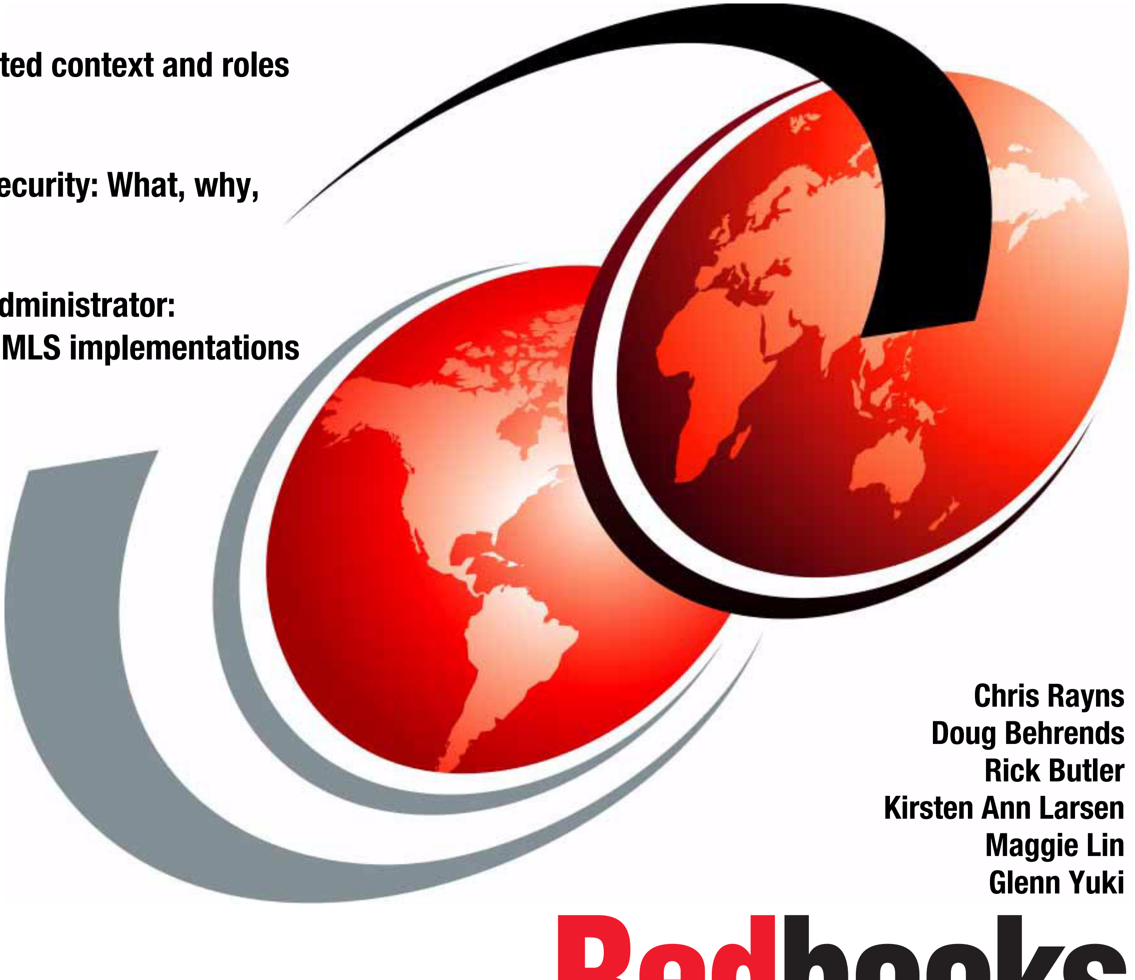
Securing DB2 and Implementing MLS on z/OS

- DB2 V9 trusted context and roles
- Multilevel security: What, why, and how
- Vanguard Administrator: Simplifying MLS implementations



Chris Rayns
Doug Behrends
Rick Butler
Kirsten Ann Larsen
Maggie Lin
Glenn Yuki

Redbooks

ibm.com/redbooks

International Technical Support Organization

Securing DB2 and Implementing MLS on z/OS

March 2007

SG24-6480-01

Note: Before using this information and the product it supports, read the information in "Notices" on page ix.

Second Edition (March 2007)

This edition applies to Version 9 of IBM DB2.

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.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

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

AIX 5L™
AIX®
CICS®
DB2 Connect™
DB2 Universal Database™
DB2®
DFSMSdss™
DFSMSrmm™
DFSMS™
DFS™
Domino®
DRDA®
eServer™
IBM®
IMS™
iSeries™
Lotus Notes®
Lotus®
MVS/ESA™
MVS™
Notes®
OMEGAMON®
OS/390®
OS/400®
PR/SM™
Print Services Facility™
Processor Resource/Systems Manager™
pSeries®
RACF®
Rational®
Redbooks (logo) ™
Redbooks™
RMF™
System z9™
System z™
Tivoli®
VTAM®
WebSphere®
xSeries®
z/OS®
z/VM®
z9™
zSeries®

The following terms are trademarks of other companies:

SAP NetWeaver, SAP, and SAP logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries.

Oracle, JD Edwards, PeopleSoft, Siebel, and TopLink are registered trademarks of Oracle Corporation and/or its affiliates.

Java, Java Naming and Directory Interface, JDBC, J2EE, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

i386, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

Preface

Today's computing environment is subject to increasing regulatory pressures and potential malicious attacks. Regulatory compliance, security, and audits are in the daily headlines and growing more prominent.The security of the information to which you have been entrusted has never been more critical. The reality of compliance is too complex. Compliance demands that you work carefully to set up a strong, comprehensive set of policies and controls. That means controls that consider operational data, financial data, unstructured data, spreadsheets, e-mail, and business intelligence data.

We have a responsibility to secure all business data and especially sensitive customer data. Security can be difficult to manage. IBM® DB2® for z/OS® already resides on one of the most secure platforms in the industry. IBM System z™ servers are routinely used by enterprises around the world to support their mission-critical applications. The mainframe's strengths in security stem in part from its history of supporting sensitive data for large enterprises, resulting in security features being built into its design for many decades. It also benefits from a system-wide approach with security capabilities built into the hardware, operating systems, databases, key middleware, and more. Its highly evolved layers and security management components give it a fundamental advantage over other systems.

The key value that we get from improved security is customer confidence.

DB2 9, z/OS Version 1.8, and RACF® provide you with additional capability while assisting you with security management.

In this IBM Redbooks™ publication, we provide an update to major existing security functionality, covering:

- RACF
- Multilevel security
- Security labels
- DB2 row-level security

We explain how to use the Vanguard Administrator software (third party) to facilitate and simplify the implementation of multilevel security (MLS) and security labels.

We also describe some significant new DB2 functionality:

- DB2 network-trusted contexts
- DBA roles

We illustrate how to use this functionality with a sample application.

The team that wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, San Jose Center.

Chris Rayns is a CICS/Security Project Leader. He is an IT Specialist at the International Technical Support Organization, Poughkeepsie Center. He writes extensively on all areas of CICS® and zSeries® Security. Before joining the ITSO, Chris worked in IBM Global Services in the U.K. as a CICS IT Specialist.

Doug Behrends is a Sr. Professional Services Consultant with Vanguard Integrity Professionals based in Las Vegas, NV. He provides his mainframe security expertise to many Fortune 500 corporations through training, security assessments, and remediation services. Before joining the Vanguard, he worked in a variety of industries, including manufacturing, insurance, and financial services. Doug has performed a wide variety of IT duties over his 25+ year IT career. He has held positions such as operations manager, DB management, office systems, and application and systems programming.

Rick Butler is a DBA Specialist and DBA Manager at Bank of Montreal (BMO) Financial Group, based in Toronto, Canada. He has 20 years of experience in designing and implementing DB2 databases for mainframe banking applications. Prior to this, he worked with network and hierarchical databases, and in London, as a programmer and designer on a foreign exchange application. He also has data warehousing experience with DB2 on IBM AIX® 5L™. He has expertise in access path statistics and DRDA®. He holds a bachelor of science degree in operations research from Université de Montréal. Rick is coauthor of *The Business Value of DB2 UDB for z/OS*, SG24-6763.

Kirsten Ann Larsen is a Senior IT Specialist and works as a DB2 database administrator with IT Delivery in IBM Denmark. She has 10 years of experience working with of DB2 for OS/390® and z/OS and has followed developments since DB2 V4. She holds an M.Sc. in computer science from Aarhus University.

Maggie Lin is a Software Engineer at the IBM Silicon Valley Lab in San Jose, California. She has been working with DB2 for z/OS for six years, with experience in system testing for DB2 for three years. She currently works in distributed database development for DB2 for z/OS. She holds a master's degree in computer science and engineering from Pennsylvania State University.

Glenn Yuki is a Software Engineer at the IBM Silicon Valley Lab in San Jose, California. He has been working with DB2 for z/OS for eight years in development and test. He holds a bachelor of science degree in mathematics from Harvey Mudd College.

Thanks to the following people for their contributions to this project:

Roger Miller, IBM Santa Teresa, is an evangelist, architect, designer, and developer who has a focus on performance, scalability, and concurrency. He has worked for 27 years on DB2 for z/OS development, product design, and strategy. He often helps customers to use the product, answers many questions, and presents frequently to user groups.

Jim Pickel
IBM Santa Teresa

Roland Grant, Mike Holmans, Patrick Kappeler, Jun Ogata, Norbert Verbestel, Matthew Yates who worked on the original version of this IBM Redbooks publication.

Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our Redbooks to be as helpful as possible. Send us your comments about this book or other Redbooks in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an e-mail to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Summary of changes

This section describes the technical changes made in this edition of the book and in previous editions. This edition may also include minor corrections and editorial changes that are not identified.

Summary of changes
for SG24-6480-01
for *Securing DB2 and Implementing MLS on z/OS*

March 2007, Second Edition

This revision reflects the addition, deletion, or modification of new and changed information described here.

New information

- Chapter 1, “What is new in security”
- Chapter 4, “Vanguard solution”
- Chapter 8, “Network trusted contexts and roles”
- Chapter 9, “A WebSphere implementation”
- Appendix A, “Trusted context syntax”
- Appendix C, “Enterprise Identity Mapping”

Changed information

- All chapters

1

What is new in security

In today's world of e-business, and data on demand from everywhere, there is an increased focus on security and privacy of information. Everyone seems to be more aware of security today. There is a general heightened awareness and interest in security and privacy. Most of us can identify information security professionals and privacy officers in our respective organizations.

We are all cognizant of the need to protect the privacy of our own identity. Improving integration and making security more robust and easier to manage is very important. Customers ask for a wide range of enhancements for security. New options for tighter security, more granularity, and more information for additional flexibility in applications and SQL are needed.

Regulatory rules are sometimes vague and ensuring compliance can require significant costs in finances and human resources. The degree of security needed and specific concerns vary.

Although there is considerable ambiguity around these requirements, the penalties for non-compliance are not ambiguous. They are very clear. They include fines, sanctions, lawsuits, possible imprisonment of top executives, and what might be the worst of all, negative publicity for the company.

The difficulty of interpreting laws into IT controls leaves many companies struggling. Individual businesses must understand legislation in the context of their enterprise and all markets they are active in and act accordingly. There is no simple remedy to automate compliance efforts and or solve compliance issues.

In this book, we discuss existing and new tools that enable you to have control over your z/OS resources, your DB2 subsystem, and your DB2 data and who can access these.

The unique capabilities of the System z technology have earned some of the highest levels of industry security certification and helped customers streamline compliance efforts on their mission-critical applications.

The mainframe's security capabilities fall into three categories. First, there is user identification and authorization, designed so that only eligible users can access their applications and data. Because for many years the mainframe has been designed to run multiple applications simultaneously on the same server, these capabilities are mature, proven technologies.

Second, there are intrusion detection services, designed to help detect and prevent unauthorized intrusion to applications or data. Third, there is encryption so that data that is being transmitted over the network or stored on tape for archival or distribution purposes can be protected, if that data gets into the wrong hands.

Security covers control of access, whether to z/OS, the DB2 subsystem, DB2 tables, data sets, DASD volumes, the TCP/IP stack, or other resources. This chapter discusses why security and privacy are of the utmost importance in order to stay in business and comply with regulatory demands.

It also offers a high-level overview of facilities available on the IBM System z platform, in DB2, and RACF to support the implementation of the appropriate security plan for your business. We describe some of these facilities in detail in later chapters; some we only mention in the overview for completeness.

In this chapter, we describe:

- The main factors driving the need for security
- Improved and new solutions for addressing security
- The goals of this book

1.1 The need for security and protection of privacy

In the past, for the applications we developed, it was enough to have data that was accurate, had integrity, and was recoverable. A few years ago, it became important to describe the data in our applications in business terms and share that metadata with internal business and systems staff. Then, there was a need to make data available to business partners and customers through new channels, over the LAN, the telephone, the Web, and wireless connections.

In the current environment, the number of databases and the amount of digital data we store are both growing dramatically. Some of this is due the needs of data warehousing and the interest in digitizing more data, even unstructured data.

Ambuj Goya, General Manager of IBM Information Management Software, made the following comments at the IBM IOD conference (October 2006):

> A Berkeley study states there was 5 Exabytes of data in the world in 2002, in 2010 the amount of data will double, not in years, months or days, but every 11 hours. We need to treat information as an asset, last year 10 million people had their identity stolen. Protect your information assets, this is critically important.

Curt Cotner, CTO for Database Servers, made the following comments at the conference:

> New generation of business leaders have grown up with new technologies, as they move into and up the Enterprise, they will expect to see collaboration and sharing of information, this will put pressure on the Enterprise. See growing need to have integrated technology and be able to easily get at any information we want. Text analytics search engine will become part of the enterprise, control will be on business user side.

However, there is a also a contrary need to restrict access to internal information within a company. Some examples include project plans and documents, configuration diagrams, marketing strategies, and intellectual property. Internal information needs to be classified and made available only to those who really need it. There can also be legal requirements to restrict access to information across divisions of a company.

More data is available to more people through more channels. The requests for data come at any time from anywhere.

Now it has become crucial for businesses to secure all of their data all of the time, whether in production, in development, at rest locally, remotely, or in transit across town or across the world.

In today's business world, there is a critical need for increased security due to:

- Compliance with security legislation. Some examples include:
  - Health Insurance Portability and Accountability Act of 1996 (HIPAA) U.S.; health care.
  - Gramm-Leach-Bliley Act of 1999 (GLBA) U.S.; financial services.
  - Sarbanes-Oxley, an act that aims to protect investors by improving the accuracy and reliability of corporate disclosures.
  - BASEL II.
  - FDA: Food and Drug Administration 21 DFR Part 11.
  - COPPA: Children's Online Privacy Protection Act of 2000.
  - In the state of California, a unique law requires that a business notify people in the event that their personal information might have been stolen.
  - Support is growing for a Federal law protecting consumers from corporate security breaches U.S.
  - OECD guidelines (Organization for Economic Co-operation and Development).
  - SEC Rule 17a-4: Records to be preserved by certain exchange members, brokers, dealers.
  - USA Patriot Act: Uniting and Strengthening America by Providing Appropriate Tools Required to Intercept and Obstruct Terrorism act of 2001.
  - Canada Pipeda act, Personal Information Protection and Electronic Documents Act.
  - EU Data Protection Directive 95/46/EC.
  - U.K. Data Protection Act (October 1998).
  - Sweden Data Protection Act (1998).
  - German Federal Data Protection Act.
  - Japanese "Protecting Personal Freedom" Act.
  - PCI Data Security Standard (DSS), owned and distributed by the PCI Security Standards Council.
- Emergence of *storage area network*s (SANs):

  The need for safely storing data in a widely accessible device has increased.

  A possible security issue presents itself as different applications and different platforms are accessing common hardware devices.

- Interest in Common Criteria:

  An international initiative to develop criteria for evaluation of IT security.

- There are many sources of breaches:

  - Sabotage
  - e-business phishing scams
  - Intrusive spyware
  - Browser weaknesses
  - Viruses
  - Computer worms
  - Spam
  - Attacks to corporate e-mail
  - System penetration
  - Web site defacement
  - Misuse
  - Telecom fraud
  - Unauthorized access
  - Notebook computer theft
  - Lost backups in transit
  - Financial fraud
  - Abuse of wireless network
  - Insider abuse
  - Theft of proprietary data
  - Denial of service attacks
  - Fake vendors acquiring customer credit information
  - Bogus wireless networks
  - SQL Injection

  This list is not exhaustive.

- And there are many reasons for not reporting breaches:

  - Unaware of law enforcement interest.
  - Choosing civil remedies.
  - Competitors might use information to their advantage.
  - Negative publicity might hurt corporate image.

When identity theft from a financial corporation occurs, a thief can use that information to acquire credit cards and take out loans based on the stolen information. A financial corporation will likely have to bear the significant cost of informing customers and changing PINs and accounts to protect their clients. They also have to attempt to regain the public trust that was lost, and in many cases, that trust is an abstract opinion. The impact to the consumer is that they must continuously monitor their credit long past the original event.

