CONNECTION definition.
3. The PEM client must add the userid to its PV signed on-to list only if a successful sign-on reply is received from the CICS PEM server. The userid has been added to the PV signed on from list in the CICS PEM server, so all subsequent attach requests from this userid can flow as signed on.

## Unsuccessful sign-on—PV connection

Figure 16 on page 181 shows the expected flows between a PEM client and CICS PEM server during an unsuccessful sign-on on a PV connection.

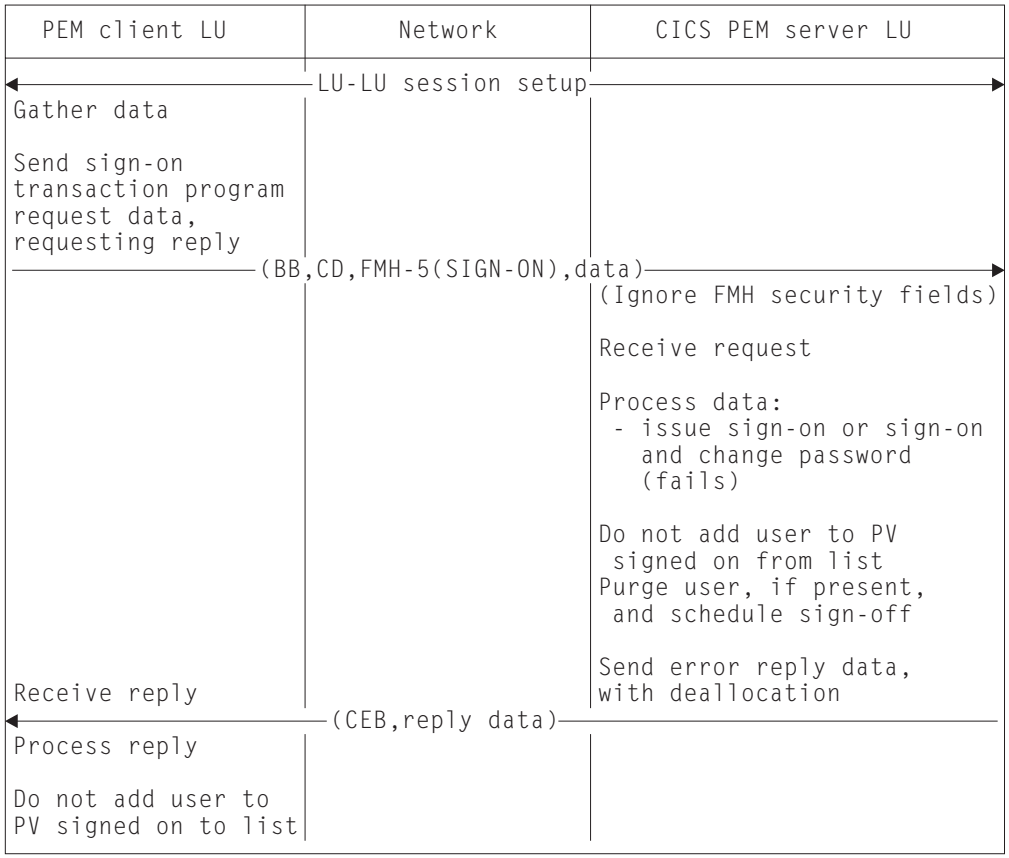



*Figure 16. Unsuccessful sign-on—PV connection*

**Note:** CICS schedules sign-off against the PEM client if a sign-on request for a userid fails, and that user is in the PV signed on from list. In this case, CICS sends the sign-off transaction program output data to the PEM client, where it is processed and the userid removed from the PV signed on to list.

## Setting up the PEM client

When setting up your PEM client, note the following:

- Use the basic (also known as **unmapped**) conversation type. In addition to sending the data you want the partner to receive, you must add control bytes (in Assembler language or C/370) to convert the data into an SNA-defined format called a **generalized data stream** (GDS). Include the keyword GDS in any EXEC CICS commands used. See the *CICS Intercommunication Guide* for introductory information on basic conversations, and the *CICS Distributed Transaction Programming Guide* for information on using them.
- The SNA service transaction program name for the sign-on transaction program is **X'06F3F0F1'**, which is also the transaction id (XTRANID) that must be used for the CICS transaction CLS4. You specify XTRANID in the CICS TRANSACTION definition.
- Run the CICS PEM server sign-on transaction as a **sync level 0** transaction. If it is initiated with a sync level other than 0, it sends an ISSUE ABEND and frees the conversation.
- Translate the userid and password into EBCDIC; if they are not in this form, the ESM cannot recognize them and issues an error. See one of the the example programs in library *CICSTS.CICS.SDFHSAMP*, described in "Introduction to

APPC password expiration management" on page 173, for an example of converting userids and passwords to EBCDIC.

If the ESM is RACF, the userid and password must also be in uppercase characters.

- On the ATTACH request for the sign-on transaction program specify SECURITY(NONE). CICS ignores any ATTACH security fields passed in the ATTACH function management header, FMH-5, for this transaction.
- CICS does not support the receipt of the PROFILE option in the sign-on transaction program. If data identifier (ID) X'00' is provided, CICS returns status value X'06' (incorrect data format) with formatting error X'0002' (precluded structure present), as described in Table 28 on page 186.
- The new password ID, X'06', is permitted and required only with the X'FF01' request data ID. If the new password is provided with a data ID other than X'FF01', CICS returns status value X'06' (incorrect data format) with formatting error X'0002' (precluded structure present), as described in Table 28 on page 186.
- CICS only supports userids and passwords up to 8 characters long. If the userid or password length (after stripping blanks and nulls) exceeds 8, CICS returns status value X'06' (incorrect data format) with formatting error X'000F' (data value out of range), as described in Table 28 on page 186.
- Program initialization parameter (PIP) data is optional on the ALLOCATE for the sign-on transaction, and is ignored if sent.
- If the sign-on transaction receives a GDS ISSUE SIGNAL command, it is ignored.
- If the CICS PEM server receives a GDS ISSUE ERROR command, it replies with ERROR and frees the conversation.
- If the CICS PEM server receives a GDS FREE command, it frees the conversation. (It does not provide diagnostic information about the type of conversation error.)
- The CICS PEM server transaction does not support the receipt of data exceeding the maximum buffer size. If the concatenation bit in the initial LL is set, the command is ignored; concatenated data is also ignored.

## Format of user data

As part of the general rules for APPC basic conversations, the user data must be in LL-ID-data format (where LL and ID are each two bytes long), and must follow the attach FMH-5 header. As described in Table 24 on page 184, the CICS DFHCLS4 program requires the user input data stream to fit into the format shown in Figure 17; if it does not, CICS rejects the data.

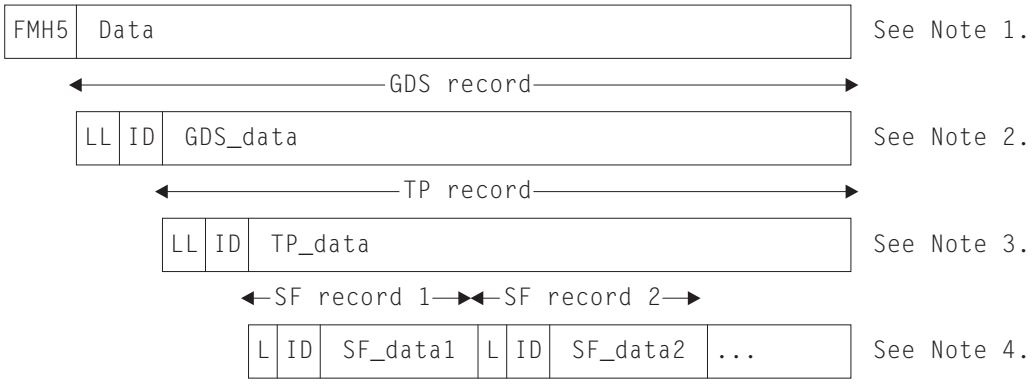



*Figure 17. Format of user data*

**Notes:**

1. This is an attach FMH-5 header with its data. Data is passed between the PEM client and the CICS PEM server via GDS variables. (For information on GDS, see the *SNA LU 6.2 Peer Protocols* manual.)
2. The **GDS record** contains GDS data in the format LL-ID-data where:
   - LL, which is two bytes long, is the length of the GDS record, including the LL and ID lengths.
   - ID, which is two bytes long, indicates what the data record represents; for example, X'1221' (sign-on data).
3. The GDS data record is itself an LL-ID-data record; in this example, a transaction program record (or **TP record**) where:
   - LL, which is two bytes long, is the length of the TP record including the LL and ID lengths.
   - ID which is two bytes long, indicates the function the TP is to perform; for example, X'FF00' (sign-on) or X'FF01' (signon and change password).
4. The TP data record is divided up into L-ID-data records (where L and ID are each **one** byte long). These are known as subfield (or **SF records**) where:
   - L is the length of the SF record, including the L and ID lengths.
   - ID indicates the subfield being passed; for example, X'01' (userid), X'02' (password), and X'06' (new password).

## PEM client input and output data

To perform the functions described in "CICS PEM server processing" on page 177, the CICS PEM server takes input data from, and sends output data to, the PEM client sign-on transaction program:

- The PEM client sends data to the CICS PEM server, as described in Table 24 on page 184.
- The CICS PEM server sends data to the PEM client, as described in Table 25 on page 184 through Table 28 on page 186.

Ensure the data conforms to the standards described in "Setting up the PEM client" on page 181, and that its format is as described in "Format of user data" on page 182. See "Sign-on with correct userid and password" on page 187 and "Sign-on with new password" on page 188 for examples of sign-on output data in GDS flows.

Basic conversation information and data are contained in the attach FMH, as described in "Format of user data" on page 182. The sign on request attaches a transaction X'06F3F0F1', which is the SNA service transaction program name for the sign-on transaction program.

### Sign-on input data sent by PEM client

Table 24 on page 184 shows the input data that the CICS PEM server needs from the PEM client sign-on transaction program. See "Sign-on with correct userid and password" on page 187 and "Sign-on with new password" on page 188 for examples of sign-on input data in GDS flows.

*Table 24. Sign-on request and data sent to CICS PEM server*

| Length (bytes) | Value | Meaning |
|---|---|---|
| 2 | X'nnnn' | Length of entire GDS data, including this 2-byte length value. |
| 2 | X'1221' | Data ID for sign-on data. |
| 2 | X'nnnn' | Length of this second (nested) data structure (length, data ID, and data), including this 2-byte length value. |
| 2 | X'FF00' or X'FF01' | Data ID for sign-on or sign-on and change password request data, respectively. (New password subfield is not permitted for X'FF00'.) |
| 1 | X'nn' | Length of subfield for userid, including this 1-byte length value. |
| 1 | X'01' | Data ID of subfield for userid. |
| X'nn'–2 | C'xxxxxxxx' | Userid. |
| 1 | X'mm' | Length of subfield for password, including this 1-byte length value. |
| 1 | X'02' | Data ID of subfield for password. |
| X'mm'–2 | C'xxxxxxxx' | Password. |
| 1 | X'pp' | Length of subfield for new password, including this 1-byte length value. |
| 1 | X'06' | Data ID of subfield for new password. |
| X'pp'–2 | C'xxxxxxxx' | New password. |

## Sign-on output data returned by CICS PEM server

Table 25 lists the sign-on output data that the CICS PEM server returns to the PEM client. See "Response to correct sign-on data" on page 189 and "Response to incorrect data format" on page 190 for examples of sign-on output data in GDS flows.

*Table 25. Sign-on output data returned to PEM client*

| Length (bytes) | Value | Required or optional | Meaning |
|---|---|---|---|
| 2 | X'nnnn' | Required | Length of entire GDS data, including this 2-byte length value. |
| 2 | X'1221' | Required | Data ID of subfield for sign-on data. |
| 2 | X'nnnn' | Required | Length of this second (nested) data structure (length, data ID, and data), including this 2-byte length value. |
| 2 | X'FF02' | Required | Data ID for sign-on reply data. |
| 1 | X'03' | Required | Length of subfield for sign-on completion status, including this 1-byte length value. |
| 1 | X'00' | Required | Data ID of subfield for sign-on completion status. |
| 1 | X'00' through X'06' | Required | Sign-on completion status—see Table 27 on page 186. |
| 1 | X'04' | Optional | Length of subfield for sign-on request formatting error, including this 1-byte length value. |
| 1 | X'01' | Optional | Data ID of subfield for sign-on request formatting error. |
| 2 | X'0000' through X'0003', X'0005' through X'0007', X'000F' | Optional | Sign-on request formatting error—see Table 28 on page 186. |

*Table 25. Sign-on output data returned to PEM client (continued)*

| Length (bytes) | Value | Required or optional | Meaning |
|---|---|---|---|
| 1 | X'0A' | Optional | Length of subfield for date and time of current successful sign-on, including this 1-byte length value. |
| 1 | X'02' | Optional | Data ID of subfield for date and time of current successful sign-on. |
| 8 | See Table 26 for format | Optional | Date and time of current successful sign-on. The date and time returned are extracted by the ESM from the user profile. |
| 1 | X'0A' | Optional | Length of subfield for date and time of last successful sign-on, including this 1-byte length value. |
| 1 | X'03' | Optional | Data ID of subfield for date and time of last successful sign-on. |
| 8 | See Table 26 for format | Optional | Date and time of last successful sign-on. The date and time returned are extracted by the ESM from the user profile. |
| 1 | X'0A' | Optional | Length of subfield for date and time password will expire, including this 1-byte length value. |
| 1 | X'04' | Optional | Data ID of subfield for date and time password will expire. |
| 8 | See Table 26 for format. | Optional | Date and time password will expire. (The date and time returned are calculated from data obtained from the ESM.) |
| 1 | X'04' | Optional | Length of subfield for revoke count, including this 1-byte length value. |
| 1 | X'05' | Optional | Data ID of subfield for revoke count. |
| 2 | X'nnnn' | Optional | Revoke count. |

## Format of date and time subfields

Table 26 lists the format of the date and time subfields that the CICS PEM server can return to the PEM client, as referred to in Table 25 on page 184. See "Response to correct sign-on data" on page 189 for an example of date and time subfields in a GDS flow.

*Table 26. Format of date and time subfields using 24-hour clock*

| Position | Meaning |
|---|---|
| 00 | Two-byte year value; for example, 1994=X'07CB'. |
| 02 | One-byte month value; January=X'01', December=X'0C'. |
| 03 | One-byte day value; first day=X'01', thirty-first day=X'1F'. |
| 04 | One-byte hour value; midnight=X'00', 23rd hour=X'17'. |
| 05 | One-byte minute value; on the hour=X'00', 59th minute=X'3B'. |
| 06 | One-byte second value; on the minute=X'00', 59th second=X'3B'. |
| 07 | One-byte 100ths of a second value; on the second=X'00', maximum=X'63'. |

**Note:** The maximum time value for a given day is 23 hours, 59 minutes, and 59.99 seconds (decimal). Midnight is 0 hours, 0 minutes, and 0 seconds on the following day.

## Sign-on completion status values returned to PEM client

Table 27 describes the sign-on completion status values (referred to in Table 25 on page 184) that the CICS PEM server can return to the PEM client in the status completion subfield in the sign-on reply data. See "Response to correct sign-on data" on page 189 for an example of sign-on completion status values in a GDS flow.

*Table 27. Sign-on completion status values returned to PEM client*

| Status value | Meaning |
|---|---|
| X'00' | All of the following conditions apply:<br>• Userid valid<br>• Password valid<br>• Password not expired or new valid password specified<br><br>When this status value is returned, the new password is set if specified, and PV processing (if used) is complete. |
| X'01' | Userid not known to the receiver. |
| X'02' | Userid valid, password incorrect. |
| X'03' | Userid valid, password correct but expired. New password must be set. |
| X'04' | Userid valid, password correct, new password not acceptable to receiving security system. |
| X'05' | Security function failure. Function not performed. |
| X'06' | Incorrect data format. Specific error is contained in the sign-on request formatting error subfield described in Table 28. |

**Note:** The CICS PEM server never returns either of the following sign-on status values to the PEM client:
- X'07'—general security error
- X'08'—password change completed, but sign-on failed.

## Sign-on request formatting errors returned to PEM client

Table 28 lists the sign-on request formatting error values (referred to in Table 25 on page 184) that the CICS PEM server can return to the PEM client. Each is a 2-byte binary value. See "Response to incorrect data format" on page 190 for an example of sign-on request formatting errors in a GDS flow.

*Table 28. Sign-on request formatting error values returned to PEM client*

| Error value | Description |
|---|---|
| X'0000' | Undefined error not described below. |
| X'0001' | Required structure absent. |
| X'0002' | Precluded structure present. |
| X'0003' | Several occurrences of a nonrepeatable structure. |
| X'0005' | Unrecognized structure present where precluded. |
| X'0006' | Length outside specified range. This value assumes that the length arithmetic balances and that the sender intended to send the structure at that length. |
| X'0007' | Length exception. Length arithmetic is out of balance. |
| X'000F' | Data value out of range. |

## Application design

Design your applications to run the sign-on transaction before any other transaction. This keeps that any password check and any password changing separate from the application's own functions. In multitasking systems, it is possible for more than one sign-on transaction to start on parallel sessions. It is important that the code dealing with application-level ALLOCATE requests, serializes the sign-on process to completion, thus ensuring both flow as signed-on.

To record the date and time of a previous successful sign-on, the CICS PEM server sign-on program extracts password data from the ESM **before** it performs sign-on. If your system uses shared userids, and two users attempt to sign on at the same time, or if a user is multitasking, the time values returned to the PEM client for the current sign-on may not be the same as the timestamp recorded on the ESM database. Remember this if you are writing an application that is to run on multiple systems, and depends on the sign-on time returned to the PEM client. (This situation should not apply on a single system, provided the sign-on process is serialized as suggested.)

If PV is being used, and the interval specified in PVDELAY is exceeded, and the userid is removed from the PV sign on from list, applications must allow for the sign-on process to be serialized again.

## Examples of PEM client and CICS PEM server user data

**General-use programming interface information**

Data is passed between the PEM client and the CICS PEM server via GDS variables. To help you check the data being sent by your PEM client, the examples that follow show:

- "Sign-on with correct userid and password"
- "Sign-on with new password" on page 188
- "Response to correct sign-on data" on page 189
- "Response to incorrect data format" on page 190.

These examples are produced by the sample PEM client program shown in the library *CICSTS13.CICS.SDFHSAMP*, described in "Introduction to APPC password expiration management" on page 173. That program uses a **partner_LU_alias** of `hostcics`, an **LU_alias** of `ps2lua`, and a **mode_name** of `lu62ss`. When writing your own PEM client program, use the values defined in your communications manager configuration.

### Sign-on with correct userid and password

Figure 18 shows a sample flow for a successful sign-on.

```
PEM hostcics ps2lua lu62ss sec2r01 drtnnom
```

*Figure 18. Sign-on with correct userid and password, no new password*

A valid userid (`sec2r01`) and password (`drtnnom`) are correctly entered. No new password is entered.

The PEM client sends the following hexadecimal user data to the CICS PEM server:

```
001A12210016FF000901E2C5C3F2D9F0F10902C4D9E3D5D5D6D4
```

This contains the following values, as described in Table 24 on page 184:

**001A** Length of the entire GDS data, including this 2-byte length value

**1221** Data ID for sign on data

**0016** Length of this second (nested) data structure (length, data ID, and data), including this 2-byte length value

**FF00** Data ID for sign-on request data

**09** Length of subfield for userid, including this 1-byte length value

**01** Data ID of subfield for userid

**E2C5C3F2D9F0F1**
Userid (SEC2R01) in EBCDIC

**09** Length of subfield for password, including this 1-byte length value

**02** Data ID of subfield for password

**C4D9E3D5D5D6D4**
Password (DRTNNOM) in EBCDIC

## Sign-on with new password

The following is an example of a successful sign-on using a new password.

```
PEM hostcics ps2lua lu62ss sec2r01 drtnnom hursley
```

A userid, password, and new password are correctly entered.

The PEM client sends the following hexadecimal user data to the CICS PEM server:

```
0231221001FFF010901E2C5C3F2D9F0F10902C4D9E3D5D5D6D40906C8E4D9E2D3C5E8
```

This contains the following values, as described in Table 24 on page 184:

**0023** Length of entire GDS variable, including this 2-byte length value

**1221** Data ID for sign

**001F** Length of this second (nested) data structure (length, data ID, and data), including this 2-byte length value

**FF01** Data ID for sign-on and change password request data

**09** Length of subfield for userid, including this 1-byte length value

**01** ID of subfield for userid

**E2C5C3F2D9F0F1**
Userid (SEC2R01) in EBCDIC

**09** Length of subfield for password, including this 1-byte length value

**02** ID of subfield for password

**C4D9E3D5D5D6D4**
Password (DRTNNOM) in EBCDIC

**09** Length of subfield for new password, including this 1-byte length value

**06** ID of subfield for new password

**C8E4D9E2D3C5E8**
New password (HURSLEY) in EBCDIC

## Response to correct sign-on data

Figure 19 shows an example of the response to the correct sign-on data being entered.

```
PEM_OK
GDS LLID
00 2d 12 21
Sign-on Reply LLID
00 29 ff 02
Sign-on Completion Status Subfield
03 00 00
Date & Time of Current Successful Sign-on Subfield
0a 02 07 ca 01 14 0d 24 31 62
Date & Time of Last Successful Sign-on Subfield
0a 03 07 ca 01 11 16 1b 23 3e
Date & Time Password Will Expire Subfield
0a 04 07 ca 02 03 00 00 00 00
Revoke Count Subfield
04 05 00 00
```

*Figure 19. Response to correct sign-on data*

The first three lines of hexadecimal user data returned to the PEM client show the following *required* values, as described in Table 25 on page 184.

**002d** Total length of the GDS variable, including this 2-byte length value

**1221** Data ID for sign-on data

**0029** Length of this second (nested) data structure (length, data ID, and data), including this 2-byte length value

**FF02** Data ID for sign-on reply data

**03** Length of subfield for sign-on completion status, including this 1-byte length value

**00** Data ID for sign-on completion status

**00** Sign-on completion status. 00 indicates that the userid and password were valid, and the password had not expired. (See Table 27 on page 186 for a list of sign-on completion status values.)

In Figure 19, the last four lines of hexadecimal user data returned to the PEM client show the following optional values, as described in Table 25 on page 184. (Note that the formatting error subfields shown in Table 25 on page 184 are not included, indicating that there are no errors.)

**0A** Length of subfield for date and time of current successful sign-on including this 1-byte length value

**02** Data ID for date and time of current successful sign-on

Date and time of current successful sign-on, as described in Table 26 on page 185:

**07CA** Year (1994)

**01** Month (January)

**14** Day (20)

**0D** Hour (13)

**24** Minutes (36)

**31** Seconds (49)

**62** Hundredths of a second (98)

**0A** Length of subfield for date and time of previous successful sign-on,

**03** Data ID for date and time of previous successful sign-on

Date and time of previous successful sign-on, as described in Table 26 on page 185:

**07CA** Year (1994)

**01** Month (January)

**11** Day (17)

**16** Hour (22)

**1B** Minutes (27)

**23** Seconds (35)

**3E** Hundredths of a second (62)

**0a** Length of subfield for date and time password will expire (including this 1-byte length value)

**04** Length of subfield for data ID for date and time password will expire

Date and time password will expire, as described in Table 26 on page 185:

**07ca** Year (1994)

**02** Month (February)

**03** Day (14)

**00** Hour (00)

**00** Minutes (00)

**00** Seconds 00)

**00** Hundredths of a second (00)

**04** Length of subfield for revoke count, including this 1-byte length value

**05** Data ID of subfield for revoke count

**0000** Revoke count. (0000 means that there have been no unsuccessful sign-on attempts since the last successful sign-on with this userid.)

## Response to incorrect data format

Figure 20 shows an example response to incorrect data being entered.

```
PEM_OK
GDS_LLID
00 0F 12 21
Sign-on Reply LLID
00 0B FF 02
Sign-on Completion Status Subfield
03 00 06
Sign-on Request Formatting Error Subfield
04 01 00 0F
```

*Figure 20. Response to incorrect data format*

The first three lines of hexadecimal user data returned to the PEM client show the following required values, as described in Table 25 on page 184:

**000F** Length of entire GDS data, including this 2-byte length value

**1221** Data ID for sign-on data

**000B** Length of this second (nested) data structure (length, data ID, and data), including this 2-byte length value

**FF02** Data ID for sign-on reply data

**03** Length of subfield for sign-on completion status, including this 1-byte length value

**00** Data ID of subfield for sign-on completion status

**06** Sign-on completion status 06 indicating incorrect data format (see Table 27 on page 186 for a list of signon completion status values.)

The last line of hexadecimal user data returned to the PEM client shows the following **optional** values, which are returned only if there is an error. (The optional values are described in Table 25 on page 184.)

**04** Length of subfield for sign-on request formatting error, including this 1-byte length value

**01** Data ID of subfield for sign-on request formatting error

**000F** Sign-on request formatting error, indicating “data value out of range” (see Table 28 on page 186 for a description of other possible formatting errors).

**End of General-use programming interface information**

# Chapter 15. Implementing LU6.1 security

This chapter tells you how to implement link security for LU6.1, and covers the following topics:



For LU6.1 links, CICS cannot check the identity of the requesting system, and the bind request is never rejected on security grounds. You are advised to use the intersystem security offered by LU6.2 links whenever possible. Note that no bind-time or user security can be applied to LU6.1 links.

## Link security with LU6.1

Link security restricts the resources that a user can access, depending on the remote system from which they are accessed. The practical effect of link security is to prevent a remote user from attaching a transaction or accessing a resource for which the link userid has no authority.

Each link between systems is given an access authority defined by a link userid. A link userid for LU6.1 is a userid defined on your sessions definition for this connection. If not defined there, the link userid is taken to be the SECURITYNAME userid specified on the connection definition. If there is no SECURITYNAME, the link userid is the local region's default userid.

You cannot function ship to CICS without having a security check. However, the security check is minimized if the two regions involved are **equivalent systems**. This term means the same for LU6.1, LU6.2 and MRO: that the link userid matches the local region's userid.

If you have equivalent systems, the resource check is made against the local region's default user. If you do not have equivalent systems, the resource check is carried out against the link userid.

If a failure occurs in establishing link security, the link is given the security of the local region's default user. This would happen if, for example, the preset session userid had been revoked.

## Specifying ATTACHSEC with LU6.1

With LU6.1 links, information about the remote user is not available for security purposes. In this case, the authority of the user is taken to be that of the link itself, and you must rely on link security alone to protect your resources.

With LU6.1, you can specify only ATTACHSEC(LOCAL) in the CONNECTION definition. Figure 21 on page 194 shows an example of doing this using CEDA.

```
CEDA DEFINE CONNECTION(name)
  GROUP(groupname)
  .
  ATTACHSEC(LOCAL)
```

*Figure 21. Defining sign-on level for user security with LU6.1*

LOCAL is the default value. It specifies that a user identifier is not required from the remote system, and, if one is received, it is ignored. Here, CICS makes the user security profile equivalent to the link security profile. You do not need to specify RACF profiles for the remote users.

## Transaction, resource, and command security with LU6.1

As in a single-system environment, links must be authorized to:

- Attach a transaction
- Access all the resources that the transaction is programmed to use.

This results in security levels called **transaction security**, **resource security**, and **command security**.

### Transaction security

As in a single-system environment, the security requirements of a transaction are specified when the transaction is defined, as described in "Chapter 5. Transaction security" on page 79.

In an LU6.1 environment, a transaction can be initiated only if the link has sufficient authority.

### Resource and command security

Resource and command security in an intercommunication environment are handled in much the same way as in a single-system environment.

CICS performs resource and command security checking only if the installed TRANSACTION definition specifies that they are required; for example, on the CEDA DEFINE TRANSACTION command, as shown in Figure 22.

```
CEDA DEFINE TRANSACTION
  .
  RESSEC(YES)
  CMDSEC(YES)
  .
```

*Figure 22. Specifying resource and command security for transactions*

If a transaction definition specifies resource security checking, using RESSEC(YES), the link must have sufficient authority for the resources that the attached transaction accesses.

If a transaction definition specifies command security checking, using CMDSEC(YES), the link must have sufficient authority for the commands (COLLECT, DISCARD, INQUIRE, PERFORM, and SET) that the attached transaction issues.

For further guidance on specifying resource and command security, see "Chapter 6. Resource security" on page 85 and "Chapter 8. CICS command security" on page 109.

### NOTAUTH exceptional condition

If a transaction tries to access a resource, but fails the resource security checks, the NOTAUTH condition is raised.

When the transaction is the CICS mirror transaction, the NOTAUTH condition is returned to the requesting transaction, where it can be handled in the usual way.

## Function shipping security with LU6.1

When CICS receives a function-shipped request, the transaction that is invoked is the **mirror transaction**. The CICS-supplied definitions of the mirror transactions all specify resource security checking, but not command security checking. This means that you are prevented from accessing the remote resources if the link does not have the necessary authority.

Note that **transaction routing** across LU6.1 links is not supported.

If the CICS-supplied definitions of the mirror transactions are not what your security strategy needs, you can change them by copying the definitions in group DFHISC into your own group, changing them, and then reinstalling them. For more information, see "Category 2 transactions" on page 128.

If you include a remote resource in your resource definitions, you can arrange for security checking to be done locally, just as if the resource were a local one. Also, the system that owns the resource can be made to apply an independent check, if it is able to receive the user identifier. You can therefore choose to apply security restrictions on both sides, on either side, or not at all.

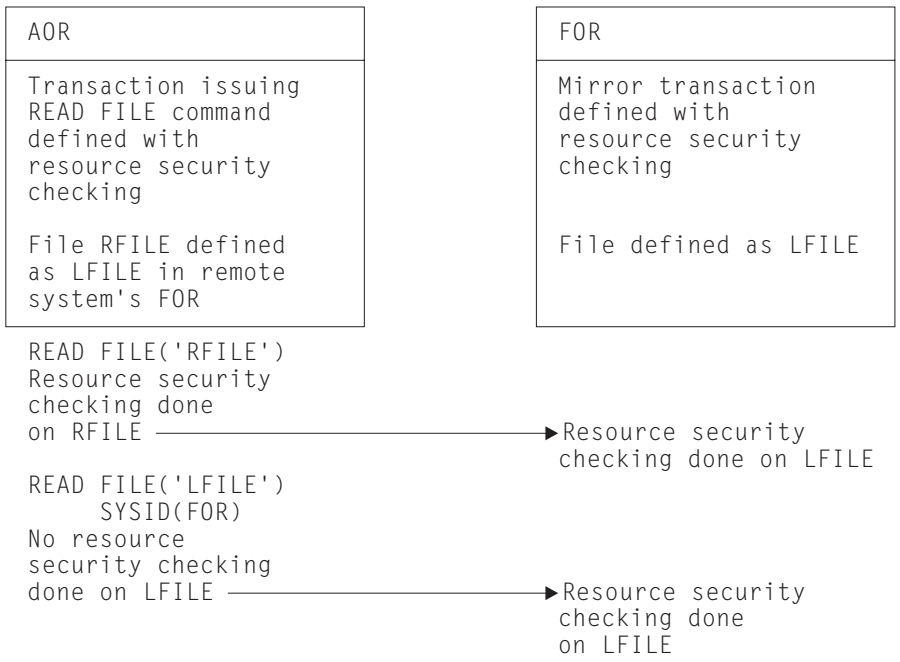



*Figure 23. Security checking done with and without SYSID*

For programming information on specifying the SYSID option, see the *CICS Application Programming Reference* manual.

## Security checking done in AOR with LU6.1

This section summarizes how security checking is done in the AOR according to how SECURITYNAME is specified in the AOR and TOR, in an LU6.1 environment.

The link userid referred to in Table 29 on page 197 is the one specified in the SECURITYNAME on the CONNECTION definition, or the USERID on the SESSIONS definition.

If a USERID is specified on the SESSIONS definition, and a link check is done, the userid used is the one on the SESSIONS definition.

Table 29 on page 197 shows how checking is done when ATTACHSEC(LOCAL) is specified.

**Neither the region userid for the TOR, nor the SECURITYNAME in the TOR's CONNECTION definition for the AOR, is relevant to security checking in the AOR.**

*Table 29. Security checking done in AOR*

| Region userid for AOR | SECURITYNAME in CONNECTION definition | USERID in SESSION definition | Checking in AOR |
|---|---|---|---|
| USERIDA | Not specified | Not specified | Check against AOR DFLTUSER |
| USERIDA | Not specified | USERIDA | Check against AOR DFLTUSER |
| USERIDA | Not specified | USERIDB | Check against USERIDB |
| USERIDA | USERIDA | Not specified | Check against AOR DFLTUSER |
| USERIDA | USERIDB | Not specified | Check against USERIDB |
| USERIDA | USERIDA | USERIDA | Check against AOR DFLTUSER |
| USERIDA | USERIDA | USERIDB | Check against USERIDB |
| USERIDA | USERIDB | USERIDA | Check against AOR DFLTUSER |
| USERIDA | USERIDB | USERIDB | Check against USERIDB |
| USERIDA | USERIDB | USERIDC | Check against USERIDC |

## Summary of resource definition options for LU6.1 security

The following is a summary of the resource definition options you need to define for LU6.1 security:

- On the CONNECTION definition:
  - ATTACHSEC, with the LOCAL option specified or allowed to default
  - SECURITYNAME
- On the SESSIONS definition:
  - USERID

For guidance on specifying CONNECTION and SESSION definitions, see the *CICS Resource Definition Guide*.

# Chapter 16. Implementing MRO security

This chapter tells you how to implement CICS multiregion operation (MRO) security, and is organized as follows:



## Security implications of choice of MRO access method

Either MVS cross-memory services or the CICS Type 3 SVC can be used for interregion communication (function shipping, transaction routing, distributed transaction processing, and asynchronous processing).

If you use cross-memory services, you lose the total separation between systems that is normally provided by separate address spaces.

The risk of accidental interference between two CICS address spaces connected by a cross-memory link is small. However, an application program in either system could access the other system's storage (subject to key-controlled protection) by using a sequence of cross-memory instructions.

If this situation would create a security exposure in your installation, use the CICS type 3 SVC for interregion communication, rather than MVS cross-memory services.

For information about how to specify the access method for MRO, see the *CICS Intercommunication Guide*.

## Bind-time security with MRO

The CICS interregion communication (IRC) facility supports MRO through the use of DFHAPPL.*applid* profiles in the FACILITY class.

There are two phases to bind security checking in DFHIRP, and these occur at:
- Logon time
- Connect time

These security checks, via RACROUTE calls to the SAF interface, are always performed, regardless of whether the or not MRO partner regions are running with external security active for CICS resource security checking (that is, for both SEC=YES and SEC=NO). In order for an MRO connection to be established

between two regions, both the logon and connect security checks in both systems must be completed successfully. This security is applied to earlier releases of CICS using the CICS/ESA 4.1 version of DFHIRP, the CICS interregion communication program.

## Logon security checking with MRO

Logon security checking is performed whenever a CICS region logs on to the CICS-supplied interregion communication (IRC) program, DFHIRP.

CICS interregion communication uses the external security manager to check that CICS regions logging on to IRC are the regions they claim to be.

Each region that uses the IRC access method must be authorized to RACF in a DFHAPPL.*applid* profile in the RACF FACILITY class. This requires the definition of a DFHAPPL.*applid* profile for each region that logs on to DFHIRP, and that each CICS region userid has UPDATE access to its own DFHAPPL.*applid* profile.

See Figure 24 for an illustration of logon checking.

### Connect security

To perform MRO connect security checking, DFHIRP checks that each CICS region in the connection has read access to its partner's DFHAPPL.*applid* profile.

When CICS Transaction Server for OS/390 Release 3 DFHIRP is installed, all regions using earlier CICS releases in the MVS image use the DFHAPPL.*applid* form of MRO connect security. In addition, the SECURITYNAME parameter on the CONNECTION definition is not used for MRO and is ignored.

To authorize the MRO partner regions for bind security purposes, you must define the appropriate DFHAPPL profiles in the RACF FACILITY class. This means that each CICS region in an MRO interregion communication link must be given access to its partner's DFHAPPL.*applid* profile with READ access authority. For example, for the CICS TOR running under userid CICSRTOR (with APPLID CICSATOR), that connects to the AOR running under userid CICSRAOR (with APPLID CICSAAOR), the RACF commands to authorize the connections are shown in Figure 24 on page 201.

You cannot specify to CICS whether or not you want connect security checking for MRO connections—CICS always issues the RACROUTE calls.