Data security is a top issue in today's business world because the customer's trust in a business depends on their perception of strong security around the data that is used in running the business.

There is a fundamental need to secure a company's data and a critical and obligatory need to protect sensitive data.The close relationship between DB2, z/OS, RACF, and the zSeries 990 platform is unique in the industry. This tight partnership is a strategic, ongoing IBM goal, a significant differentiator that enables world class security.

1.2 Improved and new facilities

The latest z/OS versions 1.7 and 1.8, as well as DB2 for z/OS in both version 8 and version 9, introduce powerful new security features, offering improved support for lifting some of the challenges related to the security issues of today.

The new features constitute a wide range of enhancements, enabling customers to choose the level of security and type of implementation that best suits their needs.

We briefly mention the major achievements to provide an overview of what is available. We discuss most of these in detail in later chapters.

Multilevel security with row-level granularity in DB2

With DB2 UDB for z/OS V8 came a powerful security enhancement, namely multilevel security (MLS) with row-level granularity. This enhancement stands on the shoulders of multilevel security on z/OS in general, but functions independently of whether MLS is implemented on the whole of z/OS.

The basic idea of MLS with row-level granularity is that any user reading or updating data in a DB2 table needs to be allowed to handle only the rows that his or her security label allows. Each row in a table is assigned a security label, and a user can read the row, only if his or her label dominates the label of the row. Similar rules apply for updating rows in a table with row-level security, only where updating within an MLS environment is concerned, other principles concerning write-down (that is, the declassification of data) influence the result of the update.

We return to both MLS itself and row-level security in much more detail in later chapters.

Network-trusted context

A powerful security enhancement in DB2 V9 for z/OS is the introduction of the network-trusted context. In itself, it supplies the ability to establish a connection as trusted when connecting to DB2 for z/OS from a certain location. Having established the connection, it provides the possibility of switching to another user ID, thus giving the opportunity of taking on the identity of this other user ID only within the trusted context.

In addition, it is possible to assign a role to a user of a trusted context. The role can be granted privileges and can therefore represent a role within the organization in the sense that it can hold the sum of privileges needed to perform a certain job or role.

These two constructs together supply security enhancements for a variety of different scenarios. ranging from any three-tier layered application such as SAP® to the daily duties of a DBA maintaining the DB2 subsystem.

The possibilities are many and varied, and we return to them in detail in later chapters. To mention some of the benefits:

- A role can be used as a single database authid that can be used to simplify administration of dynamic SQL privileges.
- The user’s authid can be used to run database transactions, so the DB2 audit is able to identify the users individually (an important capability for meeting some regulatory compliance requirements).
- The trusted context retains many of the performance benefits of connection pooling.
- The trusted context and role support can be used to implement DBA privileges that can easily be disconnected and reconnected to individual employees. This provides function similar to shared SYSADM or DBADM user IDs, but avoids the audit compliance problems associated with shared user IDs.

Improved trace filtering in DB2

New filtering options on the START TRACE command give the ability to select your trace records more carefully, thus minimizing the amount of data collected or monitoring more closely access from specific users or locations, or both. The new possibilities include:

- Roles (used through a trusted context) can be filtered with new keyword ROLE.
- Other trace filter capabilities for package name, collection name, and so on.
- New keywords are introduced to provide *exclude* trace filtering capabilities.

Audit Management Expert: A new tool for DB2

Many auditors have little familiarity with databases and even less with determining which objects to audit, starting appropriate traces, collecting and filtering, and producing reports. So, the task usually requires a lot of the DBA's time to interpret what an auditor is requesting and to gather that information. Responding to these auditors' requests can distract DBAs from more strategic responsibilities.

To address this situation, IBM recently released the newest member of the IBM DB2 regulatory compliance suite, IBM DB2 Audit Management Expert. This tool can be used by DBAs and auditors themselves to provide all the information needed for an audit while also maintaining database security.

With an easy to use graphical interface, IBM DB2 Audit Management Expert requires no DB2 expertise—allowing auditors to gather and correlate a coherent view of DB2 activity. Auditors are not required to log in to DB2 nor are they able to directly manipulate any DB2 resource, thus securing a further layer of protection for your DB2 data.

Auditors collect log and trace data in an audit repository, and then view, analyze, and generate comprehensive reports on the data. They can selectively filter SELECT, INSERT, UPDATE, and DELETE activity by user or by object and export these filters for use on another DB2 system.

The DB2 Audit Management Expert for z/OS V1.1 focuses on aiding compliance with a plethora of regulations.

Internet encryption advances

Encryption of data over the wire is made easier in z/OS V1.7. The Communications Server now offers Application Transparent Transport Layer Support (AT-TLS), which facilitates the use of SSL encryption of data on behalf of the application.

SSL encryption has been available on z/OS for a long time, but with this new facility, more applications will be able to offer this level of encryption.

DB2 V9 for z/OS makes use of this new facility and now offers SSL encryption using a new secure port.

When acting as a requester, DB2 for z/OS can request a connection using the secure port of another DB2 subsystem. When acting as a server, and from within a trusted context, SSL encryption can be required for the connection.

New separate IBM Encryption Key Manager

We also increase security through encryption. In the payment card industry, IT organizations are required to encrypt DB2 data in tables, indexes, image copies, and even logs, which can create a performance drag. With DB2 V9, you can use System z disk and tape controllers to encrypt the data at rest on these devices, and System z has advanced capabilities to centrally manage all of the encryption keys. By off loading the encryption work to the storage devices, you can save a lot of server processing power.

1.3 The goals of this book

In this book, we discuss a number of different aspects of security.

We emphasize the improvements in DB2 V8 and V9 for z/OS and RACF V1.7 and V1.8, as well as within a number of different tools. We also describe multilevel security in detail and demonstrate how you can use Vanguard Administrator to ease the implementation of MLS.

The goal is to provide an understanding of both multilevel security and recent security enhancements in DB2, and how it is possible to combine and tailor the different options to meet the security needs of your system.

There is not one solution, or one correct way to implement security on System z platform. There are a number of different elements and flavors of these elements, and they can be combined in various ways. The object of this book is to provide an understanding of some of the most important elements, giving you the framework for building the right security setup at your installation.

2

Security labels

In this chapter, we describe what security labels are, how they are used, and their association to the RACF resource classes SECLABEL, SECDATA, and SECLEVEL. We also discuss the security labels that the system automatically creates for you, how to define security labels, and how to authorize users to use security labels. We acknowledge *z/OS Planning for Multilevel Security and the Common Criteria*, GA22-7509-05, for information in this chapter.

2.1 Security labels and data classification policies

A *security label* enables an installation to classify subjects and objects according to a data classification policy, identify objects to audit based on their classification, and protect objects such that only appropriately classified subjects can access them.

Subjects and objects

In this book, a *subject* is an entity that requires access to system resources. These system resources are *objects*.

Examples of subjects are:

- Human users
- Started tasks
- Batch jobs
- z/OS UNIX® daemons

Examples of objects are:

- Data sets
- Row within a DB2 table
- Commands
- Terminals
- Printers
- DASD volumes
- Tapes

Subjects are defined to RACF, for example, a user or started task will have a RACF user ID. Objects, other than rows in a DB2 table, are also defined to RACF as either a *resource profile* or *data set profile*. The terms *subject* and *user ID* have the same meaning in this book and can be used interchangeably.

In a multilevel secure system, subjects and objects have a security label associated with them. The security label is defined to RACF in the resource class SECLABEL. Rows in a DB2 table have a security label associated with them by means of a special column in the table that contains only the eight-character security label that defines the security classification of each row in that table. A subject's security label determines whether the subject is allowed to access a particular object. An object's security label indicates the sensitivity of that object's data.

A subject is authorized to use a security label by having been permitted READ access to the resource profile in the SECLABEL class in RACF, which defines the particular security label. A TSO user can have a default security label defined in RACF if desired. The appropriate security label can then be chosen or allowed to default at logon time. Batch users can also override their default security label by specifying the SECLABEL parameter on the JOB statement when a job is submitted to the system. Other subjects, such as started tasks or z/OS UNIX daemons, must have their security label defined as their default because it is not possible to choose. Users of the (new to V9) TRUSTED CONTEXT statement can also override the DEFAULT SECURITY LABEL defined for the trusted context.

Subjects can have authorization for more than one security label, but can use only one security label at any given time. TSO users gets assigned their chosen security label at logon time, and this remains for the duration of their session. If a user needs to switch security labels, the user needs to log off and log back on again to choose another one from the TSO logon panel. See Figure 2-1 on page 19.

Access to objects (that is, data) in a multilevel secure system is controlled by both the subject and the object having a security label assigned to them. A subject can access information in an object only when the subject's security label entitles the access. The entitlement to access the data depends on the theory of dominance of the subject’s security label over the object’s security label. If the subject's security label does not have enough authority, the subject cannot access the information in the object.

In a non-multilevel secure system, access to objects or data is controlled by discretionary access control (DAC). In a multilevel secure system, access to objects or data is controlled by mandatory access control (MAC), in addition to the DAC.

A security label is used as the basis for mandatory access control decisions. By assigning security labels, the security administrator can ensure that data of a certain classification is protected from access by a user of a lesser security classification. Security labels provide the capability to maintain multiple levels of security within a system. By assigning a security label to a resource, the security administrator can prevent the movement of data from one level of security to another, that is, de-classification of data.

2.2 Mandatory access control

MAC is the principle way of restricting access to objects based on the sensitivity of the information that the object contains and the authorization of the subject to access information within that level of sensitivity. The sensitivity of objects is defined by the security administrator, not the owner of the data. The security administrator needs to be someone with the RACF system SPECIAL attribute. The security label indicates the hierarchical level of classification of the information (such as top secret, sensitive, or unclassified), and also inherently indicates to which non-hierarchical category the information belongs within that level (such as project A, project B, or project C). The security administrator also controls each subject's access to an object or data within the multilevel secure system by specifying which security labels the subject can use. A subject can access information in an object only when the subject's security label entitles the access. If the subject's security label does not have enough authority, the subject cannot access the information in the object.

Mandatory access control is based on the theory of *dominance* between security labels. This dominance is based on each security label's combination of security level and zero or more categories.

2.3 Discretionary access control

DAC is the principle of restricting access to objects based on the identity of the subject (the user or the group to which the user belongs). Discretionary access control is implemented using *access control lists*. A RACF resource profile contains an access control list that identifies the users who can access the resource and the authority (such as read or update) the user is allowed in referencing the resource. The access control list overrides the universal access list of the resource profile. The security administrator defines a profile for each object (a resource or group of resources) and updates the access control list for the profile. This type of control is *discretionary* in the sense that subjects can manipulate it, because the owner of a resource, in addition to the security administrator, can identify who can access the resource and with what authority.

2.4 Security levels and security categories

Security labels establish an association between a RACF *security level* and a set of zero or more RACF *security categories*. For example, a system might have three security levels, such as top secret, sensitive, and unclassified. It can have various security categories that span these security levels and represent individual projects or departments, for example, project A, project B, and project C. Note the following definitions:

Security level This is controlled through a resource profile called SECLEVEL in the RACF class SECDATA. It defines the hierarchical degree of sensitivity of the data. The security administrator defines in RACF each level, which consists of a name and a numerical security level (the higher the number, the higher the security level). In the previous example, you might define top secret as level 30, sensitive as level 20, and unclassified as level 10. The security administrator can define up to 254 different levels, although this is an impractical amount. We recommend keeping the setup simple and concise.

Security category This is controlled through a resource profile called CATEGORY in the RACF class SECDATA. It is non-hierarchical and further qualifies the access capability. The security administrator can define zero or more categories that correspond to some grouping within an organization that has similar security classifications.

Security label After defining the SECLEVEL and CATEGORY profiles, the security administrator defines a resource profile in the class SECLABEL for each security label. The security label is a name of up to eight uppercase alphanumeric or national characters. The national characters are $(X’5B’), #(X’7B’), and @(X’7C’). The first character cannot be numeric. Each security label name must be unique. Each SECLABEL profile defines a combination of a SECLEVEL member and zero or more members of the CATEGORY profile, which are required for that particular security label. Note that you do *not* need to define a security label for *every* possible combination of level and category.

Guideline: Although the system allows the definition of several thousand categories, up to 254 security levels, and unlimited security labels, define *only* as many as you really need. A large number of levels and categories can decrease performance, particularly at initial program load (IPL) time and for the SETROPTS RACLIST(REFRESH) command. DB2 caches security labels to avoid extra calls to RACF. The impact was measured and documented in *DB2 UDB for z/OS Version 8 Performance Topics*, SG24-6465, Section 4.9, “Row level security.” The caching works best if there are a relatively small number of security labels to be checked compared with the number of rows accessed in a long commit scope.

Security labels that the system creates

The system creates four labels automatically at initialization time:

- SYSHIGH: This label is equivalent to the highest security level defined by the security administrator and all categories defined by the security administrator. It dominates all other security labels in the system. It will always be of a higher level than any other in the system. Restrict SYSHIGH to special system-level address spaces such as consoles, and to system programmers, system operators, and system administrators.
- SYSLOW: This label is equivalent to the lowest security level defined by the security administrator and has no categories. It is dominated by all other security labels. Only use SYSLOW for resources that have no classified data content. It is appropriate to use SYSLOW for data sets that IBM supplies for which the following is true:
  - Most users only need to read them.
  - A limited number of users, such as system programmers, might need to update them. They must only do so when running at a very low classification to prevent them from accidentally putting classified data into the data sets.

  If a resource is not in a class that requires reverse mandatory access checks or equal mandatory access checks, assigning a security label of SYSLOW to the resource allows all subjects to pass a mandatory access check for read access to the resource. Subjects still must pass the discretionary access check in order to access the resource.
- SYSNONE: This label is treated as equivalent to any security label to which it is compared. Only use SYSNONE, like SYSLOW, for resources that have no classified data content. It is different from SYSLOW in that all users might need to update resources with SYSNONE, even when running at a high classification. It is intended for use on resources that must be written to at different security labels when write-down is not allowed. It is used to ensure that a user is permitted read/write access to a data set such as a catalog.

  Follow these guidelines for assigning the SYSNONE security label:
  - Use SYSNONE for a data set only when some other process (such as catalog management, or a program through program access to data sets, or PADS) mediates the user's access to ensure that no classified data is written into the data set.
  - Use SYSNONE for a z/OS UNIX file only when you limit discretionary access to the file to a specific UID, and the only access to the file is through a z/OS UNIX program with the `setuid` option that switches to that UID and ensures that no classified data is written into the data set.
  - Do not use SYSNONE for users.

- SYSMULTI: This label is considered to be equivalent to *any* defined security label. It is intended for use by:
  - Server or daemon address spaces whose implementation and documentation explicitly support multilevel security, giving them the ability to perform and separate work for users running with different security labels.
  - zFS directories that can contain data with different security classifications, such as the root directory of a file system.

  Follow these guidelines for assigning the SYSMULTI security label:

  - SYSMULTI is not an appropriate security label for a data set or z/OS UNIX file, unless access to the data set is mediated through PADS or some other mechanism to ensure that either of the following is true:
    - Users can only write, and not read.
    - Users can only read appropriate parts of the data.
  - SYSMULTI says that the resource manager has to handle the multiple SECLABELs within, so SYSMULTI is needed in DB2 for the appropriate level of a DB2 subsystem or a DB2 table that has multiple SECLABELs within it.
  - SYSMULTI is not generally appropriate for users.

2.5 Defining security labels

After you activate the SECLABEL class, users who log on without a security label can no longer access resources protected by security labels. Therefore, assign a default security label to all users before you assign a security label to a commonly accessed resource, such as SYS1.BRODCAST. Each user who does not have a default security label must specify a security label at logon time or else be denied access to the system. If you assign a security label to a resource profile while the SECLABEL class is active, the security label you assign to the users must allow the required users to access the resource.

Before defining the SECLABELs, the security administrator must also define two profiles in the RACF SECDATA resource class: one to define the security levels (SECLEVEL), and the other to define the security categories (CATEGORY) for the system.

The security labels are defined in RACF and used in a special column of the DB2 tables to define the security classification of the rows in those tables. The RACF definitions are in the SECLABEL resource class. This class is known to RACF by virtue of being defined in the RACF Class Descriptor Table, and you need to activate it after you have defined your system's security labels. Use RACLIST for the class for the best performance, which means that the profiles are held in memory.

The resource class SECLABEL must be active and have RACLIST. A subject needs at least one security label defined as its default; they can have authorization to more than one security label, but can only use one security label at a given time.

2.6 Authorizing users to access security labels

Give authorization to use a security label by permitting the user READ access to the security label profile in the resource class SECLABEL. Users can have authority to use more than one security label, but remember, as we already said, a user can only use one security label at any one time, as defined at logon time. To change the security label, the user needs to log off and log on with another security label. A user usually, but not necessarily, has a default security label defined to his or her user ID. Assign a security label to a new user when the user ID is initially defined with the ADDUSER command, or for an existing user, alter the user ID with the RACF ALTUSER command.

A security label cannot be assigned to a RACF group; only individual subjects or objects can have security labels assigned to them. However, it is important to understand that authorization to use a security label can be given by permitting a RACF group (to which the user must be connected) READ access to the security label profile in the resource class SECLABEL, that is, through the access list of that security label. See also the following tip.

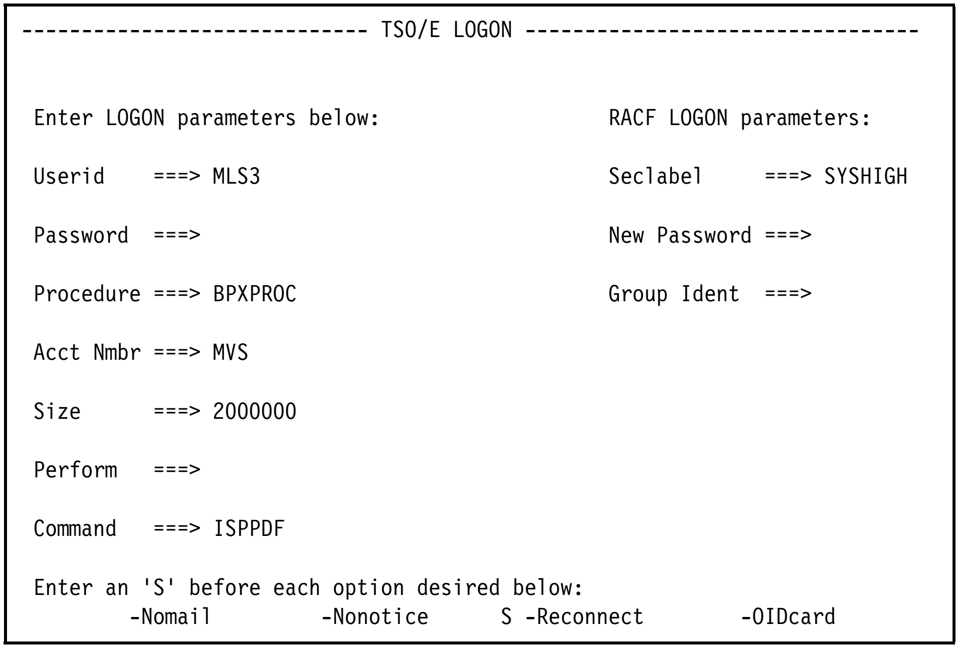
Give access to a security label by permitting the user to the security label profile in the resource class SECLABEL. A user does not necessarily have a default security label defined to that user’s ID. Do this using the RACF alter user command:

```
ALTUSER USER1 SECLABEL(XYZ)
```

When the SECLABEL class is activated, the TSO/E LOGON panel has an extra input field for specifying the security label. This shows the user’s default security label. See Figure 2-1 on page 19.

A user will be permitted access to zero or more security labels. On the logon panel, the user can specify the security label to which that user has read access. The user is governed by that security label during that session.

Tip: A useful way to administer access to SECLABELs with large numbers of users is to create a RACF group of the *same name* as the SECLABEL. Then, permit that group read access to the SECLABEL and give them read access by connecting users to the group. For numbers of users greater than 1024, consider defining the group as a universal RACF group. This use of groups keeps access lists to a manageable size and also provides an immediate indication, when listing the user, of which SECLABELs a user has access to without needing to run the RACF report utility (IRRUT100).



```
---------------------------- TSO/E LOGON --------------------------------


 Enter LOGON parameters below:                   RACF LOGON parameters:

 Userid    ===> MLS3                             Seclabel     ===> SYSHIGH

 Password  ===>                                  New Password ===>

 Procedure ===> BPXPROC                          Group Ident  ===>

 Acct Nmbr ===> MVS

 Size      ===> 2000000

 Perform   ===>

 Command   ===> ISPPDF

 Enter an 'S' before each option desired below:
         -Nomail         -Nonotice      S -Reconnect        -OIDcard
```

Figure 2-1   TSO logon panel

2.7 Using security labels

After creating and assigning security labels, the security administrator can activate the RACF SECLABEL resource class to cause the system to use the security labels for authorization checks. Then, when a user tries to access a resource, RACF checks whether the resource has a security label. If it does, RACF compares the security label of the user with that of the resource (this is mandatory access control, or MAC). If the security labels allow access, RACF then checks the access list of the profile that protects the resource (this is discretionary access control, or DAC). The decision as to whether to allow the access is based on MAC (security labels) and DAC (access lists).

To activate the SECLABEL class, the security administrator issues the command:

```
SETROPTS CLASSACT(SECLABEL) REFRESH RACLIST(SECLABEL)
```

Note that when the SECLABEL class is active, unless the security administrator has also set certain system options, a user needs a security label only if the resource the user wants to access has a security label, and resources are not required to have security labels. To increase security, the security administrator can use the SETROPTS command to set RACF system options that require that certain resources have security labels and require that any user who tries to access those resources have a security label.

2.8 Comparing security labels

When authorization checks are made to determine security label authorization, the relationship between security labels is assessed. The types of these relationships are:

- Dominance
- Equivalence
- Disjoint

One security label dominates a second security label when the following two conditions are true:

- The security level, defined in class SECDATA, that is associated with the first security label is *greater than or equal to* that of the second security label.
- The set of security categories, defined in class SECDATA, that are associated with the first security label *includes* the set of security categories associated with the second security label.

If neither security label dominates the other then the two security labels are said to be *disjoint* or *incompatible*.

The following tables provide an example of security label dominance.

Table 2-1 Security levels example

Security level	Hierarchical level
Regulatory	L3
Confidential	L2
Unclassified	L1

Table 2-2 Security categories example

Security categories
A, B, C, D, and E

Table 2-3 Security labels and their levels and categories

SECLABEL	SECLEVEL	CATEGORY
L3ABCDE	Regulatory	A, B, C, D, and E
L2BC	Confidential	B and C
L1AB	Unclassified	A and B
L1C	Unclassified	C

Here, L3ABCDE dominates L2BC because the hierarchical security level of regulatory (L3) is higher than that of confidential (L2), and L3ABCDE's non-hierarchical categories include all of L2BC's categories.

Table 2-4 Security labels showing security levels and categories

<table>
<tr><th></th><th>A</th><th>B</th><th>C</th><th>D</th><th>E</th></tr>
<tr><td>Regulatory</td><td colspan="5">SECLABEL=L3ABCDE</td></tr>
<tr><td>Confidential</td><td>No label</td><td colspan="2">SECLABEL=L2BC</td><td>No label</td><td>No label</td></tr>
<tr><td>Unclassified</td><td colspan="2">SECLABEL=L1AB</td><td>SECLABEL = L1C</td><td>No label</td><td>No label</td></tr>
</table>

In this example:

- L3ABCDE also dominates L1AB and L1C.
- L2BC dominates L1C.
- L2BC and L1AB are disjoint.

See Figure 2-2 and Figure 2-3.

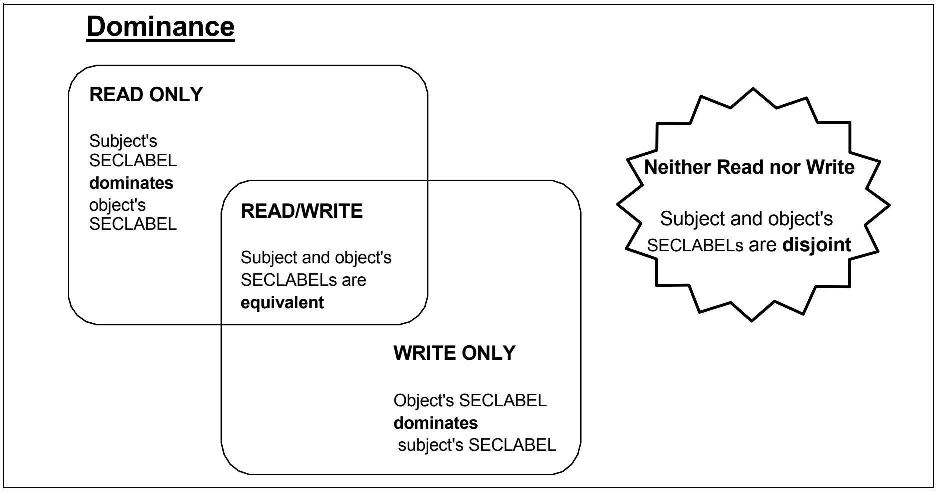

Dominance
READ ONLY
Subject's SECLABEL **dominates** object's SECLABEL
READ/WRITE
Subject and object's SECLABELs are **equivalent**
WRITE ONLY
Object's SECLABEL **dominates** subject's SECLABEL
Neither Read nor Write
Subject and object's SECLABELs are **disjoint**

Figure 2-2 SECLABEL dominance schematic (1 of 2)

This shows, in their simplest terms, the relationships between the four cases of subjects:

- Dominating the object: Read-only
- Equivalent to the object: Read/write
- Dominated by the object: Write-only
- Disjoint: None of the above

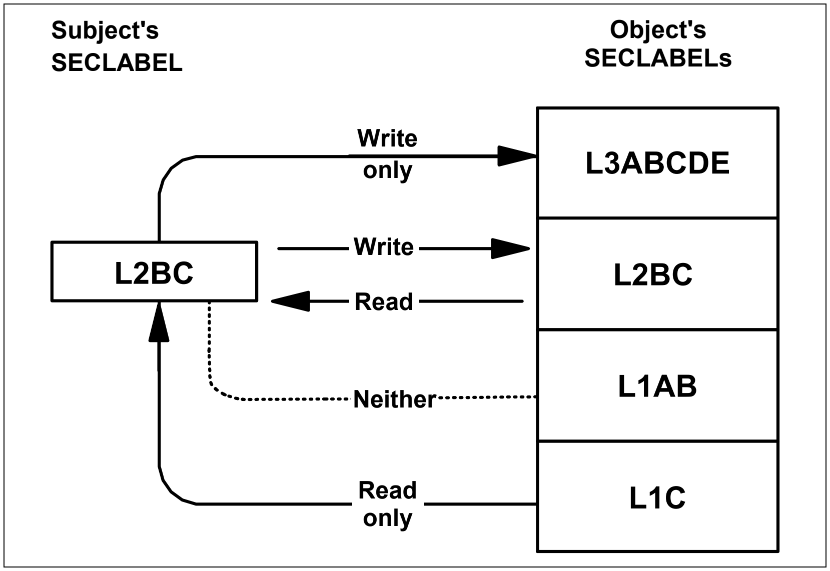


Figure 2-3 SECLABEL dominance schematic (2 of 2)

This graphically shows our example. In this example, it is important to note that the user does not have the ability to write down. In other words, MLS(FAILURES/WARNING) is not set, or MLS(FAILURES/WARNING) is set but the user does not have access to the RACF profile IRR.WRITEDOWN.BYUSER in the FACILITY class. The user has a security clearance of SENSITIVE, but only for categories B and C, and so will have access to a security label of L2BC.

2.9 Security label authorization checking

When the SECLABEL class is active on your system, and a subject requests access to a resource, RACF compares the security label of the subject with that of the resource. For a general description of these comparisons, see 2.8, "Comparing security labels" on page 20.

Table 2-5 provides an overview of the effect of the MLACTIVE settings on missing SECLABELs.

Table 2-5 Effect of MLACTIVE settings on missing SECLABELs

Environment	Missing user security label (resource security label is present)	Missing resource security label (user security label is present)	Missing both user and resource security labels
MLACTIVE(FAILURES) and resource class requires security labels.	Fail[a].	Fail.	Fail[a].
MLACTIVE(WARNING) and resource class requires security labels.	Fail.	Pass and warning message sent to security console.	Pass and warning message sent to security console.
NOMLACTIVE and resource class requires security labels.	Fail.	Pass.	Pass.
MLACTIVE(FAILURES) and resource class does not require security labels.	Fail[a].	Pass.	Pass[a].
MLACTIVE(WARNING) and resource class does not require security labels.	Fail.	Pass.	Pass.
NOMLACTIVE and resource class does not require security labels.	Fail.	Pass.	Pass.

a. In these cases, the user has a missing security label while SETROPTS MLACTIVE(FAILURES) is in effect because the user logged in without a security label before SETROPTS MLACTIVE(FAILURES) was activated. Authorization requests are passed or failed according to the entries in this table. If such a user attempts to log on to the system while SETROPTS MLACTIVE(FAILURES) was in effect, the user is not allowed to log on unless the user has access to the SYSLOW security label. Users who have access to SYSLOW at logon time when MLACTIVE(FAILURES) is active are assigned and run with SYSLOW. Security label authorization checking is performed when SECLABEL class is active and either SETROPTS MLS(FAILURES) or MLS(WARNING) is in effect

Table 2-6 provides an overview of the relationships among the SECLABEL class, SETROPTS MLS(FAILURES), SETROPTS MLACTIVE(FAILURES), and SETROPTS MLQUIET.

Table 2-6 Relationship overview

SECLABEL class	MLS (FAILURES)	MLACTIVE (FAILURES)	MLQUIET	Effect
Inactive	Off	Off	Off	Security labels have no effect on authorization checking.
Active	Off	Off	Off	RACF uses security labels and allows writing to a lower security label.
Active	On	Off	Off	RACF uses security labels and prevents writing to a lower security label (no write-down).
Active	On	On	Off	All resources must be labeled. RACF uses security labels and RACF prevents writing to a lower security label.
Active	Off	On	Off	Those resources required to have security labels by definition in the class descriptor table (CDT).
Active	Either	Either	On[a]	All attempts to access the system or resources fail (unless the attempt is made by the trusted computing base, a security administrator, or a console operator).

a. To activate SETROPTS MLQUIET, you must also enable SETROPTS MLSTABLE.

2.10 Using system-specific security labels in a sysplex

In a sysplex, it can be useful to limit the use of certain security labels to certain members of the sysplex. This allows one member of the sysplex to run work at security label A, while another handles work at security label B, keeping work separated based on security classification while still sharing the RACF database. The SETROPTS option SECLBYSYSTEM allows you to use security labels on a per-system basis.

To define system-specific security labels, the security administrator specifies on which systems a security label is to be active by adding a member list to the SECLABEL resource class profile. The member names are system SMF IDs containing 1-4 characters. For example, to define the security label named SECRET as being active only on the systems with SMF system IDs SYSA and SYSB, the security administrator can define SECRET with a command such as:

```
RDEFINE SECLABEL SECRET....ADDMEM(SYSA,SYSB)
```

If no member list of system IDs is added, the security label is considered to be active on all systems sharing the RACF database.

To activate the use of system-specific security labels, activate the SECLBYSYSTEM option and refresh the SECLABEL class:

```
SETROPTS SECLBYSYSTEM
SETROPTS RACLIST(SECLABEL) REFRESH
```

The following restrictions apply to system-specific labels:

- JES3 does not support the use of system-specific security labels. Do not activate the SECLBYSYSTEM SETROPTS option if you are using JES3.
- JES2 does not support the use of system-specific security labels for systems that perform NJE and OFFLOAD processing. These systems must have all security labels active. In addition, JES2 printers cannot process output unless the security label associated with the output is active on the system controlling the printer.
- If you define system-specific security labels, be aware that using a generic TSO system name at logon might not work, because the user cannot be allocated to a system where the user's security label is not active.
- If you use Automatic Restart Manager (ARM) to manage applications and you use system-specific security labels, ensure that the systems that you told ARM to use when restarting an application are systems that have the appropriate security label active. Otherwise, ARM might try to restart an application requiring a particular security label on a system where that security label is not active, and the application restart will fail.

2.11 Summary

Although the use of SECLABELs is required to fully implement MLS, you can see from Figure 2-2 on page 22 and Figure 2-3 on page 22 that you can implement them in phases and without the other options. This is a reasonable course of action. To accomplish DB2 row-level security, SECLABEL assignment for users is the only requirement. Like the PROTECTALL option in RACF, the most complete level of protection will come with the implementation of all of the MLS options, which we discuss in the next chapter.

3

MLS

In this chapter, we provide an overview of multilevel security (MLS) followed by a detailed description of an implementation.

3.1 MLS overview

In this chapter, we provide an overview of multilevel security. We discuss what multilevel security is and why it is used.

3.1.1 What is multilevel security?

A multilevel security system is a security environment that allows the protection of data based on both traditional discretionary access controls and controls that check the sensitivity of the data itself through mandatory access controls.

These mandatory access controls are at the heart of a multilevel security environment, which prevents unauthorized users from accessing information at a classification to which they are not authorized or changing the classification of information to which they do have access. These mandatory access controls provide a way to segregate users and their data from other users and their data regardless of the discretionary access they are given though access lists and so on.

Creating a multilevel security environment requires a combination of several software and hardware components that enforce the security requirements needed for such a system. The security-relevant portion of software and hardware components that make up this system are also known as the trusted computing base.