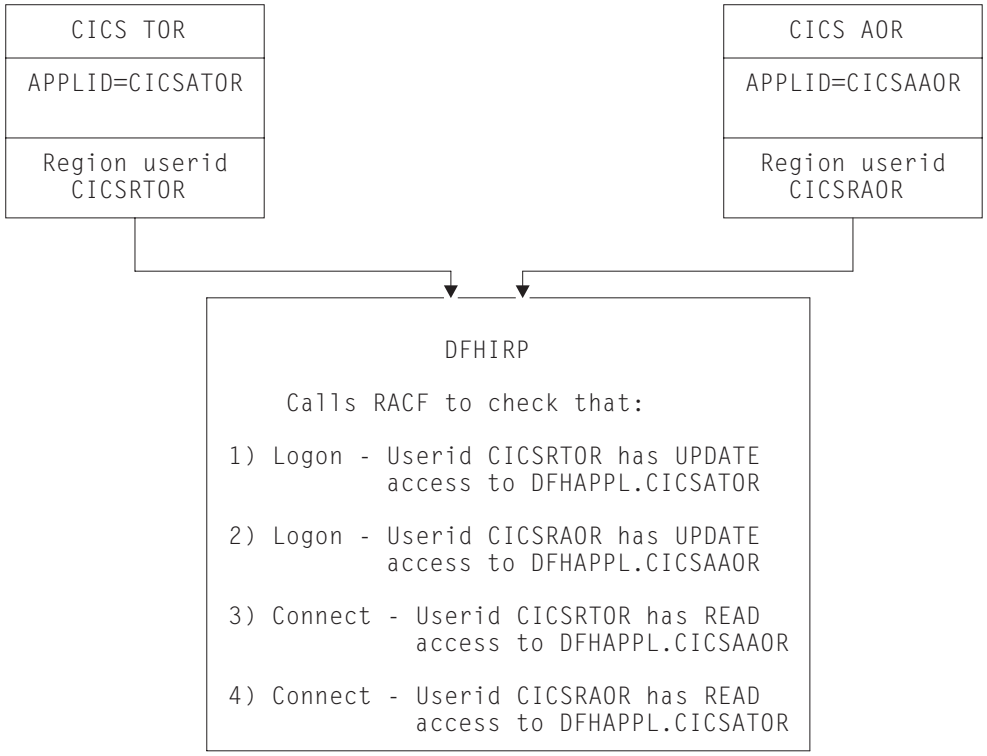



*Figure 24. Illustration of the DFHIRP logon and connect security checks*

The TOR and AOR shown in Figure 24, running under region userids CICSRTOR and CICSRAOR respectively, with APPLIDs CICSATOR and CICSAAOR, require the following RACF definitions to authorize their logon to DFHIRP:

- For the MRO logon and connect process:

```
RDEFINE FACILITY (DFHAPPL.CICSATOR) UACC(NONE)
RDEFINE FACILITY (DFHAPPL.CICSAAOR) UACC(NONE)

PERMIT DFHAPPL.CICSATOR  CLASS(FACILITY) ID(CICSRTOR) ACCESS(UPDATE)
PERMIT DFHAPPL.CICSAAOR  CLASS(FACILITY) ID(CICSRAOR) ACCESS(UPDATE)
```

- For connection:

```
PERMIT DFHAPPL.CICSAAOR  CLASS(FACILITY) ID(CICSRTOR) ACCESS(READ)
PERMIT DFHAPPL.CICSATOR  CLASS(FACILITY) ID(CICSRAOR) ACCESS(READ)
```

## Responses from the system authorization facility (SAF)

If the security profile for a specified resource is not retrieved, SAF neither grants nor refuses the access request. In this situation:

IRC rejects the logon or connect request if:

- A security manager was installed, but is either temporarily inactive or inoperative for the duration of the MVS image. This is a fail-safe action, on the grounds that, if the security manager was active, it might retrieve a profile that does not permit access.

IRC allows the logon or connect request if:

- There is no security manager installed, or

- There is an active security manager, but the FACILITY class is inactive, or there is no profile in the FACILITY class. The logon is allowed in this case because there is no evidence that you want to control access to the CICS APPLID.

Any CICS region without a specific DFHAPPL.*applid* profile, or applicable generic profile, permits all logon and connect requests. No messages are issued to indicate this. To avoid any potential security exposures, you can use generic profiles to protect all, or specific groups of, regions before, or in parallel with, security measures for specific regions. For example, specifying

```
RDEFINE FACILITY (DFHAPPL.*) UACC(NONE)
```

ensures that any region without a more specific profile is prevented from binding.

## Link security with MRO

Link security restricts the resources that a user can access, depending on the remote system from which they are accessed. The practical effect of link security is to prevent a remote user from attaching a transaction or accessing a resource for which the link userid has no authority.

Each link between systems is given an access authority defined by a link userid. A link userid for MRO is a userid defined on your sessions definition for this connection. Note that for MRO, unlike LU6.2, you can have only one sessions definition per connection, and there can be only one link userid per connection. If there is no preset session userid, the link userid is taken to be the region userid of the TOR region. The SECURITYNAME field on the connection definition is ignored for MRO.

You can never transaction route or function ship to CICS without having at least one security check, but the security checks done are minimized if the two regions involved are **equivalent systems**. This term means the same thing for LU6.1, LU6.2 and MRO: that the link userid matches the local region's userid.

If you have equivalent systems, you will always only have one security check. This will be made either against the local region's default user (for ATTACHSEC=LOCAL) or against the userid in the received FMH-5 attach request (ATTACHSEC=IDENTIFY).

If you do not have equivalent systems for ATTACHSEC=LOCAL, resource checks are done only against the link userid. For ATTACHSEC=IDENTIFY you will always have two resource checks. One check is against the link userid, and the other is against the userid received from the remote user in the attach request.

If a failure occurs in establishing link security, the link is given the same security authorization as defined for the local region's default user. This would happen, for example, if the preset session userid had been revoked.

Associate the SESSIONS definition with a RACF user profile that has access to any protected resource to which the inbound transaction needs access. See "Chapter 2. RACF facilities" on page 9 for guidance on defining profiles.

If the sign-on fails, a sign-on failure message is sent to the CSCS security destination, and the link is given the security of the DFLTUSER in the receiving system; that is, it is able to access only those resources to which the default user has access.

## Obtaining the CICS region userid

For the purposes of MRO logon and connect security checks, DFHIRP needs to know the CICS region userid under which the CICS job or task is running. DFHIRP obtains the CICS region's userid by issuing a RACROUTE REQUEST=EXTRACT macro.

If you are not using RACF as your external security manager, you must use the MVS security router exit, ICHRTX00, to customize the response from the RACROUTE REQUEST=EXTRACT macro.

CICS determines whether a security manager is present or not by examining the SAF response codes.

# User security with MRO

User security causes CICS to make a second check against a user signed on to a terminal, in addition to the link security check described in "Link security with MRO" on page 202. You should consider whether you want the extra level of security checking that user security provides.

You can specify either LOCAL, in which case the user is not checked, or IDENTIFY, in which case a userid is required, but no password is sent.

You specify the sign-on support for each connection using the ATTACHSEC operand of CONNECTION definition, as described in "User security in link definitions".

## User security in link definitions

The level of user security you require for a remote system is specified in the ATTACHSEC operand of the CONNECTION definition. Figure 25 shows an example of defining ATTACHSEC using CEDA.

CICS interprets the parameters of the ATTACHSEC operand as described here. However, special rules apply for CICS transaction routing using CRTE, as described in "CICS routing transaction, CRTE" on page 208.

```
CEDA DEFINE CONNECTION(name)
  GROUP(groupname)
  .
  ATTACHSEC(LOCAL|IDENTIFY)
```

*Figure 25. Defining sign-on level for user security*

The ATTACHSEC operand specifies the sign-on requirements for incoming requests. It has no effect on requests that are issued by your system to a remote system; these are dealt with by the remote system.

The following ATTACHSEC operands are valid with MRO:

**LOCAL**
specifies that a user identifier is not required from the remote system, and if one is received, it is ignored. Here, CICS makes the user security profile equivalent to the link security profile. You do not need to specify RACF profiles for the remote users. (LOCAL is the default value.)

Specify ATTACHSEC(LOCAL) if you think that the link security profile alone provides sufficient security for your system.

**IDENTIFY**
specifies that a user identifier is expected on every attach request. All remote users of a system must be identified to RACF.

Specify ATTACHSEC(IDENTIFY) when you know that CICS can trust the remote system to verify its users, when, for example, the remote system is another CICS.

The following rules apply to IDENTIFY:

- If a password is included in an attach request with a user identifier on a link with ATTACHSEC(IDENTIFY), CICS rejects the attach request and unbinds the session.
- If a null user identifier or an unknown user identifier is received, CICS rejects the attach request.
- If no user identifier is received, the attach is rejected unless USEDFLTUSER(YES) is specified on the connection. In this case CICS applies the security capabilities of the default user, as specified in the DFLTUSER system initialization parameter. For more information, see "CICS default user" on page 15, and "Attach-time security and the USEDFLTUSER option" on page 238.

**Note:** In the case of distributed transaction processing (DTP) transactions, you must issue a BUILD ATTACH request before the MRO SEND or CONVERSE command to include the userid of the terminal user in an attach request.

### Sign-on status

With ATTACHSEC(IDENTIFY), the remote user remains signed-on after the conversation associated with the first attach request is complete. CICS then accepts attach requests from the same user without a new sign-on until either of the following occurs:

- The period specified in the system initialization parameter USRDELAY elapses after completion of the last transaction associated with the attach request for this user.

  When you are running remote transactions, over ISC and IRC links, USRDELAY defines the length of time for which entries can remain signed onto the remote CICS region. For information on specifying USRDELAY, see the *CICS System Definition Guide*. For information on tuning, see the *CICS Performance Guide*
- The CICS system is terminated.

If you alter the RACF profile of a signed-on remote user (for example, by revoking the user), CICS continues to use the authorization established at the first attach request until the user is signed off by one of the events just described.

## Information about remote users

With MRO links, information about the user can be transmitted with the attach request from the remote system. This means that you can protect your resources not only on the basis of which remote system is making the request, but also on the basis of which actual user at the remote system is making the request.

This section describes some of the concepts associated with remote-user security, and how CICS sends and receives user information.

You will have to define your users to RACF. If a remote user is not defined to RACF, any attach requests from that remote user are rejected.

User profiles can be transmitted instead of, or in addition to, user identifiers. The profile name, if supplied, is treated as the groupid.

If the user has been added to the front-end system with a groupid explicitly specified, (for example in EXEC CICS SIGNON, or by filling in the GROUPID parameter on the on the CESN panel) this will be propogated by CICS in outbound attach FMHs for MRO links when ATTACHSEC(IDENTIFY) has been specified in the CONNECTION definition. If the groupid has been allowed to default at the time the user was originally added to the front-end system, the profile field will not be included in the outbound FMH5. If the groupid is passed to the backend system, the groupid will be used as part of ADD_USER processing on the backend. (The user ID must be defined as a member of the group passed in the ESM on the backend for the ADD_USER to be successful.)

CICS sends userids on ATTACHSEC(IDENTIFY) conversations. Table 30 shows how CICS decides which userid to send.

*Table 30. MRO attach-time user identifiers*

| Characteristics of the local task | User identifier sent by the TOR to the AOR |
|---|---|
| Task with associated terminal—user identifier | Terminal user identifier |
| Task with associated terminal—no user signed on and no USERID specified in the terminal definition | Default user identifier from the TOR |
| Task with no associated terminal or USERID, started by interval control START command (if using function shipping or DTP) | User identifier for the task that issued the START command |
| Task started with USERID option | User identifier specified on the START command |
| CICS internal system task | CICS region userid |
| Task with no associated terminal, started by transient data trigger | User identifier specified on the DCT that defines the queue |
| Task with associated terminal, started by transient data trigger | Terminal user identifier |
| Task started from PLTPI | User identifier specified by the PLTPIUSR system initialization parameter |

## New sign-on authorization processes

In earlier releases, CICS passes either its generic or specific APPLID to RACF when verifying a user's sign-on. This enables RACF, in addition to password checking, to check also that the user is authorized to signon to that CICS region.

This process is affected by the following:

1. When signing on users in the terminal-owning region, CICS passes to RACF one of the following names as the CICS APPL name:
   - The VTAM generic resources name if GRNAME is specified as a system initialization parameter

- The generic APPLID if one is specified on the APPLID system initialization parameter
- The specific APPLID if only one is specified on the system initialization parameter

The effect of this change is that you need define only one APPL profile name in the RACF database for all the CICS regions that are members of the same VTAM generic resources name. All sign-on verifications in a CICSplex, where all the terminal-owning regions have the same VTAM generic resources name, are made against the same APPL profile.

2. CICS passes the APPL name used in the sign-on process, and the NETNAME, across all MRO links (for example, from TOR to AOR, and from AOR to FOR). When signing-on the user in application-owning region and file-owning regions, where the connection definition specifies ATTACHSEC=IDENTIFY, CICS passes the terminal-owning region's APPLID and NETNAME to RACF.

   There are several benefits from this. It enables RACF 2.1 to reuse original terminal-owning region sign-on information, which is cached in VLF, when CICS is signing on the user in the application-owning region. This gives a significant improvement in performance. It also reduces the number of APPL profiles you need to maintain in the RACF database, saving on security administration. Finally, it prevents users signing on directly to an application-owning region, because the terminal-owning region APPLs are the only ones to which they are authorized.

## Transaction, resource, and command security with MRO

As in a single-system environment, users must be authorized to:

- Attach a transaction.
- Access all the resources that the transaction is programmed to use. This results in security levels called transaction security, resource security, and command security.

### Transaction security

As in a single-system environment, the security requirements of a transaction are specified when the transaction is defined, as described in "Chapter 5. Transaction security" on page 79.

In an MRO environment, two basic security requirements must be met before a transaction can be initiated:

- The link must have sufficient authority to initiate the transaction.
- The "user" who is making the request must have sufficient authority to access the system and to initiate the transaction.

### Resource and command security

Resource and command security in an intercommunication environment are handled in much the same way as in a single-system environment.

#### When resource and command security checking are performed

Resource and command security checking are performed only if the installed transaction definition specifies that they are required; for example, on the CEDA DEFINE TRANSACTION command, as shown in Figure 26 on page 207.

```
CEDA DEFINE TRANSACTION
  .
  RESSEC(YES)
  CMDSEC(YES)
  .
```

*Figure 26. Specifying resource and command security for transactions*

If a transaction specifies resource security checking, using RESSEC(YES), both the link and the user must also have sufficient authority for the resources that the attached transaction accesses.

If a transaction specifies command security checking, using CMDSEC(YES), both the link and the user must also have sufficient authority for the commands (shown in Table 11 on page 109) that the attached transaction issues.

For further guidance on specifying resource and command security, see "Chapter 6. Resource security" on page 85 and "Chapter 8. CICS command security" on page 109.

### NOTAUTH exceptional condition

If a transaction tries to access a resource, but fails the resource security checks, the NOTAUTH condition is raised.

When the transaction is the CICS mirror transaction, the NOTAUTH condition is returned to the requesting transaction, where it can be handled in the usual way.

## Transaction routing security with MRO

In transaction routing, the authority of a user to access a transaction can be tested in both the TOR and the AOR.

In the TOR, a normal test is made to ensure that the user has authority to access the transaction defined as remote, just as if it were a local transaction. This test determines whether the user is allowed to run the relay program.

In the AOR, the transaction has as its principal facility a remote terminal (the "surrogate" terminal) that represents the "real" terminal in the TOR. The way in which the remote terminal is defined (see the *CICS Intercommunication Guide*) affects the way in which user security is applied.

- If the definition of the remote terminal does not specify the USERID parameter:
  - For links with ATTACHSEC(IDENTIFY), the transaction security and resource security of the user are established when the remote user is signed on. The userid under which the user is signed on, whether explicitly or implicitly (in the DFLTUSER system initialization parameter), has this security capability assigned in the remote system.
  - For links with ATTACHSEC(LOCAL), transaction security, command security, and resource security are limited by the authority of the link.

In both cases, tests against the link security are made as described in "Link security with MRO" on page 202.

**Note:** During transaction routing, the 3-character operator identifier from the TOR is transferred to the surrogate terminal entry in the AOR. This identifier is not used for security purposes, but it may be referred to in messages and audit trails.

When transaction routing a PSB request, the following conditions must both be satisfied:

- ATTACHSEC on the connection definition must not be LOCAL (that is, it can be IDENTIFY, PERSISTENT, MIXIDPE, or VERIFY).
- PSBCHK=YES must be specified as a system initialization parameter in the remote system.

## Preset-security terminals and transaction routing

Preset-security for a terminal is determined by the specification of the USERID parameter.

When considering the security aspects of transaction routing from a preset-security terminal, remember that preset-security is an attribute of the terminal rather than of the user who is performing the transaction routing request.

During transaction routing, CICS creates a surrogate terminal in the AOR to represent the terminal at which the transaction routing request was issued. Whether the surrogate terminal has preset- security or not depends upon a number of factors:

- If a remote terminal definition exists in the AOR for the terminal at the TOR, and specifies the USERID parameter, the surrogate terminal is preset with this userid. If the USERID parameter is not coded, the surrogate terminal does not have preset-security.
- If a remote terminal definition does not exist in the AOR, the preset-security characteristics of the surrogate terminal are determined from the terminal definition shipped from the TOR. If the shipped terminal definition has preset security, the surrogate also has preset security, unless the connection to the AOR is defined with ATTACHSEC=LOCAL, in which case any preset security information shipped to the AOR is ignored.

## CICS routing transaction, CRTE

You can use the CICS routing transaction, CRTE, with MRO to run transactions that reside on a connected remote system, instead of defining these transactions as remote in the local system. CRTE is particularly useful for infrequently used transactions, or for transactions such as CEMT that reside on all systems.

Ensure that the terminal through which CRTE is invoked is defined on the remote system (or defined as “shippable” in the local system). The terminal operator needs RACF authority if the remote system is protected.

Security checking done in the AOR for transactions executed under CRTE does not depend on what is specified on ATTACHSEC, nor on the userid signed on in the TOR. Instead, security checking depends on whether the user signs on while using CRTE:

- If the user does **not** sign on, the surrogate terminal created is associated with the AOR default user. When a transaction is run, the security checks are carried out against this default user. A check is also done against the link userid to see whether the routing application itself has authority to access the resource.

- If the user **does** sign on, using the CESN transaction while running CRTE, the surrogate points to the userid of the signed-on user. For transactions attempting to access resources, security checking is done against the signed-on user's userid in the surrogate and the link userid.

For more information on CRTE, see the *CICS Supplied Transactions* manual and the *CICS Intercommunication Guide*.

## Function shipping security with MRO

When CICS receives a function-shipped request, the transaction that is invoked is the **mirror transaction**. The CICS-supplied definitions of the mirror transactions all specify resource security checking, but not command security checking. This means that you are prevented from accessing the remote resources if either the link or your user profile on the other system does not have the necessary authority.

If the CICS-supplied definitions of the mirror transactions are not what your security strategy needs, you can change them by copying the definitions in group DFHISC into your own group, changing them, and then reinstalling them. For more information, see "Category 2 transactions" on page 128.

If you include a remote resource in your resource definitions, you can arrange for security checking to be done locally, just as if the resource were a local one. Also, the system that owns the resource can be made to apply an independent check, if it is able to receive the user identifier. You can therefore choose to apply security restrictions on both sides, on either side, or not at all.

**Note:** If you specify the SYSID option on a function-shipped request, security checking is done in the remote system but is **bypassed in the local system**. Figure 27 summarizes what happens.

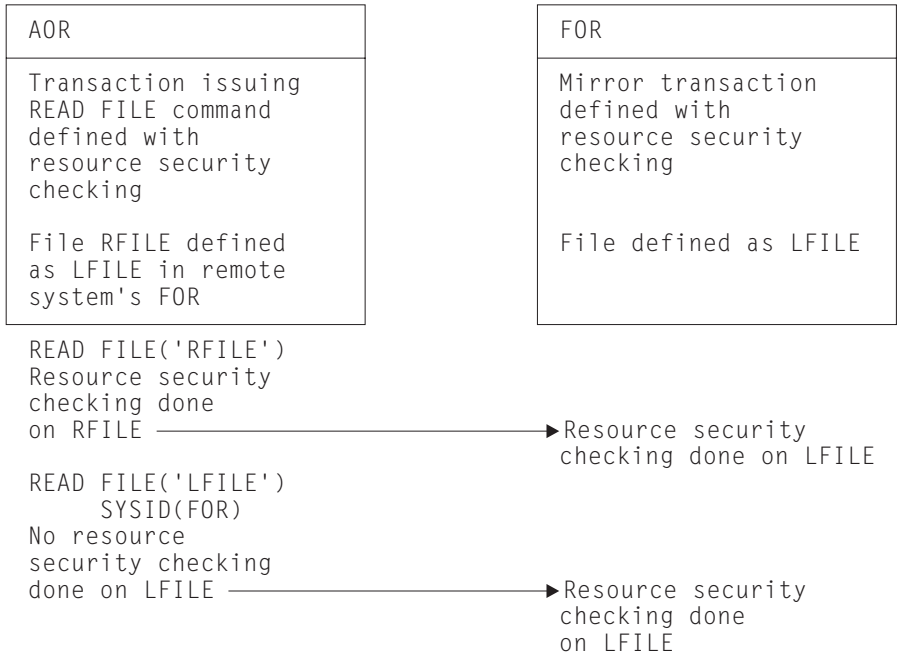



*Figure 27. Security checking done with and without SYSID*

For programming information on specifying the SYSID option, see the *CICS Application Programming Reference* manual.

## Distributed program link security with MRO

The CICS distributed program link (DPL) facility enables a program (the client program) to call a CICS program (the server program) in a remote CICS region. The client program may be a CICS program or a non-CICS program.

A CICS client program uses DPL by specifying the SYSID option on the EXEC CICS LINK PROGRAM command, or omitting the SYSID option if the REMOTESYSTEM option of the program resource definition already specifies a remote CICS region. When the SYSID option on the EXEC CICS LINK command specifies a remote CICS system, the client region does not perform any resource security checking, but leaves the resource check to be performed in the server region.

A non-CICS client program uses calls to DFHXCIS to open a line to the CICS system, and then to link to a CICS program. This is called the external CICS interface (EXCI). One of the parameters of the link call is the transaction identifier under which the server program is to run. Define this transaction to CICS as running program DFHMIRS and as using profile DFHCICSA. Another parameter of the link call is the client's userid, which is validated if the MRO connection has been defined with ATTACHSEC(IDENTIFY).

To use the userid parameter in the DFHXCIC call, the client program must have surrogate-user authority to the specified userid. This is described in more detail in the *CICS External Interfaces Guide* manual. For information about using the SURROGCHK parameter to specify surrogate user checking on DPL calls, see "Userid passed as parameter on EXCI calls" on page 106.

The client program receives a USER_ERROR error if the external CICS interface command fails the security check. However, this error can have other causes; each reason code value for a USER_ERROR response indicates whether the command can be reissued directly, or whether the pipe being used has to be closed and reopened first.

The server program is executed by a mirror transaction, in a similar way to other function-shipped CICS requests. However, the transaction name associated with the mirror depends on how the program link is invoked in the client region. You must be aware of the transaction name because normal attach security applies to the mirror transaction:

- If a transaction identifier is specified on the link request, the specified transaction name is used for the mirror.
- If the transaction is omitted from the link request, but the TRANSID option is used in the program resource definition in the client region, the name for the mirror is taken from the program's TRANSID specification.
- Otherwise, the default name of CSMI is used for the mirror transaction.

Authorize users to access the transaction name that the mirror runs under. The userids to be authorized depend on whether LOCAL or IDENTIFY attach security is being used, and are described in "Security checking done in AOR with MRO" on page 211. If you define the mirror transaction with RESSEC(YES) in the server region, authorize these userids to access the server program that is being linked to by the mirror. If the server program accesses any CICS resources, authorize the same userids to access them. If the server program invokes any SP-type commands, and the mirror transaction is defined with CMDSEC(YES) in the server region, authorize the same userids to access the commands.

If the mirror transaction cannot be attached because of security reasons, the NOTAUTH condition is not raised, but the TERMERR condition is returned to the issuing application in the client region. If the mirror transaction is successfully attached, but it is not authorized to link to the distributed program in the server region, the NOTAUTH condition is raised. The NOTAUTH condition is also raised if the server program fails to access any CICS resources for security reasons.

The server program is restricted to a DPL subset of the CICS API commands when running in a server region. The commands that are not supported include some that return security-related information. For programming information about which commands are restricted, see the *CICS Application Programming Reference* manual. For further information about DPL, refer to the *CICS Intercommunication Guide*.

## Security checking done in AOR with MRO

This section summarizes how security checking is done in the AOR.

The userid of the front-end CICS region is assigned as the default. However, if a USERID is specified on the SESSIONS definition, and a link check is done, the userid actually used is the one on the SESSIONS definition.

The region userid referred to in Table 31 through Table 32 is the USERID on the SESSIONS definition. The userid referred to in this case is the one under which the job is running. This userid is the one normally returned by the security manager domain.

### With ATTACHSEC(LOCAL) specified

Table 31 shows how checking is done in the AOR when ATTACHSEC(LOCAL) has been specified.

*Table 31. Security checking done in AOR—ATTACHSEC(LOCAL) specified*

| Region userid for AOR | Userid in session definition | Region userid for TOR | Checking in AOR |
|---|---|---|---|
| USERIDA | Not specified | USERIDA | Check against AOR DFLTUSER |
| USERIDA | USERIDA | Anything | Check against AOR DFLTUSER |
| USERIDA | Not specified | USERIDB | Check against USERIDB |
| USERIDA | USERIDB | Anything | Check against USERIDB |

### With ATTACHSEC(IDENTIFY) specified

Table 32 on page 212 shows how checking is done in the AOR when ATTACHSEC(IDENTIFY) has been specified.

*Table 32. Security checking done in AOR—ATTACHSEC(IDENTIFY) specified*

| Region userid for AOR | Userid in session definition | Region userid for TOR | Checking in AOR |
|---|---|---|---|
| USERIDA | Not specified | USERIDA | FMH-5 ATTACH check only |
| USERIDA | USERIDA | Anything | FMH-5 ATTACH check only |
| USERIDA | Not specified | USERIDB | FMH-5 ATTACH check and USERIDB |
| USERIDA | USERIDB | Anything | FMH-5 ATTACH check and USERIDB |

## Summary of resource definition options for MRO security

The following is a summary of the resource definition options you need to define for MRO security:

- On the CONNECTION definition:
  - ATTACHSEC, with either of the following options:
    - IDENTIFY
    - LOCAL
- On the SESSIONS definition:
  - USERID

For guidance on specifying CONNECTION and SESSION definitions, see the *CICS Resource Definition Guide*.

# Chapter 17. Security for data tables

This chapter describes how to provide security for CICS shared data tables and coupling facility data tables. It covers the following topics:

- Security for CICS shared data tables
- "Security for coupling facility data tables" on page 216

## Security for CICS shared data tables

To provide security for a shared data table when **cross-memory services** are used, ensure that:

- The file-owning region (FOR) that is acting as the shared data table server cannot be impersonated. See "SDT server authorization security check" on page 214 for details of how you ensure this.
- An application-owning region (AOR) cannot gain access to data that it is not meant to access. You can prevent this by checking at CONNECT time that the AOR is allowed access to the FOR and, if file security is in force, that the AOR is allowed access to the requested file.

These security checks are performed through the system authorization facility (SAF), to invoke RACF or an equivalent security manager.

**Note:** A region is still able to use data tables locally even if it does not have authority to act as a shared data table server.

The CICS shared data tables (SDT) facility reproduces the main characteristics of function-shipping security that operate at the region level, but note the following differences:

- SDT does not provide any mechanism for the FOR to perform security checks at the transaction level (there is no equivalent of ATTACHSEC(IDENTIFY) or ATTACHSEC(VERIFY)). Therefore, if you consider that the transaction-level checks performed by the AOR are inadequate for some files, ensure that those files are not associated with data tables in the FOR.
- SDT does not support any equivalent of preset security on SESSIONS, because no sessions are used.
- SDT does not pass any installation parameter list (INSTLN) information to the security user exits.

Security for CICS shared data tables is covered in the following topics:

- Security checking
- "SDT server authorization security check" on page 214
- "CONNECT security checks for AORs" on page 214.

### Security checking

You should consider the implications of the security checks before sharing a file that is associated with a data table.

SDT security makes use of existing CICS file security definitions, but it also relies on treating SDT server APPLIDs as protected resources. An SDT server's APPLID is represented by a DFHAPPL.*applid* profile in the RACF FACILITY resource class.

## SDT server authorization security check

When a region attempts to be an SDT server, it calls RACF to check whether its user ID has the required access authority to its APPLID. If the call fails, the region cannot initialize the required SDT support to be a server. This minimizes the risk that an AOR might accept **counterfeit data records** from an FOR that is not properly authorized to act as an SDT server. This check is never bypassed, even when SEC=NO is specified at system initialization.

To act as a server for a protected APPLID, an SDT FOR's userid must have UPDATE (or higher) access to its DFHAPPL.*applid* profile in the FACILITY class. In the following example definitions, the APPLID of the FOR is CICSHF01, and its user ID is CICSSDT1:

```
RDEFINE FACILITY (DFHAPPL.CICSHF01) UACC(NONE)

PERMIT DFHAPPL.CICSHF01 CLASS(FACILITY) ID(CICSSDT1) ACCESS(UPDATE)
```

The above example authorizes one FOR to act as a server with APPLID CICSHF01, running under user ID CICSSDT1. The following example shows how to authorize a group of FORs, with user IDs defined as members of group SDTGRP1, to act as SDT servers using a generic profile in the FACILITY class:

```
RDEFINE FACILITY (DFHAPPL.CICSTST*) UACC(READ)

PERMIT DFHAPPL.CICSTST* CLASS(FACILITY) ID(SDTGRP1) ACCESS(UPDATE)
```

### If SAF neither grants nor refuses an access request

If a security profile for a specified resource is not retrieved, SAF neither grants nor refuses the access request. In this situation:

- The request fails if a security manager is installed but is either temporarily inactive or inoperative for the duration of this MVS IPL. This decision is made on the grounds that had the security manager been active it might have retrieved a profile that refuses access.
- The request succeeds if:
  - There is no security manager at all.
  - There is an active security manager but the FACILITY class is undefined or inactive.
  - There is no profile covering the APPLID in question.

  The request is allowed in these cases because there is no evidence that you want to control access to the particular FOR APPLID.

## CONNECT security checks for AORs

The security checks performed at CONNECT time provide two levels of security:

- **Bind security** allows an FOR that runs without CICS file security to be able to restrict shared access to selected AORs. (Running without file security minimizes runtime overheads and the number of security definitions.)
- **File security** can be activated in the FOR if you want SDT to implement those checks that apply to the AOR as a whole.

Note that SDT provides no way of implementing those security checks that an FOR makes at the transaction level when ATTACHSEC(IDENTIFY) or ATTACHSEC(VERIFY) is used with function shipping.

## Bind security

To be allowed shared access to any of an FOR's data tables, an AOR's userid needs READ (or higher) access to the FOR's DFHAPPL.*applid* in the FACILITY class. This check is never bypassed, even when SEC=NO is specified at system initialization. In the following example definitions, three CICS AORs (userids is CICSAOR1, CICSAOR2, and CICSAOR3) all require SDT access to the FOR represented by the DFHAPPL.CICSHF01 profile:

```
PERMIT DFHAPPL.CICSHF01 CLASS(FACILITY) ID(CICSAOR1 CICSAOR2 CICSAOR3)
       ACCESS(UPDATE)
```

Cases when SAF neither grants nor refuses access are resolved in the same way as for server LOGON (see "If SAF neither grants nor refuses an access request" on page 214). If the result is a refusal, CICS does not permit shared access by the AOR to the FOR's APPLID.

Note that controlling SDT server authorization security and bind security by using different (but hierarchical) levels of access to the same resource has the following consequences:

- Any region with the same userid as a server can always bind to that server.
- It is impossible to control which userids can bind to a given APPLID without also controlling which userids can log on as servers for that APPLID.

SDT bind-time security uses different definitions from those employed by ISC and (if using preset sessions) MRO. Therefore, unless you make them consistent, SDT access might be granted when function shipping attempts are rejected, or vice versa. Both MRO and SDT use the same class and so, with ISC only, SDT CONNECT security might react to changes in security definitions either earlier or later than function shipping.

If file security is not in force in the FOR (that is, if SEC=NO or XFCT=NO was specified at system initialization), an AOR that is allowed to bind to an FOR is also allowed to access all that FOR's shared data tables.

If file security is in force, an AOR that is allowed to bind is still allowed free access if the userids of the AOR and FOR are the same (undefined userids are not considered to be the same).

## File security

After the bind-security check, and when file security is in force in the FOR, the FOR checks whether the AOR is authorized to"sign on" to the FOR. This security check is optional, and applies only when the userid of the AOR is different from that of the FOR. It is the equivalent of ATTACHSEC(LOCAL) in an MRO environment (see "User security with MRO" on page 203). The AOR also requires READ authorization to the file it is trying to access in the FOR.

To implement file security checking by the FOR:

- Initialize the FOR with system initialization parameter SEC=YES
- Authorize the AOR with READ access to the FOR's APPLID profile in the APPL general resource class
- Specify the appropriate value on the XFCT system initialization parameter
- Authorize the AOR's region user ID with READ access to the required files in the file resource profiles named on the XFCT system initialization parameter.

For example, define the APPL profile for an FOR with APPLID CICSHF01, and the PERMIT command to enable the AORs with user IDs CICSAOR1 and CICSAOR2 to sign on to CICSHF01, as follows:

```
RDEFINE APPL CICSHF01 UACC(NONE) NOTIFY(sys_admin_userid)
PERMIT CICSHF01 CLASS(APPL) ID(CICSAOR1 CICSAOR2) ACCESS(READ)
```

For information about authorizing access to files, see "Files" on page 91.

Cases when SAF neither grants nor refuses the request are resolved in the same way as for server LOGON (see "If SAF neither grants nor refuses an access request" on page 214).

If the userid is allowed to sign on to the FOR's application, the CONNECT request succeeds unless the AOR's userid is not allowed to read the specified file. Otherwise, the CONNECT request is treated in the same way as when the AOR's userid is undefined.

When file security is in force in an FOR, and the userid of the AOR is *undefined*, a CONNECT request fails unless the FOR's default userid (specified by the DFLTUSER system initialization parameter) is allowed to read the specified file.

Function shipping detects that an AOR's access to a file has been revoked when a rebuild of the file control resource class is completed in the FOR. However, if a valid connection already exists, SDT continues to allow access until something causes the connection to be broken. See "Refreshing resource profiles in main storage" on page 27.

**Caution:** If you use ISC instead of MRO for function shipping, ensure that the value of the SECURITYNAME parameter in the FOR is the same as the userid of the AOR. Otherwise, the SDT CONNECT and function shipping security checks will be inconsistent.

## Security for coupling facility data tables

CICS and MVS use RACF facilities to provide security for coupling facility data tables in the following areas:

1. Authorizing server access to a coupling facility list structure
2. Authorizing the server
3. Authorizing a CICS region's access to a coupling facility data table pool
4. Authorizing a CICS region to a CFDT
5. File resource security checking.

With the exception of items 4 and 5, which are optional, the other security checks are made automatically and are never bypassed. For items 2 and 3 in the above list, in cases when the system authorization facility (SAF) neither grants nor refuses access are resolved in the same way as the LOGON security check for CICS shared data table support (see "SDT server authorization security check" on page 214 for details).

An optional security check, which is controlled by server startup parameters, is provided for controlling access to specific tables within a coupling facility data table pool. This is described under "Authorizing a CICS region to a coupling facility data table" on page 217.

## Authorizing server access to a list structure

Each coupling facility data table server requires access to the coupling facility list structure that contains its pool of coupling facility data tables. To permit access, give the server region user ID ALTER access to a FACILITY class general resource profile called IXLSTR.*structure_name*. Structure names for coupling facility data tables take the form DFHCFLS_*poolname*.

For example, if coupling facility data tables are defined in a pool called PRODCFT1, the list structure for this pool is named DFHCFLS_PRODCFT1 in the CFRM policy. To access this list structure, the server user ID for pool PRODCFT1 requires ALTER access to the IXLSTR profile, defined as follows:

```
RDEFINE FACILITY IXLSTR.DFHCFLS_PRODCFT1 UACC(NONE)
PERMIT IXLSTR.DFHCFLS_PRODCFT1 CLASS(FACILITY) ID(server_userid) ACCESS(ALTER)
```

## Authorizing the server

When a CFDT server starts up for a given coupling facility data table pool CICS authorized cross-memory (AXM) services calls RACF to establish that it is authorized to act as a server for that pool. To authorize a coupling facility data table server to act as a server for its specified pool, give the server region user ID CONTROL access to a FACILITY class general resource profile called DFHCF.*poolname*.

For example, if the pool is PRODCFT1, define the profile and the required PERMIT statement as follows

```
RDEFINE FACILITY DFHCF.PRODCFT1 UACC(NONE)
PERMIT DFHCF.PRODCFT1 CLASS(FACILITY) ID(server_userid) ACCESS(CONTROL)
```

## Authorizing a CICS region to a CFDT pool

Each CICS region requires authorization to connect to a coupling facility data tablepool. To authorize a CICS region to connect to a server and its pool, give the CICS region UPDATE access to the server's FACILITY class profile for the pool.

For example, if the pool is PRODCFT1, define the required PERMIT statement as follows:

```
PERMIT DFHCF.PRODCFT1 CLASS(FACILITY) ID(CICS_region_userid) ACCESS(UPDATE)
```

## Authorizing a CICS region to a coupling facility data table

In addition to controlling a CICS region's access to a coupling facility data table pool, you can optionally control access to each CFDT in the pool. This security check, if active, is performed by the server each time a CICS region connects to a coupling facility data table for the first time. The resource security check is done as if for a CICS file owned by the coupling facility data table server region, using a profile defined in the general resource class specified on the SECURITYCLASS server initialization parameter. The default for this is the FCICSFCT class. For the profile name, use the table name as defined in the file resource definition.