3.1.2 Why multilevel security?

The primary arena where multilevel security is valuable is government agencies that need a security environment that keeps information classified and compartmentalized between users. In addition to the fundamental identification and authentication of users, auditing and accountability of the actions by authenticated users on these systems is provided by the security environment.

Normally in such highly secure environments, to manage the compartmentalization of information between users, each compartment is on its own system, making it difficult for classified information to spill from one system to another, because the connections between systems can be highly controlled. With multilevel security, these systems can be consolidated onto a single system, with each compartment independent of the other, so no transfer of data can occur between compartments within that system. This takes advantage of the cost savings of not having to manage multiple systems, but instead only a few, or one system.

Commercial customers might also find some features of multilevel security useful, such as to separate sensitive customer information from the general populace, or from another user. New government regulations, such as HIPAA, and corporate mergers are examples where security of information based on the information itself is important in the commercial world.

RACF, also known as the z/OS Security Server, has several options that can be turned on and off to manipulate different aspects of a multilevel security environment. Because it is possible to have some features of multilevel security on at one time (creating a partial multilevel security environment), commercial customers might find this type of environment useful to meet these needs versus running a full fledge multilevel security environment.

3.1.3 Access controls

Discretionary access control (DAC) allows access to data to be controlled by the discretion of the owner of the data (or those with administrative authority to the data). Because access can be given out at the discretion of an authorized person to the data, it is easy to unintentionally give access to data to people who do not need access to it. For example, the owner of some data gives access to a GROUP because that group of people need access to the data. Then, the group owner connects a new user to the GROUP because the new user needs access to resources that the GROUP has access to, not knowing that the GROUP also has access to sensitive data to which this new user must not have access.

Mandatory access control (MAC) imposes additional restrictions on users, so they now access data based on a comparison of the classification of the user and the classification of the data, as well as the standard DAC checking. This additional security check verifies that users can only access data and resources that their classification allows them, even though their discretionary access would allow them to access such data or resources.

Mandatory access control was implemented in RACF 1.9 with MVS/ESA™ 3.1.3 (in 1990) as part of the effort to meet the U.S. Government criteria (TCSEC B1) as specified in the *Department of Defense Trusted Computer System Evaluation Criteria*, DoD 5200.28-STD (also know as the TCSEC or Orange Book).

In addition to checking a user’s access to data based on the classification of the user and the data, the z/OS trusted computing base implementation of multilevel security provides some of these additional functions:

- The system does not allow a storage object to be reused until it is purged of residual data.

- The system enforces accountability by requiring each user to be identified and creating audit records that associate security-relevant events with the users who cause them.
- The system labels all hardcopy with security information.
- The system optionally hides the names of data sets, files, and directories from users who do not have access to those data objects.
- The system does not allow a user to declassify data by “writing-down” (that is, writing data to a lower classification than the classification at which it was read) except with explicit authorization to do so.
- The system does not allow a user to view rows of data in a DB2 table that the user is not classified to view.

3.1.4 Introduction to mandatory access control

There are several principles you need to comprehend to fully understand the interactions of MAC checking—the given classification for a particular piece of data, a resource, or a user as defined by the security label for that item. These security labels are then used to determine what access will be allowed. For example, for a user looking at some data in a file, the subject’s (in this case, the user) security label is compared to the object’s (in this case, the file) security label. A dominance or equivalence relationship is determined, if one exists, and the type of access requested by the subject onto the object is determined. Based on that, MAC to the object is determined.

Breaking this down, you can see that there are three major principles:

- Security labels
- The type of relationship between security labels (dominance, equivalence, and disjoint)
- The type of access requested (MAC)

We discuss these principles in Chapter 2, “Security labels” on page 11, but here we take a high-level look at them all.

Security labels

A security label is a combination of a hierarchical level of classification (security level) and a set of zero or more non-hierarchical categories (security category).

In Figure 3-1, we show a basic table that has a list of hierarchical levels from 1-4, where 1 is the lowest and 4 is the highest security level, and a list of non-hierarchical categories from category A to category E. Using this table, we can quickly create several security labels combining a security level and zero or more categories. Some example security labels are:

- L1A is a combination of security level 1 with category A.
- L2BCD is a combination of security level 2 with categories B, C, and D.
- L3N is a combination of security level 3 with no categories.

> **Note:** The security labels we use for this example are defined as they are for ease of use and to allow you to determine a security label's security level and categories quickly. As an example, using Figure 3-1, you can also create the additional security labels:
>
> - ARTHUR is a combination of security level 2 with category C.
> - FORD is a combination of security level 3 with categories A, C, and E.
> - MARVIN is a combination of security level 3 with no categories. This security label is also equivalent to L3N.

Figure 3-1 has several more security labels defined, and you can easily create several more based on this simple example.

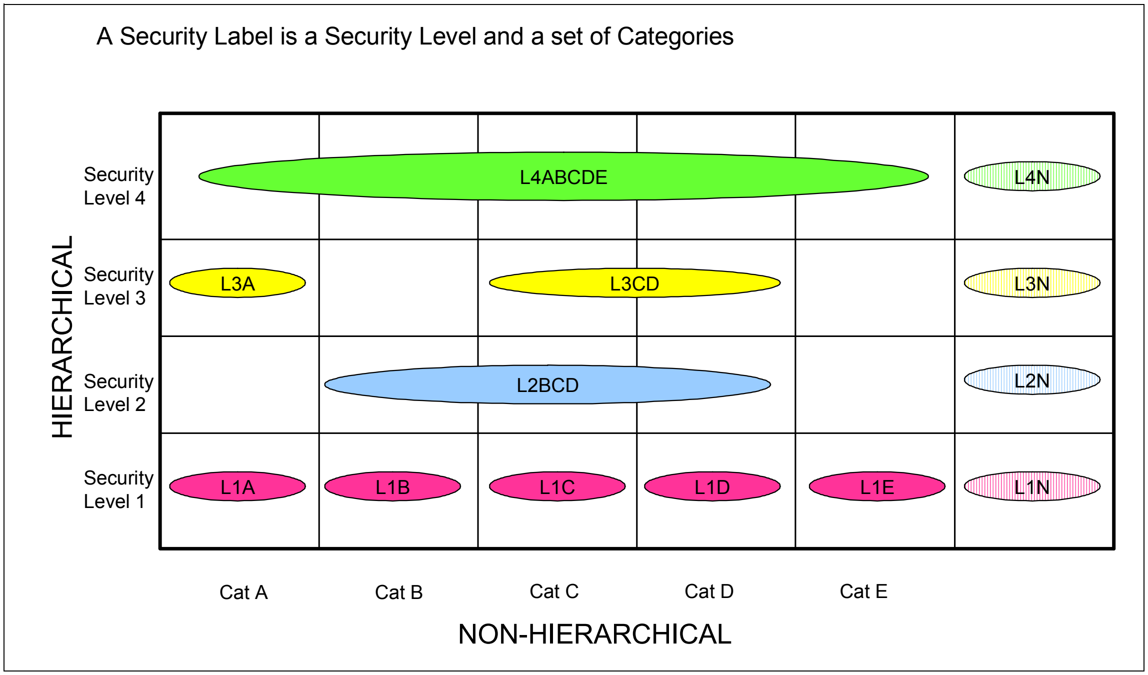


Figure 3-1 Simple security label diagram

Dominance, equivalence, and disjoint

Access is granted or rejected based on the relationship between the subject's security label and the object's security label, and the type of access that is requested. Two important relationships between security labels are dominance and equivalence. Although other relationships between two security labels can exist, in those cases, authority to the object is not allowed.

Dominance

For security label A to dominate security label B, the following must be true:

- The security level of A is greater than or equal to the security level of B.
- Security label A has at least all the categories that define security label B.

Returning to Figure 3-1 on page 31, security label L3CD, for example, dominates security labels L1N, L1C, L1D, L2N, L3N, and L3CD, because L3CD is at an equivalent or higher security level than the security labels it is dominating, and it contains all the categories that these security labels have. Notice that even though security label L2N has no categories, it is dominated by all security labels at a higher or equivalent security level and any categories.

With reverse dominance access checking, the access rules are the reverse of the access rules for dominance access checking.

Equivalence

For security label A to be equivalent to security label B, the following must be true:

- The security level of A is equal to the security level of B.
- Both security labels A and B must have the same set of categories.

Another way that two security labels are equivalent is if both security labels dominate each other.

Returning to Figure 3-1 on page 31, the only case of equivalence in the figure is if a security label is being compared to itself (that is, the subject and object security labels are the same).

Disjoint

If neither security label dominates the other, the two security labels are said to be *disjoint* or *incompatible*.

Mandatory access control

In addition to the type of relationship, as documented earlier, the type of access that is requested is also important. There are only three types of access that can be requested: read-only, read/write, and write-only.

Read-only test

The user intends to only read information. Therefore, the user needs to be able to read information at his or her classification or information at a lower classification. In terms of MAC checking, the user's security label must *dominate* the object's security label that the user intends to read. This allows the user to read information covered by that user's security label, but not information of a higher level, or outside the user's category.

Write-only test

The user intends to only write information. Therefore, the user needs be able to write information at his or her classification or information at a higher classification. In terms of MAC checking, the object's security label that the user intends to write to must *dominate* the user's security label. This allows the user to write information at his or her security label or higher, but not information of a lower level, or outside the user's category (that is, the user will not be able to declassify information).

Read/write test

The user intends to read and write information. Like the read-only case, the user's security label must be able to *dominate* the object's security label, *and*, like the write-only case, the object's security label must be able to *dominate* the user's security label. Therefore, for the user to do a read/write action to the object, the user's and object's security labels must both be *equivalent* to each other. This allows the user to read and write information only at his or her security label, but not outside his or her security label (that is, the user will not be able to declassify information).

Controlled write-down

There might be cases where you want to allow for controlled situations of write-down. The security administrator can assign a *write-down by user* privilege to individual users or groups of users that allows them to select the ability to write down. The security administrator activates and deactivates the privilege by creating the profile IRR.WRITEDOWN.BYUSER in the facility class. A user can activate write-down mode if the profile exists, the user has at least read access to it, and the FACILITY class is active and SETROPTS RACLISTed. If the user has update or higher access to the profile, write-down mode is active by default when the user enters the system. RACF provides the RACPRIV command and z/OS UNIX provides the **`writedown`** command, which allows users who are authorized to the write-down privilege to reset and query the setting of the write-down mode.

Access rules

Access rules depend on the purpose for which a subject accesses an object and whether the subject is allowed to write down.

The subject is not allowed to write down

A subject is not allowed to write down if the RACF MLS option is active and the security administrator has not set up controlled write-down and given the subject authorization to write down. This should be the case for most users in a multilevel secure environment. In this case, the access rules, depending on the purpose for which the subject accesses an object, are:

- Read-only: A subject can read an object when the subject's security label dominates the object's security label.
- Write-only: A subject can write to an object when the object's security label dominates the subject's security label.
- Read/write: A subject can read from and write to an object only if the security labels of the subject and object are equivalent.

Only the basic case is documented here—that is, a normal MAC check is being done with the "no write-down" rule in effect. With RACF, there are several other RACF options that can affect how these three different MAC tests are done, which we describe in more detail in Chapter 2, "Security labels" on page 11.

To summarize the basic MAC principles (as shown in Figure 3-2 on page 35):

- If the user's security label dominates the data's security label, the user can read the data.
- If the data's security label dominates the user's security label, the user can write to the data.
- If the user's and data's security labels are equivalent, the user can read and write to the data.
- If there is no equivalence or dominance relationship between the user's security label and data's security label, the user is denied access to the data.

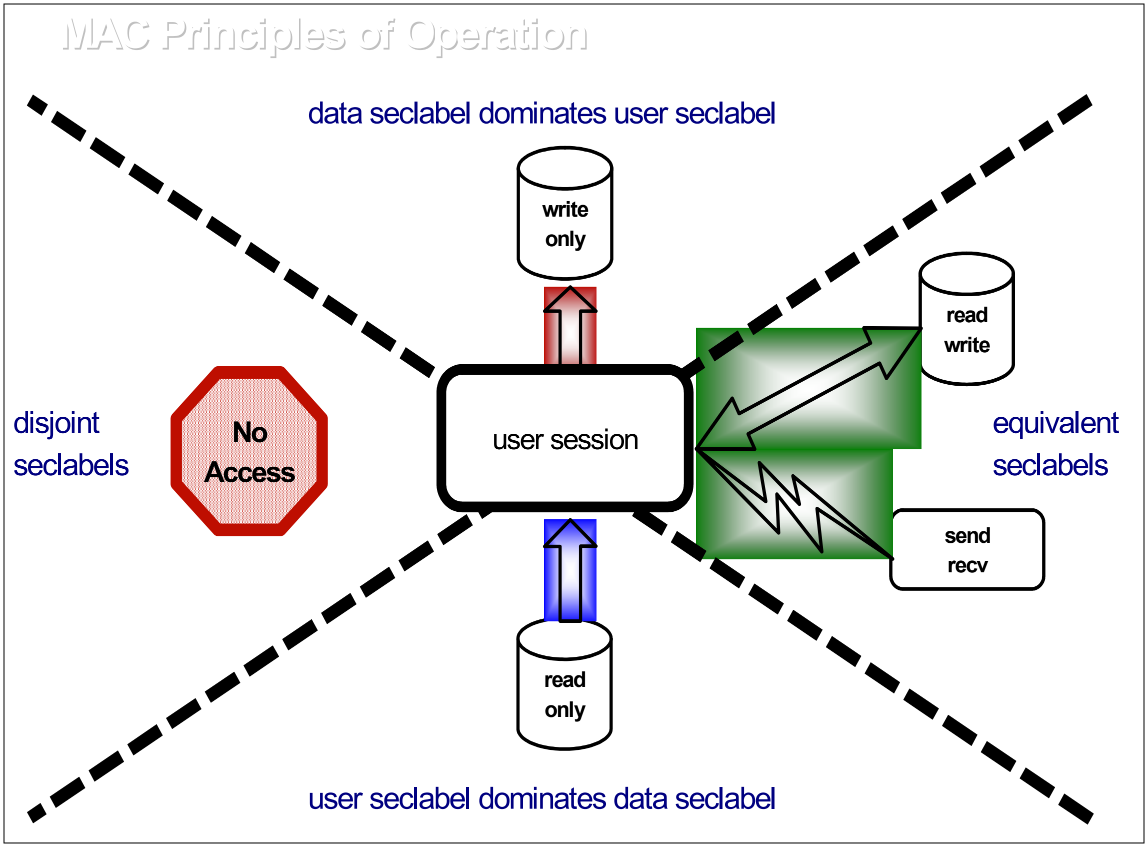

MAC Principles of Operation
data seclabel dominates user seclabel
write only
read write
disjoint seclabels
No Access
user session
equivalent seclabels
send recv
read only
user seclabel dominates data seclabel

Figure 3-2 MAC principles

3.1.5 Multilevel security in z/OS with RACF

Even though the theoretical implementation of multilevel security might seem reasonable, the actual implementation that is done by a trusted computing base might be slightly different. The trusted computing base, in this case z/OS, must make certain decisions about what defines certain actions, and then make the appropriate security decision based on the definition of that action. This is important to understand because what you might think is a certain type of MAC, the trusted computing base might have decided is a different type of MAC.

For example, in z/OS, the case of a user attempting to do a read-only or read/write action against a data set is straightforward. But because of the way z/OS handles data set processing, there is no real case, in the z/OS trusted computing base, where you can do a write-only action against a data set. This is because every time a user needs to write to a data set, z/OS must read some part of the data set before preforming the write action. As a result, the z/OS trusted computing base has decided that there is no case where a it will do a write-only test against a data set, so no such test exists.

Therefore, as you can see from this simple example, it is not only important to know how multilevel security works, but also how it is implemented by the trusted computing base.

SECLABELs

In RACF, security labels are defined in the SECLABEL class and are often called SECLABELs (versus security labels). As discussed earlier, a SECLABEL is a combination of security level (saved in the SECLEVEL profile in the SECDATA class) and a set of zero or more categories (saved in the CATEGORY profile in the SECDATA class).

Users are then granted access to any SECLABELs to which they need authority. Although a user can only have one SECLABEL active at a time for a session, at logon time, the user can request to log on with any SECLABEL for which the user has authority. Therefore, during the life of that session, the user is considered to be at the SECLABEL with which that user logged on. Resources, such as a data sets, that require a security label are given SECLABELs as well.

Now with all the SECLABELs in place and SECLABEL class activated, when a user requests access to resources (such as a user reading a data set), a MAC check is done using the user and resource SECLABELs, followed by a DAC check using the resource's access list. Note that the MAC check occurs first (after the SECLABEL class is activated), followed by the DAC check.

Multilevel security in action

Now that we have a basic idea of how multilevel security works, let's go though a simple example of multilevel security in action.

In the following example, we use the security labels defined in Figure 3-1 on page 31. We have the user MATT and provide him with the authority to the SECLABEL L4ABCDE (that is, hi has a security level of 4, and access to categories A, B, C, D, and E) and all the data sets he owns have his SECLABEL. The user ROLAND has authority to the SECLABEL L3CD, and all the data sets he owns have his SECLABEL. Finally, the user CHRIS has authority to the SECLABEL L1A, and all the data sets he owns have his SECLABEL as well. In addition, we have data sets defined to the group ITSO, and give all the data sets with its high-level qualifier a SECLABEL of L1D. Let us finally assume that all the users also have UPDATE authority to all the data sets in this example.

The user ROLAND (who will be the subject taking these actions) will be able to read and write to his own data sets (the data sets being the object receiving the actions in all these cases) because his user ID will have an equivalent SECLABEL to the data set SECLABEL of L3CD.

The user ROLAND can now only read those data sets that belong to the group ITSO, because ROLAND’s SECLABEL of L3CD dominates the ITSO’s data set SECLABEL of L1D (that is, his SECLABEL’s security level is greater than the ITSO’s security level, and his SECLABEL contains all the categories that the ITSO’s SECLABEL has).

He cannot read anything in the data sets belonging to MATT, because the security level is higher than his, and his SECLABEL has more categories than ROLAND’s SECLABEL (that is, the MAC check fails), even though his discretionary access allows him access to those data sets.

Also, ROLAND cannot access any of the data sets that belong to CHRIS. Even though ROLAND’s SECLABEL has a higher security level than CHRIS’s, the categories that ROLAND’s SECLABEL can look at are not included in the categories to which CHRIS has access.

This example is a very simple picture of multilevel security in a z/OS environment. You can also see that movement of data is very restrictive. It is also possible to implement only certain parts of multilevel security, creating a partial multilevel security environment. This reduces some of the restrictions on the movement of data, though making a less secure system (from an multilevel security point of view).

3.1.6 DB2 working with multilevel security

With the introduction of DB2 V8, DB2 actively participates in multilevel security as well. DB2 provides row-level support so that rows can be protected based on the SECLABEL of the row. This allows multiple users to access a particular table, yet allows only those users with the correct SECLABEL to read or update the particular rows to which they have access.

Additionally, with the RACF Access Control Module installed, SECLABELs can be assigned to tables or views, for example, so that only those users with the correct SECLABEL can read or update the table or view to which they have access.

3.1.7 Before turning on multilevel security

A very important part of preparing to run in a multilevel security environment, or even a partial multilevel security environment, is planning. There is no way to emphasize this more than to say that the next most important thing is planning.

To simply put it, *planning is key*.

Part of the complexity that is created by this environment is the approach that we have seen most people take toward multilevel security. Commonly, it is taking a system and breaking it down into multiple compartmental environments. Although it might be better to take an approach of imagining things as an individual compartmentalized system being incorporated into a single system, each compartment is independent of the other, such that no data movement can occur between compartments within that system, yet taking advantage of the fact that everything is on a single system so that individuals that have the security label that allows them to see information at their classification or lower can, without having to move from one compartment to another.

Another part of the complexity of establishing a working multilevel security environment (or even a partial environment) is that when turning on or off any feature of multilevel security, you are not just turning it on for an application, but for the entire MVS™ image. Even though there are several multilevel security options, these options affect the entire MVS image, not just a single application. Therefore, you not only need to consider the one application used to take advantage of multilevel security, but how that will affect the entire MVS image.

As you can see, before starting to work with a multilevel security environment (or a partial multilevel security environment), proper prior planning will help prevent problems as you start to establish this in a production environment.

3.1.8 Multilevel security vocabulary

When working with multilevel security, you need to be careful about the vocabulary. Confusion can easily arise while working in this environment, especially because there are several similar terms and terms that have different meanings in different contexts.

A very good example of this is the term MLS. One use of this term means the RACF option MLS, which activates the *no-write down* option. The term MLS has also been used as a shortcut for the term multilevel security. The intended meaning of the writer or speaker might be misunderstood.

For the purposes of this book, we use the following terms:

Multilevel security Running in an environment with all the multilevel security options turned on. This includes protecting all security-relevant resources by RACF with a security label and full no-write down protection. To be in this environment, you take all the applicable steps as documented in Chapter 3 of *Planning for Multilevel Security and the Common Criteria,* GA22-7509.

Multilevel security is also used when talking about an environment where the potential of having all the multilevel security options turned on could exist.

Partial multilevel security Running in an environment with some of the multilevel security options on.

MLS or SETR MLS This term is reserved for the RACF SETROPTS option MLS, which turns on and off the no-write down protection (also known as the *-property).

The following terms are not very confusing when seen in written text, but when spoken can easily be confused:

- SECLEVEL: A hierarchical designation for data that represents the sensitivity of the information
- SECLABEL: A name that represents the combination of a hierarchical level of classification and a set of nonhierarchical categories

3.2 Common Criteria

Standards have been set and evaluations have been (and are being) done to certify the integrity of the z/OS trusted computing base for multilevel security support. Agencies outside of IBM have set these standards and evaluated the trusted computing base, verifying the integrity of the IBM solution for multilevel security.

For the latest information, see:

http://www.ibm.com/systems/z/security/ccs_certification.html
http://www.ibm.com/security/standards/st_evaluations.shtml

> **Note:** To understand more about how to configure for Common Criteria compliance, refer to *Planning for Multilevel Security and the Common Criteria,* GA22-7509-05.

As a bit of history, the MVS 3.1.3 trusted computing base (which included RACF, JES2, JES3, TSO, VTAM®, DFP, and PSF) was formally evaluated and received the National Security Agency list of evaluated products with a B1 level of trust. Eventually, the Common Criteria and ISO 15408 superseded the older U.S. Government standards described in the Orange Book, and IBM has submitted z/OS for evaluation under the Common Criteria standard.

IBM DB2 for z/OS Version 8 is in-evaluation under the Common Criteria with a conformance claim of EAL3.

For the latest Common Criteria status, refer to

http://www.ibm.com/security/standards/st_evaluations.shtml

3.2.1 IBM System z9 EC and System z9 BC and zSeries 990 achieve prestigious EAL5 assurance certification

The IBM eServer™ zSeries 990 server received the Common Criteria Evaluation Assurance Level 5 (EAL5) certification level for the security of its logical partitions. The logical partitioning of the z990 was evaluated against the ISO 15408 Common Criteria standard, recognized in the global IT market. The z990, along with the z900 and z800, continues zSeries leadership as the only servers to have achieved this prestigious assurance level for partitioning.

This evaluation demonstrates that the zSeries server can be an essential building block for server consolidation and the integration of e-business applications and traditional corporate workloads on a single server. It provides a high degree of assurance that Processor Resource/Systems Manager™ (PR/SM™) can be configured and used in environments where separation of workloads is a requirement, but where the use of a single hardware platform is desirable for reasons of economy, flexibility, security, or management.

System z9 Enterprise Class and z9 Business Class met the certification requirements to achieve EAL5 for LPAR isolation. The logical partitioning technology on System z servers was evaluated against the Common Criteria standard ISO 15408.

3.2.2 zSeries running z/OS

z/OS V1.6 has been awarded EAL3 certification. This certification encompasses Controlled Access Protection Profile (CAPP) at EAL3 and Labeled Security Protection Profile (LSPP) at EAL3. IBM is also in evaluation for the EAL4 for z/OS V1.7 with the RACF feature.

PR/SM is designed to prevent the flow of information among logical partitions, providing highly secure isolation. This isolation allows images of zSeries-supported operating systems, including z/OS, z/VM®, and Linux® for zSeries, to run in different logical partitions on a single zSeries server. To understand the latest certification levels, refer to the following Web sites for up-to-date information:

- Common Criteria Security Certification

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

- Security Evaluations for IBM Products

  `http://www.ibm.com/security/standards/st_evaluations.shtml`

- National Security Agency and the Central Security Service

  `http://www.nsa.gov/ia/industry/niap.cfm`

3.3 Implementing MLS

In order to implement MLS for DB2, we must first implement MLS on our MVS system. This section highlights that process. The required steps are outlined in *Planning for Multilevel Security and the Common Criteria,* GA22-7509-05. For a fully compliant system from the perspective of MLS, you need to perform every applicable task in this section.

3.3.1 Background

We started with a z/OS 1.6 system and a RACF database that had been used for various demonstrations and testing. In order to have an effective book, we decided that it was necessary to fully implement MLS. To do this, we followed the book *Planning for Multilevel Security and the Common Criteria*, GA22-7509. That book describes MLS in great detail, with a small section about DB2. This book expands on the DB2 section. The DB2 examples are all based on the default tables that come with DB2, so anyone can follow the samples in an isolated LPAR if desired. In going through the book, we developed batch jobs to do the RACF definitions, and we include those jobs in Appendix B, “RACF options that control the use of security labels” on page 315. We also include the batch jobs used to do various DB2 work.

Here, we describe our efforts to implement MLS for MVS. Some of the infrastructure is required in order to exploit MLS for DB2. In order to fully protect the system, much more of the infrastructure is required. Also, the system we used was not very protected, so many of the steps suggested did not apply to our configuration.

The first thing we needed to do involved planning. After analyzing the database we were going to use, we needed to come up with our matrix of hierarchy and categories, and then decide on some security labels.

3.3.2 Defining SECLABEL names for your situation

We needed to identify all of the SECLABELs to be used in the system. The system comes with four default SECLABELs, SYSMULTI, SYSHIGH, SYSLOW, and SYSNONE, and these were to be given to certain resources in the system. We still needed to create a list of SECLABELs that would eventually represent access to various categories and levels. These labels are defined by the designers of the installation, so we based ours on the number of categories in the default tables in DB2, as described by Figure 3-3.

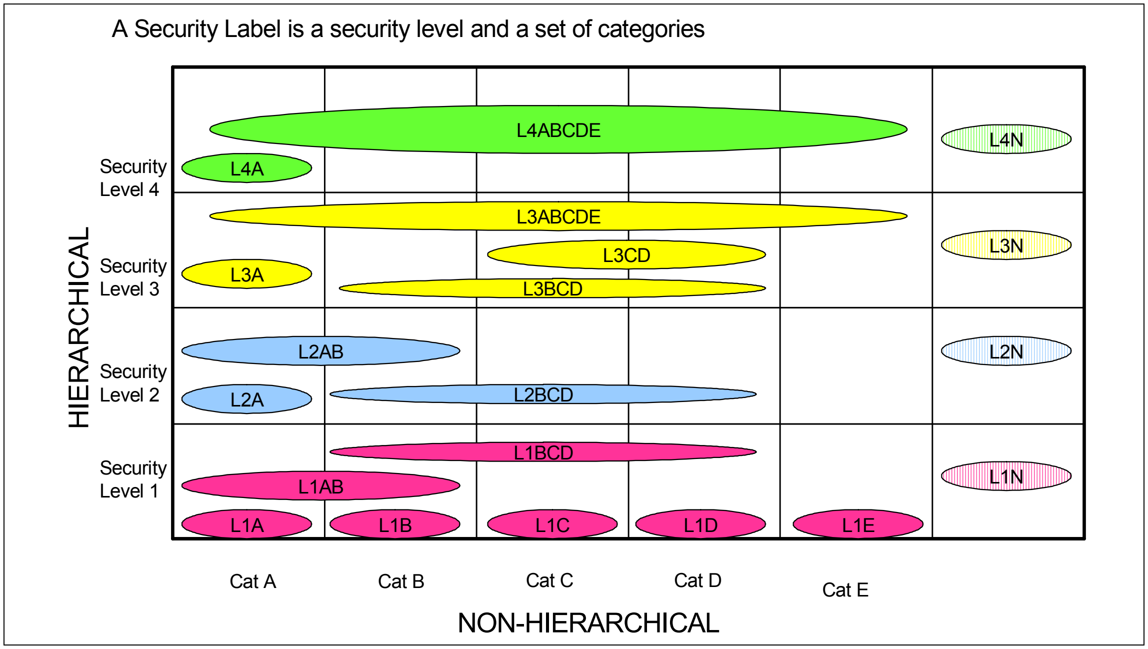


Figure 3-3 Matrix of SECLABELs used in this book

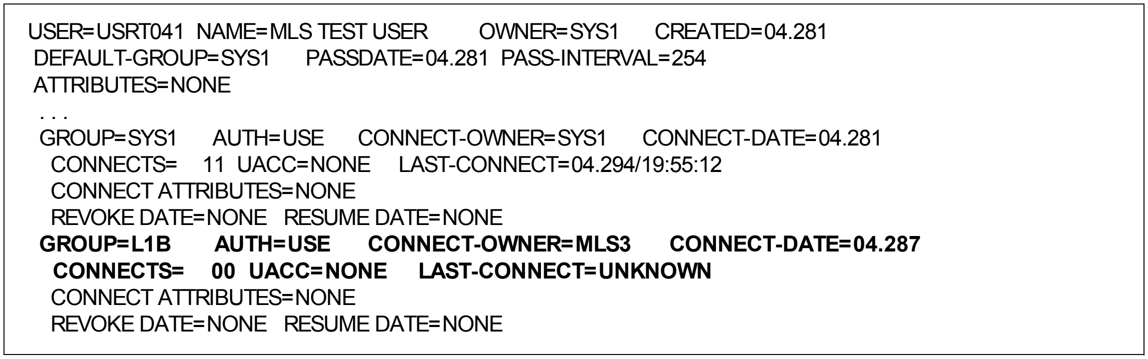
The smallest ovals along the left and lower axis of the table represent access to a single category at one security level, from the lowest to the highest level of access. For clarity, the chart does not show that there is a similar definition for every box in the chart.

The rightmost column represents a SECLEVEL that is not connected to any category. The long ovals represent a SECLABEL that describes access at one level (plus all levels below) and all the categories spanned by the oval. For example, a user ID with a SECLABEL of L3CD has access to data at the L3 level of categories C and D, plus access to those same categories at the L2 and L1 levels. For each oval, a GROUP was defined so that user IDs can be permitted access to that GROUP. For example, user ID USRT060 is allowed to use a group called L2AB. Here is the syntax of the ALTGROUP command that adds the L1B group:

```
AG L1B      SUPGROUP(SYS1) OWNER(DB2) +
DATA('FOR ACCESS TO SECLABEL OF SAME NAME AS THIS GROUP')
```

Although it would be difficult to show in the chart in the previous figure, there is no reason that a SECLABEL consisting of nonadjacent categories could not be defined.

We chose to build a group for each security label because it is easier to identify what access levels an individual user can use. In our case, there are few times where the benefit of grouping is realized, but it is a good practice to use groups. When you list a user ID, you get good documentation.



```
USER=USRT041  NAME=MLS TEST USER       OWNER=SYS1     CREATED=04.281
 DEFAULT-GROUP=SYS1     PASSDATE=04.281  PASS-INTERVAL=254
 ATTRIBUTES=NONE
  . . .
  GROUP=SYS1     AUTH=USE     CONNECT-OWNER=SYS1     CONNECT-DATE=04.281
    CONNECTS=    11  UACC=NONE     LAST-CONNECT=04.294/19:55:12
    CONNECT ATTRIBUTES=NONE
    REVOKE DATE=NONE   RESUME DATE=NONE
  GROUP=L1B      AUTH=USE     CONNECT-OWNER=MLS3     CONNECT-DATE=04.287
    CONNECTS=    00  UACC=NONE     LAST-CONNECT=UNKNOWN
    CONNECT ATTRIBUTES=NONE
    REVOKE DATE=NONE   RESUME DATE=NONE
```

Figure 3-4   Example of self-documenting using groups

It is very easy to tell how many SECLABELs this user can access. The administrator always CONNECTs users to groups (previously permitted to SECLABELs) instead of using PERMIT to permit users to SECLABELs. If the administrator used PERMIT to the SECLABEL, and wanted to know which SECLABELs this user can use, the administrator has to display all of the SECLABELs and review the list of users permitted to connect or run a RACF utility.

3.3.3 Defining resource names to RACF

Now that we determined what labels we need, we are ready to define them to RACF. Perform the following steps:

1. Define the SECLEVEL profile in the SECDATA class. For example:

   ```
   RDEFINE SECDATA SECLEVEL UACC(NONE)
   ```

2. Define your security levels as members of the SECLEVEL profile. For example, to define the security levels L1, L2, L3, and L4:

   ```
   RALTER SECDATA SECLEVEL ADDMEM(L1/10, L2/20, L3/30, L4/40)
   ```

3. Define the CATEGORY profile to the SECDATA class. For example:

   ```
   RDEFINE SECDATA CATEGORY UACC(NONE)
   ```

4. Define your security categories as members of the CATEGORY profile. For example, to define the categories CatA, CatB, CatC, CatD, and CatE:

   ```
   RALTER SECDATA CATEGORY ADDMEM(CATA, CATB, CATC, CATD, CATE)
   ```

5. Define your security labels. Use the RACF RDEFINE command to specify the SECLABEL class profiles. The profile names are the security labels and are limited to eight characters. For example, to define the security labels L4ABCDE and L3BCD:

   ```
   RDEFINE SECLABEL L4ABCDE SECLEVEL(L4)
   ADDCATEGORY(CATA,CATB,CATC,CATD,CATE) UACC(NONE)
   RDEFINE SECLABEL L3BCD SECLEVEL(L3) ADDCATEGORY(CATB,CATC,CATD)
   UACC(NONE)
   ```

In the examples in this section, we use the matrix in Figure 3-3 on page 42 as the basis of the security labels. We assigned each of our user IDs the following SECLABELs (Table 3-1).