You can optionally prefix the profile name using the server region user ID as the prefix by specifying SECURITYPREFIX=YES as a server initialization parameter. You can customize the prefix for this security check using the server initialization parameter SECURITYPREFIXID.

The coupling facility data table server performs the table security check by issuing a cross-memory mode FASTAUTH check, which requires the use of global in-storage security profiles. Access fails if a return code other than zero is received

by the server in response to the FASTAUTH check. If the external security manager does not support cross-memory mode FASTAUTH or global in-storage profiles, coupling facility data table security checks are not possible and an error message is issued at server initialization time if table security checking is specified. For information about all server initialization parameters that can be specified, see the *CICS System Definition Guide*

## File resource security checking

Normal CICS resource security for files is supported for coupling facility data tables. CICS performs the usual file resource security checks against signed-on users of transactions that access coupling facility data tables, using profiles defined in the general resource class named on the XFCT system initialization parameter.

See "Files" on page 91 for details of CICS file security.

# Part 4. Customization

This part discusses customizing the CICS-External Security Manager interface, in the following:.



This describes the CICS-RACF interface, and how customization can use the MVS router. There is also information about RACF user exits and security control points.

# Chapter 18. Customizing security processing

**Product-sensitive Programming Interface information**

This chapter introduces you to the CICS-RACF interface, and describes how the MVS router passes control to RACF. It describes how RACF exit programs can access CICS-related information. Finally, it lists the control points at which CICS invokes the external security manager (ESM). The chapter is organized as follows:

- Overview of the CICS-RACF interface
- "MVS router" on page 222
- "How ESM exit programs access CICS-related information" on page 222
- "CICS security control points" on page 223
- "Determining the userid of the CICS region" on page 225
- "Specifying user-defined resources to RACF" on page 226
- "How to bypass attach checks for non-terminal transactions" on page 228.

For programming information on customizing the CICS-ESM interface (using either RACF or a compatible user-written or vendor-supplied ESM), see the *CICS Customization Guide*.

## Overview of the CICS-RACF interface

In CICS Transaction Server for OS/390 Release 3, the only form of security CICS supports is that provided by an external security manager (ESM), such as RACF. CICS uses, by means of the RACROUTE macro, the MVS system authorization facility (SAF) interface to route authorization requests to RACF.

As shown in Figure 28, the RACROUTE macro invokes the MVS router, which invokes the RACF router, which calls the ESM (in this case, RACF).
See the *OS/390 Security Server External Security Interface (RACROUTE) Macro*

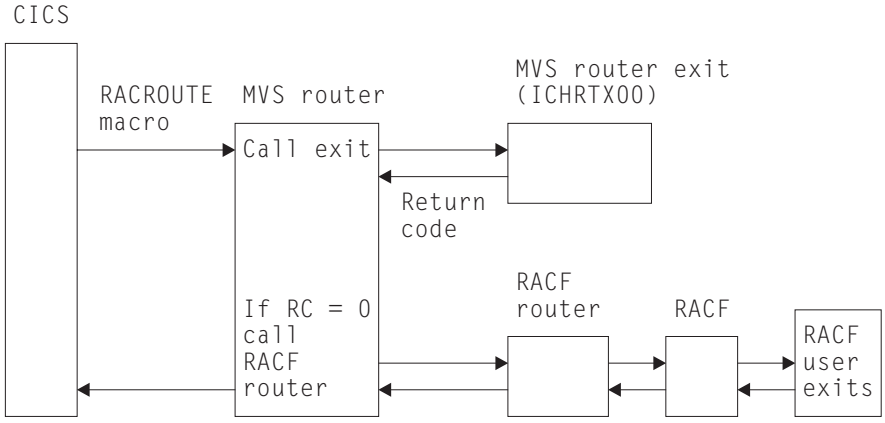



*Figure 28. MVS router exit*

*Reference* for information on how the RACROUTE macro is coded.

The control points at which CICS issues a RACROUTE macro to route authorization requests are described in "CICS security control points" on page 223.

## MVS router

The system authorization facility (SAF) provides your installation with centralized control over security processing by using a system service called the **MVS router**. The MVS router provides a common system interface for all products providing and requesting resource control. The resource-managing components and subsystems (such as CICS) call the MVS 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 use of common control functions shared across products and across systems.

If RACF is available in the system, the MVS router may pass control to the RACF router, which in turn invokes the appropriate RACF function. (The parameter information and the RACF router table, which associates router invocations with RACF functions, determine the appropriate function.) However, before calling the RACF router, the MVS router calls an optional installation-supplied security-processing exit, if one has been installed.

The system authorization facility 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, such as CICS, invoke RACF through the RACROUTE macro and SAF. Therefore, installations can modify RACF parameter lists and do customized security processing within the SAF router. For information about how to code a SAF router exit, see the *OS/390 Security Server (RACF) Messages and Codes*.

## How ESM exit programs access CICS-related information

When CICS invokes the ESM, it passes information about the current CICS environment, for use by an ESM exit program, in an **installation data parameter list**. How your exit programs access the installation data parameter list depends on the ESM you are using. The ICHxxxxx interfaces defined in Table 33 on page 223 apply only to RACF. For programming information on non-RACF interfaces, see the *CICS Customization Guide*.

### RACF user exit parameter list

If you write RACF user exits, you can find the address of the installation data parameter list directly from the RACF user exit parameter list. The name of the relevant field in the user exit parameter list varies according to the RACROUTE REQUEST type and the RACF user exit that is invoked. The relationships between REQUEST type, exit name, and field name are shown in Table 33 on page 223.

*Table 33. Obtaining the address of the installation data parameter list*

| RACROUTE REQUEST type | RACF exit | Exit list mapping macro | Parameter list field name (see Notes 1 and 2.) |
|---|---|---|---|
| VERIFY | ICHRIX01 | ICHRIXP | RIXINSTL |
| | ICHRIX02 | ICHRIXP | RIXINSTL |
| AUTH | ICHRCX01 | ICHRCXP | RCXINSTL |
| | ICHRCX02 | ICHRCXP | RCXINSTL |
| FASTAUTH | ICHRFX01 | ICHRFXP | RFXANSTL |
| | ICHRFX02 | ICHRFXP | RFXANSTL |
| LIST | ICHRLX01 | ICHRLX1P | RLX1INST |
| | ICHRLX02 | ICHRLX2P | RLX2PRPA |

**Notes:**

1. The 'xxxINSTL' field points to the installation parameter list only if you code ESMEXITS=INSTLN in the CICS system initialization parameters. The default value for this parameter is NOINSTLN, which means that no installation data is passed. (Note that ESMEXITS cannot be coded as a SIT override.)
2. RLX2PRPA contains the address of the ICHRLX01 user exit parameter list (RLX1P). Field RLX1INST of RLX1P points to the installation data parameter list.
3. There is no RACF user exit for REQUEST=EXTRACT, and no installation parameter data is passed. Any customization must be done using the MVS router exit, ICHRTX00.

For brief descriptions of RACF exits and their functions, see the *OS/390 Security Server (RACF) Security Administrator's Guide*. For full descriptions of the RACF exit parameter lists, see the *OS/390 Security Server (RACF) System Programmer's Guide*.

## Installation data parameter list

The installation data parameter list gives your ESM exit programs access to the following information:

- CICS security event being processed
- Details of the current CICS environment, as available
  - APPLID of the CICS region
  - Common work area
  - Transaction being invoked
  - Program being executed
  - CICS terminal identifier
  - VTAM LUname
  - Terminal user area
  - An 8-byte communication area, whose usage is described in the *CICS Customization Guide*.

For programming information about user-written ESMs, see the *CICS Customization Guide*.

# CICS security control points

This section summarizes the RACROUTE macros used by CICS to invoke the ESM, and the control points at which they are issued.

Some of these calls may not always be issued, because CICS reuses entries for users already signed on.

**RACROUTE**
This is the "front end" to the macros described below; it invokes the MVS router.

**RACROUTE REQUEST=VERIFY**
This macro is issued at operator sign-on (with the parameter ENVIR=CREATE), and at signoff (with the parameter ENVIR=DELETE). It creates or destroys an ACEE (access control environment element). It is issued at the following CICS control points (it is also issued (with the parameter ENVIR=VERIFY) early in normal sign-on through EXEC CICS SIGNON, but this call is ignored by RACF):

Each of the following control points relates to ENVIR=CREATE:
- Normal sign-on through EXEC CICS SIGNON
- Sign-on of the default userid DFLTUSER
- Sign-on of preset-security terminal
- Sign-on of MRO session
- Sign-on of LU6.1 session
- Sign-on of LU6.2 session
- Sign-on for XRF tracking of any of the above
- Sign-on associated with the userid on an attach request (for all operands of ATTACHSEC except LOCAL).

Each of the following control points relates to ENVIR=DELETE:
- Normal sign-off through EXEC CICS SIGNOFF
- Sign-off when deleting a terminal
- Sign-off when TIMEOUT expires
- Signoff when USRDELAY expires
- Sign-off of MRO session
- Sign-off of LU6.1 session
- Sign-off of LU6.2 session
- Sign-off for XRF tracking of any of the above.
- Sign-off associated with the userid on an attach request (for all operands of ATTACHSEC except LOCAL).

**RACROUTE REQUEST=VERIFYX**
This macro creates and deletes an ACEE in a single call. It is issued at the following control points:
- Sign-on, as an alternative to VERIFY, when an optimized sign-on is performed for subsequent attach sign-ons across an LU6.2 link with ATTACHSEC(VERIFY) or ATTACHSEC(PERSISTENT).
- When an invalid password or PassTicket is presented, or EXEC CICS VERIFY PASSWORD is issued.

**RACROUTE REQUEST=FASTAUTH**
This macro is issued during resource checking, on behalf of a user who is identified by an ACEE. It is the high-performance form of REQUEST=AUTH, using in-storage resource profiles, which does not cause auditing to be performed. It is issued at the following CICS control points:
- When attaching a local transaction
- When checking link security for transaction attach
- Transaction validation for an MRO task
- CICS resource checking
- Link security check for a CICS resource

- Transaction validation for EDF
- Transaction validation for the transaction being tested (by EDF)
- DBCTL PSB scheduling resource security check
- DBCTL PSB scheduling link security check
- Remote DL/I PSB scheduling resource check
- When checking a surrogate user authority
- QUERY SECURITY with the RESTYPE option.

**RACROUTE REQUEST=AUTH**

This macro provides a form of resource checking with a larger pathlength, and causes auditing to be performed. It is used as follows:

- After a call to FASTAUTH indicates an access failure that requires logging.
- When a QUERY SECURITY request with the RESCLASS option is used. This indicates a request for a resource for which CICS has not built in-storage profiles.

**RACROUTE REQUEST=LIST**

This macro is issued to create and delete the in-storage profile lists needed by REQUEST=FASTAUTH. (One REQUEST=LIST macro is required for each resource class.) It is issued at the following CICS control points:

- When CICS security is being initialized
- When an EXEC CICS PERFORM SECURITY REBUILD command is issued
- When XRF tracks either of these events.

**RACROUTE REQUEST=EXTRACT**

This macro is issued (with the parameters SEGMENT=CICS,CLASS=USER, with the parameters and with the SEGMENT=BASE,CLASS=USER to obtain the national language and user name) at all the following control points:

- Normal sign-on through EXEC CICS SIGNON
- Sign-on of the default userid DFLTUSER
- Sign-on of preset security terminal
- Sign-on of MRO session
- Sign-on of LU6.1 session
- Sign-on of LU6.2 session
- Sign-on for XRF tracking of any of the above
- Sign-on associated with the userid on an attach request (for all operands of ATTACHSEC except LOCAL).

It can be used to verify the user's password when an entry in the user table is reused within the USRDELAY period.

It is also issued (with the parameters SEGMENT=SESSION,CLASS=APPCLU) during verification of LU6.2 bind security, at the CICS control point for bind of an LU6.2 sessions.

**Note:** There is no RACF user exit for REQUEST=EXTRACT, and no installation parameter data is passed. Any customization must be done using the MVS router exit, ICHRTX00. For a detailed description of these macros, see the *OS/390 Security Server External Security Interface (RACROUTE) Macro Reference*.

## Determining the userid of the CICS region

CICS makes use of the userid of the region in which it runs for the following purposes:

- To prefix resource names if SECPRFX=YES is specified. For more information about the SECPRFX system initialization parameter, see "SECPRFX" on page 56.
- As the user to be checked for category 1 transactions. For more information, see "Category 1 transactions" on page 126.
- As the default PLTPI user for PLTPI non-terminal security, if a PLTPIUSR is not specified in the system initialization parameter.
- For SURROGAT checking (for example, authority to use the PLTPI and default userids).
- For MRO bind security. For more information, see "Chapter 16. Implementing MRO security" on page 199.

CICS obtains the region userid by invoking the external security manager, which extracts it from the RACF control blocks relevant for the job. The security domain and MRO-bind security each obtain the region userid by issuing a RACROUTE REQUEST=EXTRACT macro. To customize the response from this macro, and thus the security identification of a CICS region, use the MVS security router exit, ICHRTX00.

## Specifying user-defined resources to RACF

If you want to use the QUERY SECURITY command with the RESCLASS option, you may need to create user-defined resources within user-defined classes to represent the non-CICS resources that you want to query. To do this, add entries to the RACF class descriptor table (CDT) and to the RACF router table. Then, you must activate the new classes, define your resources in the new classes, and finally grant your users access to the resources. To improve the performance of QUERY SECURITY, also consider loading the new resource profiles into virtual storage.

### Adding new resource classes to the class descriptor table

The RACF class descriptor table has a system-defined part, and an installation-defined part named ICHRRCDE. You add new resource classes to ICHRRCDE by coding the ICHERCDE macro. For example, to add to the CDT a new class $FILEREC, and a corresponding (optional) group class $GILEREC, add the following macros to ICHRRCDE:

```
$FILEREC ICHERCDE CLASS=$FILEREC,      Entity or Member class          *
               GROUP=$GILEREC,                                         *
               ID=192,                                                 *
               MAXLNTH=17,                                             *
               RACLIST=ALLOWED,                                        *
               FIRST=ALPHANUM,                                         *
               OTHER=ANY,                                              *
               POSIT=42,                                               *
               OPER=NO,                                                *
               DFTUACC=NONE

$GILEREC ICHERCDE CLASS=$GILEREC,      Group class                     *
               MEMBER=$FILEREC,                                        *
               ID=191,                                                 *
               MAXLNTH=17,                                             *
               FIRST=ALPHANUM,                                         *
               OTHER=ANY,                                              *
               POSIT=42,                                               *
               OPER=NO,                                                *
               DFTUACC=NONE
```

Add the same classes to the RACF router table, ICHRFR01, by coding the ICHRFRTB macro:

```
ICHRFRTB CLASS=$FILEREC,ACTION=RACF
ICHRFRTB CLASS=$GILEREC,ACTION=RACF
```

Both the ICHERCDE and ICHRFRTB macros are described in the *OS/390 Security Server (RACF) Macros and Interfaces* manual.

When you have recreated the two modules ICHRRCDE and ICHRFR01, re-IPL your MVS system to bring them into use.

## Activating the user-defined resource classes

Once you have installed the new classes in your system, it is necessary to activate them in RACF before they can be used. This has to be done by a user with system-SPECIAL authority, who enters the following commands under TSO:

```
SETROPTS CLASSACT($FILEREC)
SETROPTS GENERIC($FILEREC)
```

To improve the performance of QUERY SECURITY, you should load the new resource profiles into virtual storage by using the RACLIST option. The RACLIST option is **required** if you are using the group class, because the connection between the group class and the entity class is resolved by RACLIST:

```
SETROPTS RACLIST($FILEREC)
```

You need to issue the SETROPTS commands for the entity class $FILEREC, because the group class $GILEREC has the same POSIT number.

## Defining resources within the new class

Resources within the new classes have to be defined by a user with system-SPECIAL authority, or with CLAUTH authority in the new class. CLAUTH authority is granted by issuing the following TSO command:

```
ALTUSER userid CLAUTH($FILEREC)
```

If you have the required authority, you can create the new resources by issuing the following TSO commands:

```
RDEFINE $FILEREC PAYFILE.SALARY   UACC(NONE)
RDEFINE $FILEREC PAYFILE.TAXBAND  UACC(NONE)
RDEFINE $GILEREC PERSONAL.DETAILS ADDMEM( PERSONAL.DEPT,     +
                                          PERSONAL.MANAGER,  +
                                          PERSONAL.PHONE)    +
                                  UACC(READ)
```

Now you are ready to authorize users to use the new resources. Assume that PAYROLL is the name of a group of users who are to be permitted to update all the pay and personal details fields in an employee record. The following TSO commands grant UPDATE access to all users in the group:

```
PERMIT PAYFILE.SALARY   CLASS($FILEREC) ID(PAYROLL) ACCESS(UPDATE)
PERMIT PAYFILE.TAXBAND  CLASS($FILEREC) ID(PAYROLL) ACCESS(UPDATE)
PERMIT PERSONAL.DETAILS CLASS($FILEREC) ID(PAYROLL) ACCESS(UPDATE)
```

If you had previously loaded the profiles by using the RACLIST option, refresh the profiles in virtual storage by issuing the command:

```
SETROPTS RACLIST($FILEREC) REFRESH
```

### Designing applications to use the user-defined resources

This topic gives an example of how you might design applications to make use of the user-defined resources.

Your applications use CICS file control in the normal way to read records from the pay and personal details file. Because you are controlling individual fields within each record, you may not need to apply resource security at the file level, so your transactions can be defined with RESSEC(NO). After reading the file record, but before displaying the results, you use QUERY SECURITY to determine whether the user has the authority to access the particular field within the record. For instance, before displaying the salary amount, you issue:

```
EXEC CICS QUERY SECURITY RESCLASS('$FILEREC')
                         RESID('PAYFILE.SALARY')
                         RESIDLENGTH(14)
                         READ(read_cvda)
```

Then, depending on the value returned in `read_cvda`, your application either displays the salary or a message stating that the user is not authorized to display it. Likewise, as part of a transaction that updates a person's telephone number, you issue:

```
EXEC CICS QUERY SECURITY RESCLASS('$FILEREC')
                         RESID('PERSONAL.PHONE')
                         RESIDLENGTH(14)
                         UPDATE(update_cvda)
```

If the value returned in `update_cvda` indicates that the user has UPDATE access, the transaction can continue and update the telephone number in the file. Otherwise, it should indicate that the user is not authorized to update the telephone number.

## How to bypass attach checks for non-terminal transactions

CICS always performs a transaction-attach security check for each transaction attach, even when the transaction has no associated terminal. Although this generally gives greater control over who can initiate transactions, it is different from the behavior of releases of CICS before CICS/ESA 4.1. The following suggests how you can bypass transaction-attach security checks for non-terminal transactions while continuing to keep full transaction-attach security for terminal-attached transactions.

CICS always performs the transaction-attach resource check using `RACROUTE REQUEST=FASTAUTH`, so you need only to provide an ICHRFX01 user exit. The

ICHRFX01 routine must issue a zero return code to indicate that the resource check processing is to continue, or a return code of 8 to indicate that the check is to be regarded as successful.

So that the ICHRFX01 exit can determine the circumstances under which it is called, specify ESMEXITS=INSTLN in the SIT for the CICS regions for which you want to control transaction-attach security. Then your ICHRFX01 routine should do the following:

1. Obtain the address of the CICS installation data parameter list, as described in "How ESM exit programs access CICS-related information" on page 222. If this address is zero, either the caller of the RACROUTE macro is not CICS, or it is a CICS region whose behavior you do not wish to modify; so exit with a return code of zero.
2. Use the DFHXSUXP macro to map the fields in the installation data parameter list.
3. Confirm that the installation data was created by CICS, by checking that UXPDFHXS is equal to 'DFHXS'. If it is not, exit with a return code of zero.
4. Examine field UXPPHASE in the installation data. If it is not equal to USER_ATTACH_CHECK (X'40'), this is not a transaction attach, so exit with a return code of zero.
5. Examine field UXPTERM in the installation data. If it is nonzero, this is a terminal-related transaction attach, so exit with a return code of zero.
6. If UXPPHASE is USER_ATTACH_CHECK and UXPTERM is zero, then a non-terminal transaction is being attached. Exit with a return code of 8 to indicate to RACF that this check is successful. The function RACROUTE REQUEST=FASTAUTH then completes with a return code of zero, and CICS continues with the attach of the non-terminal transaction.

## Global user exits in signon and signoff

CICS provides the XSNON global user exit in EXEC CICS SIGNON processing and the XSNOFF global user exit in EXEC CICS SIGNOFF processing. These exits do not allow you to affect the result of the sign-on or sign-off, but notify you when the userid associated with a terminal changes. The exits are further described in the *CICS Customization Guide*.

**End of Product-sensitive Programming Interface information**

# Part 5. Migration and coexistence

This part describes the security implications in migrating from earlier releases of CICS/ESA to CICS Transaction Server for OS/390 Release 3. It consists of two chapters:

- "Chapter 19. Migration considerations" on page 233 describes the migration implications of certain security-related features introduced in CICS/ESA 4.1 and earlier releases. It also covers the mixing of internal and external security in an MRO environment, the use of preset-security terminals, and the CESN transaction.
- "Chapter 20. Coexistence with previous CICS releases" on page 241 describes various aspects of coexistence from a security viewpoint, including MRO, system initialization parameters, transaction resource definitions, and timeout values.

# Chapter 19. Migration considerations

This chapter considers the security aspects of migrating from earlier releases of CICS to CICS Transaction Server for OS/390 Release 3. Note that CICS/ESA 3.3 security is essentially the same as CICS/ESA 3.2.1 security:



## UPDATE access authority in CICS/ESA 3.1.1

In CICS releases before CICS/ESA Version 3, the only access authority recognized by CICS is READ. These CICS releases do not distinguish levels of access authority (for example, between READ and UPDATE authority).

CICS differentiates between requests to read data those that attempt to update data. It conveys the application program's intent when issuing RACROUTE authorization requests, so that RACF can provide the required response to either type of access intent for any particular terminal user. In releases of CICS earlier than 3.1.1., a user with READ access to a CICS filename can perform update as well as read activity on the designated file. The authorization is essentially only a "use" or "not use" distinction.

You can alter existing PERMIT commands to specify ACCESS(UPDATE) as appropriate. When these altered profiles are used with earlier CICS releases, access is granted when either READ or UPDATE authority is in effect for access to the resource described by the profile. This enhancement, introduced in CICS/ESA 3.1.1, to correctly map the access intent within the CICS application to the RACROUTE REQUEST=FASTAUTH request issued by CICS, is not dependent on any RACF release. The enhancement is in effect when CICS/ESA 3.1.1 is running with EXTSEC=(YES,,UPDATE) specified as a system initialization parameter.

In CICS/ESA 3.2.1 and CICS/ESA 3.3, specifying SEC=MIGRATE causes CICS to ignore any distinction between READ and UPDATE access intent. SEC=MIGRATE is no longer supported in CICS. For more information, see "SEC" on page 56.

For releases before CICS/ESA 3.1.1, there are alternatives that can provide the level of application control required even though CICS does not distinguish READ from UPDATE. If an update function within an application is performed either by a program or under a transaction code that is not required for the read-only

function, program security or transaction security can be used to determine whether a given terminal user can perform an update function in the earlier CICS releases.

One other possibility that involves modification of the application, but one that is potentially very simple, is to define an alias transaction code to be used when the update function is to be performed (assuming that a single program performs both the inquiry and update processing). The only change required within the application program itself is to add a test before performing an update to ensure that EIBTRNID contains the transaction code intended to permit updating. When the applications are driven through menu panels it is frequently possible to make the introduction of this alias transaction code completely transparent to the users of the application.

## Removal of internal security in CICS/ESA 3.2.1

In CICS releases before CICS/ESA 3.2.1, for any resource type for which RACF authorization was not requested (for example, XFCT=NO in the system initialization parameters), CICS reverted to the CICS internal resource security level checking mechanism, as though RSLC=YES had been specified for the transaction.

This mechanism is based on the RSLKEY values defined for the terminal user in the sign-on table and the RSL value specified in the relevant resource control table entry. Unless the CICS resource definition, for a filename for example, has RSL(PUBLIC) specified or the RSL(*nn*) matches one of the RSLKEYs associated with the signed-on terminal user, then the NOTAUTH condition is raised following the execution of an EXEC CICS READ FILE(*filename*) ... command. For all resource definitions for which the X*name* class is NO, and which may be accessed by any transaction for which RSLC=EXTERNAL is specified, specify either RSL=PUBLIC or RSL=*n*, where *n* is a security key other than 0.

## Removal of internal LU6.2 bind time security

You do not use BINDPASSWORD in a CSD CONNECTION definition for LU6.2 bind time security validation. Instead, create RACF APPCLU profiles, and specify XAPPC=YES on the SIT to maintain validated links.

## Use of CICS segment in RACF user profiles in CICS Transaction Server for OS/390 Release 3

If you are migrating to this release of CICS from a release earlier than CICS/ESA 3.2.1, you can use the DFHSNMIG migrate utility to migrate your existing sign-on table (SNT) to the CICS segment of RACF user profiles. See "Sign-on table migration utility" and the *CICS Operations and Utilities Guide* for information about the DFHSNMIG utility.

The CICS segment of the user profile contains data for CICS terminal users. In earlier releases of CICS, you provided this information in a CICS SNT. For information on the order in which CICS searches for the operator information, see "Obtaining CICS-related data for the default user" on page 74.

### Sign-on table migration utility

The sign-on table migration utility, DFHSNMIG, is provided to help you migrate CICS terminal-user data from an SNT to the CICS segment of a RACF user's

profile. For each user entry in the SNT it creates a CLIST of RACF commands, generating either an ADDUSER or an ALTUSER command as appropriate for each SNT user entry. Because the DFHSNT macro is no longer supplied, assemble the SNTs assembled using the pre-CICS 4.1 DFHSNT macro.

DFHSNMIG can be found as an APF-authorized program in CICSTS13.CICS.SDFHAUTH, and must be run from an APF-authorized library. If you invoke the program from TSO, add its name to the list of authorized program names in the AUTHPGM NAMES section in the IKJTSOxx member of SYS1.PARMLIB.

The DFHSNMIG utility creates a CLIST of ADDUSER and ALTUSER commands to define CICS users to RACF. These commands do not specify the default RACF group each user should belong to. You might want to edit the CLIST created by DFHSNMIG to add DFLTGRP information. See "Defining terminal users and user groups to RACF" on page 75 for an example of specifying DFLTGRP on the ADDUSER command.

Figure 29 shows an example sign-on table entry. In this example, `OLDUSER` is an existing RACF-defined userid, and `NEWUSER` is a userid that has not previously been defined to RACF. `DFHSNT TYPE=(ENTRY,DEFAULT)` is a default entry, for which DFHSNMIG will *not* create an entry.

```
SNT     DFHSNT TYPE=INITIAL
*
        DFHSNT TYPE=ENTRY,                                              *
               USERID=OLDUSER,                                          *
               OPIDENT=OLD,                                             *
               OPPRTY=255,                                              *
               OPCLASS=(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,   *
               19,20,21,22,23,24),                                      *
               NATLANG=K,                                               *
               XRFSOFF=FORCE
*
        DFHSNT TYPE=ENTRY,                                              *
               USERID=NEWUSER,                                          *
               OPIDENT=NEW,                                             *
               OPPRTY=100,                                              *
               TIMEOUT=20,                                              *
               OPCLASS=(10)
*
        DFHSNT TYPE=(ENTRY,DEFAULT),                                    *
               OPIDENT=XXX,                                             *
               TIMEOUT=10
*
        DFHSNT TYPE=FINAL
         END
```

*Figure 29. Sample sign-on table entry*

Figure 30 on page 236 shows an example of output from DFHSNMIG, which has changed the SNT shown in Figure 29 into entries for the RACF database. For more information about running DFHSNMIG, see the *CICS Operations and Utilities Guide*.

```
/*-------------------------------------------------------*/
/*                                                       */
/* Migration of DFHSNT. (Created by DFHSNMIG utility.)   */
/* This CLIST will add CICS attributes into your RACF    */
/* database.  Please note that keywords are for RACF 1.9 */
/* and will not work against earlier versions of RACF.   */
/*                                                       */
/* You may need to edit this file before executing the   */
/* CLIST under a TSO userid that has SPECIAL authority.  */
/*                                                       */
/* ADDUSER: Asks RACF to create a new entry for the user.*/
/* ALTUSER: Adds CICS attributes to an existing RACF user.*/
/*                                                       */
/* Userid   - Identifier for user.                       */
/* LANGUAGE - Preferred language:  ENU = English (US)    */
/*                                 JPN = Japanese        */
/* The CICS attributes are:                              */
/* OPCLASS  - Operator Class                             */
/* OPIDENT  - Operator Identifier                        */
/* OPPRTY   - Operator Priority                          */
/* TIMEOUT  - Timeout Value                              */
/* XRFSOFF  - FORCE or NOFORCE                           */
/*                                                       */
/*-------------------------------------------------------*/
/*-------------------------------------------------------*/
/* Details for                                 OLDUSER   */
/*-------------------------------------------------------*/
ALTUSER OLDUSER                                          +
  LANGUAGE(PRIMARY(JPN))                                 +
  CICS(                                                  +
     OPCLASS(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,  +
      19,20,21,22,23,24)                                 +
     OPIDENT(OLD)                                        +
     OPPRTY(255)                                         +
     TIMEOUT(0)                                          +
     XRFSOFF(FORCE)                                      +
  )
/*-------------------------------------------------------*/
/* Details for                                 NEWUSER   */
/*-------------------------------------------------------*/
ADDUSER NEWUSER                                          +
  CICS(                                                  +
     OPCLASS(1,10)                                       +
     OPIDENT(NEW)                                        +
     OPPRTY(100)                                         +
     TIMEOUT(20)                                         +
     XRFSOFF(NOFORCE)                                    +
  )
/*-------------------------------------------------------*/
/* 00000002 entries successfully processed.              */
/*-------------------------------------------------------*/
```

*Figure 30. Example of output from DFHSNMIG*

## Goodnight transaction

By specifying your own GNTRAN transaction, you can use the CICS API to control the TIMEOUT operation. For example, your transaction could display a screen that prompts for the password. Specifying the EXEC CICS ASSIGN USERID request obtains the userid, and EXEC CICS VERIFY PASSWORD( ) USERID ( ) would validate the input. Based on the response, the user could remain signed on or be signed off.

By default CICS uses the CESF transaction to sign-off a user terminal. The goodnight transaction is not available for a surrogate terminal that is timed out during a CRTE session. Sign-off occurs with a loss of the security capabilities the terminal previously had, leaving a DFHSN1200 message in the log.

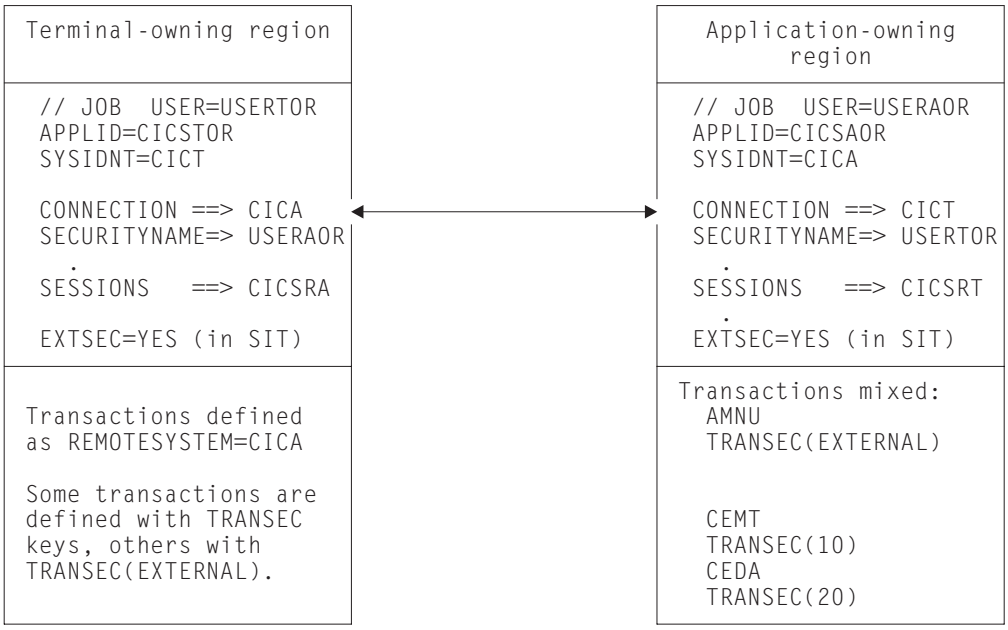
## Migrating to RACF on CICS Version 2

In preparation for migrating to CICS TS Release 3 from CICS MVS Version 2, consider migrating to RACF on your existing release as the first step. When you do this, consider the following:

- Can you convert a region in its entirety to RACF, or, because of the size of the conversion task, will you have to convert regions progressively, and run with some internal and some external security?
- Are you using MRO? If so, you can mix internal and external security while you gradually complete the migration process.

Internal and external security can be mixed in the same region, but there are some pitfalls to be avoided. Avoid trying to use the CICS-supplied CEDF transaction specified with a different form of security from that of a transaction running under CEDF. For example, if you try to run CEDF with external security to debug transactions protected with CICS internal security, CEDF abends with abend code AED3.

### Mixing internal and external security in an MRO environment




**Note:** The SESSIONS definition installed in CICA no longer specifies security keys, so there is no preset security on the link between the TOR and AOR. (Preset security with an external security manager is not supported in releases of CICS Version 2.)

## Installing preset-security terminals

In CICS Version 3, the authority to install terminals with preset userids took the form of ALTER authority to the TERMINAL resource in the CCICSCMD resource class. In CICS (CICS Transaction Server for OS/390) change these RACF definitions to use READ authority to the *userid*.DFHINSTL resource in the SURROGAT resource class.

To define a surrogate user to RACF:

1. Define a resource in the SURROGAT resource class for each transaction user. The name of the resource is:
   - *userid*.DFHSTART for authority to start a transaction with the START command.
   - *userid*.DFHINSTL for authority to:
     - Install terminals with a preset userid
     - Start TD trigger-level transactions
     - Run PLTPI transactions at CICS startup.
2. Add the surrogate user to the access lists for those resources, with READ authority.

For more information about surrogate users defined to RACF, see "Chapter 7. Surrogate user security" on page 103.

## Signing off with CESN

In CICS Version 3, CESN signed off any signed-on user as soon as the transaction identifier was entered, and then presented a sign-on panel. For example, you were signed off by first entering CESN, and then pressing F3 when the signon panel is displayed. CICS Transaction Server for OS/390 Release 3, however, does not always sign off a user when CESN is entered.

In CICS Version 3, CESN signs off any signed-on user only when a new signon attempt is made. Alternatively, the signed-on user is signed off if the CESN transaction identifier is entered with operands (for example USERID=*userid*).

## APPC password expiry management

When you successfully verify your password with CICS Transaction Server for OS/390 Release 3, you are **not** signed on in the target CICS system.

If your CICS connection specifies persistent verification, a successful password verification will cause you to be added to the LUIT table. If no other attaches are received, you will receive a CLS3 transaction flow after the PVDELAY interval.

## Attach-time security and the USEDFLTUSER option

In CICS Version 2 and Version 3, a remote user who was not signed on would not have an associated userid. This caused an LU6.2 protocol violation. In order to ignore this violation. In CICS Transaction Server for OS/390 Release 3, coding USEDFLTUSER on the connection indicates that the default user can be used. The following types of incoming attach FMH-5 are accepted by CICS Transaction Server for OS/390 Release 3 only if the USEDFLTUSER option is coded on the connection:

- An FMH-5 with an ATTACHSEC of IDENTIFY not containing a userid subfield, for example, from a CICS/MVS or a CICS/VSE® system

- An FMH-5 with an ATTACHSEC of VERIFY containing userid and password subfields that have zero-length, for example, from certain non-EBCDIC based systems

If you do not specify the USEDFLTUSER option in these exceptions, the expected protocol violation occurs, a message is generated, and the attach fails.

## Transaction-attach security for non-terminal transactions

In CICS Version 2 and Version 3 transaction-attach security for non-terminal transactions was not required. In CICS Transaction Server for OS/390 Release 3 when transaction security is active (SEC=YES and XTRAN is not NO) CICS **always** checks the authority of **any** userid to attach a transaction.

In particular, when transactions are started by the EXEC CICS START TRANSID command (with neither a TERMID nor a USERID operand) they inherit the userid associated with the starting transaction, and the inherited userid must be authorized to attach the started transaction.

Therefore you may need to authorize your users to attach transactions that they previously had authority only to start. Alternatively, you can customize RACF so that the transaction-attach check is always successful for non-terminal transactions, as described in "How to bypass attach checks for non-terminal transactions" on page 228.