Table 3-1 Matrix of security labels

	Cat A	Cat B	Cat C	Cat D	Cat E	
USRT041		L1				
USRT044						SYSLOW
USRT045	L4	L4	L4	L4	L4	
USRT051						SYSHIGH
USRT052	L3	L3	L3	L3	L3	
USRT053		L3	L3	L3		
USRT055		L3	L3	L3		

	Cat A	Cat B	Cat C	Cat D	Cat E	
USRT056						L3
USRT057						L3
USRT060	L2	L2				
USRT066		L1	L1	L1		
USRT067		L1	L1	L1		
USRT068	L1	L1				
USRT069	L1	L1				
USRT070			L2			

Each task that runs in the system also has to run with a user ID. The system level task's user IDs (JES, SMF, VTAM, XCF, and so on) were assigned a SECLABEL, as suggested by the *Planning for Multilevel Security and the Common Criteria,* GA22-7509 book.

The next step is to update the user IDs in RACF. To assign the SECLABEL to the individual users, we alter the RACF-defined user ID. For example:

```
ALTUSER USRT070 SECLABEL(L2C)
```

3.3.4 Defining the attributes of resources

Data sets also need security labels. When MLACTIVE(FAILURES) is active, if a data set is not protected by a profile, or if the profile that protects a data set does not have a security label assigned to it, every attempt to access the data set fails. Therefore, you need to ensure that every data set is protected by a profile in the DATASET class, and that every profile in the DATASET class has a security label, before you activate MLACTIVE(FAILURES).

To find all the profiles with (and without) SECLABELs, issue this SEARCH command:

```
SR CLASS(DATASET) NOMASK CLIST('LD DA(' ') ALL GEN') GEN
```

The command produces a data set in the form of an executable CLIST. The default name is *tsohlq*.EXEC.RACF.CLIST. Execute the output (assuming PREFIX ON to pick up your HLQ):

```
EXEC EXEC.RACF.CLIST
```

You will be presented with a long list of profiles. The ones that indicate NO SECLABEL need to be updated or those data sets will not be available to the system or anyone who is using a SECLABEL. You need to do this twice, the second time for DISCRETE profiles (substitute NONGENERIC for GEN in the previous search command). We assigned most all of the profiles to SYSLOW for our testing, but you might want to be more careful with your choices.

Next, we gave the system resources security labels. Table 3-2 identifies the resource classes that, if not assigned a security label, will fail when MLACTIVE is in FAILURES mode. In our system, many of the classes did not have profiles. So if the class did not have any profiles, we added them first. Then, all profiles in these classes were given a SECLABEL.

Table 3-2 Resource classes that need a security label

APPCPORT APPCSERV APPCTP APPL DATASET DEVICES DIRECTRY[a] DSNADM DSNR	FILE[a] GDSNBP GDSNCL GDSNDB GDSNJR GDSNPN GDSNSC GDSNSG GDSNSM	GDSNSP GDSNTB GDSNTS GDSNUF MDSNBP MDSNCL MDSNDB MDSNJR MDSNPN	MDSNSC MDSNSG MDSNSM MDSNSP MDSNTB MDSNTS MDSNUF SERVER SERVAUTH	TAPEVOL TERMINAL VMMAC[a] VMMDISK[a] VMSEGMT[a] WRITER

a. These classes are specific to VM.

In addition, we added the following SECLABELs to the following classes:

```
RDEF  TSOPROC   *      UACC(READ)  SECLABEL(SYSNONE)
RDEF  ACCTNUM   *      UACC(READ)  SECLABEL(SYSNONE)
RDEF  OPERCMDS  *      UACC(READ)  AUDIT(ALL) SECLABEL(SYSNONE)
RDEF  CONSOLE   *      UACC(READ)  SECLABEL(SYSNONE)
RDEF  TSOAUTH   *      UACC(READ)  SECLABEL(SYSNONE)
RDEF  TSOAUTH   TESTAUTH   UACC(READ) SECLABEL(SYSNONE)
```

In addition to assigning security labels to profiles in the DATASET class, there are other steps you can take to both protect your system as well as implement MLS:

- Ensure that user data is erased from the volume after is has been deleted.
- Protect access to volumes by restricting profiles in DASDVOL.
- Take steps to protect data on tape.
- Protect temporary data sets.
- Protect catalogs.

We did not do any of these things in our test. However, for a fully compliant MLS system, you need to take these steps. On a production system, these things are probably already protected. We chose to skip these steps because they do not affect the DB2 MLS discussion. We also did not implement the console logon requirement for the same reason.

3.3.5 Notes from the MLS book

The *Planning for Multilevel Security and the Common Criteria*, GA22-7509 book describes the next steps in the process of setting up the system software for MLS. It describes things to consider for the components of MVS that support MLS. Each topic requires in-depth knowledge of the component and how it is used in your environment. At the end of each section, there is a checklist of the tasks to consider. The book describes the considerations for each of the following functional areas: DFS™, SMS, JES, MVS, PSF, RACF, RMF™, SDSF, TCP/IP, TSO/E, VTAM, and finally, z/OS UNIX System Services. If applicable, a list of restrictions for the task is included.

We had little to consider from these sections. Our system did not have DFS installed; our SMS environment was minimal; and the JES protections would be needed in a real system, but did not affect the DB2 testing. For MVS, we updated the SCHEDxx member to remove the NOPASS option. We made no changes to RACF, RMF, or SDSF. There is a chapter in this book about TCPIP (Chapter 5, “MLS as applied to TCP/IP communications” on page 85), but for the initial work we did not implement any of the recommendations. For TSO/E, we added the profiles mentioned earlier, and no changes were needed for VTAM. The only changes to UNIX System Services was the introduction of the zFS and the labeling of files for one user who had the ability to log on at more than one SECLEVEL. This was a test only, and not part of the DB2 work. We discuss the UNIX System Services changes later.

System tasks that we did customize

For dump services, we had two methods of securing dumps. The first is with a data set rule, and the second is to control how much data can be dumped by dump services. We chose the second. First, we defined a FACILITY class, and then allowed access to the group that is in charge of creating dumps:

```
RDEFINE FACILITY IEAABD.DMPAKEY UACC(NONE)
PE IEAABD.DMPKEY CLASS(FACILITY) ACCESS(READ) ID(system_access_group)
```

The group with SYSPROG type access was granted READ.

z/OS Unix System Services

Because DB2 did not require any customization of UNIX System Services, we did not perform any of the steps in this section. For discussion purposes only, here is a list of some items to consider if your users use UNIX System Services.

HFS does not fully support SECLABELs, so plan a migration to zFS. We recommend that you refer to *z/OS Distributed File Service zSeries File System Implementation z/OS V1R7*, SG24-6580. Chapters 2 and 3 are very helpful if you do not have any experience with the new file system. Also, after giving a file a SECLABEL, that SECLABEL cannot be changed. An extension of this concept is that each user will need a zFS at each SECLABEL for which that user can use. There is a discussion of how to use the automount function and system symbols to facilitate mounting the different file systems. Also important is the concept that any aggregate (that is, the MVS data set that contains a UNIX file system) needs a matching SECLABEL to the mount point to which it is mounted. Therefore, the mount points for individuals' separate zFS (one for each SECLABEL) match the SECLABEL of the MVS data set being mounted by the automounter. In addition, there is a hierarchy of how SECLABELs are assigned to aggregates and the files they contain under various situations, as shown in Example 3-1.

Example 3-1 Hierarchy of how SECLABELs are assigned

```
Security labels for zFS & their contents

If SECLABEL class is active, and a profile exists, and the profile
defines a SECLABEL
   Then AGGREGATE gets a SECLABEL
Then, when
FILE SYSTEMS ARE CREATED WITHIN THE AGGREGATE
   The root of the file system in the aggregate gets the SECLABEL of
the aggregate


If SECLABEL inactive or no profile exists or profile does not define
SECLABEL
   Then AGGREGATE doesn't get a SECLABEL
Then, when
FILE SYSTEMS ARE CREATED WITHIN THE AGGREGATE
   If MLFSOBJ is active, the user's SECLABEL is assigned
   If MLFSOBJ is inactive, no security label is assigned
```

In the event that no security labels are assigned to files within an aggregate, the **chlabel** shell command can be used to assign a label, but this command cannot change an existing label.

Other steps for customizing UNIX System Services for an MLS environment include disabling **cron** for general use, because it does not check security labels, and using DFSMSdss to perform all backups, because **tar** and **pax** also do not support security labels.

System tasks that we did not customize

In this section, we review system tasks that we did not customize.

Notes about the DFSMS checklist in the "Planning for Multilevel Security and the Common Criteria," GA22-7509 book

Protect SMS. In Chapter 14 of the *DFSMS/MVS DFSMSdfp Storage Administration Reference*, SC26-4920, there is a discussion about the steps required. We defined a SECLABEL for the control data sets. Use the PROGRAM class to protect ISMF functions, along with STGADMIN FACILITY class command and keyword profiles.

Do not use the object access method (OAM) to access to OAM objects, as described in *z/OS DFSMS Object Access Method Planning, Installation, and Storage Administration Guide for Object Support*, SC35-0426. Note that OAM functions in support of system-managed or automated manual tape libraries do support an MLS configuration, because the OAM support is at the tape volume level instead of the data set level. See *z/OS DFSMS Object Access Method Planning, Installation, and Storage Administration Guide for Tape Libraries*, SC35-0427, for more information.

For DFSMSdss™, protect the administrator functions from being used. See Chapter 5 of *z/OS DFSMSdss Storage Administration Guide*, SC35-0423, for a complete discussion of the functions and corresponding RACF profile names available.

The name-hiding function, MLNAMES(ACTIVE), restricts users to seeing data set names for which their SECLABEL dominates. This is accomplished by defining a facility class:

```
RDEFINE FACILITY STGADMIN.IFG.READVTOC.volser UACC(NONE)
```

And then, if needed, give the user read access, which will allow that user to see the entire data set list:

```
PERMIT STGADMIN.IFG.READVTOC.volser CLASS(FACILITY) ID(userid)
ACCESS(READ)
```

Be aware that if a user knows the exact name of a data set for which that user does not have access, and that user uses the fully qualified name in a LISTCAT ENTRY command or ISPF 3.4 panel, the name-hiding function will not block the listing or displaying of the data set name.

Configure DFSMSrmm™ to erase tapes prior to returning them to the scratch pool. To do this, add the following entry into the EDGRMMxx PARMLIB member:

```
SECCLS NUMBER(30) NAME(CC) DESCRIPTION(‘CONF’) MASK(‘**’) SMF(N)
MESSAGE(N) ERASE(Y)
```

See *z/OS DFSMShsm Implementation and Customization Guide*, SC35-0418, for more details. Also set the DFSMSrmm parmlib option TPRACF to AUTOMATIC or PREDEFINED so that rmm ensures that all tape volumes have a profile in the RACF TAPEVOL class.

Protect tape labels from being changed by defining a FACILITY class called STGADMIN.EDG.LABEL.* and STGADMIN.EDG.NOLABEL.* with a UACC of none. Also use EXPDTCHECK(Y) of the VLPOOL parmlib command to prevent expiration dates in tape labels being overridden. Finally, set up your MVS automation to reply to the following WTORs with a "M" to unload from the following message numbers: IEC507D, IEC354D, and IEC704A.

Some notes from the JES2 checklist

Assign a SECLABEL of SYSMULTI to the user ID under which JES2 runs.

Protect JES commands. Chapter 7 of the *z/OS JES2 Initialization and Tuning Guide*, SA22-7532, discusses security considerations, including MLS, in detail. The specific checklist for a MLS-compliant system is discussed in Chapter 3 of *Planning for Multilevel Security and Common Criteria*, GA22-7509-05. These steps are a combination of administrative rules and RACF definitions. Note the following items:

- Protect the JESNEWS data set with READ in the JESSPOOL resource class.
- Assign SECLABEL SYSLOW to the job that creates JESNEWS.
- Protect the SYSLOG with a profile in the JESSPOOL class.
- Use profiles in the DATASET or TAPEVOL classes to protect the system data sets.
- Protect all NJE and RJE input sources with an appropriate SECLABEL defined in each profile of the RACF WRITER class.
- Ensure that only PSF printers manage output with SECLABELs.
- Remove any modifications made to the JES source code, or any installation routines added to your JES library.
- Specify AUTH=(DEVICE=NO,JOB=NO,SYSTEM=NO) on the RDRnnn and INITRDR initialization statements to prevent system-level commands from being entered through job streams.
- If you want to restrict jobs to running on certain systems based on security classification, set up system-specific security labels.

JES3 requires very similar actions. See the *Planning for Multilevel Security and Common Criteria*, GA22-7509-05 book for the list.

MVS notes

We discussed the MVS changes we made, but here we mention some of the recommendations from the MVS checklist.

As mentioned before, we did not define console logon or protect EMCS or subsystem consoles.

We have a OPERCMDS * profile, with a UACC of read. Obviously, a production system would be more restricted.

Protect LLA with a generic profile in the FACILITY class and permit the operators access to the profile so that they can recycle LLA:

```
RDEFINE FACILITY CSVLLA.* UACC(NONE)
PERMIT CSVLLA.* CLASS(FACILITY) ID(userid or groupname) ACCESS(UPDATE)
```

Create DATASET profiles for the system data sets.

> **Note:** If you install a ServerPac and you have a chance to name all of the system data sets, you might want to consider a naming convention that allows you to group-like level data sets for the purpose of simplified profiling.

You can define data sets accessed by all users (such as SYS1.LINKLIB) with a SECLABEL of SYSLOW and a UACC of read. For other data sets (such as SYS1.PARMLIB), choose the same SECLABEL, but a UACC of NONE, and grant specific users access. For any data set that has the possibility of containing data from any SECLABEL, assign a SECLABEL of SYSHIGH. The *Planning for Multilevel Security and Common Criteria*, GA22-7509-05 book cites a list of such data sets:

- Log data sets
- Dump data sets
- Trace data sets
- SMF data sets
- Page and Swap data sets
- Spool data sets
- Dump Analysis and Elimination (DAE) data sets
- Spool Offload data sets
- JES checkpoint data sets
- PSF security libraries (overlay, font, page segment, security definitions)
- XCF couple data sets
- SMS configuration data sets

The catalog must have a SECLABEL of SYSNONE.

Protect the APF list, its data sets, and related commands with FACILITY class profiles:

```
CSVAPF.*
CSVAPF.MVS.SETPROG.FORMAT.STATIC
CSVAPF.MVS.SETPROG.FORMAT.DYNAMIC
```

Add seven FACILITY class profiles to protect the dynamic exits facility:

```
CSVDYNEX.*.DEFINE
CSVDYNEX.*.*
CSVDYNEX.*.UNDEFINE
CSVDYNEX.*.ATTRIB
CSVDYNEX.LIST
CSVDYNEX.*.CALL
CSVDYNEX.*.RECOVER
```

Protect global resource serialization (GRS) services.

Restrict GQSCAN with a FACILITY profile, and grant READ access to any unauthorized (not key 0) users:

```
RDEFINE FACILITY ISG.QSCANSERVICES.AUTHORIZATION UACC(NONE)
```

For JCL, make sure that all JOB cards have (or inherit) a user ID entry. Jobs with the SECLABEL keyword run under that label; otherwise, the job inherits the SECLABEL of the user. The following sample JOB card illustrates both requirements:

```
//TESTER1 JOB (SYS10000),USER=TESTER1,
//        CLASS=A,MSGCLASS=T,MSGLEVEL=(1,1),REGION=4096K,
//        SECLABEL=TOPSEC
```

There is a statement to remove all installation-written exit routines or modifications that you added to your system.

Prevent the use of any APPC/MVS servers on the system, and prevent the use of multitrans APPC/MVS transaction programs (TPs):

```
RDEFINE APPCSERV * UACC(NONE)
SETROPTS CLASSACT(APPCSERV) RACLIST(APPCSERV)
RDEFINE FACILITY APPCMVS.TP.MULTI UACC(NONE)
```

PSF

Only PSF provides the capability to print security information about all hardcopy output and provide an auditing mechanism designed to capture any attempt to circumvent the printing of the security information. Assuming you already have PSF installed, the following list notes the things you might need to alter in order to take advantage of the capabilities of PSF in an MLS environment:

- Set up PSF print labeling:
  - Create the security libraries.
  - Install the PSF exit routines APSUX01S and APSUX02S.
  - Modify the PSF startup procedure to reference the security libraries and to define print labeling.
  - Authorize users allowed to override print labeling.
  - Authorize operators allowed to override (with JES commands) the printing of separator pages.
  - Activate the RACF PSFMPL resource class.
- Set audit options for PSF.
- Disable the direct printing subsystem (DPSS) if it is defined.