# Chapter 20. Coexistence with previous CICS releases

This chapter covers the following topics:



## Coexistence overview

This chapter discusses differences in how you implement RACF security on releases of CICS earlier than CICS Transaction Server for OS/390 Release 3. You can use RACF with the following earlier releases:

- **CICS/OS/VS Version 1 Release 7 in an MVS/370 or MVS/XA environment**: This release of CICS supports RACF 1.6 and later, upward compatible, releases. See the *CICS/OS/VS Installation and Operations Guide* for information about using RACF with CICS OS/VS 1.7.
- **CICS/MVS 2.1 in an MVS/XA™ or MVS/ESA environment**: This release of CICS supports RACF 1.6 and later, upward compatible, releases. See the *CICS/MVS Operations Guide* for information about using RACF with CICS/MVS 2.1.
- **CICS/ESA Version 3 Release 1 in an MVS/ESA environment**: This release of CICS supports RACF 1.8.1 and later, upward compatible, releases. For information about using RACF with CICS/ESA 3.1.1, see the *CICS System Definition Guide*.
- **CICS/ESA 3.2.1 in an MVS/ESA environment**: This release of CICS supports RACF 1.8.1 and later, upward compatible, releases, but exploits some functions specific to RACF 1.9.
- **CICS/ESA 3.3 in an MVS/ESA environment**: This release of CICS supports RACF 1.8.1 and later, upward compatible, releases, but exploits some functions specific to RACF 1.9.
- **CICS/ESA 4.1 in an MVS/ESA environment**: This release of CICS supports RACF 1.9 and later, upward compatible, releases, but exploits some functions specific to RACF 2.1.
- **CICS Transaction Server Releases 1 and 2**: These releases of CICS support RACF 2.1 and later, upward compatible, releases, but exploits some functions specific to RACF 2.2.

The listed releases of CICS before CICS/ESA 4.1, together with the stated releases of RACF, support all the security functions described in the earlier chapters of this book for CICS Transaction Server for OS/390 Release 3, except as shown in Table 34 on page 242.

*Table 34. CICS releases in which some security-related functions are not available*

| Function | Releases in which function is not available |
|---|---|
| Default userid | **CICS OS/VS 1.7, CICS/MVS 2.1, and CICS/ESA 3.1.1** You cannot use RACF to define a CICS default userid. The security attributes for unsigned-on users are predetermined by CICS. |
| CICS segment | **CICS OS/VS 1.7, CICS/MVS 2.1, and CICS/ESA 3.1.1** You cannot use RACF to define CICS terminal operator data. To specify operator characteristics, define appropriate entries in the CICS sign-on table (SNT). |
| Command security | **CICS OS/VS 1.7 and CICS/MVS 2.1** You cannot use RACF to perform CICS command security checking |
| LU 6.2 session security | **CICS OS/VS 1.7, CICS/MVS 2.1, and CICS/ESA 3.1.1** You cannot use RACF to control LU 6.2 session security. Use the CICS internal mechanisms by specifying a bind password on the appropriate connection definition. |
| Preset security | **CICS OS/VS 1.7, CICS/MVS 2.1, and CICS/ESA 3.1.1** You cannot use RACF for preset security on terminals and sessions. Use the CICS internal mechanisms by specifying the appropriate security keys for the terminals and sessions. |
| QUERY SECURITY | **CICS OS/VS 1.7 and CICS/MVS 2.1** You cannot query the user's security access to RACF-protected resources using this CICS API command. This command, introduced in CICS/ESA 3.1.1 for CICS-managed resources, was extended in CICS/ESA 3.2.1 to permit user applications to query user-defined resources. |
| SECURITYNAME | **CICS/ESA 4.1** You cannot use this in MRO CONNECTION definitions. |
| BINDPASSWORD | **CICS/ESA 4.1** You cannot use this in CONNECTION definitions. |
| Automatic resolution of resource profiles refreshed by SETROPTS RACLIST | **Releases earlier than CICS/ESA 4.1**. |
| PassTicket support | You cannot generate PassTickets with releases earlier than CICS/ESA 4.1, but you can **use** them in sign-on in any release. |
| Reuse of RACF user profiles in VLF between MRO regions | **Releases earlier than CICS/ESA 4.1** Only available with RACF 2.1. and later releases |
| Surrogate user support | **Releases earlier than CICS/ESA 4.1** |
| VERIFY CHANGE PASSWORD | **Releases earlier than CICS/ESA 4.1** |
| USEDFLTUSER | **Releases earlier than CICS/ESA 4.1** |

In addition to the functions not supported, there are differences in the way you specify RACF security on CICS resource definitions, and on system initialization parameters. This chapter explains these differences, and how your CICS regions can coexist in the RACF security environment with earlier releases.

## System initialization parameters

If you are using a pre-CICS/ESA 3.2.1 version of CICS, use the system initialization parameter EXTSEC to specify that you want to use RACF. This parameter is described in the following sections for the earlier releases. The security parameters introduced in CICS/ESA 3.2.1 (SEC, SECPRFX, DFLTUSER, ESMEXITS, XAPPC), described in "Chapter 3. CICS data set and system security" on page 37, should not present any coexistence difficulties.

## Transaction resource definitions

You might find it necessary to have earlier releases of CICS coexisting with this release in the same MVS image, possibly because you cannot migrate all of your CICS regions to in their entirety. If so, you might want to share resource definitions from the same CSD. How you can do this is discussed in the following section, which compares the various resource definition parameters. Some of the obvious differences are illustrated by the CEDA display panels shown in the figures that follow.

Note in Figure 31 that the internal security attributes of earlier releases are shown, even though internal security is not supported in CICS/ESA 3.2.1 and later releases. This is to enable you to share the CSD between different releases of CICS, and continue to maintain the resource definitions for the earlier releases. To update any resource definitions being shared between releases, use the CICS ALTER function.

```
OBJECT CHARACTERISTICS                              CICS RELEASE = 0410
 CEDA  VIew
```

```
  SECURITY
   RESSec         : No                No | Yes
   Cmdsec         : No                No | Yes
   Extsec         : No                No | Yes
   TRANsec        : 01                1-64
   RSl            : 00                0-24 | Public
                                                        APPLID=CICSTOR
PF 1 HELP 2 COM 3 END      6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

*Figure 31. View of part of a CICS Transaction Server for OS/390 Release 3 transaction resource definition*

In the CEDA panel for CICS/ESA 3.1.1 shown in Figure 32, the security keywords are changed from those used in CICS OS/VS 1.7 and CICS/MVS 2.1, but the supported function is the same. For details of the changes, compare Figure 32 with Figure 33.

```
OBJECT CHARACTERISTICS
 CEDA  VIew
```

```
  SECURITY
   TRANSEC        : YES               Yes| External
   TRANSECNum     : 01                1-64
   RESSec         : No                No | Yes | External
   RESSECNum      : 00                0-24 | Public
   Cmdsec         : No                No | Yes | External
                                                        APPLID=CICSTOR
PF 1 HELP     3 END        6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

*Figure 32. View of part of a CICS/ESA 3.1.1 transaction resource definition*

```
 OBJECT CHARACTERISTICS
  CEDA  VIew
```

```
   SECURITY
    Extsec         : No                No | Yes
    TRANsec        : 01                1-64
    RSL            : 00                0-24 | Public
    RSLC           : No                No | Yes | External
                                                    APPLID=CICSTOR
 PF 1 HELP      3 END       6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

*Figure 33. View of part of a CICS/MVS 2.1 and CICS/OS/VS 1.7 transaction resource definition*

## Transaction-attach security coexistence

If you are sharing the CSD (for example, in an MRO environment where the AORs are at the CICS/ESA 3.2.1 level or later, and the TOR is at an earlier release level), you are recommended to use external security for all connected regions. For attach-time transaction security, this means the security attributes shown in the first row in Table 35. If you cannot easily convert your old regions to use RACF, you can still share the same resource definitions, even though the earlier releases are using CICS internal security. To use RACF on CICS/ESA 3.2.1, or later, and internal security on an earlier release, define the transaction security attributes as shown in the second row in the table.

*Table 35. Transaction-attach security definitions across releases*

| **Security attribute on transaction resource definition** | | | **Action taken by the CICS releases indicated in the column headings** |
|---|---|---|---|
| **CICS/OS/VS 1.7 and CICS/MVS** | **CICS/ESA 3.1.1** | **CICS/ESA 3.2.1 or later** | |
| EXTSEC(YES) | TRANSEC (EXTERNAL) | None | For all the releases, CICS calls RACF to verify whether the user is permitted to invoke the transaction. |

*Table 35. Transaction-attach security definitions across releases (continued)*

| Security attribute on transaction resource definition | | | Action taken by the CICS releases indicated in the column headings |
|---|---|---|---|
| **CICS/OS/VS 1.7 and CICS/MVS** | **CICS/ESA 3.1.1** | **CICS/ESA 3.2.1 or later** | |
| EXTSEC(NO)<br>TRANSEC(*nn*) | TRANSEC(YES)<br>TRANSECNUM(*nn*) | None | In CICS/ESA 3.2.1 and later releases, CICS calls RACF to verify whether the user is permitted to invoke the transaction.<br><br>In all earlier releases, CICS uses its own internal security mechanisms, comparing the transaction security number from the resource definition (nn) with the terminal user's security keys, to determine whether the user is permitted to invoke the transaction. |
| **Note:** To alter resource definitions in a CSD that is being shared between CICS and any earlier release, always make the changes using the CICS ALTER command (in compatibility mode). See Figure 34 for an example of a CEDA panel after pressing PF2. | | | |

```
OBJECT CHARACTERISTICS                              COMPATIBILITY MODE
 CEDA  VIew
```

```
  SECURITY
   RESSec          : No                No | Yes | External
   Cmdsec          : No                No | Yes | External
   Extsec          : No                No | Yes
   TRANsec         : 01                1-64
   RSl             : 00                0-24 | Public
                                                        APPLID=CICSTOR
PF 1 HELP 2 COM 3 END      6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

*Figure 34. A CICS Transaction Server for OS/390 Release 3 transaction resource definition in compatibility mode*

## Resource security coexistence

If you are sharing the CSD (for example, in an MRO environment where the AORs are at the CICS/ESA 3.2.1 or later level and the TOR is at an earlier release level), you are recommended to use external security for all connected regions. For resource security, this means specify resource security attributes. If you cannot easily convert your old regions to use RACF, you can still share the same resource definitions, even though the earlier releases are using CICS internal security. To use

RACF on CICS/ESA 3.2.1 and later releases, and internal security on an earlier release, define the transaction's resource security attributes as shown in the second row in the table.

## Extending timeout values

In any one MVS image, there can be many CICS regions, but only one RACF system in which the TIMEOUT values are stored. These values affect only CICS/ESA 3.2.1 and later systems.

However, if you are using RACF 2.1, and you run CICS regions with different levels of CICS, there are coexistence issues to consider. For example, a user may have a TIMEOUT value defined as 0101 (one hour and one minute). In the CICS region, the value will be treated correctly. But in the regions running CICS/ESA 3.3 or CICS/ESA 3.2.1, the hours will be truncated and the value will be treated as one minute.

The situation is even worse if the TIMEOUT value is set as 0200 (two hours). The user in the CICS Transaction Server for OS/390 Release 3 region will be timed out after two hours, while in the regions running earlier levels of CICS, the user will never be timed out. We recommend that in most cases you use the minutes value that you used on earlier levels of CICS. For upward compatibility, both CICS and RACF accept a value of 0060. You can then add the number of hours, as required, in the CICS region. If the hours value is greater than zero, the minutes value cannot exceed 59.

See the *OS/390 Security Server (RACF) Command Language Reference* for information about entering the extended timeout values, and the *OS/390 Security Server (RACF) General User's Guide*for information about how to list them. In addition, the *OS/390 Security Server (RACF) System Programmer's Guide*gives information on configuring your panels to use either the old values or the extended values.

## MRO bind security with multiple CICS releases in the same MVS

With CICS, DFHIRP resides within MVS and outside the individual CICS regions. Therefore, when CICS DFHIRP is installed, all CICS regions in the MVS image, regardless of their release level, use the DFHAPPL.*applid* form of MRO connect security. The CICS Transaction Server for OS/390 Release 3 implementation of MRO security is therefore forced upon any regions coexisting with a CICS Transaction Server for OS/390 Release 3 region. For more information, see "Chapter 16. Implementing MRO security" on page 199

When CICS DFHIRP is installed, all regions using earlier CICS releases in the MVS image use the DFHAPPL.*applid* form of MRO connect security. In addition, the SECURITYNAME parameter on the CONNECTION definition is obsolete and is ignored. It no longer has any effect on bind-time or link security. This means that there can be a reduced level of security on MRO links, unless you specify otherwise. Any CICS region with neither a specific DFHAPPL.*applid* profile (nor an applicable generic profile) ceases to have MRO connect security.

CICS does not issue any message to indicate this change.

You can use generic profiles to protect all regions or specific groups of regions against potential security exposures. They can be used before or in parallel with security measures for specific regions. Use the RDEFINE statement in RACF to

establish profiles for the links to be protected. For example, specifying the following would ensure that all CICS regions were subject to connect-time security:

```
RDEFINE FACILITY (DFHAPPL.*) UACC(NONE)
```

Authority to use a link can then be specified using the PERMIT command in RACF.

You still define regions to DFHIRP as described in "Logon security checking with MRO" on page 200.

## Removal of internal LU6.2 bind time security

The BINDPASSWORD in a CSD CONNECTION definition is not used for LU6.2 bind time security validation. Instead, create RACF APPCLU profiles, and specify XAPPC=YES on the SIT to maintain validated links.

## Transactions that use the JOURNALNUM option

Transactions that contain EXEC CICS WRITE JOURNALNUM can still be translated in CICS Transaction Server for OS/390 Release 3, and the commands still execute, but EXEC CICS WRITE JOURNALNAME is the preferred form. If resource security applies to a transaction executing WRITE JOURNALNUM, the journal number is prefixed with 'DFHJ' before the security check is applied. Thus, writing to journal number 2 requires UPDATE access to the resource DFHJ02. This is exactly compatible with previous releases.

# Part 6. Problem determination

This section consists of the chapter "Chapter 21. Problem determination in a CICS-RACF security environment" on page 251, and considers the following aspects of problem determination in a CICS-RACF security environment as follows:

# Chapter 21. Problem determination in a CICS-RACF security environment

This chapter provides information to help you find the causes of access authority problems. It covers the following topics:



## Resolving problems when access is denied incorrectly

When a user requires access to a protected resource (such as a CICS transaction) and RACF denies the requested access, you will often have to analyze the problem before deciding what action to take.

The basic points to ensure are that:

- CICS is using RACF for this particular kind of resource.
- You know which profile RACF is using to check the user's authority.
- You know which userid CICS is using for the authorization check.

For each security violation, up to three messages are issued:

- CICS issues an authority message to the terminal user (or returns a "not authorized" return code to an application).
- CICS sends a message DFHXS1111 to the CSCS transient data destination.
- RACF sends an ICH408I message to the CICS region's job log and to the security console.

  For a brief description of message ICH4081, see "RACF message ICH408I" on page 255. (For complete descriptions of this and all other RACF messages, see the *OS/390 Security Server (RACF) Messages and Codes* manual.)

If message ICH408I is issued for an authorization failure, RACF is active. The message text itself indicates the userid for which the authorization check was done and the name of the RACF profile that was used for the check.

When issued because of a CICS-originated authorization check, the RACF sends the ICH408I message to the CICS region's job log. Most CICS authorization messages also go to the CSCS transient data queue, except DFHIR and DFHZC messages, which go to the CSMT transient data queue.

**Note:** You can use the CICS-supplied message domain global user exit, XMEOUT, to reroute CICS-issued authorization messages. (For example, you can send them to the same console as the ICH408I messages.) For programming information about using XMEOUT, see the *CICS Customization Guide*.

If no profile exists for a particular resource, RACF returns a "profile not found" indication to CICS. CICS issues message DFHXS1111 with a SAF return code of X'00000004' and an ESM code of X'00000000'. **No ICH408I message is issued in this case.** The RLIST command issues a message stating that no profile was found.

**Note:**

- The RLIST command shows the profile as it exists in the RACF database, which might not necessarily be the same as the in-storage copy that CICS uses.
- When you have determined which RACF profile is denying access, see the *OS/390 Security Server (RACF) Security Administrator's Guide*for a description of authorization checking for RACF-protected resources. The following describe some further steps to take in resolving "access denied" problems.

## Is CICS using RACF for this particular kind of resource?

- Is CICS using an external security manager (ESM)?

  Make sure that CICS is using an ESM. If it is not using an ESM, it issues message DFHXS1102.
- Is security checking done for the particular general resource class? Message DFHXS1105 tells you if the class named on an X*name* parameter has been initialized.

  **Note:** If message DFHXS1105 is not there, ensure that the SEC=YES system initialization parameter is specified for the region.

  Check the appropriate CICS initialization parameter for the resource. For example: for transactions, this is the XTRAN parameter.

## Which profile is RACF using?

- Check the RACF message ICH408I for the name of the profile that RACF used.
- If CICS prefixing is in effect for the CICS region involved, the prefix specified is used as the first qualifier of RACF resource profiles (or member names).
  - Make sure that you have specified the correct prefix as part of resource profile names (on the RDEFINE command) and as member names on the ADDMEM operand.
  - Make sure the CICS job is running under the correct prefix if SECPRFX=YES is specified.
  - Make sure that an installation-written SAF exit is not changing the effective userid under which the CICS region is running.

  **Note:** The name of the resource in message ICH408I includes the prefix if SECPRFX=YES is specified in the system initialization parameters.
- Is CICS using current copies of the RACF resource profiles?

  If you have created, changed, or deleted a resource profile, the in-storage profile does not reflect the change until one of the following is completed:
  - SETROPTS GENERIC(*class-name*) REFRESH (a generic profile has been changed).
  - SETROPTS RACLIST(*classname*) REFRESH

  For more information about refreshing resource profiles, see "Refreshing resource profiles in main storage" on page 27.

  **Note:** If RACF is unable to process the PERFORM SECURITY REBUILD command (for example, because of an abend), CICS may terminate,

depending on the circumstances of the abend. For more information on this command, see "Refreshing resource profiles in main storage" on page 27.

- You can use the TSO RLIST command to determine which profile (or profiles) protect the resource. See Which profile is used to protect the resource?.

## Which userid did CICS supply for the authorization check?

Check to see if the user reporting the problem has signed on to CICS. If the user has not signed on to CICS, one of the following could be occurring:

- If you are using preset-terminal security, the authorization could be related to that terminal's userid.
- The user could be trying to operate as the CICS default user (without signing on to CICS).
- If the transaction was initiated by a START command, the userid could be inherited from the transaction issuing the START, or specified on the START command itself.
- If the transaction was initiated as the trigger transaction associated with a transient data queue, the userid could have been specified in the DCT for the queue.
- If the program is running as a PLTPI program, the userid could be specified in the PLTPIUSR system initialization parameter.

**Note:** RACF message ICH408I identifies the userid, as supplied by CICS to RACF, for which the authorization failed.

For help in identifying the user, see Table 1 on page 12.

## Which profile is used to protect the resource?

If you are using generic profiles (and you are **not** using resource group profiles), only the most specific profile is used. For example, if the following profiles exist:

```
**
C*
CE*
CEDA
```

CEDA is the profile that is used to control access to the CEDA transaction. If you delete profile CEDA and refresh the in-storage copies, CE* is used.

**Note:** This assumes CICS prefixing is not used and generic profile checking **is** used. (That is, that the RACF command SETROPTS GENERIC(*class_name*) has been issued for the class.)

If resource group profiles have been defined in the relevant class (for example, profiles in the GCICSTRN class), it is possible that more than one profile is used in determining a user's access. To determine which profiles protect the resource, enter the RLIST command with the RESGROUP operand. Be sure to specify the **member class** on the RLIST command. For example:

```
RLIST TCICSTRN transaction-name RESGROUP
```

If prefixing is in effect for this CICS region, include the prefix on the resource name specified on the RLIST command:

```
RLIST TCICSTRN prefix.transaction-name RESGROUP
```

Note that these examples use the member class TCICSTRN, not the resource group class GCICSTRN.

The AUDITOR attribute enables users to list all profiles that are defined, but does not authorize them to change those profiles. We recommend you give AUDITOR access to system programmers who need to see all profiles (for example, those who are doing problem determination) instead of system-SPECIAL access.

As a result of issuing RLIST with RESGROUP, you might see:

- A brief listing of the resource group profile that protects the resource. See Figure 35.
- A slightly longer listing showing the member profile as well as the resource group profile. See Figure 36.
- A "profile not found" message, if no profile is found that protects the resource. See Figure 37 on page 255.
- A "not authorized" message, if If a profile exists, but you are not authorized to list the profile. See Figure 38 on page 255.

```
rlist tcicstrn cemt resgroup
CLASS      NAME
-----      ----
TCICSTRN   CEMT
GROUP CLASS NAME
----- ----- ----
GCICSTRN
RESOURCE GROUPS
-------- ------
CAT2
```

*Figure 35. Output of RLIST command with RESGROUP: resource group profile*

```
rlist tcicstrn cemt resgroup
 CLASS      NAME
 -----      ----
 TCICSTRN   CEMT
 GROUP CLASS NAME
 ----- ----- ----
 GCICSTRN
 RESOURCE GROUPS
 -------- ------
 CICSCAT2A
 LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
 -----  --------   ----------------  -----------  -------
  00    PUB01           NONE              ALTER    NO
 INSTALLATION DATA
 -----------------
 NONE
 APPLICATION DATA
 ----------------
 NONE
 AUDITING
 --------
 FAILURES(READ)
 NOTIFY
 ------
 NO USER TO BE NOTIFIED
```

*Figure 36. Output of RLIST command with RESGROUP: several profiles*

**Note:** When you are using resource group profiles, more than one profile might be used at the same time. If the resource is protected by more than one profile, you are strongly urged to delete all other occurrences of the resource name. Use the DELMEM operand on the RALTER command to remove the resource name from existing resource group profiles. Use the RDELETE command **with care** to delete profiles from the member class.

```
rlist tcicstrn dfhcicsm.cemt resgroup
 ICH13003I DFHCICSM.CEMT NOT FOUND
```

*Figure 37. Output of RLIST command with RESGROUP: profile not found*

**Note:** If you get the "profile not found" message, make sure that generic profile processing is in effect for the specified class. (SETROPTS LIST will show this.) If, indeed, no profile exists, create a suitable profile and ensure that the appropriate users and groups have access.

```
rlist tcicstrn dfhcicsm.cemt resgroup
 ICH13002I NOT AUTHORIZED TO LIST DFH*
```

*Figure 38. Output of RLIST command with RESGROUP: authorization message*

The "not authorized" message identifies the name of the profile preventing you from having access. You can ask the RACF security administrator (who has the system-SPECIAL attribute and can therefore list the profile) to investigate the problem.

Some possible solutions are:

- The profile is not needed and can be deleted.
- You can be made OWNER of the profile.
- A more specific generic profile can be created, and you or your group can be made OWNER of the new profile.
- If the profile is a discrete profile, you can be given ALTER access to the profile.
- You can be assigned the AUDITOR attribute.

For a description of the authority needed to list a general resource profile, see the description of the RLIST command in the *OS/390 Security Server (RACF) Command Language Reference*.

## RACF message ICH408I

For a complete description of RACF message ICH408I, see the *OS/390 Security Server (RACF) Messages and Codes* manual.

The first line of message ICH408I identifies a user who had an authorization problem. The other lines of the message describe the request the user was issuing and the reason for the failure.

Consider the following example:

```
ICH408I USER(JONES   ) GROUP(DEPT60  ) NAME(M.M.JONES          )
ICH408I   FLA32 CL(FCICSFCT)
ICH408I   INSUFFICIENT ACCESS AUTHORITY
ICH408I   FROM F%A* (G)
ICH408I   ACCESS INTENT(UPDATE )  ACCESS ALLOWED(READ   )
```

This message can be interpreted as follows:

User JONES, a member of group DEPT60, whose name is M.M.JONES, had INSUFFICIENT ACCESS AUTHORITY to resource FLA32, which is in class FCICSFCT.

**Note:** If the class shown is a resource group class, the profile might be in the class shown or in the related member class. For example, if GCICSTRN appears, check TCICSTRN also. If HCICSFCT appears, check FCICSFCT also. For a list of all the IBM-supplied class names, see Table 3 on page 26. For a list of the installation-defined class names that are in use on your installation, see your RACF system programmer, or issue the SETROPTS LIST command.

The RACF profile protecting the resource is F%A*. "(G)" indicates that F%A* is a generic profile.

The access attempted by user JONES was UPDATE, but the access allowed by RACF was READ. Therefore, user JONES was denied access.

A DFHXS1111 message would also be issued for this access attempt:

```
DFHXS1111 26/09/95 15:34:01 CICSSYS1 Security violation by user JONES
          at netname D2D1 for resource FLA32 in class
          TCICSTRN. SAF codes are (X'00000008',X'00000000'). ESM codes
          are (X'00000008',X'00000000').
```

The SAF and ESM codes come from RACROUTE REQUEST=AUTH.

To find the cause of the violation, issue the RLIST command with AUTHUSER specified to list the profile indicated in the ICH408I message. The AUTHUSER operand displays the access list, as shown in Figure 39.

```
rlist fcicsfct f%a* authuser
CLASS      NAME
-----      ----
FCICSFCT   F%A* (G)
GROUP CLASS NAME
----- ----- ----
HCICSFCT
LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    CICSADM         NONE              ALTER    NO
                           .
                           .
                           .
                           .
USER      ACCESS
----      ------
DEPTA     UPDATE
JONES     READ
GROUPX    NONE
SYSPROG   ALTER
```

*Figure 39. Requesting a display of the access list*

In this profile, user JONES has an explicit entry in the access list. If the userid itself does not appear in the access list, check for one of JONES's connect groups. To list the groups to which JONES is connected, issue LISTUSER JONES Other

specifications in the profile (such as security level or security category) might cause access to be denied. For a complete description, see the *OS/390 Security Server (RACF) Security Administrator's Guide*.

**Note:** If NOTIFY(CICSADM) were specified in this profile, TSO userid CICSADM would receive immediate notification of failed attempts to access resources protected by the profile.

## Resolving problems when access is allowed incorrectly

There could be many reasons why a user might have access to a protected resource, even when you think that the user should **not** have that access. Here are some checks that you can make to investigate this kind of situation:

- Confirm which userid the user is signed on as. (Make sure the user has actually signed on and is not acting as the CICS default user.) You can ask the user to sign off, then sign on again. You can also ask the user to issue EXEC CICS ASSIGN or EXEC CICS INQUIRE TERMINAL, which can be issued with the CECI transaction (or a user-written transaction).
- Make sure that the SEC system initialization parameter is SEC=YES for the CICS region the user is signed on to.

  If SEC=NO is specified, users can access any resource.
- If the user is running a transaction that communicates with other regions such as application-owning regions (AORs) or file-owning regions (FORs), make sure that the SEC system initialization parameter is SEC=YES for those regions.
- Is prefixing correct?
  - Has the CICS JOB been submitted by the correct USER?
  - Is SECPRFX set correctly?
  - Has an installation-written SAF exit been used to return a different CICS region userid when RACROUTE=EXTRACT has been specified?
- Depending on the resource, make sure that RESSEC(YES) is specified for each transaction that might access that resource.
- Is the appropriate X*name* CICS system initialization parameter correctly set?

  For example, if it is a file control request, is XFCT=YES or XFCT=*value* specified? If the X*name* parameter specifies a value other than YES or NO, does the value show the correct installation-defined class name?
- Is the transaction exempt from transaction security? (For information on transactions that may have been defined in this way, see "Category 3 transactions" on page 133.)
- Does the transaction have the correct RESSEC and CMDSEC options?
- Check that the RESSEC setting on the MIRROR transaction is correct.
- If the resource is temporary storage, are you using the correct TST? Check:
  - The DFHTST TYPE=SECURITY entry in the TST
  - That TST entries are in the correct order

  If you are using TSMODEL resource definition, check the SECURITY option of the model.
- If intersystem communication is involved, check the following:
  - Is a SECURITY REBUILD required (on this or on the remote system)?
  - If ATTACHSEC=LOCAL is specified, does the SECURITYNAME userid have access to the resource?
  - Is ATTACHSEC=IDENTIFY specified?

  - Are 'equivalent systems' causing link security to be bypassed
  - Is the remote system using the same RACF database?
- Do you have any RACF installation exits?
- To check the profile that you think protects the resource, use the checklist provided in the *OS/390 Security Server (RACF) Security Administrator's Guide*.

## CICS initialization failures related to security

From CICS/ESA 4.1, if SEC=YES is specified, external security is *required*. If external security cannot be provided, CICS cannot be initialized.

Figure 40 on page 259 shows an example of a failure to initialize.

If security initialization fails:

- Examine the DFHXS1106 message return codes. In the example shown in Figure 40 on page 259, SAF return code X'00000004' and reason code X'00000000' were issued:

  A return code of X'00000004' indicates that an error occurred in the MVS security router (RACROUTE). See the RACROUTE macro reference in "CICS security control points" on page 223.
- Check the CICS startup options, in particular the X*name* system initialization parameters. Make sure that:
  - The class is defined to RACF and is active (use the SETROPTS LIST command to check this).
  - The class is defined in the router table. To do this, examine the installation source for ICHRFR01 for any installation-defined classes. (The description of the ICHFRTB macro in the *OS/390 Security Server (RACF) Macros and Interfaces*manual includes a listing of the IBM-supplied module, ICHRFR0X.)

Figure 40 on page 259 shows that XPPT=UNKNOWN has been specified. This causes CICS to try to use a class called MUNKNOWN. MUNKNOWN has not been defined to the MVS router, or to the RACF CDT.

### RACF abends

If a RACF abend occurs, see the *OS/390 Security Server (RACF) Messages and Codes*manual and the *OS/390 Security Server (RACF) Diagnosis Guide*for further guidance.

### SAF or RACF installation exits

Check if any SAF or RACF installation exits are causing initialization (RACLIST requests to fail.

### CICS default user fails to sign on

Figure 41 on page 260 shows an example of a CICS job log when the DFLTUSER fails to sign on. CICS is started with SEC=YES and DFLTUSER=ORMAN. User profile ORMAN has not been defined to RACF.

This CICS region cannot be initialized because, with SEC=YES specified, external security is required and the default user must be defined to RACF.

```
DFHPA1927  IYCTZCCA AKPREQ=0
DFHPA1927  IYCTZCCA APPLID=IYCTZCCA
DFHPA1927  IYCTZCCA CSDFRLOG=NO
DFHPA1927  IYCTZCCA CSDRECOV=NONE
DFHPA1927  IYCTZCCA FCT=NO
DFHPA1927  IYCTZCCA SIT=Tø
DFHPA1927  IYCTZCCA START=INITIAL
DFHPA1927  IYCTZCCA GMTEXT='SSYS NOOR CIC5100M SYSTEM with RACF'
DFHPA1927  IYCTZCCA GRPLIST=USERLIST
DFHPA1927  IYCTZCCA PLTPI=NO
DFHPA1927  IYCTZCCA SEC=YES
DFHPA1927  IYCTZCCA XCMD=NO
DFHPA1927  IYCTZCCA XDCT=1CVFDCT
DFHPA1927  IYCTZCCA XFCT=1CVFFCT
DFHPA1927  IYCTZCCA XJCT=1CVFJCT
DFHPA1927  IYCTZCCA XPCT=1CVFPCT
DFHPA1927  IYCTZCCA XPPT=UNKNOWN
DFHPA1927  IYCTZCCA XPSB=NO
DFHPA1927  IYCTZCCA XTST=1CVFTST
DFHPA1927  IYCTZCCA XTRAN=1CVFTRN
DFHPA1927  IYCTZCCA CICSSVC=212
DFHPA1927  IYCTZCCA SRBSVC=211
DFHPA1927  IYCTZCCA .END
DFHPA1103  IYCTZCCA END OF FILE ON SYSIN.
+DFHTR0103 TRACE TABLE SIZE IS 64K
+DFHSM0122I IYCTZCCA Limit of DSA storage below 16MB is 5,120K.
+DFHSM0123I IYCTZCCA Limit of DSA storage above 16MB is 20M.
+DFHSM0113I IYCTZCCA Storage protection is not active.
+DFHSM0126I IYCTZCCA Transaction isolation is not active.
+DFHDM0101I IYCTZCCA CICS is initializing.
+DFHLG0101I IYCTZCCA Log manager domain initialization has started.
+DFHSI1500 IYCTZCCA CICSTS13.CICS. Startup is in progress.
+DFHXS1100I IYCTZCCA Security initialization has started.
+DFHDU0304I IYCTZCCA Transaction Dump Data set DFHDMPA opened.
+DFHSI1501I IYCTZCCA Loading CICS nucleus.
+DFHXS1105 IYCTZCCA Resource profiles for class A1CVFPCT have been built.
+DFHDU03041 IYCTZCCA Transaction Dump Data set DFHDMPA opened.
+DFHXS1105 IYCTZCCA Resource profiles for class D1CVFDCT have been built.
+DFHXS1105 IYCTZCCA Resource profiles for class F1CVFFCT have been built.
+DFHXS1105 IYCTZCCA Resource profiles for class J1CVFJCT have been built.
+DFHXS1106 IYCTZCCA
 Resource profiles could not be built for class MUNKNOWN. CICS is
 terminated. SAF codes are (X'00000004',X'00000000'). ESM codes are
 (X'00000000',X'00000000').
+DFHDU0303I IYCTZCCA Transaction Dump Data set DFHDMPA closed.
+DFHKE1800 IYCTZCCA ABNORMAL TERMINATION OF CICSTS13.CICS IS COMPLETE.
IEF450I SSYTZCCA CICS - ABEND=S000 U1800 REASON=00000000
                            TIME=13.55.26
$HASP395 SSYTZCCA ENDED
```

*Figure 40. Security initialization failure*

```
DFHPA1927  IYCTZCCE SEC=YES
DFHPA1927  IYCTZCCE XUSER=YES
DFHPA1927  IYCTZCCE DFLTUSER=ORMAN
DFHPA1927  IYCTZCCE XCMD=NO
DFHPA1927  IYCTZCCE XDCT=1CVFDCT
DFHPA1927  IYCTZCCE XFCT=1CVFFCT
DFHPA1927  IYCTZCCE XJCT=1CVFJCT
DFHPA1927  IYCTZCCE XPCT=1CVFPCT
DFHPA1927  IYCTZCCE XFCT=1CVFPPT
DFHPA1927  IYCTZCCE XPSB=NO
DFHPA1927  IYCTZCCE XTST=1CVFTST
DFHPA1927  IYCTZCCE XTRAN=1CVFTRN
DFHPA1927  IYCTZCCE CICSSVC=212
DFHPA1927  IYCTZCCE SRBSVC=211
DFHPA1927  IYCTZCCE .END
DFHPA1103  IYCTZCCE END OF FILE ON SYSIN.
+DFHTR0103 TRACE TABLE SIZE IS 64K
+DFHSM0122I IYCTZCCE Limit of DSA storage below 16MB is 5,120K.
+DFHSM0123I IYCTZCCE Limit of DSA storage above 16MB is 20M.
+DFHSM0113I IYCTZCCE Storage protection is not active.
+DFHSM0126I IYCTZCCE Transaction isolation is not active.
+DFHDM0101I IYCTZCCE CICS is initializing.
+DFHLG0101I IYCTZCCE Log manager domain initialization has started.
+DFHSI1500 IYCTZCCE CICSTS13.CICS Startup is in progress                    .
+DFHXS1100I IYCTZCCE Security initialization has started.
+DFHDU0304I IYCTZCCE Transaction Dump Data set DFHDMPA opened.
+DFHSI1501I IYCTZCCE Loading CICS nucleus.
+DFHXS1105 IYCTZCCE Resource profiles for class A1CVFPCT have been built.
+DFHXS1105 IYCTZCCE Resource profiles for class D1CVFDCT have been built.
+DFHXS1105 IYCTZCCE Resource profiles for class F1CVFFCT have been built.
+DFHXS1105 IYCTZCCE Resource profiles for class J1CVFJCT have been built.
+DFHXS1105 IYCTZCCE Resource profiles for class M1CVFPPT have been built.
+DFHXS1105 IYCTZCCE Resource profiles for class S1CVFTST have been built.
+DFHXS1105 IYCTZCCE Resource profiles for class T1CVFTRN have been built.
+DFHXS1105 IYCTZCCE Resource profiles for class SURROGAT have been built.
ICH4081 USER(ORMAN   ) GROUP(      ) NAME(???
  LOGON/JOB INITATION - USER AT TERMINAL      NOT RACF-DEFINED
IRR012I VERIFICATION FAILED. USER PROFILE NOT FOUND.
+DFHXS1104 IYCTZCCE
 Default security could not be established for userid ORMAN.  The
 security domain cannot continue, so CICS is terminated.  SAF codes are
 (X'00000004',X'00000000'). ESM codes are (X'00000004',X'00000000').
+DFHDU0303I IYCTZCCE Transaction Dump Data set DFHDMPA closed.
+DFHKE1800 IYCTZCCE ABNORMAL TERMINATION OF CICSTS13.CICS IS COMPLETE.
IEF450I SSYTZCCE CICS - ABEND=S000 U1800 REASON=00000000
                           TIME=13.55.26
$HASP395 SSYTZCCE ENDED
```

*Figure 41. Example of CICS job log if DFLTUSER fails to sign on*

## Revoked user attempting to sign on

The following example sequence illustrates what happens when a revoked user attempts to sign on:

1. User USR001 attempts to sign on using CESN. However, the user is revoked. The user sees the following on the terminal:

   ```
   DFHCE3546 Your signon userid has been revoked. Signon is terminated.
   ```

2. A RACF ICH408I message is sent to the CICS region's job log:

   ```
   ICH408I USER(USR001  ) GROUP(GRP001  ) NAME(AUSER          )
     LOGON/JOB INITIATION - REVOKED USER ACCESS ATTEMPT
   ```

This message indicates that user USR001, whose name as recorded in the RACF user profile is AUSER, and whose current RACF connect group is GRP001, attempted to sign on.