See the *Print Services Facility for z/OS Security Guide*, S544-3291, for more information.

RACF considerations

If you implement multilevel security in a sysplex without sysplex communications enabled, some SETROPTS commands are not automatically propagated to the other systems in the sysplex. You need to know when this happens and then issue the command on all members of the sysplex. This causes excessive ENF signaling, and until the commands are reissued, the in-storage security label information is out of sync. We suggest using sysplex communications to avoid this scenario.

Note the following restrictions. Define entries in the global access checking table for only those resources that do not require security label checking or access list checking, do not require any audit records, or both. The only entries in this table must have a maximum SECLABEL of SYSLOW with a UACC of READ. Another restriction has to do with IEC.TAPERING. The suggestion is to not define this FACILITY class profile. If you use the IEC.TAPERING support to allow users to read from tapes that are enabled for writing (when the users are only authorized to read), the system software cannot prevent knowledgeable users from also writing on any files on the tapes.

RMF considerations

If name hiding is implemented, it is necessary to restrict access to Monitor II type 79 reports and Monitor III set-of-samples, because they might contain data set names, and access checking is not performed by RACF or RMF to determine if the user is authorized to see the names. Access can be limited by issuing these commands:

```
RDEFINE FACILITY ERBSDS.* UACC(NONE)
PERMIT ERBSDS.* CLASS(FACILITY) ID(userid) ACCESS(READ)
SETROPTS CLASSACT(FACILITY) RACLIST(FACILITY)
```

Also, no authorization is performed for the resource ERBSDS.MON3DATA when data is accessed through the DDS LDAP interface. To prevent unauthorized access to RMF performance data, do not specify the RMF LDAP back end ERB6LBCK in the LDAP server configuration file. Leaving this out of the configuration file disables RMF LDAP requests in general.

SDSF considerations

Use of ISFPARMS does not support MLS, so if you use it, you need to migrate to using SAF authentication.

TSO considerations

The process to ensure that TSO is completely secure of the possibility of any kind of spillage or crossover between SECLABELs results in a lengthy checklist of administrative tasks and RACF definitions. Consider the following items:

- Define TSO/E users in the RACF database instead of using SYS1.UADS.
- Every user must get a SECLABEL at logon. In line-mode, if the user does not specify a security label, the default for the terminal is used. If it is not available, the user’s default is used. In full-screen mode, the user’s label is displayed from the TSO segment of the RACF profile. If the user has access to more than one label, and chooses to overtype that other label, the other label is stored in the TSO segment for use at the next full screen logon.
- All logon attempts must be audited. MVS writes a type 30 record for successful attempts, and RACF writes a type 80 record for failed attempts.
- Protect user messages by using user-specific log data sets instead of SYS1.BROADCAST. *z/OS TSO/E Customization*, SA22-7783, has a section in Chapter 34 called “Storing messages in separate user logs” that describes this process. It is not a new process, and it is not MLS specific, but it provides the MLS support to allow for mandatory checking so that the receiver of a message always dominates the SECLABEL of the sender. The net of it is to make the following changes.

Update SYS1.PARMLIB(IKJTSOxx). From the sample provided in SYS1.SAMPLIB:

```
SEND                      /* SEND COMMAND DEFAULTS */      +
   OPERSEND(ON)             /*                    */      +
   USERSEND(ON)             /*                    */      +
   SAVE(ON)                 /*                    */      +
   CHKBROD(OFF)             /*                    */      +
   LOGNAME(*)               /*                    */      +
   USEBROD(ON)              /*                    */      +
   MSGPROTECT(OFF)          /*                    */      +
   SYSPLEXSHR(OFF)          /*                    */      +
   BROADCAST(DATASET(SYS1.BRODCAST)                         +
             TIMEOUT(5) PROMPT)
```

Make the following changes:

```
LOGNAME(LOGNAME)         /* HLQ OF LOG DATASET  */      +
USEBROD(OFF)             /* NO BRODCAST FOR MSG */      +
MSGPROTECT(ON)           /* PROTECT LOG DATASET */      +
```

Make the following definitions in RACF:

```
AG LOGNAME SUPGROUP(SYS1) OWNER(SYS1)
ADDSD  'LOGNAME.**'   UACC(READ) SECLABEL(SYSHIGH)
RDEFINE SMESSAGE ** UACC(READ)
SETROPTS RACLIST(SMESSAGE) REFRESH
```

And activate the DIRAUTH class (if not already active).

Create a log data set to receive messages by issuing the LISTBC command. After you have a log data set, if DAC and MAC allow you to send or receive the message, your log data set holds the message until needed.

- Control access to spool data sets through TSO/E commands. For output, use commands such as:

```
SETR CLASSACT(JESSPOOL)
RDEF JESSPOOL *.*.*.** OWNER(SYS1) UACC(NONE)
PE *.*.*.**  CLASS(JESSPOOL) ID(USRT053 USRT068 USRT069) ACC(READ)
SETROPTS RACLIST(JESSPOOL)
```

For TRANSMIT and RECEIVE, MLS is supported. A recipient's security label must dominate the security label of the sender, and the transmitted data sets always have the security label of the sender. The only setup considerations relate to two data sets, the log data set and the NAMES data set:

  - Define the log data set (*user*.LOG.MISC) at the user's most common security label, and if the user needs to transmit or receive at a different level, the user needs to specify a different log data set with the LOGDATASET keyword on the TRANSMIT or RECEIVE command.

  - Define the NAMES data set (*user*.NAMES.TEXT) with a security label that is the lowest to which the user has access.

- Control who can submit and cancel jobs with the JESJOBS resource class. By defining profiles in this class, the security administrator can control what job names can be started or cancelled through TSO. By default, users can cancel only the jobs that they submit, but can be granted access to other job names that they will then be able to cancel.
- The default TSO exit IKJEFF53 is not sufficient for JESSPOOL and JESJOBS classes. Use the one in SYS1.SAMPLIB instead.
- Carefully set up generic TSO system names so that there is not an opportunity for a user with a security label to be logged on to a system that does not have the user's security label active.
- The TSO/E Information Center Facility is a set of panels that enable you to use certain services and products. The services and products that you can use are displayed on a main menu panel similar to ISPF. It does not support security requirements for multilevel security.

VTAM setup

There are two aspects of VTAM that need definitions in an MLS system, application authorization and auditing:

- Define a profile in the VTAMAPPL resource class (which must be active in a MLS-compliant system) with a UACC of none:

  ```
  SETROPTS CLASSACT(VTAMAPPL))
  RDEFINE VTAMAPPL * UACC(NONE)
  ```

- Require auditing with the following command:

  ```
  SETROPTS LOGOPTIONS(ALWAYS(VTAMAPPL))
  ```

z/OS UNIX System Services

z/OS UNIX System Services needs several modifications and administrative procedures to function correctly in an MLS environment:

- The following IDs must use the SECLABEL SYSMULTI: The OMVS user ID, the user ID associated with the zFS address space, and the user ID associated with the BPXAS procedure.
- Convert any HFS that needs to have security labels to zFS. HFS supports security labels only in read-only mode.

- For any user authorized to log on at more than one SECLABEL, that user needs to have a home directory and default program at each SECLABEL. The automounter and the use of symbols can make this process easier to administer. Define mount points in the /u directory for each SECLABEL defined in your system, including SYSHIGH and SYSLOW, and have the automounter mount the user's file system at those points.
- Disable **`cron`** for general users, because it does not support security labels.
- Change your installation's procedures to use DFSMSdss to create backups and perform restores of any zFS or HFS instead of **`pax`** or **`tar`**. These two utilities do not support security labels.
- The UUCP (UNIX-to-UNIX copy program) group of commands does not support multilevel security.
- Do not define the profile BPX.SAFFASTPATH in the FACILITY class, because it bypasses RACF in some instances for performance reasons, and you must not bypass RACF in a multilevel system.

4

Vanguard solution

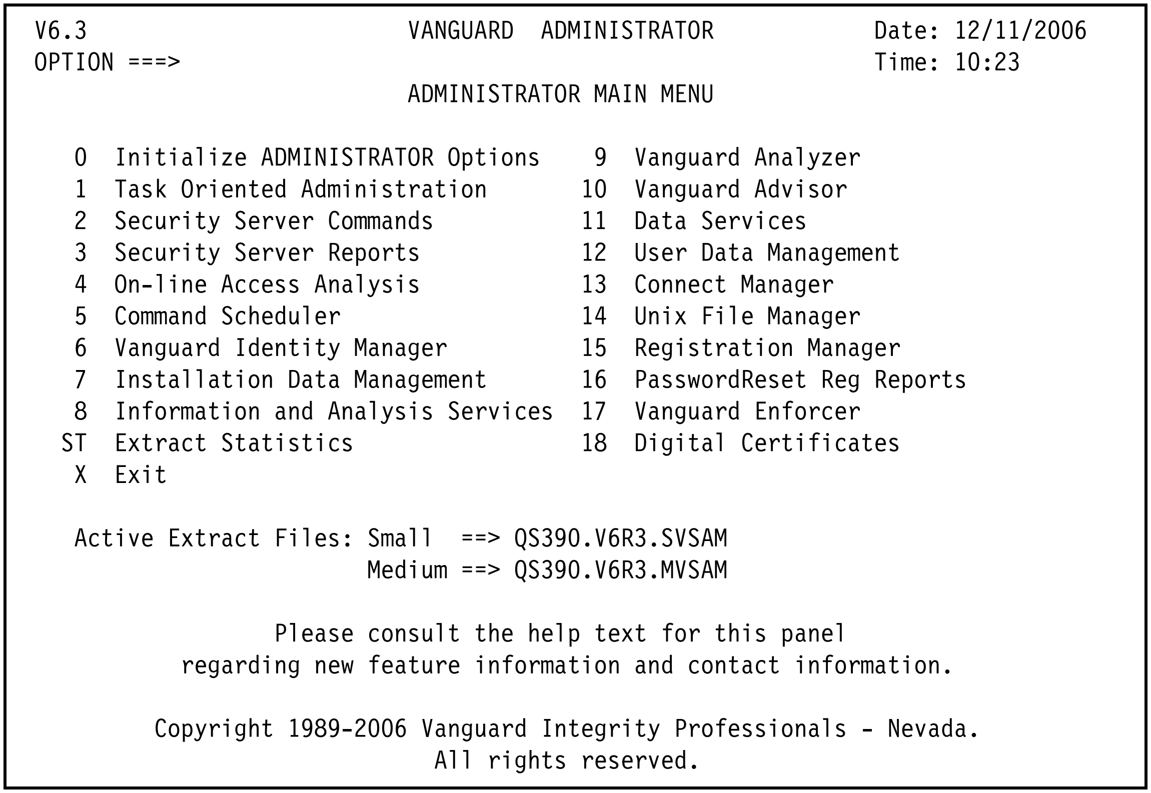
In this chapter, we discuss how you can use Vanguard Administrator to ease the the migration of your RACF environment from non-multilevel access controls to a fully defined system.

4.1 Setting up the basics

As mentioned earlier, the key to a successful implementation of MLS is planning. For our example, we determined that we have three SECLEVELS (Unclassified, Corporate Confidential, and Regulatory). We also define five categories of resources (payroll, credit card, source code, financials, and general). From this matrix of choices, we then determine what SECLABEL resources need to be defined.

4.2 Basic Vanguard Administrator navigation

First, we provide a few basics about Vanguard Administrator before we get started. Figure 4-1 on page 61 shows the Administrator Main Menu. We concentrate our solution using options 2 and 3. Administrator allows “fast pathing” from the main menu, although you can proceed down the menu tree one level at a time. Throughout this chapter, we direct you to particular functions using this method. As an example, to reach the User Reports selection panel, specify option 3.1.



```
V6.3                        VANGUARD  ADMINISTRATOR           Date: 12/11/2006
OPTION ===>                                                   Time: 10:23
                            ADMINISTRATOR MAIN MENU

   0  Initialize ADMINISTRATOR Options    9  Vanguard Analyzer
   1  Task Oriented Administration       10  Vanguard Advisor
   2  Security Server Commands           11  Data Services
   3  Security Server Reports            12  User Data Management
   4  On-line Access Analysis            13  Connect Manager
   5  Command Scheduler                  14  Unix File Manager
   6  Vanguard Identity Manager          15  Registration Manager
   7  Installation Data Management       16  PasswordReset Reg Reports
   8  Information and Analysis Services  17  Vanguard Enforcer
  ST  Extract Statistics                 18  Digital Certificates
   X  Exit

   Active Extract Files: Small  ==> QS390.V6R3.SVSAM
                         Medium ==> QS390.V6R3.MVSAM

                  Please consult the help text for this panel
           regarding new feature information and contact information.

         Copyright 1989-2006 Vanguard Integrity Professionals - Nevada.
                              All rights reserved.
```

Figure 4-1 Administrator Main Menu

4.2.1 Standard masking

The RACF database contains a sizable amount of information. Vanguard Administrator (Administrator for short) enables you to specify extensive search criteria, thus narrowing the search to only the data you want to see. Use the Masking Fields section of a panel to specify simple search criteria. Using the Masking Fields section of a panel to produce a report is referred to as standard masking. Standard masking is used when you can produce the information you need by filling in the Masking Fields section of the panel. Standard masking fields are processed as a logical AND condition. When you specify text strings such as profile names, installation data, and classes as masking values, you can use the generic characters * and % to further refine your search.

4.2.2 Enhanced masking

Use enhanced masking expressions when you need to create a more sophisticated search by connecting search statements with a logical operator. Expressions that consist of multiple elements combined with ANDs, ORs or NOTs use Boolean logic. Enhanced masking includes support for the logical OR, AND, and NOT as well as nested parenthesis.

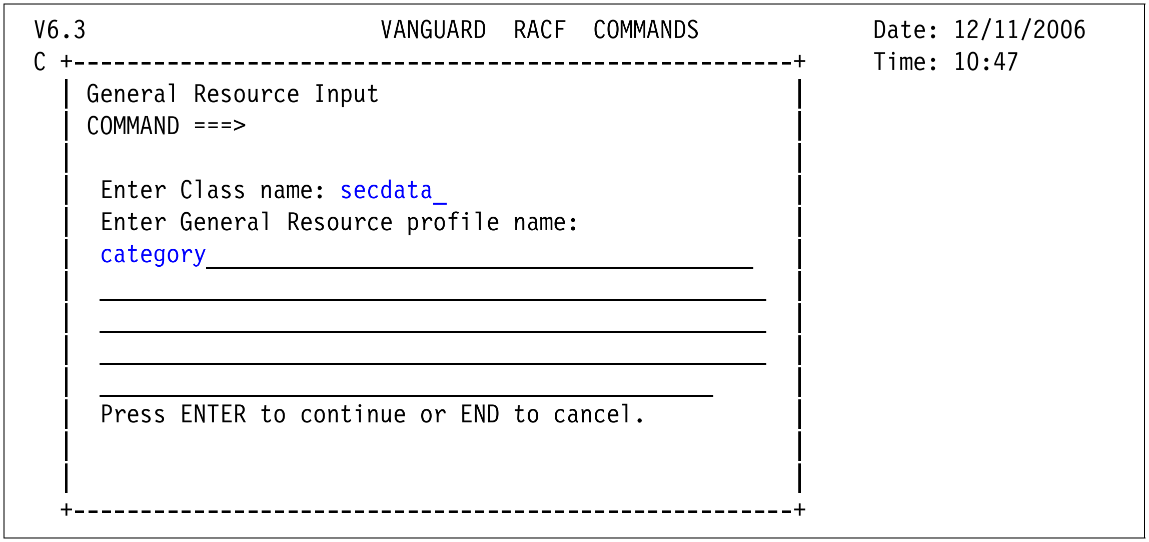
4.2.3 QuickGen

The QuickGen feature enables the user to generate free-form output (usually RACF command oriented) from any Security Server report panel. Although similar in approach to the RACF command SEARCH MASK CLIST, QuickGen functionality goes beyond the SEARCH MASK CLIST command capabilities. Because QuickGen capabilities are coupled with the extensive search capabilities in Vanguard Administrator, you can quickly generate commands or output based on a wide variety of informational needs.

4.3 Setting up the basic components

For our first task, we set up our SECDATA profiles that define the security levels and categories. We use the Security Server Commands for General Resources. After specifying option 2.4 from the main menu, the panel shown in Figure 4-2 on page 63 opens.

We specify the class (SECDATA) and profile name (CATEGORY) and press Enter (Figure 4-2).