3. A CICS message is sent to the CSCS transient data queue:

```
DFHSN1120 26/09/95 12:20:24 CICSSYS1 Signon at netname D2D1
with userid USR001 failed because the userid has been revoked.
```

## User has insufficient authority to access a resource

Now let us consider user USR001, who has signed on successfully with current connect group GRP001. User USR001 attempts unsuccessfully to use transaction CEMT, which is protected by profile CAT2 in class GCICSTRN (the resource group class for CICS transactions), because XTRAN=YES is specified in the CICS system initialization parameters.

1. The terminal user received the following CICS message:

```
DFHAC2033 26/09/95 15:18:44 CICSSYS1 You are not authorized to use
transaction CEMT. Check that the transaction name is correct.
```

2. A RACF ICH408I message is sent to the CICS region's job log:

```
ICH408I USER(USR001  ) GROUP(GRP001  ) NAME(AUSER         )
ICH408I   CEMT CL(TCICSTRN)
ICH408I   INSUFFICIENT ACCESS AUTHORITY
ICH408I   ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
```

This message indicates that user USR001, whose name as recorded in the RACF user profile is AUSER, and whose current RACF connect group is GRP001, attempted to use the CEMT transaction. To do this, AUSER needs to have at least READ access to the profile protecting the CEMT transaction. However, RACF determined that AUSER had **no** access authority.

3. A CICS message is sent to the CSCS transient data queue:

```
DFHXS1111 26/09/95 13:30:41 CICSSYS1 CEMT Security violation
          by user USR001 at netname D2D1 for resource CEMT in class
          TCICSTRN. SAF codes are (X'00000008',X'00000000'). ESM codes
          are (X'00000008',X'00000000').
```

The following message is also sent to the CSMT transient data queue:

```
DFHAC2003 26/09/95 15:18:44 CICSSYS1 Security violation has been
detected term id = D2D1, trans id = CEMT, userid = USR001.
```

4. Which profile protects CEMT?

It appears from the ICH408I message that profile CEMT in class TCICSTRN protects CEMT. However, this is not necessarily the case. A resource group profile (in class GCICSTRN) might protect CEMT. In fact, in this case, there is no profile named CEMT. If a system-SPECIAL or AUDITOR user issues the SEARCH command with CLASS(TCICSTRN) specified, no profile named CEMT would appear.

To determine which profile was actually used, you must issue the RLIST command with the RESGROUP operand as follows:

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

In this case, issue the following:

```
RLIST TCICSTRN CEMT RESGROUP
```

**Note:** If prefixing is used for this CICS region, specify the prefix on the resource-name in the RLIST command.

RACF displays the following:

```
CLASS      NAME
-----      ----
TCICSTRN   CEMT
GROUP CLASS NAME
----- ----- ----
GCICSTRN
RESOURCE GROUPS
-------- ------
CAT2
```

The profiles in class GCICSTRN that protect CEMT are shown under RESOURCE GROUPS in the command output. In this case, only one profile (CAT2) protects profile CEMT.

**Note:** If a profile in class TCICSTRN protected CEMT, that profile's contents would be added to the output of RLIST.

5. To determine how profile CAT2 protects CEMT, list that profile with the AUTHUSER operand specified on the RLIST command:

```
RLIST GCICSTRN CAT2 AUTHUSER
```

RACF displays the following:

```
CLASS      NAME
-----      ----
GCICSTRN   CAT2
MEMBER CLASS NAME
------ ----- ----
TCICSTRN
RESOURCES IN GROUP
--------- -- -----
CDBC
CDBI
CBRC
CEDA
CEMT
CETR
LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
.
.
.
NOTIFY
------
NO USER TO BE NOTIFIED
USER      ACCESS   ACCESS COUNT
----      ------   ------ -----
DEPTA     ALTER       000000
USR001    NONE        000000
-------- -------  ------------ -------- -------------------------
NO ENTRIES IN CONDITIONAL ACCESS LIST
```

## CICS region user ID access problem

CICS security initialization can fail if the CICS region user ID does not have access to the necessary Category 1 transactions. A message similar to the following is shown:

```
DFHXS1103I CICSAPPL Default security for userid CICSUSER has been established.
DFHDU0304I CICSAPPL Transaction dump data set DFHDMPA opened.
DFHXS1111 CICSAPPL
 01/18/99 16:05:15 CICSAPPL ???? Security violation by user TESTRGN for resource CATA
in class TCICSTRN.SAF codes are
(X'00000004',X'00000000').ESM codes are (X'00000004',X'00000000').
DFHXS1113 CICSAPPL
The region userid cannot access system transaction CATA.  CICS will terminate.
SAF codes are (X'00000004',X'00000000').ESM codes are (X'00000004',X'00000000').
```

When this occurs the SAF and return codes (X'00000004') indicate that no profile in the TCICSTRN (or GCICSTRN) class is protecting the resource CATA. To resolve this a suitable profile for CATA must be defined in the TCICSTRN (or GCICSTRN) class, and CICS region user ID TESTRGN must be given at least READ access. For guidance you can use example CLIST DFH$CAT1 which is in library CICSTS13.CICS.SDFHSAMP (see "Category 1 transactions" on page 126).

After adding a suitable profile, you must issue the command:

```
SETROPTS RACLIST (TCICSTRN) REFRESH
```

Even if you have also added the profile to the GCICSTRN class, you still only issue this command for TCICSTRN. If you do NOT issue this SETROPTS command, and you restart CICS, initialization will fail again with the same error, because RACF will not be using the updated definitions.

## Password expiry management problem determination

If you are running a CICS-APPC PEM environment, and are not receiving the expected responses, check the following possible sources of errors in the sign-on transaction program:

- The function management header (FMH) may be in error; check that:
  - The conversation type being used is **basic**.
  - The XTRANID in the CICS TRANSACTION definition for CLS4 is X'06F3F0F1'. (See "Setting up the PEM client" on page 181.)
  - The CICS PEM server sign-on transaction is running as a **synclevel 0** transaction. (See "Setting up the PEM client" on page 181.)
- The user data may be in error; check that:
  - Valid lengths are being sent. (See Table 24 on page 184, Table 25 on page 184, and "Format of user data" on page 182.)
  - Userids and passwords are sent in uppercase EBCDIC. (See "Setting up the PEM client" on page 181.)
  - GDS variables (required in basic conversations) are being used: (See "Format of user data" on page 182.)

**Note:** If the CICS PEM server receives an error in the FMH or user data, it sends an ISSUE ERROR to the PEM requester, and terminates without an abend. If this happens, it is likely that there is an error in the flow. For examples of valid flows, see "Examples of PEM client and CICS PEM server user data" on page 187.

## Execution diagnostic facility (EDF)

The execution diagnostic facility (EDF) **cannot** be used to check DFHCLS4, for security reasons, because user passwords would be displayed on the EDF screens.

# Part 7. CICSPlex SM security

This part describes how to implement security for CICSPlex SM. It contains the following chapters

- "Chapter 22. Implementing CICSPlex SM security" on page 267 explains how to implement RACF security for CICSPlex SM
- "Chapter 23. Invoking a user-supplied external security manager" on page 303 provides information on using a SAF-compliant external security manager other than RACF.
- "Chapter 24. Writing an API security exit" on page 307 describes how to write an API security exit and describes the role of the default security routine, EYU9XESV.
- "Chapter 25. Example tasks: security" on page 313 provides examples of typical security setup tasks that you can use as a model for your own.

# Chapter 22. Implementing CICSPlex SM security

This chapter explains how to implement RACF security for CICSPlex SM. The first section provides general information to help you determine who needs access to the various CICSPlex SM functions. The remaining sections provide detailed information on defining CICSPlex SM class names, using resource names, activating security, and refreshing RACF profiles.

**Note:** For information on using a SAF-compliant external security manager (ESM) other than RACF, refer to "Chapter 23. Invoking a user-supplied external security manager" on page 303.

The following steps are required to implement RACF for CICSPlex SM:

1. Decide who needs access to CICSPlex SM (see page 267).
2. Review the general security requirements for CICSPlex SM (see page 270).
3. Create RACF profiles for the CICSPlex SM data sets (see page 270).
4. Define the CICSPlex SM started tasks to RACF (see page 271).
5. If CICS transaction security is active in a CMAS , define the CICSPlex SM transactions to RACF (see page 271).
6. If CICS transaction security is active in a MAS running CICS/ESA 4.1 or later, define the CICSPlex SM transactions to RACF (see page 272).
7. Create RACF profiles for the CAS functions and PlexManager views (see page 274).
8. Create RACF profiles for the CICSPlex SM views (see page 276).
9. If desired, activate simulated security checking using the CICSSYS, CPLEXDEF, or MAS views (see page 293).
10. Activate security in the CMASs and MASs using the CICSPlex SM and CICS SIT security parameters (see page 294).

## Determining who needs access to the CICSPlex SM views

To determine who needs access to the CICSPlex SM views, consider the following questions. You can use the security matrix in Table 36 on page 268 to record your answers to these questions. The matrix can then be used as the basis for creating PERMIT statements.

**What groups of users will use CICSPlex SM?**
Your enterprise probably already has several user groups defined to RACF. The groups that typically require access to CICSPlex SM include systems programming, operations, the help desk, applications programming, and performance monitoring. These groups are used as column headings in the security matrix. You can supply their corresponding RACF group IDs. (If necessary, you can ignore, replace, or add groups to the matrix as appropriate for your enterprise.)

**Which CICSPlex SM views will each group need access to?**
The CICSPlex SM views are grouped by functionality: configuration, topology, workload management, real-time analysis, operations, monitoring, business application services, and CICSplex management. Not all view groups are appropriate for all users. Certain groups of users will require access to a subset of views. For example, the systems programming

group might require access to all views, while the help desk group might only need access to one or two. The view groups are listed vertically on the left side of the matrix, along with the high-level qualifier of their CICSPlex SM resource names.

**What type of access does each RACF group need?**

After deciding who should have access to what, you should prohibit universal access to all of the views. You can then selectively permit read, update, or alter access to specific view groups. To complete the matrix, specify READ, UPDATE, or ALTER access for each RACF group that needs access to a group of views.

**Note:** For Business Application Services (BAS), users need UPDATE access to create and update resource definitions. However, they need ALTER access to install resource definitions in CICS systems.

*Table 36. Security matrix*

| RACF group → CICSPlex SM view group ↓ | System Programming ID( ) | Operations ID( ) | Help Desk ID( ) | Application Programming ID( ) | Performance ID( ) |
|---|---|---|---|---|---|
| Configuration CONFIG | | | | | |
| Topology TOPOLOGY | | | | | |
| Workload Management WORKLOAD | | | | | |
| Real-Time Analysis ANALYSIS | | | | | |
| Operations OPERATE | | | | | |
| Monitor MONITOR | | | | | |
| Business Application Services BAS | | | | | |
| PlexManager BBM.PLEXMGR | | | | | |

Table 37 is a sample of a completed security matrix for a production CICSplex:

*Table 37. Sample security matrix*

| RACF group → CICSPlex SM view group ↓ | System Programming ID(SYSPGRP) | Operations ID(OPSGRP) | Help Desk ID(HELPGRP) | Application Programming ID(APPLGRP) | Performance ID(PERFGRP) |
|---|---|---|---|---|---|
| Configuration CONFIG | UPDATE | | | | |
| Topology TOPOLOGY | UPDATE | UPDATE | READ | | |
| Workload Management WORKLOAD | UPDATE | | | READ | |
| Real-Time Analysis ANALYSIS | UPDATE | UPDATE | READ | | READ |
| Operations OPERATE | ALTER | UPDATE | READ | READ | READ |
| Monitor MONITOR | UPDATE | READ | | | READ |
| Business Application Services BAS | ALTER | ALTER | | UPDATE | |
| PlexManager BBM.PLEXMGR | UPDATE | | | | |

First you need to ensure that the CPSMOBJ class is active and that generic profiles can be defined:

```
SETROPTS CLASSACT(CPSMOBJ)
SETROPTS GENERIC(CPSMOBJ)
SETROPTS GENCMD(CPSMOBJ)
```

Then you need to create a RACF profile to protect all of the views and action commands for all CICSPlex SM functions:

```
RDEF CPSMOBJ ** UACC(NONE) OWNER(admin_group) NOTIFY(admin_user)
```

CPSMOBJ is the CICSPlex SM member class. This class is predefined for RACF Version 2.1 and later. The double asterisks indicate that all of the CICSPlex SM views are included in this RDEF statement.

Next, using the information in the sample matrix, you can permit access to the specific view groups. For example, the systems programming group requires update access to all of the view groups and ALTER access to the BAS views. This can be accomplished with just three PERMIT statements:

```
PERMIT ** CLASS(CPSMOBJ)  ID(SYSPGRP) ACCESS(UPDATE)
PERMIT BBM.PLEXMGR.** CLASS(FACILITY) ID(SYSPGRP) ACCESS(UPDATE)
PERMIT BAS.** CLASS(CPSMOBJ) ID(SYSPGRP) ACCESS(ALTER)
```

The double asterisks indicate that all of the CICSPlex SM views are affected by this PERMIT statement.

The following PERMIT statements grant the appropriate access to all of the topology views for the operations and help desk groups:

```
PERMIT TOPOLOGY.** CLASS(CPSMOBJ) ID(OPSGRP)  ACCESS(UPDATE)
             PERMIT TOPOLOGY.** CLASS(CPSMOBJ) ID(HELPGRP) ACCESS(READ)
```

For the workload management views:

```
PERMIT WORKLOAD.** CLASS(CPSMOBJ) ID(APPLGRP) ACCESS(READ)
```

For the real-time analysis views:

```
PERMIT ANALYSIS.** CLASS(CPSMOBJ) ID(OPSGRP)  ACCESS(UPDATE)
PERMIT ANALYSIS.** CLASS(CPSMOBJ) ID(HELPGRP) ACCESS(READ)
PERMIT ANALYSIS.** CLASS(CPSMOBJ) ID(PERFGRP) ACCESS(READ)
```

For the operations views:

```
PERMIT OPERATE.** CLASS(CPSMOBJ) ID(OPSGRP)  ACCESS(UPDATE)
PERMIT OPERATE.** CLASS(CPSMOBJ) ID(HELPGRP) ACCESS(READ)
PERMIT OPERATE.** CLASS(CPSMOBJ) ID(APPLGRP) ACCESS(READ)
PERMIT OPERATE.** CLASS(CPSMOBJ) ID(PERFGRP) ACCESS(READ)
```

For the monitor views:

```
PERMIT MONITOR.** CLASS(CPSMOBJ) ID(APPLGRP) ACCESS(READ)
PERMIT MONITOR.** CLASS(CPSMOBJ) ID(PERFGRP) ACCESS(READ)
```

For the business application services views:

```
PERMIT BAS.** CLASS(CPSMOBJ) ID(OPSGRP) ACCESS(ALTER)
PERMIT BAS.** CLASS(CPSMOBJ) ID(APPLGRP) ACCESS(UPDATE)
```

For simplicity, these PERMIT statements grant access to broad groups of views by using the double asterisks in the resource names. However, if desired, you can use more specific resource names in your PERMIT statements. Refer to "Specifying CICSPlex SM resource names in profiles" on page 276 for details.

Using your own completed security matrix and the information in the remainder of this chapter, you can create as many profiles as necessary for your enterprise. "Chapter 25. Example tasks: security" on page 313 provides detailed profile examples.

## General requirements for CICSPlex SM security

You should review your RACF configurations to ensure that the following minimum requirements are met:

- The user ID associated with the coordinating address space (CAS) must have:
  - Authority to define and initialize the MVS subsystem for the CAS. See "Specifying CAS and PlexManager resource names in profiles" on page 274 for more details.
  - UPDATE access to the BBIPARM data set.
  - READ access to the BBSECURE data set.
- Each CICSPlex SM address space (CMAS) must have authority to connect to a CAS and attach a service point, which establishes the product and context a user can access.
- The IDs for all users expected to use CICSPlex SM should be defined to RACF in each MVS system in which there is a CMAS. For each individual user, the ID must be the same for each MVS system.
- User access authority to CICSPlex SM definitions and CICS commands and resources should be defined to RACF in a consistent manner in all MVS systems used by CICSPlex SM.

In addition, you should be aware that, in the CMAS address space, a security environment is created for the user specified in the CICS system initialization table (SIT) parameter DFLTUSER associated with the MAS.

## Creating profiles for the CICSPlex SM data sets

You should restrict access to CICSPlex SM data sets using RACF data set protection. Use the following guidelines:

- Prohibit universal access, by specifying UACC(NONE).
- Ensure that minimum access to the data sets is authorized for the RACF USERID assigned to the:
  - CAS started task.
  - Each CMAS job (or started task).
  - Each MAS.
  - All individuals allowed to use CICSPlex SM via the CICSPlex SM EUI and API (both system administrators and end users).

Table 38 lists the CICSPlex SM data sets and the minimum access that should be granted to each type of user ID.

*Table 38. Access by user ID for CICSPlex SM data sets*

| Data set name | CAS | CMAS | MAS | System Admin. | Individual User |
|---|---|---|---|---|---|
| SYS1.CICSTS13.CPSM.SEYULPA | NONE | NONE | READ | UPDATE | NONE |
| SYS1.CICSTS13.CPSM.SEYULINK | NONE | READ | NONE | UPDATE | NONE |
| CICSTS13.CPSM.SEYUAUTH | READ | READ | READ | UPDATE | READ |
| CICSTS13.CPSM.SEYULOAD | NONE | READ | READ | UPDATE | NONE |
| CICSTS13.CPSM.SEYUPARM | READ | READ | READ | UPDATE | NONE |
| CICSTS13.CPSM.SEYUCMOD | NONE | NONE | NONE | UPDATE | NONE |

*Table 38. Access by user ID for CICSPlex SM data sets (continued)*

| Data set name | CAS | CMAS | MAS | System Admin. | Individual User |
|---|---|---|---|---|---|
| CICSTS13.CPSM.SEYUCOB | NONE | NONE | NONE | UPDATE | READ |
| CICSTS13.CPSM.SEYUC370 | NONE | NONE | NONE | UPDATE | READ |
| CICSTS13.CPSM.SEYUDEF | NONE | READ | READ | UPDATE | READ |
| CICSTS13.CPSM.SEYUADEF | READ | READ | NONE | UPDATE | NONE |
| CICSTS13.CPSM.SEYUVDEF | READ | READ | NONE | UPDATE | NONE |
| CICSTS13.CPSM.SEYUCLIB | NONE | NONE | NONE | UPDATE | READ |
| CICSTS13.CPSM.SEYUMLIB | NONE | NONE | NONE | UPDATE | READ |
| CICSTS13.CPSM.SEYUPLIB | NONE | NONE | NONE | UPDATE | READ |
| CICSTS13.CPSM.SEYUTLIB | NONE | NONE | NONE | UPDATE | READ |
| CICSTS13.CPSM.SEYUINST | NONE | NONE | NONE | UPDATE | NONE |
| CICSTS13.CPSM.SEYUJCL | NONE | NONE | NONE | UPDATE | NONE |
| CICSTS13.CPSM.SEYUMAC | NONE | NONE | NONE | UPDATE | READ |
| CICSTS13.CPSM.SEYUOS2 | NONE | NONE | NONE | UPDATE | NONE |
| CICSTS13.CPSM.SEYUPL1 | NONE | NONE | NONE | UPDATE | READ |
| CICSTS13.CPSM.SEYUPROC | NONE | NONE | NONE | UPDATE | READ |
| CICSTS13.CPSM.SEYUSAMP | NONE | NONE | NONE | UPDATE | READ |
| CICSTS13.CPSM.EYUSDEF | NONE | NONE | NONE | UPDATE | UPDATE |
| CICSTS13.CPSM.EYUDREP | NONE | UPDATE | NONE | UPDATE | NONE |
| CICSTS13.CPSM.EYUIPRM | UPDATE | NONE | NONE | UPDATE | NONE |

For more details about RACF data set protection, see the *OS/390 Security Server (RACF) Security Administrator's Guide*.

## Defining the CICSPlex SM started tasks

For the CAS (a started task) and CMAS (when it is run as a started task), you must associate the appropriate procedure names with a suitably authorized USERID. This is normally achieved using the STARTED general resource class, or the RACF ICHRIN03 tables. The names of the associated USERIDs need not match the names of the procedures. Each USERID must have the appropriate level of access to all of the data sets referenced in the cataloged procedures.

For additional information about the STARTED class, see the *OS/390 Security Server (RACF) Security Administrator's Guide*. For more information about ICHRIN03, see the *OS/390 Security Server (RACF) System Programmer's Guide*.

**Note:** If the USERID and group name that you assign are not defined to RACF, the started tasks will execute with only the limited authority of an undefined user. In this case, the address space will be able to access protected resources only if the universal access authority (UACC) for the resource is sufficient to allow the requested operation.

## Defining the CICSPlex SM transactions in a CMAS

If transaction–attach security is active in a CMAS (that is, SEC=YES and XTRAN=YES|*classname* are specified in the SIT), you must define to RACF the CICSPlex SM transactions that run in a CMAS. The following is a list of the transaction IDs that you must define to RACF for CICSPlex SM.

- BMLT
- LCPP
- LECI
- LECR
- LECS
- LEEI
- LEER
- LEMI
- LEMS
- LENS
- LMIR
- LNCI
- LNCS
- LNMI
- LNMS
- LPDG
- LPLK
- LPLT
- LPRT
- LPSC
- LPSM
- LRLT
- LSRT
- LWTM
- MCCM
- MCTK
- MMEI
- MMIS
- MMST
- PEAD
- PELT
- PMLT
- PNLT
- PPLT
- PRLT
- PRPR
- PSLT
- TICT
- TIRT
- TIST
- TSMH
- TSPD
- TSSC
- TSSJ
- WMCC
- WMGR
- WMLA
- WMQB
- WMQM
- WMQS
- WMSC
- WMWC
- WMWT
- WSCL
- WSLW
- XDBM
- XDNC
- XDND
- XDNE
- XDNR
- XDNS
- XDSR
- XLEC
- XLEV
- XLNX
- XLST
- XQST

**Note:** A list of these transactions is also contained in the CSD group EYU140G0.

The region userid, and any userid that may be specified on the PLTPIUSR SIT parameter, must have authority to attach these transactions. In addition, and depending on the security attributes specified for any CMTCMDEF or CMTPMDEF, any userids which may flow from connected CMASs or remote MASs should have authority to attach these transactions. See "Part 3. Intercommunication security" on page 145 for information on intercommunication security.

The following transactions are supplied for debugging purposes under the guidance of IBM support personnel, and are associated with a terminal:

CODB
COD0
COD1
COD2
COLU

They must be defined to RACF if transaction security is active, regardless of the CICS/ESA release running as the CMAS. Authority to initiate these transactions should be restricted to only those users who may become involved in working with IBM to resolve CICSPlex SM problems.

The COSD transaction allows a terminal user to shut down a CMAS. Access to this transaction should be granted only to those users who may need to shut down a CMAS

## Defining the CICSPlex SM transactions in a MAS

For MASs capable of running with an external security manager, it may be necessary to define the CICSPlex SM transactions which run in the MAS to the ESM.

If transaction–attach security is active in a MAS running CICS/ESA 4.1 or later (that is, SEC=YES and XTRAN=YES|*classname* are specified in the SIT), you must define to RACF the following transactions in the appropriate class:

COIE
COIR
COI0
COND
CONL
CONM
CORT
COWC

The region userid, and any userid that may be specified on the PLTPIUSR SIT parameter, should be given READ access to these transactions.

For CICS/MVS, CICS/ESA, and CICS TS for OS/390, users who may initiate the MAS agent code using transaction COLM (for a local MAS), or transaction CORM (for a remote MAS), should also be given access to these transactions. Users who may enter dynamic transactions in a CICSPlex SM workload management requesting region must have READ access to the COWC transaction. For more information about creating a CMAS to CMAS link, see For more information about CMAS link definition, see *CICSPlex SM Administration*.

For CICS/MVS, CICS/ESA, and CICS TS for OS/390, users who may invoke the CICSPlex SM debugging transactions should be given READ access to the following transactions:

CODB
COD0
COD1
COD2
COLU

For remote MASs running CICS/MVS, CICS/ESA, or CICS TS for OS/390, define the link manager transactions to RACF. These are:

COI1
COI2
COI3
COI4

The security attributes of the CONNECTION/SESSION pair defined for the link to the CMAS define which users are authorized to run these transactions. See “Part 3. Intercommunication security” on page 145 for information on intercommunication security.

The COSH transaction allows a terminal user to stop MAS agent code execution. Access to this transaction should be restricted to those users who may need to stop the MAS in this way.

## Specifying CAS and PlexManager resource names in profiles

The simplest way to secure the CAS is to control access to the TSO signon procedure or CLIST used to access CICSPlex SM, as described in the *CICS Transaction Server for OS/390 Installation Guide*. This is sufficient for most enterprises. However, you can provide further control over the CAS by creating RACF profiles using the resource names described in Table 39.

To control access to the CAS functions and PlexManager views, you create profiles in the RACF FACILITYclass. Table 39 lists the resource names that you should use in these profiles. In all cases, define READ access to these resources. The following variable names are used in Table 39 to illustrate resource names. When you define your profiles, replace these variable names with the actual value(s) used on your system(s).

**Note:** You must define a profile in order for it to have a level of protection. If no profile exists, resources are unprotected.

**context**
The context being accessed. For PlexManager views, this is the MVS image SMF ID; for CICSPlex SM views, it is the CICSPlex SM context.

**smfid** The SMF ID of the MVS system on which the CAS or CMAS is running.

**ssid** The CAS MVS subsystem ID.

*Table 39. Resource names used by specific functions*

| Function | Resource name | Class name |
|---|---|---|
| For the CAS and the CMAS to define an MVS/ESA subsystem: | | |
| Define the SSCT for the CAS | SUBSYS.ssid.DEFINE | FACILITY |
| For the CAS to initialize as an MVS/ESA subsystem: | | |
| Define, initialize, and use an SSCT | SUBSYS.ssid.INIT | FACILITY |
| For any user or CMAS connecting to the CAS: | | |
| Connect to CAS | BBM.ssid.CN | FACILITY |
| For a user opening a window to a particular context or changing to a new context: | | |
| Access to a service point | **For CAS:** BBM.smfid.PLEXMGR.context.TA<br>**For CMAS:** BBM.smfid.CPSM.context.TA | FACILITY |
| When a CMAS attaches a service point for a context: | | |
| Attach a service point | BBM.smfid.CPSM.context.TC | FACILITY |
| To allow access to the PlexManager views and actions: | | |
| Access to any PLEXMGR specific secured action (currently only CASDEF). | BBM.PLEXMGR.smfid.AA | FACILITY |
| Access to the CASACT view | BBM.PLEXMGR.smfid.CYAD0.OD | FACILITY |
| Access to the CASDEF view | BBM.PLEXMGR.smfid.CYAB0.OD | FACILITY |
| Access to any CASDEF view action | BBM.PLEXMGR.smfid.CYAB0.AO | FACILITY |
| Access to the DIAGSYS view | BBM.PLEXMGR.smfid.CZZ01.OD | FACILITY |
| Access to the DIAGSESS view | BBM.PLEXMGR.smfid.CZZ02.OD | FACILITY |
| Access to PLEX view or PLEXOVER view | BBM.PLEXMGR.smfid.CCE92.OD | FACILITY |
| To allow access to the views and actions which can be accessed from either PlexManager or CICSPlex SM: | | |

*Table 39. Resource names used by specific functions (continued)*

| Function | Resource name | Class name |
|---|---|---|
| Access to any PLEXMGR secured action from the shared views. | **For CAS:**<br>BBM.PLEXMGR.smfid.COMMON.AA<br>**For CMAS:**<br>BBM.CPSM.context.COMMON.AA | FACILITY |
| Access to the VIEWS view | **For CAS:**<br>BBM.PLEXMGR.smfid.MCE90.OD<br>**For CMAS:**<br>BBM.CPSM.context.MCE90.OD | FACILITY |
| Access to the SCREENS view | **For CAS:**<br>BBM.PLEXMGR.smfid.MCE95.OD<br>**For CMAS:**<br>BBM.CPSM.context.MCE95.OD | FACILITY |
| Access to the DIAGMSG view | **For CAS:**<br>BBM.PLEXMGR.smfid.MYA40.OD<br>**For CMAS:**<br>BBM.CPSM.context.MYA40.OD | FACILITY |
| Access to any DIAGMSG view action | **For CAS:**<br>BBM.PLEXMGR.smfid.MYA40.AO<br>**For CMAS:**<br>BBM.CPSM.context.MYA40.AO | FACILITY |
| Access to a specific DIAGMSG view action (ON or OFF) | **For CAS:**<br>BBM.PLEXMGR.smfid.msgsdaid.MYA40.OA<br>**For CMAS:**<br>BBM.CPSM.context.msgsdaid.MYA40.OA<br>where, for msgsdaid, you substitute one of the following values, which appear on the DIAGMSG view on the line where ON or OFF is specified:<br>**GEMM**<br>Extended Message Mode<br>**LEMM** Extended Message Mode<br>**LSEMM**<br>Security Extended Message Mode<br>**LESTR** Extended Security Trace<br>**GESTR**<br>Extended Security Trace<br>**LSSTR** Simple Security Trace<br>**GSSTR**<br>Simple Security Trace<br>**GSSM** Safe Security Message Display<br>**WSXASTR**<br>Extended Authorization Simple Trace | FACILITY |

## Specifying CICSPlex SM resource names in profiles

This section provides the resource names for CICSPlex SM views to be used in RACF profiles. Refer to "Chapter 25. Example tasks: security" on page 313 for profile examples.

You can create RACF profiles for CICSPlex SM views for a specific CICS system, a group of CICS systems, or all systems comprising a CICSplex.

CICSPlex SM views are divided into groups that reflect the functions they perform. Within each functional group, the views are divided by their type. Functional groups can be even further qualified with the addition of a context and, for some groups, a scope. You can control access to a specific set of views (and their associated action commands) by identifying the set in a profile, using the following resource name format:

```
function.type.context[.scope]
```

where:

**function**
The name of the CICSPlex SM function to be affected:

| Function | Meaning |
|---|---|
| **ANALYSIS** | Real-time analysis |
| **BAS** | Business application services |
| **CONFIG** | CMAS configuration |
| **MONITOR** | Resource monitoring |
| **OPERATE** | Operations |
| **TOPOLOGY** | CICSplex configuration |
| **WORKLOAD** | Workload management |

**type** The specific or generic name of an area that qualifies the CICSPlex SM function to be affected. The specific names are:

| Type | Meaning |
|---|---|
| **AIMODEL** | CICS AIMODEL |
| **CONNECT** | CICS connections |
| **DB2DBCTL** | DB2/DBCTL resources and subsystems |
| **DEF** | CICSPlex SM definitions |
| **DOCTEMP** | Document templates |
| **ENQMODEL** | CICS global enqueue models |
| **EXIT** | CICS exits |
| **FEPI** | CICS FEPI resources |
| **FILE** | CICS files |
| **JOURNAL** | CICS journals |

**PARTNER**
CICS partners
**PROFILE**
CICS profiles
**PROGRAM**
CICS programs
**REGION**
CICS region data
**RQMODEL**
CICS request models
**TASK** CICS active tasks
**TCPIPS**
TCP/IP services
**TDQUEUE**
CICS transient data queues
**TERMINAL**
CICS terminals
**TRAN** CICS transactions
**TSQUEUE**
CICS temporary storage queues
**UOW** CICS units of work

The type must be valid for the specified function. Table 40 on page 279 lists the valid `function.type` combinations.

**context**
The specific or generic name of the CMAS or CICSplex to be affected by the designated `function` and `type`. If the `function` is CONFIG or TOPOLOGY, the `context` must be a CMAS. For all other functions, the `context` must be a CICSplex.

**scope** The specific or generic name of a CICS system within the CICSplex identified as the context or the CICSplex itself. This value is ignored when either:

- The context is a CMAS
- The type is DEF

for CICSPlex SM definitions.

**Notes:**

1. In this section only, the term `scope` means CICS systems. It does ***not*** mean the scope (CICS system groups) you have defined as part of the CICSPlex SM environment, nor does it refer to a BAS logical scope.
2. To include all of the systems comprising a CICS system group when their names do not match a generic system name, you must establish a profile for each system.

## Using asterisks in resource names

To reduce the number of profiles you need to define, you can use * (one asterisk) and ** (two consecutive asterisks) to represent one or more entries. Use of one or two asterisks is optional.

**Note:** Before using asterisks in profile definitions, ensure that generics have been activated for the relevant class:

```
SETROPTS GENERIC(CPSMOBJ,CPSMXMP)
```

The following examples demonstrate how asterisks can be used:

**OPERATE.*.EYUPLX01.EYUPLX01**
: Indicates that all views and action commands associated with any type valid within the OPERATE function are to be recognized when the context and scope are EYUPLX01.

**OPERATE.PROGRAM.****
: Indicates that all views and action commands associated with the PROGRAM type within the OPERATE function are to be recognized, regardless of the current context and scope.

**OPERATE.****
: Indicates that all views and action commands associated with any type valid within the OPERATE function are to be recognized, regardless of the current context and scope.

**
: Indicates that *all* views and action commands associated with *any* type valid within *any* function are to be recognized, regardless of the current context and scope.

## Valid resource name combinations

Table 40 on page 279 lists the valid function and type combinations and the set of general views associated with each combination. Summary and detail views are not listed in this table, but are included in the sets of views.

**Note to CICSPlex SM API users:** You can use the function and type combinations in Table 40 on page 279 when creating profiles to control access to resource tables from the CICSPlex SM API. A view name usually, but not always, matches the name of its corresponding resource table. In Table 40 on page 279, if the two names differ, the view name is listed first and is followed by the resource table name enclosed in parentheses. Resource tables that do not have a corresponding view, but are accessible via the CICSPlex SM API, are listed in Table 41 on page 289.

*Table 40. Function and type combinations for resources accessible via the CICSPlex SM EUI or API*

| Function.Type | View (Resource Table) Usage |
|---|---|
| ANALYSIS.DEF | **APACTV**<br>Display analysis definitions associated with an analysis point specification |
| | **ACTNDEF (ACTION)**<br>Create, display, and maintain action definitions |
| | **APCMAS**<br>Display analysis point specification to CMAS |
| | **APSPEC**<br>Create, display, and maintain analysis point specifications |
| | **EVALDEF**<br>Create, display, and maintain evaluation definitions |
| | **EVENT**<br>Display changes in the status of a CICSplex |
| | **EVENTDTL**<br>Display evaluation definitions associated with an analysis definition that caused an event |
| | **RTAACTV**<br>Display analysis and status definitions in CICS systems |
| | **RTADEF**<br>Create, display, and maintain analysis definitions |
| | **RTAGROUP**<br>Create, display, and maintain analysis groups |
| | **RTAINAPS**<br>Display analysis groups in analysis point specifications |
| | **RTAINGRP**<br>Display analysis and status definitions in analysis groups |
| | **RTAINSPC**<br>Display analysis groups in analysis specifications |
| | **RTASPEC**<br>Create, display, and maintain analysis specifications |
| | **STATDEF**<br>Create, display, and maintain status definitions |
| BAS.CONNECT | **CONNDEF**<br>Install connection definitions. |
| | **SESSDEF**<br>Install session definitions. |
| BAS.DB2DBCTL | **DB2CDEF**<br>Install DB2 connection definitions |
| | **DB2EDEF**<br>Install DB2 entry definitions |
| | **DB2TDEF**<br>Install DB2 transaction definitions |

*Table 40. Function and type combinations for resources accessible via the CICSPlex SM EUI or API (continued)*

| Function.Type | View (Resource Table) | Usage |
|---|---|---|
| BAS.DEF | **CONNDEF** | Create, display, maintain, and install connection definitions. |
| | **DB2CDEF** | Create, display, maintain, and install DB2 connection definitions. |
| | **DB2EDEF** | Create, display, maintain, and install DB2 entry definitions. |
| | **DB2TDEF** | Create, display, maintain, and install DB2 transaction definitions. |
| | **DOCDEF** | Create, display, maintain, and install document template definitions. |
| | **ENQMDEF** | Create, display, maintain, and install enqueue models definitions. |
| | **FENODDEF** | Create, display, maintain, and install FEPI node definitions. |
| | **FEPOODEF** | Create, display, maintain, and install FEPI pool definitions. |
| | **FEPRODEF** | Create, display, maintain, and install FEPI property set definitions. |
| | **FETRGDEF** | Create, display, maintain, and install FEPI target definitions. |
| | **FILEDEF** | Create, display, maintain, and install file definitions. |
| | **FSEGDEF** | Create, display, and maintain OS/2 key file segment definitions. |
| | **JRNLDEF** | Create, display, maintain, and install journal definitions. |
| | **JRNMDEF** | Create, display, maintain, and install journal model definitions. |
| | **LSRDEF** | Create, display, maintain, and install LSR pool definitions. |
| | **MAPDEF** | Create, display, maintain, and install mapset definitions. |
| | **PARTDEF** | Create, display, maintain, and install partner definitions. |
| | **PROCDEF** | Create, display, maintain, and install process type definitions. |
| | **PROFDEF** | Create, display, maintain, and install profile definitions. |
| | **PROGDEF** | Create, display, maintain, and install program definitions. |
| | **PRTNDEF** | Create, display, maintain, and install partition set definitions. |
| | **RASGNDEF** | Create, display, and maintain resource assignments |

*Table 40. Function and type combinations for resources accessible via the CICSPlex SM EUI or API (continued)*

| Function.Type | View (Resource Table) Usage |
|---|---|
| BAS.DEF cont | **RASINDSC**<br>Display resource assignments in descriptions |
| | **RASPROC**<br>Display resource assignment process |
| | **RDSCPROC**<br>Display resource description process |
| | **RESDESC**<br>Create, display, maintain, and install resource descriptions |
| | **RESGROUP**<br>Create, display, maintain, and install resource groups |
| | **RESINDSC**<br>Display resource groups in descriptions |
| | **RESINGRP**<br>Display resource definitions in groups |
| | **RQMDEF**<br>Create, display, maintain, and install request model definitions. |
| | **SESSDEF**<br>Create, display, maintain, and install session definitions |
| | **SYSRES**<br>Display CICS system resources |
| | **TCPDEF**<br>Create, display, maintain, and install TCP/IP service definitions. |
| | **TDQDEF**<br>Create, display, maintain, and install transient data queue definitions |
| | **TERMDEF**<br>Create, display, maintain, and install terminal definitions |
| | **TRANDEF**<br>Create, display, maintain, and install transaction definitions |
| | **TRNCLDEF**<br>Create, display, maintain, and install transaction class definitions |
| | **TYPTMDEF**<br>Create, display, maintain, and install typeterm definitions |
| BAS.FILE | **FILEDEF**<br>Install file definitions. |
| BAS.JOURNAL | **JRNLDEF**<br>Install journal definitions. |
| | **JRNMDEF**<br>Install journal model definitions. |
| BAS.PARTNER | **PARTDEF**<br>Install partner definitions. |

*Table 40. Function and type combinations for resources accessible via the CICSPlex SM EUI or API (continued)*

| Function.Type | View (Resource Table) Usage |
|---|---|
| BAS.PROFILE | **PROFDEF**<br>Install profile definitions. |
| BAS.PROGRAM | **MAPDEF**<br>Install map set definitions.<br>**PROGDEF**<br>Install program definitions.<br>**PRTNDEF**<br>Install partition set definitions. |
| BAS.REGION | **LSRDEF**<br>Install LSR pool definitions.<br>**TRNCLDEF**<br>Install transaction class definitions. |
| BAS.TDQUEUE | **TDQDEF**<br>Install transient data queue definitions. |
| BAS.TERMINAL | **TERMDEF**<br>Install terminal definitions.<br>**TYPTMDEF**<br>Install typeterm definitions. |
| BAS.TRAN | **TRANDEF**<br>Install transaction definitions. |
| CONFIG.DEF | **CICSPLEX**<br>Display and manage CMAS in CICSplex<br>**CMAS** Display and manage active CMASs<br>**CMASPLEX**<br>Display CICSplexes for a CMAS<br>**CMTCMDEF**<br>Create, display, and maintain CMAS links<br>**CMTCMLNK**<br>Display active CMAS links<br>**CMTPMDEF**<br>Create, display, and maintain remote MAS links<br>**CMTPMLNK**<br>Display active remote MAS links<br>**CPLEXDEF**<br>Create, display, and maintain CICSplex definitions<br>**CPLXCMAS**<br>Display CMAS to CICSplex |

*Table 40. Function and type combinations for resources accessible via the CICSPlex SM EUI or API (continued)*

| Function.Type | View (Resource Table) Usage |
|---|---|
| MONITOR.CONNECT | **MCONNECT**<br>ISC and MRO connections |
| | **MMODENAME**<br>LU 6.2 modenames |
| MONITOR.DB2DBCTL | **MDB2THRD**<br>DB2 threads |
| MONITOR.DEF | **MONACTV**<br>Display Active and Pending monitor definitions |
| | **MONDEF**<br>Create, display, and maintain monitor definitions |
| | **MONGROUP**<br>Create, display, and maintain monitor groups |
| | **MONINGRP**<br>Create, display, and maintain monitor definitions in monitor groups |
| | **MONINSPC**<br>Create, display, and maintain monitor groups in monitor specifications |
| | **MONSPEC**<br>Create, display, and maintain monitor specifications |
| MONITOR.FEPI | **MFECON (MFEPICON)**<br>FEPI connections |
| MONITOR.FILE | **MCMDT**<br>Data tables |
| | **MLOCFILE**<br>Local files |
| | **MREMFILE**<br>Remote files |
| MONITOR.JOURNAL | **MJOURNAL**<br>Journals |
| MONITOR.PROGRAM | **MPROGRAM**<br>Programs |

*Table 40. Function and type combinations for resources accessible via the CICSPlex SM EUI or API (continued)*

| Function.Type | View (Resource Table) Usage |
|---|---|
| MONITOR.REGION | **MCICSDSA**<br>Dynamic storage areas |
| | **MCICSRGN**<br>CICS systems |
| | **MLSRPBUF**<br>LSRPOOL buffer pool |
| | **MLSRPOOL**<br>LSRPOOL |
| | **MTRNCLS (MTRANCLS)**<br>Transaction classes |
| MONITOR.TDQUEUE | **MINDTDQ**<br>Indirect transient data queues |
| | **MINTRATDQ**<br>Intrapartition transient data queues |
| | **MREMTDQ**<br>Remote transient data queues |
| | **MTDQGBL**<br>Global intrapartition transient data queues |
| | **MXTRATDQ**<br>Extrapartition transient data queues |
| MONITOR.TERMINAL | **MTERMNL**<br>Terminals |
| MONITOR.TRAN | **MLOCTRAN**<br>Local transactions |
| | **MREMTRAN**<br>Remote transactions |
| OPERATE.AIMODEL | **AIMODEL**<br>Auto install models |
| OPERATE.CONNECT | **CONNECT**<br>ISC connections |
| | **MODENAME**<br>LU 6.2 modenames |

*Table 40. Function and type combinations for resources accessible via the CICSPlex SM EUI or API (continued)*

| Function.Type | View (Resource Table) Usage |
|---|---|
| OPERATE.DB2DBCTL | **DB2CONN** DB2 connection<br>**DB2ENTRY** DB2 entry<br>**DB2TRN** DB2 transaction<br>**DBCTLSS** DBCTL subsystem<br>**DB2SS** DB2 subsystem<br>**DB2THRD** DB2 threads<br>**DB2TRAN** DB2 transactions |
| OPERATE.DOCTEMP | **DOCTEMP** Document templates |
| OPERATE.ENQMODEL | **ENQMODEL** Enqueue models |
| OPERATE.EXIT | **EXITGLUE** Global user exits<br>**EXITTRUE** Task-related user exits |
| OPERATE.FEPI | **FECONN** FEPI connections<br>**FENODE** FEPI nodes<br>**FEPOOL** FEPI pools<br>**FEPROP** FEPI property sets<br>**FETRGT** FEPI targets |
| OPERATE.FILE | **CMDT** Data tables<br>**DSNAME** Data sets<br>**LOCFILE** Local files<br>**REMFILE** Remote files |

*Table 40. Function and type combinations for resources accessible via the CICSPlex SM EUI or API (continued)*

| Function.Type | View (Resource Table) Usage |
|---|---|
| OPERATE.JOURNAL | **DSKJRNL**<br>Disk journal<br>**JOURNAL**<br>Journals<br>**JRNLMODL**<br>Journal models<br>**JRNLNAME**<br>System logs and general logs<br>**SMFJRNL**<br>SMF journals<br>**STREAMNM**<br>MVS log streams<br>**TAPJRNL**<br>Tape journals<br>**VOLUME**<br>Tape journal volumes |
| OPERATE.PARTNER | **PARTNER**<br>CICS partners |
| OPERATE.PROCTYPE | **PROCTYP**<br>Process types |
| OPERATE.PROFILE | **PROFILE**<br>CICS profiles |
| OPERATE.PROGRAM | **PROGRAM**<br>Programs<br>**RPLLIST**<br>DFHRPL data sets |

*Table 40. Function and type combinations for resources accessible via the CICSPlex SM EUI or API (continued)*

| Function.Type | View (Resource Table) Usage |
|---|---|
| OPERATE.REGION | **CICSDSA** Dynamic storage areas<br>**CICSRGN** CICS systems<br>**LSRPBUF** Buffer usage for LSR pools<br>**LSRPOOL** LSR pools<br>**REQID** Timed requests<br>**SYSDUMP** System dump codes<br>**TRANCLS (TRANCLAS)** Transaction classes<br>**TRANDUMP** Transaction dump codes |
| OPERATE.RQMODEL | **RQMODEL** Request models |
| OPERATE.TASK | **TASK** Active tasks |
| OPERATE.TCPIPS | **TCPIPS** TCP/IP services |
| OPERATE.TDQUEUE | **EXTRATDQ** Extrapartition transient data queues<br>**INDTDQ** Indirect transient data queues<br>**INTRATDQ** Intrapartition transient data queues<br>**QUEUE** Transient data queues<br>**REMTDQ** Remote transient data queues<br>**TDQGBL** Intrapartition transient data queue usage |
| OPERATE.TERMINAL | **TERMNL** Terminals |
| OPERATE.TRAN | **LOCTRAN** Local transactions<br>**REMTRAN** Remote transactions<br>**TRAN** Transactions |

*Table 40. Function and type combinations for resources accessible via the CICSPlex SM EUI or API (continued)*

| Function.Type | View (Resource Table) Usage |
|---|---|
| OPERATE.UOW | **UOWDSNF**<br>Display shunted units of work |
| | **UOWENQ**<br>Display enqueues for executing units of work |
| | **UOWLINK**<br>Display links for unit of work |
| | **UOWORK (UOW)**<br>Display executing units of work |
| TOPOLOGY.DEF | **CICSGRP (CSYSGRP)**<br>Create, display, and maintain CICS system groups |
| | **CICSSYS (CSYSDEF)**<br>Create, display, and maintain CICS systems |
| | **MAS** Display CICS systems in a CICSplex |
| | **MONSCOPE**<br>Create, display, and maintain monitor systems and monitor system groups in monitor specifications |
| | **PERIODEF**<br>Display period definitions |
| | **RTASCOPE**<br>Display analysis specifications assigned a scope |
| | **SYSGRPC**<br>Create, display, and maintain the contents of CICS system groups |
| | **SYSLINK**<br>Display information about the links that exist between CICS systems. |
| | **WLMSCOPE**<br>Display workload systems and workload system groups in specifications |

*Table 40. Function and type combinations for resources accessible via the CICSPlex SM EUI or API (continued)*

| Function.Type | View (Resource Table) Usage |
|---|---|
| WORKLOAD.DEF | **DTRINGRP**<br>Display transactions in transaction groups |
| | **TRANGRP**<br>Create, display, and maintain transaction groups |
| | **WLMATAFF**<br>Display and discard active affinities |
| | **WLMATGRP**<br>Display and discard active transaction groups |
| | **WLMATRAN**<br>Display and discard transaction directory |
| | **WLMAWAOR**<br>Display active AORs in a workload |
| | **WLMAWDEF**<br>Display and discard active workload definitions |
| | **WLMAWORK**<br>Display active workloads |
| | **WLMAWTOR**<br>Display active AORs in a workload |
| | **WLMDEF**<br>Create, display, and maintain workload definitions |
| | **WLMGROUP**<br>Create, display, and maintain workload groups |
| | **WLMINGRP**<br>Display workload definitions in groups |
| | **WLMINSPC**<br>Display workload groups in workload specifications |
| | **WLMSPEC**<br>Create, display, and maintain workload specifications |

Table 41 lists those resource table accessible via the API only.

*Table 41. Function and type combinations for resources accessible via the API only*

| Function.Type | Resource Table Usage |
|---|---|
| ANALYSIS.DEF | **CMDMPAPS**<br>Resource table only. Identify the role of a primary CMAS |
| | **CMDMSAPS**<br>Resource table only. Identify the role of a secondary CMAS |
| | **LNKSRSCG**<br>Describe the link between a CICS system group and an analysis specification |
| | **LNKSRSCS**<br>Describe the link between a CICS system and an analysis specification |
| | **STAINGRP**<br>Resource table only. Identify the membership relation of a status definition in an RTAGROUP |

*Table 41. Function and type combinations for resources accessible via the API only (continued)*

| Function.Type | Resource Table | Usage |
|---|---|---|
| BAS.DEF | **CONINGRP** | Describe the membership of a connection definition in a resource group |
| | **DOCINGRP** | Describe the membership of a document template definition in a resource group |
| | **D2CINGRP** | Describe the membership of a DB2 connection definition in a resource group |
| | **D2EINGRP** | Describe the membership of a DB2 entry definition in a resource group |
| | **D2TINGRP** | Describe the membership of a DB2 transaction definition in a resource group |
| | **ENQINGRP** | Describe the membership of an ENQ/DEQ model definition in a resource group |
| | **FILINGRP** | Describe the membership of a file definition in a resource group |
| | **FNOINGRP** | Describe the membership of a FEPI node definition in a resource group |
| | **FPOINGRP** | Describe the membership of a FEPI pool definition in a resource group |
| | **FPRINGRP** | Describe the membership of a FEPI property set definition in a resource group |
| | **FSGINGRP** | Describe the membership of a file key segment definition in a resource group |
| | **FTRINGRP** | Describe the membership of a FEPI target definition in a resource group |
| | **JRMINGRP** | Describe the membership of a journal model definition in a resource group |
| | **JRNINGRP** | Describe the membership of a journal definition in a resource group |
| | **LSRINGRP** | Describe the membership of an LSR pool definition in a resource group |
| | **MAPINGRP** | Describe the membership of a map set definition in a resource group |
| | **PARINGRP** | Describe the membership of a partner definition in a resource group |
| | **PGMINGRP** | Describe the membership of a program definition in a resource group |
| | **PRCINGRP** | Describe the membership of a process type definition in a resource group |
| | **PRNINGRP** | Describe the membership of a partition set definition in a resource group |
| | **PROINGRP** | Describe the membership of a profile definition in a resource group |

*Table 41. Function and type combinations for resources accessible via the API only (continued)*

| Function.Type | Resource Table | Usage |
|---|---|---|
| BAS.DEF cont | **RQMINGRP** | Describe the membership of a request model definition in a resource group |
| | **SESINGRP** | Describe the membership of a session definition in a resource group |
| | **TCLINGRP** | Describe the membership of a transaction class definition in a resource group |
| | **TCPINGRP** | Describe the membership of a TCPIP Service definition in a resource group |
| | **TDQINGRP** | Describe the membership of a transient data queue definition in a resource group |
| | **TRMINGRP** | Describe the membership of a terminal definition in a resource group |
| | **TRNINGRP** | Describe the membership of a transaction definition in a resource group |
| | **TSMINGRP** | Describe the membership of a temporary storage model definition in a resource group |
| | **TYPINGRP** | Describe the membership of a typeterm definition in a resource group |
| CONFIG.DEF | **CMASLIST** | Resource table only. Describe a CMAS and its connection to characteristics |
| MONITOR.DEF | **LNKSMSCG** | Describe the link between a CICS system group and a monitor specification |
| | **LNKSMSCS** | Describe the link between a CICS system and a monitor specification |
| | **POLMON** | Resource table only. Describe a monitor definition in a specific CICS system |
| OPERATE.AIMODEL | **CRESAIMD** | Describe an instance of an autoinstalled terminal model within a CICS system |
| OPERATE.CONNECT | **CRESCONN** | Describe an instance of an ISC connection within a CICS system |
| | **CRESMODE** | Describe an instance of an LU6.2 modename within a CICS system |
| OPERATE.DB2DBCTL | **CRESDB2C** | Describe an instance of a DB2 connection within a CICS system |
| | **CRESDB2E** | Describe an instance of a DB2 entry within a CICS system |
| | **CRESDB2T** | Describe an instance of a DB2 transaction within a CICS system |

*Table 41. Function and type combinations for resources accessible via the API only (continued)*

| Function.Type | Resource Table | Usage |
|---|---|---|
| OPERATE.DOCTEMP | **CRESDOCT** | Describe an instance of a document template within a CICS system |
| OPERATE.ENQMODEL | **CRESENQM** | Describe an instance of an ENQ/DEQ model within a CICS system |
| OPERATE.EXIT | **CRESGLUE** | Describe an instance of a global user exit within a CICS system |
| | **CRESTRUE** | Describe an instance of a task-related user exit within a CICS system |
| OPERATE.FEPI | **CRESFECO** | Describe an instance of a FEPI connection within a CICS system |
| | **CRESFEND** | Describe an instance of a FEPI node within a CICS system |
| | **CRESFEPO** | Describe an instance of a FEPI pool within a CICS system |
| | **CRESFETR** | Describe an instance of a FEPI target within a CICS system |
| OPERATE.FILE | **CRESDSNM** | Describe an instance of a data set within a CICS system |
| | **CRESFILE** | Describe an instance of a file within a CICS system |
| OPERATE.JOURNAL | **CRESJRNL** | Describe an instance of a journal within a CICS system |
| | **CRESJRNM** | Describe an instance of a journal name within a CICS system |
| OPERATE.PARTNER | **CRESPART** | Describe an instance of a partner table within a CICS system |
| OPERATE.PROCTYPE | **CRESPRTY** | Describe an instance of a process type within a CICS system |
| OPERATE.PROFILE | **CRESPROF** | Describe an instance of a profile within a CICS system |
| OPERATE.PROGRAM | **CRESPRGM** | Describe an instance of a program within a CICS system |
| OPERATE.REGION | **CRESSDMP** | Describe an instance of a system dump code within a CICS system |
| | **CRESTDMP** | Describe an instance of a transaction dump code within a CICS system |

*Table 41. Function and type combinations for resources accessible via the API only (continued)*

| Function.Type | Resource Table | Usage |
|---|---|---|
| OPERATE.RQMODEL | **CRESRQMD** | Describe an instance of a request within a CICS system |
| OPERATE.TCPIPS | **CRESTCPS** | Describe an instance of a TCPIP service within a CICS system |
| OPERATE.TDQUEUE | **CRESTDQ** | Describe an instance of a transient data queue within a CICS system |
| OPERATE.TERMINAL | **CRESTERM** | Describe an instance of a terminal within a CICS system |
| OPERATE.TRAN | **CRESTRAN** | Describe an instance of a transaction within a CICS system |
| OPERATE.TSQUEUE | **CRESTSMD** | Describe an instance of a temporary storage queue within a CICS system |
| TOPOLOGY.DEF | **CSGLCGCG** | Describe the link of a CICS system group to an outer system group |
| | **CSGLCGCS** | Describe the link of a CICS system to a system group |
| WORKLOAD.DEF | **LNKSWSCG** | Describe the link between a CICS system group and a workload specification |
| | **LNKSWSCS** | Describe the link between a CICS system and a workload specification |

## Activating simulated CICS security

When you create RACF profiles using the CICSPlex SM resource classes to permit access to the operations and monitoring views, CICSPlex SM determines which views a user can access. However, CICSPlex SM cannot determine if that user is authorized to access the CICS resources represented within the view.

You can enhance the security provided by your CICSPlex SM profiles by activating *simulated CICS security checking*. Simulated security uses your existing RACF profiles to control access to CICS resources and/or CICS commands. It is available only for the operations and monitor views. When using this combination of profiles, your CICSPlex SM profiles determine which sets of views can be accessed and your CICS resource profiles determine which resources within the view can be accessed. For example, you can create a CICSPlex SM profile that allows a user to issue the file view commands and any associated action commands, and then have CICS simulated security determine which files the user is authorized to access.

To activate or deactivate simulated security checking, use the CICSSYS view (for a single CICS system) or CPLEXDEF view (for multiple systems). You can indicate whether you want CICS resource checking, CICS command checking, or both, to

occur. CICS resource checking controls which resources are displayed in a view. CICS command checking controls what commands can be used within the view. The CICSSYS and CPLEXDEF views are described in the *CICSPlex SM Administration*.

To activate or deactivate simulated security checking temporarily for an active CICS system, use the MAS view (as described in the *CICSPlex SM Operations Views Reference* book).

**Notes:**

1. Refer to Table 42 on page 295 for important information on how the CICSPlex SM and CICS security parameters can affect simulated security.
2. Simulated security involves significantly more processing overhead than using only CICSPlex SM profiles and will have a negative impact on performance.

## Simulated CICS security checking exemptions

There may be certain individuals who need not be subject to simulated security checking. There may also be certain CICS resources that are sufficiently protected by CICSPlex SM profiles and, therefore, do not need to be involved in security checking. You can exempt these individuals and resources from simulated CICS security checking using the CICSPlex SM CPSMXMP resource class.

To create exemption profiles use the resource name format described in "Specifying CICSPlex SM resource names in profiles" on page 276.

For example, you might want to define an exemption profile that allows the individuals comprising the group EYUGRP2 to bypass security checking for all views and action commands associated with the TERMINAL type within the MONITOR function, when the context is EYUPLX01 and the scope is EYUMAS1A:

```
PERMIT  MONITOR.TERMINAL.EYUPLX01.EYUMAS1A  /* Resource name    */+
        CLASS(CPSMXMP)                      /* Class name       */+
        ACCESS(UPDATE)                      /* Access           */+
        ID(EYUGRP2)                         /* User or group    */+
                                            /*  granted access  */
```

Exemption bypasses only the simulated CICS security checks, not the basic CICSPlex SM resource checks. For example, if a user does not have RACF authority to issue the CICS command `CEMT INQ FILE`, you can enable that user to achieve the same result by creating a profile in the exemption class that allows the user to issue the equivalent CICSPlex SM command `LOCFILE`.

# Activating security parameters

To activate security for CICSPlex SM, you must:

- Specify the CICSPlex SM parameter SEC in the EYUPARM data set or member defined in the JCL used to start the CMAS and MAS, as described in the *CICS Transaction Server for OS/390 Installation Guide*.
- Specify the CICS parameter SEC= in the CICS system initialization parameters table used to start the MAS, as described in the *CICS Transaction Server for OS/390 Installation Guide*.

Together these parameters determine what security checking is performed. Table 42 on page 295 explains the possible parameter combinations.

*Table 42. Parameters controlling security checking*

| CMAS (CICSPlex SM parameter) | MAS (CICS SIT parameter) | Explanation |
|---|---|---|
| SEC(YES) | SEC=YES | Both view selection checking and simulated security checking can occur, depending on the settings in the CICSSYS and CPLEXDEF views. This means that after CICSPlex SM determines whether a user can display a particular view, simulated security determines what information can be provided in the view. |
| SEC(YES) | SEC=NO | View selection occurs; simulated security checking **does not** occur even if it is requested in the CICSSYS or CPLEXDEF views. This means that after CICSPlex SM determines whether a user can display a designated view, no simulated security checking is performed to determine what information is to be provided in that view. |
| SEC(NO) | SEC=YES | CICSPlex SM does not allow the MAS to connect to the CMAS. This prevents a MAS that has requested security from connecting to a CMAS that cannot provide security. |

**Note:** CICSPlex SM honors any of the CICS SIT parameters XCMD, XDCT, XFCT, XJCT, XPCT, and XPPT; that is, CICSPlex SM includes or excludes the designated commands and resources from security checking. For each MAS, you can specify YES, NO, or CLASS NAME for these CICS SIT parameters. However, for the CMAS, you *must* specify NO for each of these CICS SIT parameters.

## Verifying CICSPlex SM global security parameters

The CICSPlex SM default global security parameters are contained in member BBMTSS of the data set defined by the BBACTDEF DD statement in the CAS procedure. Changes, or overrides, to the default security parameters should be placed in member BBMTSS00 of the data set defined by the BBSECURE DD statement in the CAS procedure.

Member BBMTSS contains the following external security manager (ESM) statement:

```
ESM ESMTYPE(RACF)      /* ESM TYPE IS RACF                        */
    ESMUID(REQUIRE)    /* ESM-DEFINED USERIDS ARE REQUIRED        */
    ESMGRINH(ALLOW)    /* ALWAYS ALLOW GROUP IDENT INHERITANCE    */
    PRODUCTS(CPSM)     /* SECURITY FOR PRODUCT CPSM               */
    ;
```

The ESM parameters are as follows:

**ESMTYPE(esmtype)**
Specify:

**RACF** RACF (or another SAF-compatible ESM) is used on the MVS system.

**NONE**
To bypass security.

Refer to "Overriding RACF security" on page 296 for details.

**ESMUID(REQUIRE)**
Specifies that ESM user ID processing is required.

**ESMGRINH(grinhopt)**
Controls inheritance of the ESM GROUP IDENT for a user ID from an extracted security environment to a target system (that is, whether a user ID ESM GROUP IDENT on one system is to be used when signing the user ID on t another system where cross-system CAS-to-CAS communication is required). Specify the following values:

**ALLOW**
The user ID ESM GROUP IDENT is inherited.

**IGNORE**
The user ID ESM GROUP IDENT is not inherited.

**Note:** IBM MVS security and integrity guidelines state that when a security environment in inherited from one address space to another (such as CAS on another MVS system) the ESM GROUP IDENT must be propagated. The CICSPlex SM CAS-to-CAS interface adheres to this requirement. However, for those customers that define an identically-named user profile on all systems, but do not define identical ESM GROUP IDENTs, CICSPlex SM provides the option to ignore the ESM GROUP IDENT when cross-system CAS-to-CAS communication is required. It is strongly recommended, however, that you abide by the MVS security and integrity guidelines and continue to use the distributed statement of ESMGRINH(ALLOW)

**PRODUCTS(CPSM)**
Specifies that CICSPlex SM is the product for which security processing is being performed.

Verify that in the CAS startup JCL the BBSECURE DD statement identifies the library containing a member named BBMTSS00 and that this member contains at least the following:

```
ESM
    ESMUID(REQUIRE)     /* ALL USERIDS MUST BE DEFINED TO ESM   */
;
```

## Overriding RACF security

By default, CICSPlex SM specifies RACF as its external security manager. If you plan to use an external security manager that is **not** SAF-compatible, you can bypass security for the entire subsystem by adding the following statement to the member BBMTSS00 in the data set defined by BBSECURE DD statement in the CAS procedure:

```
ESMTYPE(NONE)
```

Specifying ESMTYPE(NONE) effectively bypasses security for the entire subsystem. During CAS initialization, a SAF-compatible call is made to the ESM using the following parameters:

**Entity name**
BBMSS.ESMTYPE.NONE

**Class name**
FACILITY

The ESM (or a user-supplied MVS router exit) must exist to authorize ESMTYPE(NONE), and the CAS must be permitted UPDATE access to it. The ESM (or user-supplied MVS router exit) must respond to this request with a return code of zero (0). Otherwise CAS installation will be terminated. See "Chapter 23. Invoking a user-supplied external security manager" on page 303 for more details about the MVS router exit.

## Refreshing RACF profiles

To eliminate unnecessary I/O to the RACF database, CICSPlex SM requires RACF to create copies of several resource classes.

- During CMAS initialization, global copies of RACF profiles in the CPSMOBJ, GCPSMOBJ, and CPSMXMP resource classes are created.
- During MAS initialization, global copies of the RACF profiles for the CICS resource classes used in the MAS are created.

Once these global copies are created, changes to the profiles in the RACF database do not take affect until they are refreshed by the following RACF command:

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

## CICSPlex SM security checking sequence

A user can issue a single CICSPlex SM command that causes data to be gathered about or an action to be performed against one or more CICS systems comprising a CICSplex. These CICS systems can reside in different MVS images.

When a user issues a request, the request is directed to the CMAS that manages the target CICS system or systems. Figure 42 on page 300 and Figure 43 on page 301 are flowcharts showing the procedure followed by CICSPlex SM to evaluate the security requirements of a request from a user. Here is a description of that procedure:

*Step 1*:

**CICSPlex SM determines whether CICSPlex SM rules allow the request to be processed.**

- When the request *can* be processed, CICSPlex SM goes to Step 2.
- When the request *cannot* be processed, CICSPlex SM terminates the request and issues an error message.

*Step 2*:

**CICSPlex SM determines whether simulated CICS security checking is to be performed.**

- When simulated CICS security checking *is* to be performed, CICSPlex SM goes to Step 3.
- When simulated CICS security checking *is not* to be performed, CICSPlex SM goes to Step 9.

*Step 3*:

**CICSPlex SM determines whether the user is exempt from simulated CICS security checking.**

- When the user *is* exempt from simulated CICS security checking, CICSPlex SM goes to Step 9.
- When the user *is not* exempt from simulated CICS security checking, CICSPlex SM goes to Step 4.

*Step 4*:

**CICSPlex SM determines whether simulated CICS command checking is to be performed.**

- When simulated CICS command checking *is* to be performed, CICSPlex SM goes to Step 5.
- When simulated CICS command checking *is not* to be performed, CICSPlex SM goes to Step 6.

*Step 5*:

**CICSPlex SM determines whether the user is allowed to process the command.**

- When the user *is* allowed to process the command, CICSPlex SM goes to Step 6.
- When the user *is not* allowed to process the command, CICSPlex SM terminates the request and issues an error message.

*Step 6*:

**CICSPlex SM determines whether the request is an action (not a request for information).**

- When the request *is* an action, CICSPlex SM goes to Step 7.
- When the request *is not* an action, CICSPlex SM goes to Step 9.

*Step 7*:

**CICSPlex SM determines whether simulated CICS resource checking is to be performed.**

- When simulated CICS resource checking *is* to be performed, CICSPlex SM goes to Step 8.
- When simulated CICS resource checking *is not* to be performed, CICSPlex SM goes to Step 9.

*Step 8*:

**CICSPlex SM determines whether the user is allowed access to information about the resource.**

- When the user *is* allowed access to information about the resource, CICSPlex SM goes to Step 9.
- When the user *is not* allowed access to information about the resource, CICSPlex SM terminates the request and issues an error message.

*Step 9*:

**CICSPlex SM performs the action or gets the information.**

CICSPlex SM then goes to Step 10.

*Step 10*:

**CICSPlex SM determines whether the request is an action (not a request for information).**

- When the request *is* an action, CICSPlex SM returns the results of the action.
- When the request *is not* an action, CICSPlex SM goes to Step 11.

*Step 11*:

**CICSPlex SM determines whether simulated CICS security checking is to be performed.**

- When simulated CICS security checking *is* to be performed, CICSPlex SM goes to Step 12.
- When simulated CICS security checking *is not* to be performed, CICSPlex SM returns the requested information in the appropriate view.

*Step 12*:

**CICSPlex SM determines whether the user is exempt from simulated CICS security checking.**

- When the user *is* exempt from simulated CICS security checking, CICSPlex SM returns the requested information in the appropriate view.
- When the user *is not* exempt from simulated CICS security checking, CICSPlex SM goes to Step 13.

*Step 13*:

**CICSPlex SM determines whether simulated CICS resource checking is to be performed.**

- When simulated CICS resource checking *is* to be performed, CICSPlex SM goes to Step 14.
- When simulated CICS resource checking *is not* to be performed, CICSPlex SM returns the requested information in the appropriate view.

*Step 14*:

**CICSPlex SM determines whether the user is allowed access to information about the resource.**

- When the user *is* allowed access to information about the resource, CICSPlex SM goes to Step 15.
- When the user *is not* allowed access to information about the resource, CICSPlex SM excludes the requested information from the appropriate view.

*Step 15*:

**CICSPlex SM determines whether information for another resource is requested.**

- When information for another resource *is* requested, CICSPlex SM goes to Step 14.
- When information for another resource *is not* requested, CICSPlex SM returns the requested information in the appropriate view.

No further security checking is required.

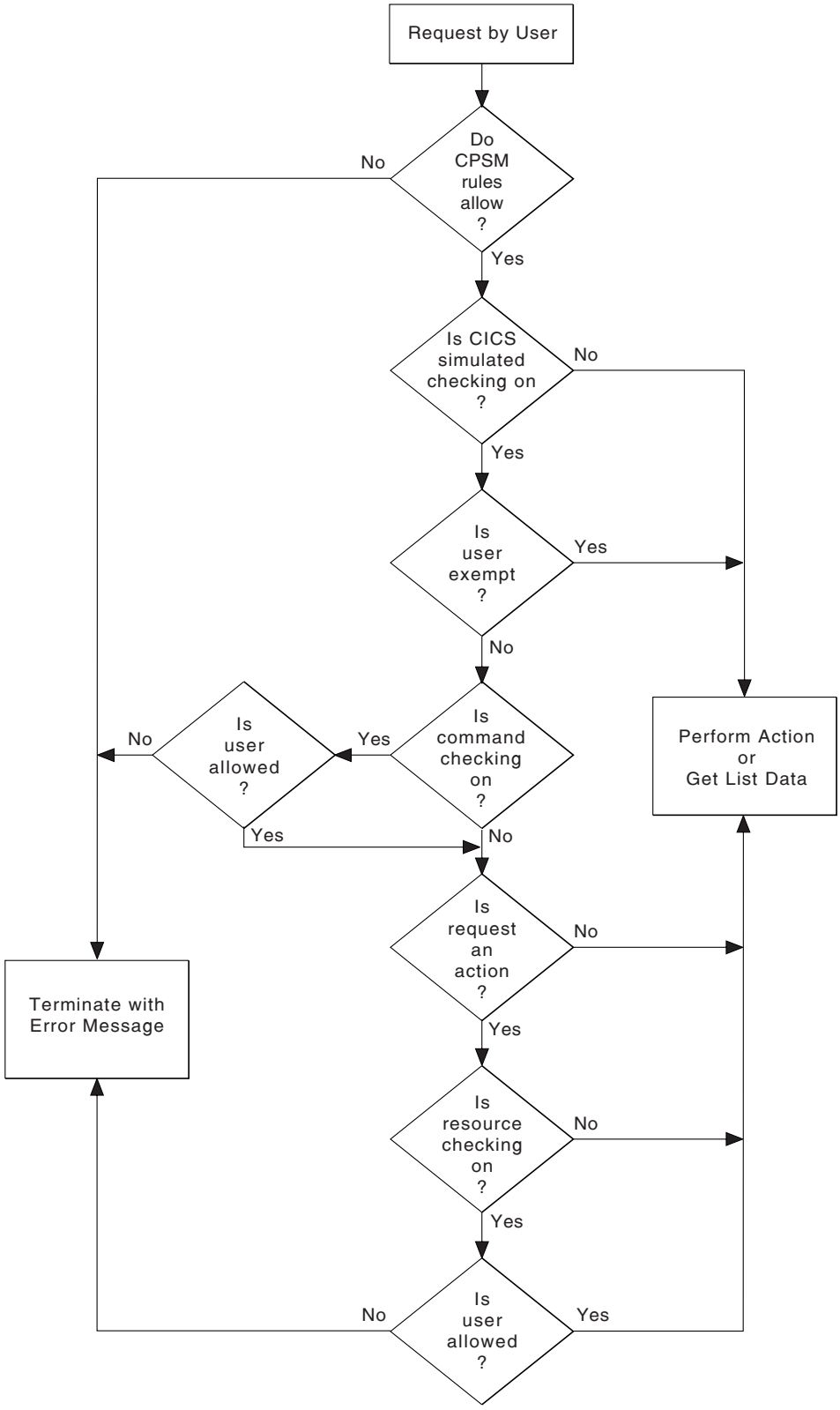



*Figure 42. Flowchart of CICSPlex SM security checking sequence - part 1*

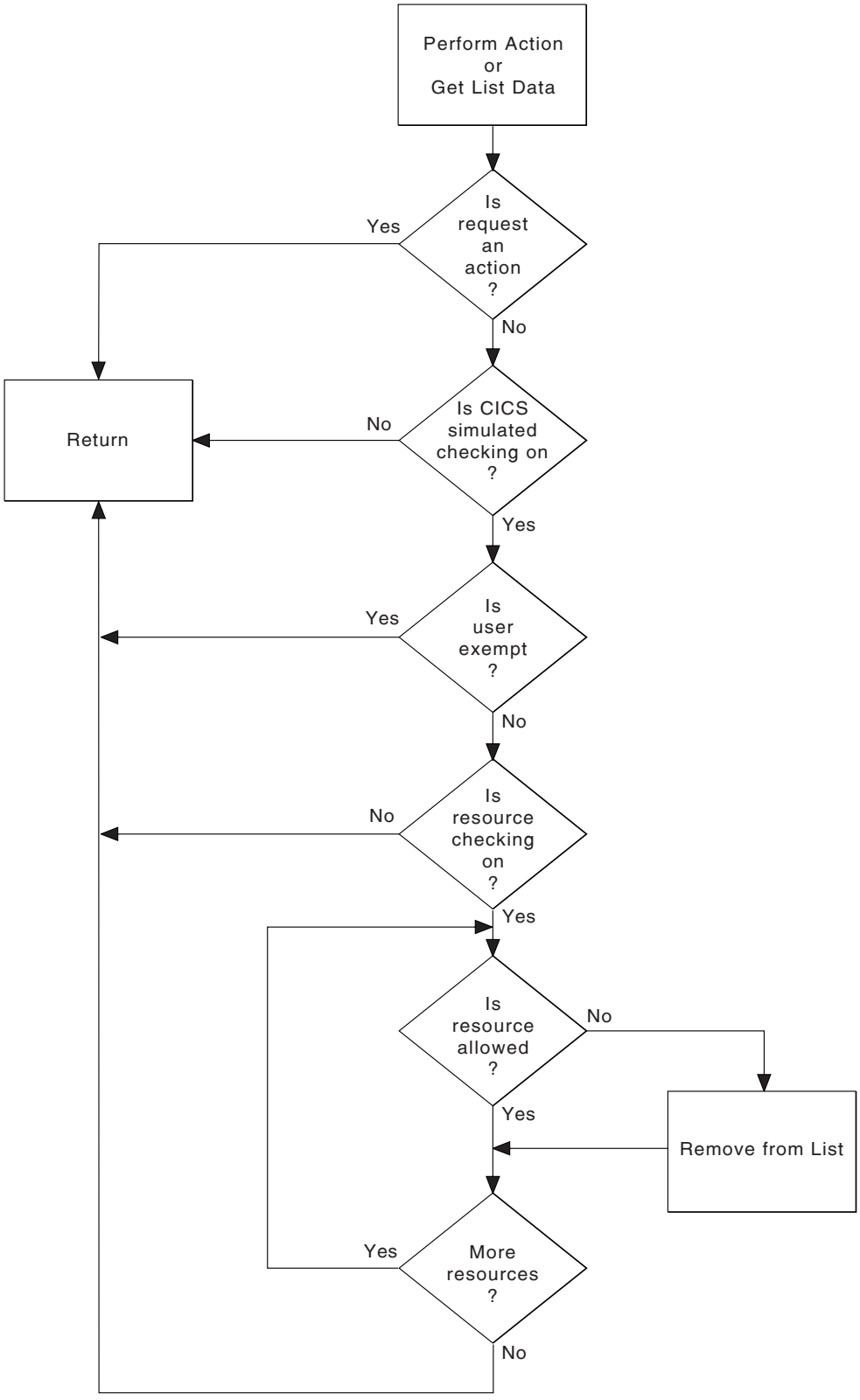



*Figure 43. Flowchart of CICSPlex SM security checking sequence - part 2*

# Chapter 23. Invoking a user-supplied external security manager

CICSPlex SM provides an interface to an external security manager (ESM), which can be user-supplied or can be the Resource Access Control Facility (RACF) program product. This chapter gives an overview of the CICSPlex SM-ESM interface, and describes how you can use the MVS router exit to pass control to a user-supplied ESM. Finally, it lists the control points at which CICSPlex SM invokes the ESM.

Be aware that upon return from any user-supplied program, CICSPlex SM must always receive control in primary-space translation mode, with the original contents of all access registers restored, and with all general-purpose registers restored(except for those which provide return codes or linkage information). For information about translation modes, refer to the *IBM ESA/390 Principles of Operation* manual.

**Note:** This chapter is intended primarily for non-RACF users. For definitive information about security processing using RACF, refer to "Chapter 22. Implementing CICSPlex SM security" on page 267.

## An overview of the CICSPlex SM-ESM interface

CICSPlex SM security uses, via the RACROUTE macro, the MVS system authorization facility (SAF) interface to route authorization requests to the ESM. Normally, if RACF is present, the MVS router passes control to it. However, you can modify the action of the MVS router by invoking the router exit. The router exit can be used, for example, to pass control to a user-supplied or vendor-supplied ESM. (If you want to use your own security manager, you must supply an MVS router exit routine.)

The control points at which CICSPlex SM issues a RACROUTE macro to route authorization requests are described in "CICSPlex SM security control points" on page 305.

## The MVS router

SAF provides your installation with centralized control over security processing, by using a system service called the MVS router. The MVS router provides a common system interface for all products providing resource control. The resource-managing components and subsystems (such as CICSPlex SM) call the MVS 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 use of common control functions shared across products and across systems.

If RACF is available in the system, the MVS router may pass control to the RACF router, which in turn invokes the appropriate RACF function. (The parameter information and the RACF router table, which associates router invocations with RACF functions, determine the appropriate function.) However, before calling the RACF router, the MVS router calls an optional, installation-supplied security-processing exit, if one has been installed.

## The MVS router exit

The MVS router provides an optional installation exit that is invoked whether or not RACF is installed and active on the system. If your installation does not use RACF, you can use the router exit to pass control to your own ESM. If you do use RACF, you could use the exit for preprocessing before RACF is invoked.

The MVS router exit routine is invoked whenever CICSPlex SM (or another component of your system) issues a RACROUTE macro. The router passes a parameter list (generated by the RACROUTE macro) to the exit routine. In addition, the exit receives the address of a 150-byte work area.

On entry to the exit routine, register 1 contains the address of the area described in Table 43.

*Table 43. Area addressed by register 1, on entry to exit routine*

| Offset | Length | Description |
|---|---|---|
| 0 | 4 | **Parameter list address:** points to the MVS router parameter list. (See "The MVS router parameter list".) |
| 4 | 4 | **Work area address:** points to a 150-byte work area that the exit can use. |

The exit must be named ICHRTX00 and must be located in the link pack area (LPA).

### The MVS router parameter list

The MVS router parameter list is generated when the RACROUTE macro is issued, and describes the security processing request by providing the request type. If the router exit routine exists, the router passes the parameter list to this exit. (If it does not exist, and if RACF is active, the router passes the parameter list to the RACF router.)

You can map the MVS router parameter list using the ICHSAFP macro. Its format is shown in the *MVS/ESA Diagnosis: Data Areas* manual.

### Router exit return codes

Your exit routine must return a return code in register 15. The hexadecimal values of the return code are shown in Table 44 on page 305.

*Table 44. MVS router exit return codes*

| Code | Meaning |
|---|---|
| 0 | The exit has completed successfully. Control proceeds to the RACF front-end routine for further security processing and an invocation of RACF. |
| C8 | The exit has completed successfully. The MVS router translates this return code to a router return code of '0' and returns control to the issuer of the RACROUTE macro (CICSPlex SM), bypassing RACF processing. (See the next section.) |
| CC | The exit has completed successfully. The MVS router translates this return code to a router return code of '4' and returns control to CICSPlex SM, bypassing RACF processing. (See the next section.) |
| D0 | The exit has completed successfully. The MVS router translates this return code to a router return code of '8' and returns control to CICSPlex SM, bypassing RACF processing. (See the next section.) |
| Other | If the exit routine sets any return code other than those described above, the MVS router returns control directly to CICSPlex SM and passes the untranslated code as the router return code. Further RACF processing is bypassed. |

### Passing control to a user-supplied ESM

Normally, a caller (such as CICSPlex SM) invokes the MVS router and passes it request type, requester, and subsystem parameters via the RACROUTE exit parameter list. Using these parameters, the MVS router calls the router exit which, on completing its processing, passes a return code to the router. If the return code is '0', as defined above, the router invokes RACF. RACF reports the result of that invocation to the router by entering return and reason codes in register 15 and register 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.

If your installation does not use RACF, you can make the MVS router exit pass control to an alternative ESM. However, if you do so you must still provide CICSPlex SM with the RACF return and reason codes that it expects to receive. You set the router exit return code, as defined in Table 44, so that RACF is not invoked; and you simulate the results of a RACF invocation by coding the exit so that it places the RACF return and reason codes in the first and second fullwords of the router input parameter list. RACF return and reason codes are documented in the *MVS/ESA Authorized Assembler Programming Reference* manual.

## CICSPlex SM security control points

All RACROUTE macros are issued from a CMAS. Macros required to support simulated CICS security checking are issued from the CMAS to which the target MAS is connected.

The following list summarizes the RACROUTE macros used by CICSPlex SM to invoke the ESM, and the control points at which they are issued.

**RACROUTE**
The "front end" to the macros described below, it invokes the MVS router. If RACF is not present on the system, RACROUTE can route to an alternative ESM, via the MVS router exit.

**RACROUTE REQUEST=VERIFY**

Issued at user signon (with the parameter ENVIR=CREATE), and at user sign-off (with parameter ENVIR=DELETE) to a CMAS. For ISPF end-user interface requests, signon calls are made during window creation in the CMAS that supports the named context. Sign-off calls are made when the window is closed. This macro creates or destroys an access control environment element (ACEE). It is issued at the following CICSPlex SM CMAS control points:

ISPF end-user interface user connection to a CMAS
API CONNECT thread creation
Single system image command routing
ISPF end-user interface user disconnect from a CMAS
API DISCONNECT thread termination

**RACROUTE REQUEST=FASTAUTH**

Issued during resource checking, on behalf of a user who is identified by an ACEE. It is the high-performance form of REQUEST=AUTH, using in-storage resource profiles, and is issued at the following CICSPlex SM CMAS control points:

Simulated CICS security checking
View selection / API security

**RACROUTE REQUEST=AUTH**

This is a higher path length form of resource checking and is issued during CAS / PLEXMGR security checking. It may also be called to perform logging and auditing after a REQUEST=FASTAUTH.

**RACROUTE REQUEST=LIST**

Issued to create and delete the in-storage profile lists needed by REQUEST=FASTAUTH. (One REQUEST=LIST macro is required for each resource class.) It is issued at the following CICSPlex SM CMAS control points:

When CICSPlex SM security is being initialized for a MAS
When the CMAS or CMASD sceurity action command (SEC) is issued.

For a detailed description of these macros, see the *OS/390 Security Server (RACF) Macros and Interfaces* manual.

# Chapter 24. Writing an API security exit

**This chapter contains Product-sensitive Programming Interface Information.**

CICSPlex SM provides a security validation exit that allows you to control access to a CMAS from application programming interface (API) programs. The security routine is called when security is active in a CMAS, but the environment in which the API program is running does not provide security of its own. CICSPlex SM attempts to extract user authorization data from the environment. If authorization data does not exist, the security routine is called.

## The supplied security routine

A default security routine called EYU9XESV is provided in the CICSTS13.CPSM.SEYUSAMP samples library. The copy book that maps the input parameter block is called EYUBXESV and is provided in CICTS13.CPSM.SEYUMAC.

By default, EYU9XESV processing is quite basic. EYU9XESV is called during both API connect and disconnect processing on the CMAS. At API connect time, EYU9XESV sets the USERID field to the default CICS user ID for the CMAS (the DFLT_UID value). EYU9XESV then returns, accepting the connection. At API disconnect time, EYU9XESV sets the RESPONSE field to OK and returns.

**Note:** The EYU9XESV security routine is supplied only in System/390 Assembler language. Any customization that you perform on EYU9XESV must be done in Assembler language.

## The security routine environment

The EYU9XESV security routine is loaded during CMAS initialization. EYU9XESV can reside in the CMAS STEPLIB, the MVS linklist, or the LPA library. If EYU9XESV cannot be loaded, all API connect requests that require its use are automatically rejected.

EYU9XESV receives control in the following processing environment:

- Supervisor state
- PSW key 0
- Primary address space control (ASC) mode
- Non-cross-memory mode
- 31-bit addressing mode.

On entry to the security exit, the general registers are set as follows:

- Register 0 is undefined
- Register 1 contains the address of the EYUBXESV parameter block
- Registers 2 through 12 are undefined
- Register 13 contains the address of a 72-byte save area
- Register 14 contains the return address
- Register 15 contains the address of the exit entry point.

Access registers AR0 through AR15 contain zeroes (0).

## Customizing the security routine

To customize the default security processing for API programs, you can modify the EYU9XESV security routine supplied with CICSPlex SM. On entry to the security routine, register 1 contains the address of the EYUBXESV parameter block. You can use the information provided in this parameter block to decide whether or not to grant an API program access to a CMAS. However, note that you cannot use the CICSPlex SM API from within EYU9XESV itself.

### API connect processing

During API connect processing, the security exit parameter block identifies the connection type. You can use the type field to identify the origin of the API connection. The following fields are also provided for all connection types:

- The thread token for the API connection, which is unique within the MVS/ESA image where the CMAS is running
- The USER value from the API CONNECT command
- The SIGNONPARM value from the API CONNECT command
- The default CICS user ID for the CMAS.

**Note:** The REXX API program passes the USER and SIGNONPARM values to the security exit as 8-byte fields. If either of the values is less than 8 characters, the field is padded with blank spaces (X'40').

For connections that originate from a MAS (that is, the API program is running in a CICS system), the following data fields are set:

- CICS SYSID
- CICS task number of the task that issued the connect
- CICS terminal ID of the task, if any.

For connections that originate from somewhere other than a MAS, the jobname of the Job, started task, or TSO address space is provided.

Using this input, your security routine can accept or reject the connection. If the connection is accepted, you must provide one of the following:

- The address of an accessor environment element (ACEE)
- A user ID for the connecting application.

If you provide both an ACEE address and a user ID, security information for the user is extracted from the ACEE and the user ID is ignored.

Your security routine can also provide a four-byte user token that will be maintained by CICSPlex SM for the life of the API program. This token is returned to the exit during API disconnect processing.

Your security routine should set the RESPONSE and REASON values from the CONNECT command prior to exiting.

### API disconnect processing

During API disconnect processing, the security exit is called to perform any resource cleanup or termination processing that may be required.

The security exit parameter block for API disconnection contains the user ID associated with the API program. The user ID is the same one returned by API connect processing, if the security routine returned a user ID. If the security

routine returned the address of an ACEE, it is the user ID contained in the ACEE. In addition, the parameter block contains the API thread token and the user token, if one was specified.

## The security routine parameter block

To map the security routine parameter block, you can use the EYUBXESV copy book provided in CICSTS13.CPSM.SEYUMAC. Figure 44 on page 310 illustrates the layout of the EYUBXESV parameter block.

```
*--------------------------------------------------------------------*
*                                                                    *
* COPY BOOK NAME = EYUBXESV                                          *
*                                                                    *
* DESCRIPTIVE NAME = %PRODUCT Site Security user validation exit     *
*                    parameter block                                 *
*                                                                    *
*    COPYRIGHT = Licensed Materials - Property of IBM                *
*                5695-081                                            *
*                (C) Copyright IBM Corp. 1995, 1997                  *
*                All Rights Reserved                                 *
*                                                                    *
*                US Government Users Restricted Rights - Use,        *
*                duplication or disclosure restricted by GSA ADP     *
*                Schedule Contract with IBM Corp.                    *
*                                                                    *
* STATUS = %CP00                                                     *
*                                                                    *
* FUNCTION =                                                         *
*       Parameter block passed to the Site Security user validation  *
*       exit program (EYU9XESV)                                      *
*                                                                    *
* LIFETIME =                                                         *
*       The site Security user validation exit parameter block is    *
*       obtained and released by the caller of the site security     *
*       user validation exit program caller                          *
*                                                                    *
* STORAGE CLASS =                                                    *
*       The XESV is acquired from private storage in the address space*
*       from which the call to the site security user validation exit *
*       program is made                                              *
*       It is key 0 storage acquired from subpool 252                *
*                                                                    *
* LOCATION =                                                         *
*       Register 1 on entry to the site security user validation exit *
*       program                                                      *
*                                                                    *
* NOTES :                                                            *
*   DEPENDENCIES = S/370                                             *
*   RESTRICTIONS = None                                              *
*   MODULE TYPE = Control Block Definition                           *
*   PROCESSOR = Assembler                                            *
*                                                                    *
*--------------------------------------------------------------------*
*                                                                    *
* CHANGE ACTIVITY :                                                  *
*                                                                    *
*      $SEG(EYUBXESV),COMP(ENVIR),PROD(%PRODUCT):                    *
*                                                                    *
*   PN= REASON REL YYMMDD BDXIII : REMARKS                           *
*  $01  Reserved for APAR fix                                        *
*  $02  Reserved for APAR fix                                        *
*  $03  Reserved for APAR fix                                        *
*  $D1  Reserved for DCR                                             *
*  $D2  Reserved for DCR                                             *
*  $D3  Reserved for DCR                                             *
*  $H1  Reserved for hardware support                                *
*  $H2  Reserved for hardware support                                *
*  $H3  Reserved for hardware support                                *
*  $L0  SM1    %S0 950406 BDEJWB : BASE RELEASE                      *
```

*Figure 44. The EYUBXESV parameter block (Part 1 of 3)*

```
*  $L1  Reserved for line item                                          *
*  $L2  Reserved for line item                                          *
*  $L3  Reserved for line item                                          *
*  $P1  Reserved for PTM                                                *
*  $P2  Reserved for PTM                                                *
*  $P3  Reserved for PTM                                                *
*                                                                       *
*-----------------------------------------------------------------------*
EYUBXESV            DSECT ,
XESV_PREFIX         DS   0CL20         Prefix
XESV_SLENGTH        DS   AL2           Structure Length
XESV_ARROW          DS   C             ">" delimiter
XESV_NAME           DS   CL8           "EYUBXESV"
XESV_BLANK          DS   C             " "
XESV_PGMNAME        DS   CL8           "EYU9XESV"
XESV_PFX_LEN        EQU  *-XESV_PREFIX Length of prefix
XESV_FUNCTION       DS   XL1           Function Code
XESV_FUNC_CONN      EQU  1              Exit Called during Connect
XESV_FUNC_DSCO      EQU  2              Exit called during Disconnect
                    DS   XL3           Reserved
XESV_RESPONSE       DS   F             Response Code
XESV_RESP_OK        EQU  0              Good response code
XESV_RESP_REJECT    EQU  4              Exit rejects Connect
XESV_RESP_ERROR     EQU  8              Error in Connect/Disconnect
XESV_REASON         DS   F             Reason Codes
*-----------------------------------------------------------------------*
*  Reasons for a Response of Connect REJECT                             *
*-----------------------------------------------------------------------*
XESV_REAS_APIUSER   EQU  4              Connect Invalid API_UID
XESV_REAS_APIDATA   EQU  8              Connect Invalid API_DATA
XESV_REAS_APIEXP    EQU  12             Connect API_UID expired
*-----------------------------------------------------------------------*
*  Reasons for a Response of Connect/Disconnect ERROR                   *
*-----------------------------------------------------------------------*
XESV_REAS_NOSTG     EQU  4              Exit could not obtain storage
XESV_PARAMETERS     DS   0C
*-----------------------------------------------------------------------*
*  The Connection Parameters are as follows                             *
*-----------------------------------------------------------------------*
XESV_CONN_TYPE      DS   XL1           Connector Environment type
XESV_CONN_LMAS      EQU  1              LMAS
XESV_CONN_TSOE      EQU  2              TSO/E Address Space
XESV_CONN_BATCH     EQU  3              BATCH or STC Address Space
XESV_CONN_OS2RMAS   EQU  4              OS/2 RMAS
                    DS   XL3           Reserved
*-----------------------------------------------------------------------*
*  For All Connectors, the following fields apply                       *
*-----------------------------------------------------------------------*
XESV_CONN_TOKEN     DS   XL4           The unique Connection Token for x
                                       the connecting application
XESV_CONN_API_UID   DS   CL8           Userid specified on the API     X
                                       CONNECT verb (zeros if none)
XESV_CONN_API_DATA  DS   CL8           User Data specified on the API  X
                                       CONNECT verb in the Signonparm  X
                                       parameter (zeros if none)
XESV_CONN_DFLT_UID  DS   CL8           Default Userid specified for theX
                                       CMAS
```

*Figure 44. The EYUBXESV parameter block (Part 2 of 3)*

```
*-------------------------------------------------------------------*
*  For MAS connectors, the following fields apply                   *
*-------------------------------------------------------------------*
XESV_CONN_SYSID     DS   CL4           MAS SYSID
XESV_CONN_TASKN     DS   PL4           Task Number of Task issuing the x
                                       Connect
XESV_CONN_TERMID    DS   CL4           If A terminal facility, the     x
                                       CICS TERMID of the facility
*-------------------------------------------------------------------*
*  For OS2/RMAS Connectors, the following fields apply              *
*-------------------------------------------------------------------*
XESV_CONN_LINKU     DS   CL8           Userid Associated with CMAS to  x
                                       RMAS Communications Link.  Blankx
                                       if none
*-------------------------------------------------------------------*
*  For ESA connectors, the following fields apply                   *
*-------------------------------------------------------------------*
XESV_CONN_JOBNAME   DS   CL8           Job Name
*-------------------------------------------------------------------*
*  One of the following two fields must be set by the exit program.   *
*  Their values are undefined on input for Connect validation.        *
*                                                                     *
*  The CONN_SECENV field, if set, must contain the address of the     *
*  accessor environment element (ACEE) as an output of the Connect    *
*  Validation.                                                        *
*                                                                     *
*  The CONN_USERID field, if set, must contain the Userid to use      *
*  as an output of the Connect Validation.                            *
*                                                                     *
*  If both fields are set by connect validation, the SECENV address   *
*  will be used                                                       *
*                                                                     *
*  If neither of the fields is set by connect validation, the         *
*  connection will be rejected.                                       *
*                                                                     *
*  The CONN_UTOKEN field may be set to a token provided by the        *
*  user exit program.  Its contents will be provided as input to      *
*  the disconnect function.  If the exit creates resources, for       *
*  example, an ACEE, this token might be used as a resource referent  *
*  so that the resources may be released during the disconnect        *
*  call.                                                              *
*-------------------------------------------------------------------*
XESV_CONN_SECENV    DS   A             Accessor Environment Element    X
                                       (ACEE)
XESV_CONN_USERID    DS   CL8           Userid of Connected User
XESV_CONN_UTOKEN    DS   XL4           Usertoken provided by exit      x
                                       program.
                    ORG  XESV_PARAMETERS
*-------------------------------------------------------------------*
*  The Disconnect Parameters are as follows                         *
*-------------------------------------------------------------------*
XESV_DSCO_USERID    DS   CL8           The USERID of the disconnection x
                                       application
XESV_DSCO_TOKEN     DS   XL4           The unique Connection Token for x
                                       the Disconnecting application
XESV_DSCO_UTOKEN    DS   XL4           UserToken provided by exit      x
                                       program on the Connect call.
                    ORG  ,
XESV_SIZE           EQU  *-EYUBXESV    Length of structure
```

*Figure 44. The EYUBXESV parameter block (Part 3 of 3)*

# Chapter 25. Example tasks: security

This chapter provides examples of typical security setup tasks that you can use as a model for your own.

Here are some general points that apply to all of the RACF examples in this chapter:

- Each RACF command shown in these task examples must be issued once against every RACF database in your CICSPlex SM configuration. So, if there are two unconnected RACF databases, one on MVS1 and one on MVS2, each RACF command must be issued twice (once on each system).
- In all of the RACF command examples, strings in lowercase must be replaced by values suitable for your own enterprise. For example, you must replace the string `admin_user` with the USERID of the administrator responsible for security of the relevant CICSPlex SM resources.
- All of the RACF task examples use the enhanced generic naming facility (**) of RACF. If you don't use this at your enterprise, see the RACF documentation for information about creating equivalent profiles.
- Operations and administration RACF groups have been used in these examples: we recommend that you create such groups.

We're going to start by creating some RACF profiles to protect all CICSPlex SM functions and resources. When we've done this, we'll permit access selectively to particular users of particular resources.

## Protect all CICSPlex SM resources

To create the RACF profile to protect all CICSPlex SM resources, do the following:

1. Ensure that the CPSMOBJ class is active and that generic profiles can be defined:

   ```
   SETROPTS CLASSACT(CPSMOBJ) GENERIC(CPSMOBJ)
   ```

2. Create a RACF profile to protect all views and action commands for all CICSPlex SM functions:

   ```
   RDEF CPSMOBJ ** UACC(NONE) OWNER(admin_group) NOTIFY(admin_user)
   ```

   This command defines a profile (**) that RACF treats as matching all `CPSMOBJ` resource entity names, and which therefore protects all CICSPlex SM resources; it also specifies that `admin_user` is to be notified of any violations.

3. The next step is very similar to Step 2: we define one RACF profile for each CICSplex in the configuration. Each profile will protect all CICSPlex SM functions and resources for that CICSplex. The purpose of doing this is to give you more flexibility in granting access to CICSplex-specific resources. In this example, we have two CICSplexes, and so create two RACF profiles:

   ```
   RDEF CPSMOBJ *.*.PLXPROD1.* UACC(NONE) OWNER(admin_group) +
        NOTIFY(admin_user)
   RDEF CPSMOBJ *.*.PLXPROD2.* UACC(NONE) OWNER(admin_group) +
        NOTIFY(admin_user)
   ```

   Note that you can't replace Step 2 with multiple CICSplex-specific profiles: such profiles won't necessarily protect CICSplexes that you create later, nor can

they protect CICSPlex SM functions whose context is the CMAS rather than the CICSplex. For example, the CONFIG views would be left unprotected if you didn't also perform Step 2 on page 313.

4. In Step 3 on page 313 we protected all CICSPlex SM functions and resources at the CICSplex level. In this step, we're going to define profiles to control access to the CICSPlex SM CONFIG and TOPOLOGY definition functions, so that we can selectively permit any "special" users, such as administrators, the access they need. (Anyone who has update access to these two functions can alter the CICSplex configuration, and so access must be limited.)

```
RDEF CPSMOBJ CONFIG.DEF.** UACC(NONE) OWNER(admin_group)
RDEF CPSMOBJ TOPOLOGY.DEF.** UACC(NONE) OWNER(admin_group)
```

Now that we've controlled access to CICSPlex SM functions and resources, we can begin to grant access to particular users or groups of users.

## Give CICSPlex SM operators appropriate authorizations

CICSPlex SM operators need access, at least, to all of the OPERATE views. In this example, we'll show you how to give CICSPlex SM operators update access to all OPERATE views and read access to the MONITOR views. This will allow operators to look at monitor data, but not to create or change monitor definitions.

1. Give CICSPlex SM operators update access to the OPERATE views:

```
RDEF CPSMOBJ OPERATE.** OWNER(admin_group) UACC(NONE)
PE OPERATE.** CLASS(CPSMOBJ) ID(ops_group) A(UPDATE)
```

2. Give CICSPlex SM operators read access to the MONITOR views:

```
RDEF CPSMOBJ MONITOR.** UACC(NONE) OWNER(admin_group)
PE MONITOR.** CLASS(CPSMOBJ) ID(ops_group) A(READ)
```

In both steps, you can see that we begin by creating a RACF profile to protect the resource, and then grant access to users in group `ops_group`.

## Give a user read access to all transactions on MVS system A

In this example, we show you how to give user PAYUSR1 read access to all transactions (via the CICSPlex SM LOCTRAN, LOCTRAND, LOCTRANS, REMTRAN, REMTRAND, REMTRANS, TRAN, and TRANS views) running on CICS systems on MVS system A. In the example, we have three CICS systems (say, CICSAA01, CICSAA02, and CICSAA03) which all belong to CICSplex PLXPROD1.

1. Define the appropriate RACF profile:

```
RDEF CPSMOBJ OPERATE.TRAN.PLXPROD1.CICSAA0* UACC(NONE) +
     OWNER(admin_group)
```

2. Give user PAYUSR1 read access to all transactions on MVS system A:

```
PE OPERATE.TRAN.PLXPROD1.CICSAA0* CLASS(CPSMOBJ) I(PAYUSR1) A(READ)
```

## Allow a user to change a named transaction in any AOR

In this example, we'll allow user PAYUSR1 to update transaction AMNU running on any AOR in CICSplex PLXPROD1 (consisting of the three CICS systems in the example above).

1. Activate simulated CICS security.

   Simulated CICS security, which tells CICSPlex SM to honor CICS security definitions, can be used to protect transaction definitions. You can activate

simulated CICS security from the CPLEXDEF view (for the CICSplex); from the CICSSYS view (for a MAS at MAS startup); or from the MAS view (for a running MAS).

2. Give user PAYUSR1 update access to the OPERATE.TRAN views:

```
PE OPERATE.TRAN.PLXPROD1.CICSAA0* CLASS(CPSMOBJ) +
   ID(PAYUSR1) A(UPDATE)
```

3. If necessary (such a profile will usually already have been defined), define a RACF profile to protect transaction AMNU:

```
RDEF ACICSPCT AMNU     +
     UACC(NONE)        +
     OWNER(admin_group)
```

   (For more information about this step, see the *CICS-RACF Security Guide*.)

4. Give user PAYUSR1 update access to transaction AMNU:

```
PE AMNU CLASS(ACICSPCT) ID(PAYUSR1) A(UPDATE)
```

   If you use a class other than (the CICS default of) ACICSPCT, you must specify its name in place of ACICSPCT.

5. Verify that the MASs have SIT parameter XPCT=YES.

In this example, we've had to give PAYUSR1 update access to the transaction views, and then to transaction AMNU itself. Both authorizations are necessary.

## Prevent a user from changing programs in a CICSplex

This example shows how to prohibit user PAYUSR1 from updating programs in any MAS belonging to CICSplex PLXPROD1.

1. Define a RACF profile to protect the PROGRAM views:

```
RDEF CPSMOBJ OPERATE.PROGRAM.PLXPROD1.* +
     UACC(NONE) OWNER(admin_group)
```

2. Give user PAYUSR1 read access to programs:

```
PE OPERATE.PROGRAM.PLXPROD1.* CLASS(CPSMOBJ) I(PAYUSR1) A(READ)
```

   Or, if you prefer, you can give PAYUSR1 no access to programs:

```
PE OPERATE.PROGRAM.PLXPROD1.* CLASS(CPSMOBJ) I(PAYUSR1) A(NONE)
```

## Allow a system administrator to create CICSPlex SM definitions

This example shows how to authorize a system administrator to create definitions for workload management, real-time analysis, and resource monitoring.

1. Create RACF profiles to protect WLM, RTA, and MON definition views:

```
RDEF CPSMOBJ WORKLOAD.DEF.** UACC(NONE) +
     OWNER(admin_group)
RDEF CPSMOBJ ANALYSIS.DEF.** UACC(NONE) +
     OWNER(admin_group)
RDEF CPSMOBJ MONITOR.DEF.** UACC(NONE) +
     OWNER(admin_group)
```

2. Allow user SYSADM to create and update WLM, RTA, and MON definitions:

```
PE WORKLOAD.DEF.** CLASS(CPSMOBJ) I(SYSADM) A(UPDATE)
PE ANALYSIS.DEF.** CLASS(CPSMOBJ) I(SYSADM) A(UPDATE)
PE MONITOR.DEF.** CLASS(CPSMOBJ) I(SYSADM) A(UPDATE)
```

# Part 8. Appendixes

# Appendix A. National Language

This appendix contains the language codes with which the user can specify a preferred language if that language is defined in their CICS system.

Specify a language request is specified in the following RACF command:

```
ALTUSER userid LANGUAGE(PRIMARY(language-code) SECONDARY(language-code))
```

For PRIMARY or SECONDARY, you can specify one of the language codes under "IBM code" in Table 45.

CICS attempts to use the PRIMARY language for a user if it corresponds to a language suffix in the NATLANG system initialization parameter. Otherwise it attempts to use the SECONDARY language. If neither the PRIMARY nor the SECONDARY language corresponds to a NATLANG value, the language must be provided from elsewhere. See "Obtaining CICS-related data at signon" on page 75.

**Note:** CICS ignores the RACF default national language defined by the command:

```
SETROPTS LANGUAGE(PRIMARY(...) SECONDARY(...))
```

In CICS, you can use only the languages listed in Table 45. Languages other than ENU, CHNS, and JPN are available only if you provide translated message tables for them, using the message editing utility program, and then specify the CICS language suffix in the NATLANG system initialization parameter. See the *CICS Operations and Utilities Guide* for information on creating translated message tables.

*Table 45. CICS language suffixes*

| Suffix | IBM Code | Language name |
|---|---|---|
| A | ENG | United Kingdom English |
| B | PTB | Brazilian Portuguese |
| C | CHS | Simplified Chinese |
| D | DAN | Danish |
| E | ENU | US English |
| F | FRA | French |
| G | DEU | German |
| H | KOR | Korean |
| I | ITA | Italian |
| J | ISL | Icelandic |
| K | JPN | Japanese |
| L | BGR | Bulgarian |
| M | MKD | Macedonian |
| N | NOR | Norwegian |
| O | ELL | Greek |
| P | PTG | Portuguese |
| Q | ARA | Arabic |
| R | RUS | Russian |

*Table 45. CICS language suffixes (continued)*

| Suffix | IBM Code | Language name |
|---|---|---|
| S | ESP | Spanish |
| T | CHT | Traditional Chinese |
| U | UKR | Ukrainian |
| V | SVE | Swedish |
| W | FIN | Finnish |
| X | HEB | Hebrew |
| Y | SHC | Serbo-Croatian (Cyrillic) |
| Z | THA | Thai |
| 1 | BEL | Byelorussian |
| 2 | CSY | Czech |
| 3 | HRV | Croatian |
| 4 | HUN | Hungarian |
| 5 | PLK | Polish |
| 6 | ROM | Romanian |
| 7 | SHL | Serbo-Croatian (Latin) |
| 8 | TRK | Turkish |
| 9 | NLD | Dutch |

# Appendix B. Resource and command check cross reference

This appendix provides a complete command and resource check cross reference.

*Table 46. Resource and command check cross reference*

| EXEC CICS COMMAND | Resource Check | | | Check class=XCMD | |
|---|---|---|---|---|---|
| | **Class** | **Access** | **Resource** | **Access** | **Resource** |
| ABEND | | | | | |
| ACQUIRE | | | | UPDATE | TERMINAL |
| ACQUIRE for BTS(see note 9) | XFCT | UPDATE | BTS repository file | | |
| ADDRESS | | | | | |
| ALLOCATE | | | | | |
| ASKTIME | | | | | |
| BIF DEEDIT | | | | | |
| BUILD ATTACH | | | | | |
| CANCEL (see note 1) | XPCT | READ | transid | | |
| CANCEL for BTS | XFCT | UPDATE | BTS repository file | | |
| CHANGE PASSWORD | | | | | |
| CHANGE TASK | | | | | |
| COLLECT FILE | XFCT | READ | file | READ | STATISTICS |
| COLLECT JOURNALNAME | XJCT | READ | journal | READ | STATISTICS |
| COLLECT JOURNALNUM | XJCT | READ | journal | READ | STATISTICS |
| COLLECT PROGRAM | XPPT | READ | program | READ | STATISTICS |
| COLLECT STATISTICS | | | | READ | STATISTICS |
| COLLECT TDQUEUE | XPCT | READ | tdqueue | READ | STATISTICS |
| COLLECT TRANSACTION | XDCT | READ | transid | READ | STATISTICS |
| CONNECT PROCESS | | | | | |
| CONVERSE | | | | | |
| CREATE CONNECTION (see note 2) | | | | ALTER | CONNECTION |
| CREATE DB2CONN (see note 3) | | ALTER | | | DB2CONN |
| CREATE DB2ENTRY (see note 3) | XDB2 | ALTER | db2entry | ALTER | DB2ENTRY |
| CREATE DB2TRAN (see note 3) | XDB2 | ALTER | db2tran | ALTER | DB2TRAN |
| CREATE DOCTEMPLATE | | | | ALTER | DOCTEMPLATE |

*Table 46. Resource and command check cross reference (continued)*

| EXEC CICS COMMAND | Resource Check | | | Check class=XCMD | |
|---|---|---|---|---|---|
| | Class | Access | Resource | Access | Resource |
| CREATE ENQMODEL | | | | ALTER | ENQMODEL |
| CREATE FILE | XFCT | ALTER | file | ALTER | FILE |
| CREATE JOURNALMODEL | | | | ALTER | JOURNALMODEL |
| CREATE LSRPOOL | | | | ALTER | LSRPOOL |
| CREATE MAPSET | XPPT | ALTER | mapset | ALTER | MAPSET |
| CREATE PARTITIONSET | XPPT | ALTER | partitionset | ALTER | PARTITIONSET |
| CREATE PARTNER | | | | ALTER | PARTNER |
| CREATE PROCESSTYPE (see note 10) | | | | ALTER | PROCESSTYPE |
| CREATE PROFILE | | | | ALTER | PROFILE |
| CREATE PROGRAM | XPPT | ALTER | program | ALTER | PROGRAM |
| CREATE REQUESTMODEL | | | | ALTER | REQUESTMODEL |
| CREATE SESSIONS (see note 3) | | | | ALTER | SESSIONS |
| CREATE TCPIPSERVICE | | | | ALTER | TCPIPSERVICE |
| CREATE TDQUEUE (see note 3) | XDCT | ALTER | tdqueue | ALTER | TDQUEUE |
| CREATE TERMINAL (see note 3) | | | | ALTER | TERMINAL |
| CREATE TRANCLASS | | | | ALTER | TCLASS |
| CREATE TRANSACTION | XPCT | ALTER | transid | ALTER | TRANSACTION |
| CREATE TSMODEL | | | | ALTER | TSMODEL |
| CREATE TYPETERM | | | | ALTER | TYPETERM |
| DEFINE ACTIVITY(see note 7) (see note 9) | XFCT | UPDATE | BTS repository file | | |
| DEFINE PROCESS (see note 7) (see note 9) | XFCT | UPDATE | BTS repository file | | |
| DELAY | | | | | |
| DELETE | XFCT | UPDATE | file | | |
| DELETE ACTIVITY (see note 9) | XFCT | UPDATE | BTS repository file | | |
| DELETEQ TD | XDCT | UPDATE | tdqueue | | |
| DELETEQ TS (see note 4) | XTST | UPDATE | tsqueue | | |
| DEQ | | | | | |
| DISABLE PROGRAM | XPPT | UPDATE | program | UPDATE | EXITPROGRAM |

*Table 46. Resource and command check cross reference (continued)*

| EXEC CICS COMMAND | Resource Check | | | Check class=XCMD | |
|---|---|---|---|---|---|
| | **Class** | **Access** | **Resource** | **Access** | **Resource** |
| DISCARD AUTINSTMODEL | | | | ALTER | AUTINSTMODEL |
| DISCARD CONNECTION | | | | | |
| DISCARD DB2CONN | | | | ALTER | DB2CONN |
| DISCARD DB2ENTRY | XDB2 | ALTER | db2entry | ALTER | DB2ENTRY |
| DISCARD DB2TRAN | XDB2 | ALTER | db2tran | ALTER | DB2TRAN |
| DISCARD DOCTEMPLATE | | | | ALTER | DOCTEMPLATE |
| DISCARD ENQMODEL | | | | ALTER | ENQMODEL |
| DISCARD FILE | XFCT | ALTER | file | ALTER | FILE |
| DISCARD JOURNALMODEL | | | | ALTER | JOURNALMODEL |
| DISCARD JOURNALNAME | XJCT | ALTER | journal | ALTER | JOURNALNAME |
| DISCARD PARTNER | | | | ALTER | PARTNER |
| DISCARD PROCESSTYPE (see note 10) | | | | ALTER | PROCESSTYPE |
| DISCARD PROFILE | | | | ALTER | PROFILE |
| DISCARD PROGRAM | XPPT | ALTER | program | ALTER | PROGRAM |
| DISCARD REQUESTMODEL | | | | ALTER | REQUESTMODEL |
| DISCARD TCPIPSERVICE | | | | ALTER | TCPIPSERVICE |
| DISCARD TDQUEUE | XDCT | ALTER | tdqueue | ALTER | TDQUEUE |
| DISCARD TERMINAL | | | | ALTER | TERMINAL |
| DISCARD TRANCLASS | | | | ALTER | TCLASS |
| DISCARD TRANSACTION | XPCT | ALTER | transid | ALTER | TRANSACTION |
| DISCARD TSMODEL | | | | ALTER | TSMODEL |
| DOCUMENT | | | | | DOCUMENT |
| DUMP TRANSACTION | | | | | |
| ENABLE PROGRAM | XPPT | UPDATE | program | UPDATE | EXITPROGRAM |
| ENDBR (see note 5) | | | | | |
| ENQ | | | | | |
| ENTER TRACENUM | | | | | |
| EXTRACT | | | | | |
| EXTRACT EXIT | XPPT | READ | program | UPDATE | EXITPROGRAM |
| FEPI | | | | | FEPI |
| FORMATTIME | | | | | |

*Table 46. Resource and command check cross reference (continued)*

| EXEC CICS COMMAND | Resource Check | | | Check class=XCMD | |
|---|---|---|---|---|---|
| | **Class** | **Access** | **Resource** | **Access** | **Resource** |
| FREE | | | | | |
| FREEMAIN | | | | | |
| GDS | | | | | |
| GETMAIN | | | | | |
| HANDLE ABEND PROGRAM | XPPT | READ | program | | |
| HANDLE AID | | | | | |
| HANDLE CONDITION | | | | | |
| IGNORE CONDITION | | | | | |
| INQUIRE ACTIVITYID (see note 9) | XFCT | READ | BTS repository file | | |
| INQUIRE AUTINSTMODEL | | | | READ | AUTINSTMODEL |
| INQUIRE AUTOINSTALL | | | | READ | AUTOINSTALL |
| INQUIRE CFDTPOOL | | | | READ | CFDTPOOL |
| INQUIRE CONNECTION | | | | READ | CONNECTION |
| INQUIRE CONTAINER(see note 9) | XFCT | READ | BTS repository file | | |
| INQUIRE DB2CONN | | | | READ | DB2CONN |
| INQUIRE DB2ENTRY | XDB2 | READ | db2entry | READ | DB2ENTRY |
| INQUIRE DB2TRAN | XDB2 | READ | db2tran | READ | DB2TRAN |
| INQUIRE DELETSHIPPED | | | | READ | DELETSHIPPED |
| INQUIRE DOCTEMPLATE | | | | READ | DOCTEMPLATE |
| INQUIRE DSNAME | | | | READ | DSNAME |
| INQUIRE DUMPDS | | | | READ | DUMPDS |
| INQUIRE ENQMODEL | | | | READ | ENQMODEL |
| INQUIRE EXCI | | | | | |
| INQUIRE EVENT(see note 9) | XFCT | READ | BTS repository file | | |
| INQUIRE EXITPROGRAM | XPPT | READ | program | READ | EXITPROGRAM |
| INQUIRE FILE | XFCT | READ | file | READ | FILE |
| INQUIRE IRC | | | | READ | IRC |
| INQUIRE JOURNALMODEL | | | | READ | JOURNALMODEL |
| INQUIRE JOURNALNAME | XJCT | READ | journal | READ | JOURNALNUM |

*Table 46. Resource and command check cross reference (continued)*

| EXEC CICS COMMAND | Resource Check | | | Check class=XCMD | |
|---|---|---|---|---|---|
| | **Class** | **Access** | **Resource** | **Access** | **Resource** |
| INQUIRE JOURNALNUM | XJCT | READ | DFHJnn | READ | JOURNALNUM |
| INQUIRE MODENAME | | | | READ | MODENAME |
| INQUIRE MONITOR | | | | READ | MONITOR |
| INQUIRE NETNAME | | | | READ | TERMINAL |
| INQUIRE PARTNER | | | | READ | PARTNER |
| INQUIRE PROCESS(see note 9) | XFCT | READ | BTS repository file | | |
| INQUIRE PROCESSTYPE (see note10) | XPTT | | | READ | PROCESSTYPE |
| INQUIRE PROFILE | | | | READ | PROFILE |
| INQUIRE PROGRAM | XPPT | READ | program | READ | PROGRAM |
| INQUIRE REQID (see note 8) | XPCT | READ | transid | READ | REQID |
| INQUIRE REQUESTMODEL | | | | READ | REQUESTMODEL |
| INQUIRE RRMS | | | | READ | RRMS |
| INQUIRE STATISTICS | | | | READ | STATISTICS |
| INQUIRE STORAGE | | | | READ | STORAGE |
| INQUIRE STREAMNAME | | | | READ | STREAMNAME |
| INQUIRE SYSDUMPCODE | | | | READ | SYSDUMPCODE |
| INQUIRE SYSTEM | | | | READ | SYSTEM |
| INQUIRE TASK | | | | READ | TASK |
| INQUIRE TCLASS | | | | READ | TCLASS |
| INQUIRE TCPIP | | | | READ | TCPIP |
| INQUIRE TCPIPSERVICE | | | | READ | TCPIPSERVICE |
| INQUIRE TDQUEUE | XDCT | READ | tdqueue | READ | TDQUEUE |
| INQUIRE TERMINAL | | | | READ | TERMINAL |
| INQUIRE TIMER (see note 9) | XFCT | READ | BTS repository file | | |
| INQUIRE TRACEDEST | | | | READ | TRACEDEST |
| INQUIRE TRACEFLAG | | | | READ | TRACEFLAG |
| INQUIRE TRACETYPE | | | | READ | TRACETYPE |
| INQUIRE TRANCLASS | | | | READ | TCLASS |
| INQUIRE TRANDUMPCODE | | | | READ | DUMPCODE |
| INQUIRE TRANSACTION | XPCT | READ | program | READ | TRANSACTION |

*Table 46. Resource and command check cross reference (continued)*

| EXEC CICS COMMAND | Resource Check | | | Check class=XCMD | |
|---|---|---|---|---|---|
| | **Class** | **Access** | **Resource** | **Access** | **Resource** |
| INQUIRE TSMODEL | | | | READ | TSMODEL |
| INQUIRE TSPOOL | | | | READ | TSPOOL |
| INQUIRE TSQUEUE (see note 4) | XTST | READ | tsqueue | READ | TSQUEUE |
| INQUIRE TSQNAME (see note 4) | XTST | READ | tsqname | READ | TSQUEUE |
| INQUIRE UOW | | | | READ | UOW |
| INQUIRE UOWDSNFAIL | | | | READ | UOWDSNFAIL |
| INQUIRE UOWENQ | | | | READ | UOWENQ |
| INQUIRE UOWLINK | | | | READ | UOWLINK |
| INQUIRE VTAM | | | | READ | VTAM |
| INQUIRE WEB | | | | READ | WEB |
| ISSUE | | | | | |
| LINK | XPPT | READ | program | | |
| LINK ACQPROCESS (see note 9) | XFCT | UPDATE | BTS repository file | | |
| LINK ACTIVITY / ACQACTIVITY (see note 9) | XFCT | UPDATE | BTS repository file | | |
| LOAD | XPPT | READ | program | | |
| MONITOR | | | | | |
| PERFORM DELETSHIPPED | | | | UPDATE | DELETSHIPPED |
| PERFORM DUMP | | | | UPDATE | DUMP |
| PERFORM RESETTIME | | | | UPDATE | RESETTIME |
| PERFORM SHUTDOWN | | | | UPDATE | SHUTDOWN |
| PERFORM STATISTICS | | | | UPDATE | STATISTICS |
| POINT | | | | | |
| POP HANDLE | | | | | |
| POST | | | | | |
| PURGE MESSAGE | | | | | |
| PUSH HANDLE | | | | | |
| QUERY SECURITY (see note 6) | | | | | |
| READ | XFCT | READ | file | | |
| READ PREV (see note 5) | | | | | |
| READ NEXT (see note 5) | | | | | |

*Table 46. Resource and command check cross reference (continued)*

| EXEC CICS COMMAND | Resource Check | | | Check class=XCMD | |
|---|---|---|---|---|---|
| | **Class** | **Access** | **Resource** | **Access** | **Resource** |
| READQ TD | XDCT | UPDATE | tdqueue | | |
| READQ TS (see note 4) | XTST | READ | tsqueue | | |
| RECEIVE | | | | | |
| RELEASE | XPPT | READ | program | | |
| RESET ACTIVITY (see note 9) | XFCT | UPDATE | BTS repository file | | |
| RESET ACQPROCESS (see note 9) | XFCT | UPDATE | BTS repository file | | |
| RESETBR (see note 5) | | | | | |
| RESYNC ENTRYNAME | | | | UPDATE | EXITPROGRAM |
| RETRIEVE | | | | | |
| RETURN / RETURN ENDACTIVITY(see note 9) (see note 11) | XFCT | UPDATE | BTS repository file | | |
| REWRITE | XFCT | UPDATE | file | | |
| ROUTE | | | | | |
| RUN (see note 9) (see note 11) | XFCT | UPDATE | BTS repository file | | |
| RUN / ASYNCH SYNC (see note 9) | XFCT | UPDATE | DFHLRQ file | | |
| SEND | | | | | |
| SET AUTOINSTALL | | | | UPDATE | AUTOINSTALL |
| SET CONNECTION | | | | UPDATE | CONNECTION |
| SET DB2CONN | | | | UPDATE | DB2CONN |
| SET DB2ENTRY | XDB2 | UPDATE | db2entry | UPDATE | DB2ENTRY |
| SET DB2TRAN | XDB2 | UPDATE | db2tran | UPDATE | DB2TRAN |
| SET DELETSHIPPED | | | | UPDATE | DELETSHIPPED |
| SET DOCTEMPLATE | | | | UPDATE | DOCTEMPLATE |
| SET ENQMODEL | | | | UPDATE | ENQMODEL |
| SET DSNAME | | | | UPDATE | DSNAME |
| SET DUMPDS | | | | UPDATE | DUMPDS |
| SET ENQMODEL | | | | UPDATE | ENQMODEL |
| SET FILE | XFCT | UPDATE | file | UPDATE | FILE |
| SET IRC | | | | UPDATE | IRC |
| SET JOURNALNAME | XJCT | UPDATE | journal | UPDATE | JOURNALNAME |
| SET MODENAME | | | | UPDATE | MODENAME |
| SET MONITOR | | | | UPDATE | MONITOR |
| SET NETNAME | | | | UPDATE | TERMINAL |
| SET PROCESSTYPE (see note 10) | XPTT | | | UPDATE | PROCESSTYPE |

*Table 46. Resource and command check cross reference (continued)*

| EXEC CICS COMMAND | Resource Check | | | Check class=XCMD | |
|---|---|---|---|---|---|
| | **Class** | **Access** | **Resource** | **Access** | **Resource** |
| SET PROGRAM | XPPT | UPDATE | program | UPDATE | PROGRAM |
| SET REQUESTMODEL | | | | UPDATE | REQUESTMODEL |
| SET STATISTICS | | | | UPDATE | STATISTICS |
| SET SYSDUMPCODE | | | | UPDATE | SYSDUMPCODE |
| SET SYSTEM | | | | UPDATE | SYSTEM |
| SET TASK | | | | UPDATE | TASK |
| SET TCLASS | | | | UPDATE | TCLASS |
| SET TCPIP | | | | UPDATE | TCPIP |
| SET TCPIPSERVICE | | | | UPDATE | TCPIPSERVICE |
| SET TDQUEUE (see note 3) | XDCT | UPDATE | tdqueue | UPDATE | TDQUEUE |
| SET TERMINAL | | | | UPDATE | TERMINAL |
| SET TRACEDEST | | | | UPDATE | TRACEDEST |
| SET TRACEFLAG | | | | UPDATE | TRACEFLAG |
| SET TRACETYPE | | | | UPDATE | TRACETYPE |
| SET TRANCLASS | | | | UPDATE | TCLASS |
| SET TRANDUMPCODE | | | | UPDATE | TRANDUMPCODE |
| SET TRANSACTION | XPCT | UPDATE | transid | UPDATE | TRANSACTION |
| SET TSMODEL | | | | UPDATE | TSMODEL |
| SET TSQNAME | XTST | UPDATE | tsqueue | UPDATE | TSQUEUE |
| SET TSQUEUE | XTST | UPDATE | tsqname | UPDATE | TSQUEUE |
| SET UOW | | | | UPDATE | UOW |
| SET UOWLINK | | | | UPDATE | UOWLINK |
| SET VTAM | | | | UPDATE | VTAM |
| SET WEB | | | | UPDATE | WEB |
| SIGNOFF | | | | | |
| SIGNON | | | | | |
| SPOOLCLOSE | | | | | |
| SPOOLOPEN | | | | | |
| SPOOLREAD | | | | | |
| SPOOLWRITE | | | | | |
| START (see note 7) | XPCT | READ | transid | | |
| STARTBR | XFCT | READ | file | | |
| STARTBROWSE ACTIVITY (see note 9) | XFCT | READ | BTS repository file | | |
| STARTBROWSE CONTAINER (see note 9) | XFCT | READ | BTS repository file | | |
| STARTBROWSE EVENT (see note 9) | XFCT | READ | BTS repository file | | |

*Table 46. Resource and command check cross reference (continued)*

| EXEC CICS COMMAND | Resource Check | | | Check class=XCMD | |
|---|---|---|---|---|---|
| | **Class** | **Access** | **Resource** | **Access** | **Resource** |
| STARTBROWSE PROCESS (see note 9) | XFCT | READ | BTS repository file | | |
| SUSPEND (see note 9) | XFCT | UPDATE | BTS repository file | | |
| SYNCPOINT | | | | | |
| TCPIP | | | | | |
| UNLOCK | | | | | |
| VERIFY PASSWORD | | | | | |
| WAIT | | | | | |
| WAIT JOURNALNAME | XJCT | READ | journal | | |
| WAIT JOURNALNUM | XJCT | READ | journal | | |
| WAITCICS | | | | | |
| WEB | | | | | |
| WRITE | XFCT | UPDATE | file | | |
| WRITE JOURNALNAME | XJCT | UPDATE | journal | | |
| WRITE JOURNALNUM | XJCT | UPDATE | DFHJnn | | |
| WRITE OPERATOR | | | | | |
| WRITEQ TD | XDCT | UPDATE | tdqueue | | |
| WRITEQ TS (see note 4) | XTST | UPDATE | tsqueue \| tsqname | | |
| XCTL | XPPT | READ | program | | |

**Notes:**

1. CANCEL does two checks. One is done against the transaction specified on the CANCEL command, and the other is done against the transaction associated with the reqid you are canceling (where applicable).
2. The CREATE CONNECTION command is subject to command security checking when you define a connection, for example; CREATE CONNECTION(con1) *Attribute*(...). However, when you use the CREATE CONNECTION COMPLETE or CREATE CONNECTION DISCARD command, no command security checking is performed unless you have been authorized to use COMPLETE and DISCARD. COMPLETE and DISCARD can only be used by those authorized to perform CREATE CONNECTION(con1) and CREATE SESSIONS(ses1) commands. Otherwise, ILLOGIC is returned.
3. An install surrogate user check can also occur.
4. A security check is performed when a DFHTST TYPE=SECURITY macro has been coded in the TST with a name that matches the TSname, or, if RDO is in use for TST and TSMODELS, and security is active for the model matching that queue.
5. No security check is performed, because the STARTBR command must be issued before this command and a security check is issued on the STARTBR command.
6. The QUERY SECURITY command is not controlled by resource or command checks, but it can cause them to be issued.

7. A start surrogate user check can also occur.
8. The resource check for the transid is only done if the reqid is associated with a transaction.
9. CICS business transaction services (BTS) application programming commands
10. CICS business transaction services commands that are subject to command security. All other CICS business transaction services commands are not subject to command-level security.
11. Any BTS commands that use timing operands will access the BTS LRQ file

# Glossary

For definitions of terms not in this glossary, see the *IBM Dictionary of Computing*, New York: McGraw-Hill, 1994.

**access.** The ability to obtain the use of a protected resource.

**access authority.** An authority that relates to a request for a type of access to protected resources. In RACF, the access authorities are: NONE, EXECUTE, READ, UPDATE, CONTROL, and ALTER.

**access list.** Synonym for standard access list. See also **conditional access list**.

**ACEE (accessor environment element).** A description of the current user including userid, current connect group, user attributes, and group authorities. An ACEE is constructed during user identification and verification.

**AOR (application-owning region).** A CICS address space whose primary purpose is to manage application programs. It receives transaction routed requests from a terminal-owning region (TOR). It may also contain file-related resources in a system that does not have a data-owning region (DOR). See also **DOR (data-owning region)** and **TOR (terminal-owning region)**.

**APPC (advanced program-to-program communication).** The implementation of the LU6.2 architecture. It is one of the intersystem communication (ISC) protocols that CICS uses.

**ASIF.** Access Security Information Field.

**ASIS.** Access Security Information Subfields.

**attribute.** See **user attribute**.

**AUDIT request.** The issuing of the RACROUTE macro with REQUEST=AUDIT specified. An AUDIT request is a general-purpose security audit request that can be used to audit a specified resource name and action.

**AUTH request.** The issuing of the RACROUTE macro with REQUEST=AUTH specified. The primary function of an AUTH request is to check a user's authorization to a RACF-protected resource or function.

**authority.** The right to access objects, resources, or functions. See **access authority**, **group authority**, and **class authority**.

**authorization checking.** The action of determining whether a user is permitted access to a protected resource. RACF performs authorization checking as a result of a RACHECK or FRACHECK request.

**automatic data set protection (ADSP).** A user attribute that causes all permanent data sets created by the user to be automatically defined to RACF with a discrete RACF profile.

**base segment.** Synonym for RACF segment.

**bind.** Refers to the SNA BIND command used to establish SNA sessions between systems, and to the CICS connection request used to establish multiregion operation (MRO) sessions for interregion communication. See also **bind-time security**.

**bind-time security.** In LU6.2 and MRO, the level of security applied when a request to establish a session is received from, or sent to, a remote system. Used to verify that the remote system is really the system it claims to be. Also known, in SNA terms, as **session security**. See also **bind**, **link security** and **user security**.

**BWO (backup while open).** A means of taking backups of VSAM files that CICS is concurrently updating.

**cache structure.** A coupling facility structure that contains data accessed by systems in a sysplex. MVS provides a way for multiple systems to determine the validity of copies of the cache structure data in their local storage.

**category.** Specifies the recommended security specifications needed for both the CICS transaction definitions and the corresponding RACF profiles.

**CEDF.** A CICS-supplied transaction for initiating the execution diagnostic facility program (DFHEDFP). It allows CICS commands issued by an application program to be traced online.

**CICS default userid.** The userid assigned to a terminal before the user signs on to CICS, and after the user signs off.

**CICS region userid.** The userid assigned to a CICS region at CICS initialization. It is specified **either** in the RACF started procedures table when CICS is started as a started task, **or** on the USER parameter of the JOB statement when CICS is started as a job.

**CICS segment.** The portion of a RACF user profile containing data for CICS.

**class.** A collection of RACF-defined entities: that is, users, groups, or resources (including general resources) that have similar characteristics. The class

names are USER, GROUP, DATASET, and the classes that are defined in the class descriptor table. See also **general resource** and **CDT (class descriptor table)**.

**Class descriptor table (CDT).** A RACF table consisting of an entry for each class except the USER, GROUP, and DATASET classes. The table is generated by invoking the ICHERCDE macro once for each class. See **CDT (class descriptor table)**.

**CLAUTH (class authority).** An authority that allows a user to define RACF profiles in a class defined in the class descriptor table. A user can have class authority to one or more classes.

**conditional access list.** An access list within a resource profile that associates a condition with a userid or group id and the corresponding access authority. If a user does not otherwise have the requested access, a conditional access list entry can allow access if the specified condition is true. For example, for program access to data sets, the condition is that the user must be executing the program specified in the access list. See also **access list** and **standard access list**.

**coupling facility.** The hardware element that provides high-speed caching, list processing, and locking functions in a sysplex.

**cross-memory services.** Services that apply to more than one private address space. Cross-memory services use the MVS common system area (CSA) storage for control blocks, not for data transfer. MVS requires that an address space using cross-memory services is non-swappable.

**current connect group.** The group with which a user is associated, for access checking purposes, during a terminal session or batch job. If a user does not specify a group on CICS sign-on, the user's default group is used.

A user may specify a group name when signing on. In this case, the group name specified becomes the current group.

**data security.** The protection of data from unauthorized disclosure, modification, or destruction, whether accidental or intentional.

**data set profile.** A profile that provides RACF protection for one or more data sets. The information in the profile can include the data set profile name, profile owner, universal access authority, access list, and other data. See **discrete profile** and **generic profile**.

**data sharing group, RACF..** A collection of one or more instances of RACF in a sysplex that have been identified to XCF and assigned to the group defined for RACF sysplex data sharing.

**default group.** In RACF, the group specified in a user profile that is the default current connect group.

**delegation.** The act of giving other users or groups authorities to perform RACF operations.

**discrete profile.** A resource profile that can provide RACF protection for only a single resource. For example, a discrete profile can protect only a single data set or minidisk.

**DOR (data-owning region).** A CICS address space whose primary purpose is to manage files and databases. Also known as a file-owning region (FOR). See also **AOR (application-owning region)** and **TOR (terminal-owning region)**.

**dynamic parse.** A method of parsing TSO commands according to syntax given in an external file.

**EDF (execution diagnostic facility).** A mechanism for debugging CICS transactions by displaying the results of CICS commands.

**entity.** A user, group, or resource (for example, a CICS resource) that is defined to RACF.

**entity class.** A resource class that contains individual resources rather than groups of resources.

**equivalent systems.** CICS regions having identical region userids. In regions connected by MRO, the link security userid can be the same as the userid of the region being connected to. In LU6.1 and LU6.2, the link security userid has to be the same as the userid belonging to the CICS region.

**EXCI (external CICS interface).** An application programming interface (API) that enables an MVS client program to call to call a program running in a CICS/ESA 4.1 system, and to pass and receive data using a communications area. The CICS program is invoked as if linked-to by another CICS program via a distributed program link (DPL) request.

**explicit sign-on.** Sign-on initiated through EXEC CICS SIGNON.

**explicit sign-off.** Sign-off initiated through EXEC CICS SIGNOFF.

**field-level access checking.** The RACF facility by which a security administrator can control access to fields or segments in a RACF profile.

**FOR (file-owning region).** A CICS address space whose primary purpose is to manage CICS files and data tables, especially shared data tables.

**FRACHECK request.** The issuing of the FRACHECK macro or the RACROUTE macro with REQUEST=FASTAUTH specified. The primary function of a FRACHECK request is to check a user's authorization to a RACF-protected resource or function. A FRACHECK request uses only in-storage profiles for faster performance. See also **authorization checking**.

**general resource.** Any system resource, other than an MVS data set, that is defined in the RACF class descriptor table (CDT). On MVS, general resources include DASD volumes, tape volumes, load modules, terminals, IMS and CICS transactions and other CICS resources, and installation-defined resource classes. See also **class**.

**general resource profile.** A profile that provides RACF protection for one or more general resources. The information in the profile can include the general resource profile name, profile owner, universal access authority, access list, and other data.

**generic profile.** A resource profile that can provide RACF protection for one or more resources. The resources protected by a generic profile have similar names and identical security requirements. For example, a generic data set profile can protect one or more data sets.

**global access checking.** The ability to allow an installation to establish an in-storage table of default values for authorization levels for selected resources. RACF refers to this table before performing normal RACHECK processing, and grants the request without performing a RACHECK if the requested access authority does not exceed the global value. Global access checking can grant the user access to the resource, but it cannot deny access.

**group.** A collection of RACF-defined users who can share access authorities for protected resources.

**group authority.** An authority that describes which functions a user can perform in a group. The group authorities are USE, CREATE, CONNECT, and JOIN.

**group data set.** On MVS, a RACF-protected data set in which either the high-level qualifier of the data set name or the qualifier supplied by an installation exit routine is a RACF group name.

**groupid (group identifier).** A string of one to eight characters that identifies a group to RACF. The first character must be A through Z, #, $, or @. The rest can be A through Z, #, $, @, or 0 through 9.

**group profile.** A profile that defines a group. The information in the profile includes the group name, profile owner, and users in the group.

**group terminal option.** A RACF function that allows users within a group to log on only from those terminals to which they have been specifically authorized.

**group-related user attribute.** A user attribute assigned at the group level that allows the user to control the resource, group, and user profiles associated with the group and its subgroups. Some of the group-related user attributes are group-SPECIAL, group-AUDITOR, and group-OPERATIONS.

**implicit sign-on.** Sign-on other than by means of CESN, CESF or EXEC CICS SIGNON

**implicit sign-off.** Sign-off other than by means of CESN, CESF or EXEC CICS SIGNOFF

**intersystem communication (ISC).** A protocol for communication between CICS regions using telecommunication.

**inventory control block(ICB).** The first block in a RACF database. The ICB contains a general description of the database.

**LANGUAGE segment.** The portion of a RACF profile containing information about the national language in which the user receives messages.

**link pack area (LPA).** (1) An area of main storage containing reenterable routines from system libraries. Their presence in main storage saves loading time. (2) An area of virtual storage that contains reenterable routines that are loaded at IPL time and can be used concurrently by all tasks in the system.

**link security.** A mechanism that limits one system's authorization to attach transactions and access resources in another. It works by signing on each end of a session to RACF when the session is bound. Each half-session then has the access requirements of a user, whose user profile is applied when a transaction is attached and whenever that transaction accesses a protected resource. See also **bind-time security**.

**list-of-groups checking.** A RACF option that allows a user to access all resources available to all groups of which the user is a member, regardless of the user's current connect group. For any particular resource, RACF allows access based on the highest access among the groups of which the user is a member.

**logging.** The recording of data about specific events.

**logon.** In CICS, the act of establishing a session with VTAM. Contrast with **sign-on**.

**logical unit (LU).** A port providing formatting, state synchronization, and other high-level services through which an end user communicates with another end user over an SNA network.

**MRO.** Communication between CICS systems in the same processor without the use of SNA networking facilities. See multiregion operation.

**multiregion operation.** Communication between CICS systems in the same processor without the use of SNA networking facilities.

**MVS.** Multiple virtual storage. Implies MVS/370, MVS/XA, or MVS/ESA.

**NetView® segment.** The portion of a RACF profile containing NetView logon information.

**NFS.** Network file system.

**OIDCARD (operator identification card).** A small card with a magnetic stripe encoded with unique characters and used to verify the identity of a terminal operator to RACF.

**owner.** The user or group who creates a profile, or is named the owner of a profile. The owner can modify, list, or delete the profile.

**PassTicket.** A password substitute that can be used only once and is valid only for a 10 minute interval between creation and use.

**password.** In computer security, a string of characters known to a computer system and to a user, who must specify it to gain full or limited access to the system and to the data stored within it. In RACF, the password is used to verify the identity of the user.

**persistent verification (PV).** PV is an APPC term that represents a level of conversation security between two logical units (LUs). PV provides a way of reducing the number of password transmissions by eliminating the need to provide a user ID and password on each attach (allocate) during multiple conversations between a user and a partner LU. The user is verified during the signon process and remains verified until the user has been signed off the partner LU.

**port of entry (POE).** The name and type of device from which a user signs on. CICS recognizes only TERMINALs and CONSOLEs.

**POSIT.** A keyword in the ICHERCDE macro that determines the position of a resource class in the RACF class descriptor table (CDT). All classes with the same POSIT value are controlled together by the SETROPTS command.

**preset-terminal security.** When a CICS region is started, the signing on of selected terminals as "users" whose userids are permanently associated with the terminal. Persons using these terminals have the authorizations given to the terminals.

**profile.** Data that describes the significant characteristics of a user, a group of users, or one or more computer resources. See also **connect profile**, **data set profile**, **directory profile**, **discrete profile**, **file profile**, **generic profile**, **general resource profile**, **group profile**, and **user profile**.

**profile list.** A list of profiles indexed by class (for general resources) or by the high-level qualifier (for DATASET profiles) and built in storage by the RACF routines.

**protected resource.** A resource that is defined to RACF for the purpose of controlling access to the resource. This book is primarily concerned with CICS resources. Some other resources that can be protected by RACF include DASD and tape data sets, DASD volumes, tape volumes, terminals, IMS transactions, IMS transaction groups, and any other resources defined in the class descriptor table.

**RACF (Resource Access Control Facility).** An IBM licensed product that provides for access control by identifying and verifying users to the system, authorizing access to protected resources, logging detected authorized and unauthorized attempts to enter the system, and logging detected accesses to protected resources.

**RACF database.** A collection of interrelated or independent data items stored together without unnecessary redundancy, to serve the Resource Access Control Facility (RACF).

**RACF-protected.** Pertaining to a resource that has either a discrete profile or an applicable generic profile. A data set that is RACF-protected by a discrete profile must also be RACF-indicated.

**RACF report writer.** A RACF function that produces reports on system use and resource use from information found in the RACF SMF records.

**RACF segment.** The portion of a RACF profile that contains basic information needed to define a user, group, or resource to RACF. Also called base segment.

**RACHECK request.** The issuing of the RACHECK macro or the RACROUTE macro with REQUEST=AUTH specified. The primary function of a RACHECK request is to check a user's authorization to a RACF-protected resource or function. See also **authorization checking**.

**RACINIT request.** The issuing of the RACINIT macro or the RACROUTE macro with REQUEST=VERIFY or REQUEST=VERIFYX specified. A RACINIT request is used to verify the authority of a user to enter work into the system.

**RACROUTE macro.** An assembler macro that provides an means of calling RACF to provide security functions. See also **FRACHECK request**, **RACHECK** request, and **RACINIT** request.

**remote user.** A user from another region.

**resource class.** See **resource group class**.

**resource group class.** A RACF class in which resource group profiles can be defined. A resource group class is related to another class, sometimes called a "member class". For example, resource group class GTERMINL is related to resource member class TERMINAL. See also **resource group profile**.

**resource member class.** See **resource group class**.

**resource group profile.** A general resource profile in a resource group class. A resource group profile can provide RACF protection for one or more resources with **unlike** names. See also **resource group class**.

**resource profile.** A profile that provides RACF protection for one or more resources. User, group, and connect profiles are not resource profiles. The information in a resource profile can include the data set profile name, profile owner, universal access authority, access list, and other data. Resource profiles can be discrete profiles or generic profiles. See **discrete profile** and **generic profile**.

**SAF (MVS System Authorization Facility).** An MVS interface invoked by CICS to communicate with an external security manager, such as RACF.

**scoping.** A mechanism for controlling multiple sign-on of the same userid to one or more CICS regions.

**segment.** A portion of RACF profile containing logically related fields. See **CICS segment, LANGUAGE segment, SESSION segment**, and **RACF segment**.

**session security, SNA.** See **bind-time security**.

**SESSION segment.** The portion of a RACF profile in the APPCLU class containing data used to control the establishment of sessions between logical units under LU 6.2.

**sign-on.** In CICS, to perform user identification and verification. Contrast with **logon**.

**SIT (system initialization table).** A table containing user-specified data that controls a system initialization process.

**SMF.** System Management Facility, a component of MVS for recording management data.

**SNSCOPE.** See **scoping**.

**SP commands.** The subset of CICS API commands (COLLECT, DISCARD, INQUIRE, PERFORM, and SET) that require the special CICS translator option, SP, and for which command security checking can be done.

**standard access list.** A list within a profile of all authorized users and their access authorities. Synonymous with access list. See also **conditional access list**.

**started transaction.** A transaction started via a CICS START command.

**surrogate terminal.** A logical representation of a terminal that is physically connected to another CICS region.

**surrogate user.** A user who is authorized to start work on behalf of another user. A surrogate user has authority to submit jobs for, or start CICS transactions for, or associate CICS resources with, the other user without needing to supply that user's password.

**sysplex.** A systems complex, consisting of multiple MVS images coupled together by hardware elements and software services. When multiple MVS images are coupled using XCF, which provides the services to form a sysplex, they can be viewed as a single entity.

**system authorization facility (SAF).** An MVS component that provides a central point of control for security decisions. It either processes requests directly or works with RACF or another security product to process them.

**system management facility (SMF).** A component of MVS for recording management data.

**Systems Network Architecture (SNA).** The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of, networks. The structure of SNA allows the end users to be independent of, and unaffected by, the specific facilities used for information exchange.

**TDQ.** System messages that CICS produces are commonly sent to Transient Data Queues, either intrapartition or extrapartition. For more information about TDQ, see the *CICS/ESA Resource Definition Guide*.

**TOR (terminal-owning region).** A CICS address space whose primary purpose is to manage terminals. See also **AOR (application-owning region)** and **DOR (data-owning region)**.

**UACC (universal access authority).** The default access authority that applies to a resource if the user or group is not specifically permitted access to the resource. The universal access authority can be any of the access authorities.

**user.** A person who requires the services of a computing system.

**user attribute.** In RACF, the extraordinary privileges, restrictions, and processing environments assigned to a user. The user attributes are SPECIAL, AUDITOR, CLAUTH, OPERATIONS, GRPACC, ADSP, and REVOKE. In CICS, the attributes of a user obtained from the CICS segment of the user profile, namely OPCLASS, OPIDENT, OPPRTY, TIMEOUT, and XRFSOFF.

**user data set.** On MVS, a data set defined to RACF in which either the high-level qualifier of the data set name or the qualifier supplied by an installation exit routine is a RACF userid.

**user identification and verification.** The acts of identifying and verifying a RACF-defined user to the system during logon or batch job processing.

**user name.** One to twenty alphanumeric characters that represent a RACF-defined user.

**user profile.** A description of a RACF-defined user that includes the userid, user name, default group name, password, profile owner, user attributes, and other information. A user profile can include information for subsystems such as CICS, DFP, and TSO. See also **CICS segment**.

**user security.** The facilities for, or action of, verifying that a user is authorized (1) to run a transaction, and (2) to access the resources and use the commands that a transaction invokes.

**userid (user identifier).** A string of characters that uniquely identifies a user to a system. On CICS, a userid is one to eight alphanumeric characters. On TSO, userids cannot exceed seven characters and must begin with an alphabetic, #, $, or @ character.

**verification.** The act of confirming that a user is eligible to use a RACF-defined userid. RACF identifies the user by the userid, and verifies the user by the password (or PassTicket) or operator identification card (OIDCARD) supplied during sign-on processing, or the password supplied on a batch by JOB statement.

**VLF.** Virtual Lookaside Facility, a service offered by MVS that makes it possible to create and retrieve named data objects, such as members of a partitioned data set, in virtual storage. VLF uses data spaces to keep large amounts of data in virtual storage. RACF uses VLF to keep information about signed-on users in storage.

**XCF.** Cross-system coupling facility.

**Xname resource classes.** The general resource classes that CICS uses based on X**name** system initialization parameters. For example, if XTRAN=YES is specified, TCICSTRN and GCICSTRN are used.

**Xname system initialization parameters.** CICS system initialization parameters: XAPPC, XCMD, XDCT, XFCT, XJCT, XPCT, XPPT, XPSB, XTRAN, and XTST, which are related to resource security checking, as the X**name** parameters.

**XRF (extended recovery facility).** A software function that minimizes the impact of various system failures on users by transferring activity to an alternate system in the same MVS image or a different one.

# Index

## Special Characters



## A



## B



## C

# D

# E



# F



# G



# H



# I

## J



## K



## L



## M



## N



## O



## P

# Q



# R

# S

## T



## U

# V



# W



# X

# Sending your comments to IBM

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book.

Please limit your comments to the information in this book and the way in which the information is presented.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

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

You can send your comments to IBM in any of the following ways:

- By mail, to this address:

  Information Development Department (MP095)
  IBM United Kingdom Laboratories
  Hursley Park
  WINCHESTER,
  Hampshire
  SO21 2JN
  United Kingdom
- By fax:
  - From outside the U.K., after your international access code use 44–1962–870229
  - From within the U.K., use 01962–870229
- Electronically, use the appropriate network ID:
  - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
  - IBMLink™: HURSLEY(IDRCF)
  - Internet: idrcf@hursley.ibm.com

Whichever you use, ensure that you include:

- The publication number and title
- The topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.

Program Number: 5655-147

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC33-1701-33

Spine information:

CICS TS for OS/390 CICS RACF Security Guide Release 3