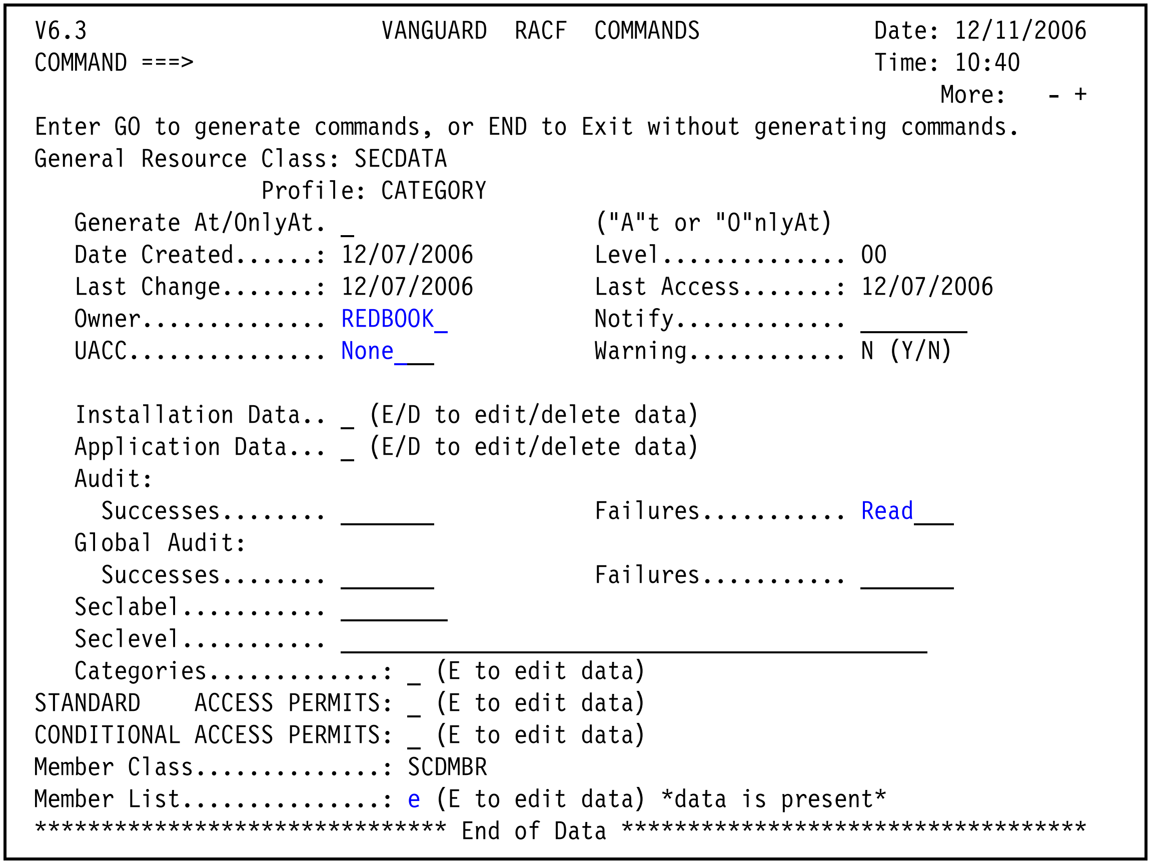


```
V6.3                      VANGUARD  RACF  COMMANDS              Date: 12/11/2006
C +------------------------------------------------------+      Time: 10:47
  | General Resource Input                               |
  | COMMAND ===>                                         |
  |                                                      |
  |  Enter Class name: secdata_                          |
  |  Enter General Resource profile name:                |
  |  category__________________________________________  |
  |  __________________________________________________  |
  |  __________________________________________________  |
  |  __________________________________________________  |
  |  ______________________________________________      |
  |  Press ENTER to continue or END to cancel.           |
  |                                                      |
  |                                                      |
  +------------------------------------------------------+
```

Figure 4-2 Specify the SECDATA category profile

A simple “fill in the blanks” panel opens to specify the parameters used to create the RACF commands to define the CATEGORY profile (Figure 4-3). Complete the OWNER and UACC fields and press Enter. As you will see, the panel remains so that you can continue to make needed changes. If you want to add installation data (a highly recommended practice), simply place an E at the Installation Data field and a 256 byte field opens that you can complete. After entering the Installation Data, press PF3 to return to this panel.

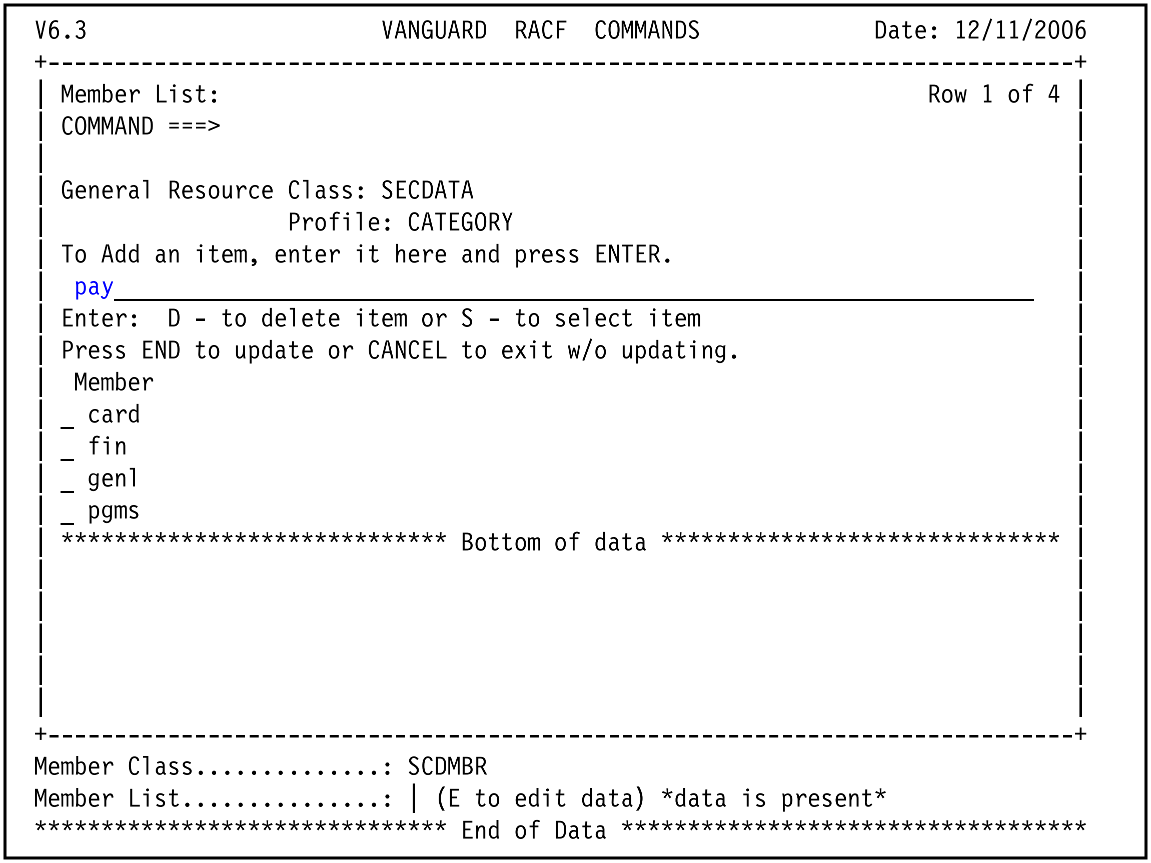
The final step here is to specify the various categories we will use. Place an E in the Member List field and press Enter.



```
V6.3                     VANGUARD  RACF  COMMANDS             Date: 12/11/2006
COMMAND ===>                                                  Time: 10:40
                                                                   More:   - +
Enter GO to generate commands, or END to Exit without generating commands.
General Resource Class: SECDATA
                 Profile: CATEGORY
   Generate At/OnlyAt. _                  ("A"t or "O"nlyAt)
   Date Created......: 12/07/2006         Level............. 00
   Last Change.......: 12/07/2006         Last Access.......: 12/07/2006
   Owner............. REDBOOK_            Notify............ ________
   UACC.............. None___             Warning........... N (Y/N)

   Installation Data.. _ (E/D to edit/delete data)
   Application Data... _ (E/D to edit/delete data)
   Audit:
     Successes........ _______            Failures.......... Read___
   Global Audit:
     Successes........ _______            Failures.......... _______
   Seclabel.......... ________
   Seclevel.......... ____________________________________________
   Categories............: _ (E to edit data)
STANDARD    ACCESS PERMITS: _ (E to edit data)
CONDITIONAL ACCESS PERMITS: _ (E to edit data)
Member Class.............: SCDMBR
Member List..............: e (E to edit data) *data is present*
******************************* End of Data ***********************************
```

Figure 4-3   Input profile parameters

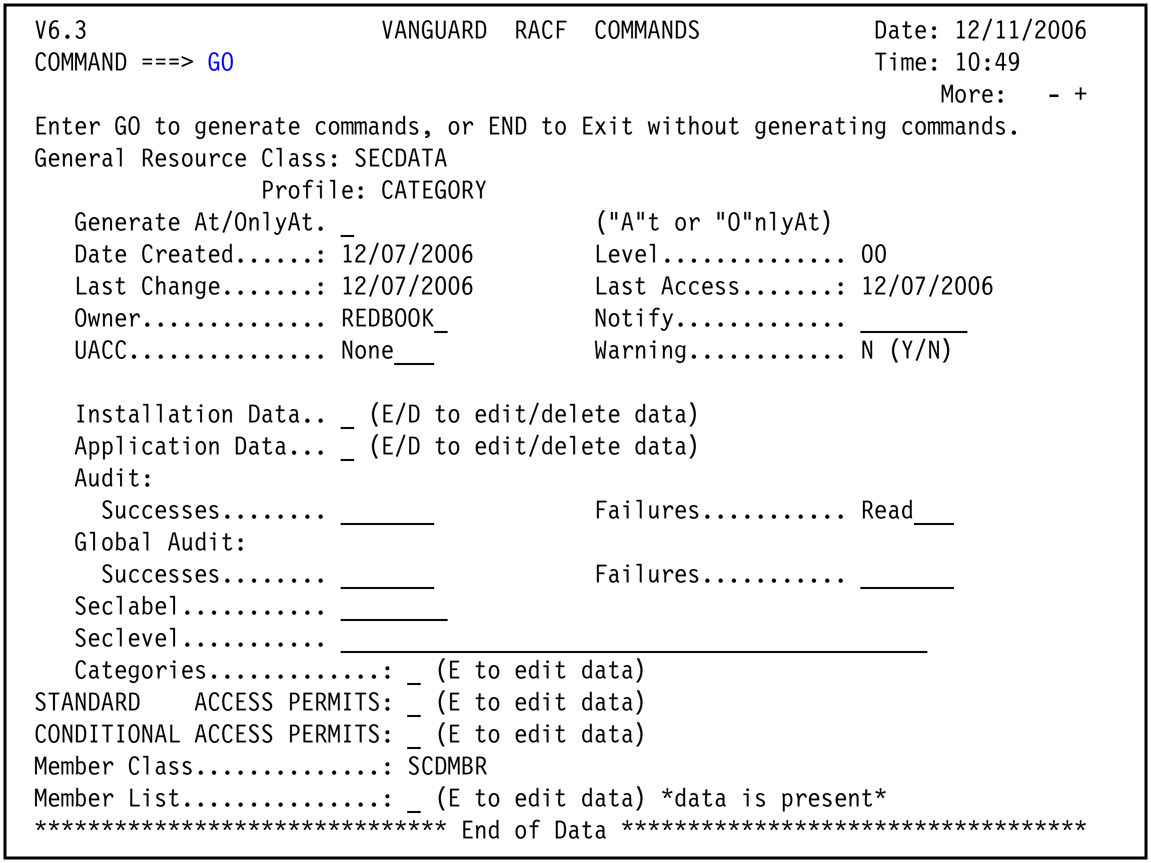
Now we specify the various categories we want to use by typing each of them into the Add item field and pressing Enter (Figure 4-4). This adds them to the member list at the bottom of the panel. Continue this until all categories have been specified. If you make a mistake and need to remove one of the categories you entered, just place a D in front of the member you want to remove and press Enter. After specifying all the categories specified, press PF3 to return to the previous panel.



```
V6.3                    VANGUARD  RACF  COMMANDS             Date: 12/11/2006
+-----------------------------------------------------------------------------+
| Member List:                                                     Row 1 of 4 |
| COMMAND ===>                                                                |
|                                                                             |
| General Resource Class: SECDATA                                             |
|                  Profile: CATEGORY                                          |
| To Add an item, enter it here and press ENTER.                              |
|  pay_____________________________________________________________________   |
| Enter:  D - to delete item or S - to select item                            |
| Press END to update or CANCEL to exit w/o updating.                         |
|  Member                                                                     |
| _ card                                                                      |
| _ fin                                                                       |
| _ genl                                                                      |
| _ pgms                                                                      |
| ***************************** Bottom of data ****************************** |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
+-----------------------------------------------------------------------------+
Member Class..............: SCDMBR
Member List...............: | (E to edit data) *data is present*
****************************** End of Data ***********************************
```

Figure 4-4   Add members

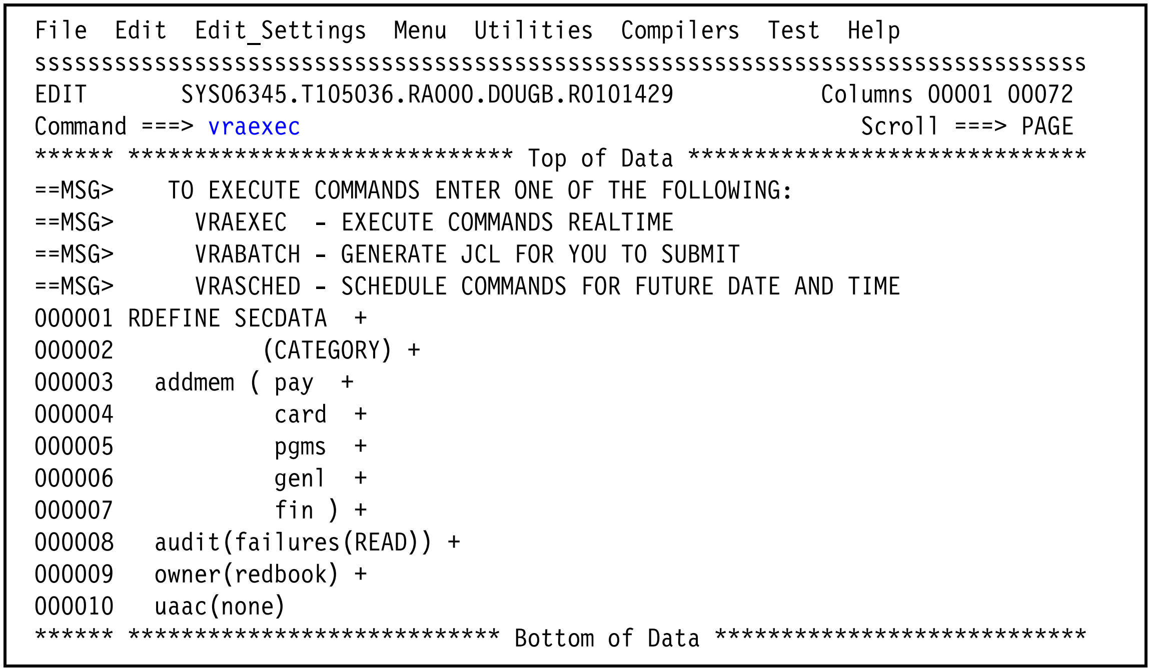
Satisfied that we have specified all the parameters needed, we are now ready to generate the RACF commands to create our CATEGORY profile. Type GO at the command prompt and press Enter (Figure 4-5).



```
V6.3                      VANGUARD  RACF  COMMANDS             Date: 12/11/2006
COMMAND ===> GO                                                Time: 10:49
                                                                    More:   - +
Enter GO to generate commands, or END to Exit without generating commands.
General Resource Class: SECDATA
                 Profile: CATEGORY
   Generate At/OnlyAt. _                  ("A"t or "O"nlyAt)
   Date Created......: 12/07/2006         Level............. 00
   Last Change.......: 12/07/2006         Last Access.......: 12/07/2006
   Owner............. REDBOOK_            Notify............ ________
   UACC.............. None___             Warning........... N (Y/N)

   Installation Data.. _ (E/D to edit/delete data)
   Application Data... _ (E/D to edit/delete data)
   Audit:
     Successes........ _______            Failures.......... Read___
   Global Audit:
     Successes........ _______            Failures.......... _______
   Seclabel.......... ________
   Seclevel.......... ____________________________________________
   Categories............: _ (E to edit data)
STANDARD    ACCESS PERMITS: _ (E to edit data)
CONDITIONAL ACCESS PERMITS: _ (E to edit data)
Member Class.............: SCDMBR
Member List..............: _ (E to edit data) *data is present*
******************************* End of Data ***********************************
```

Figure 4-5 Generate commands

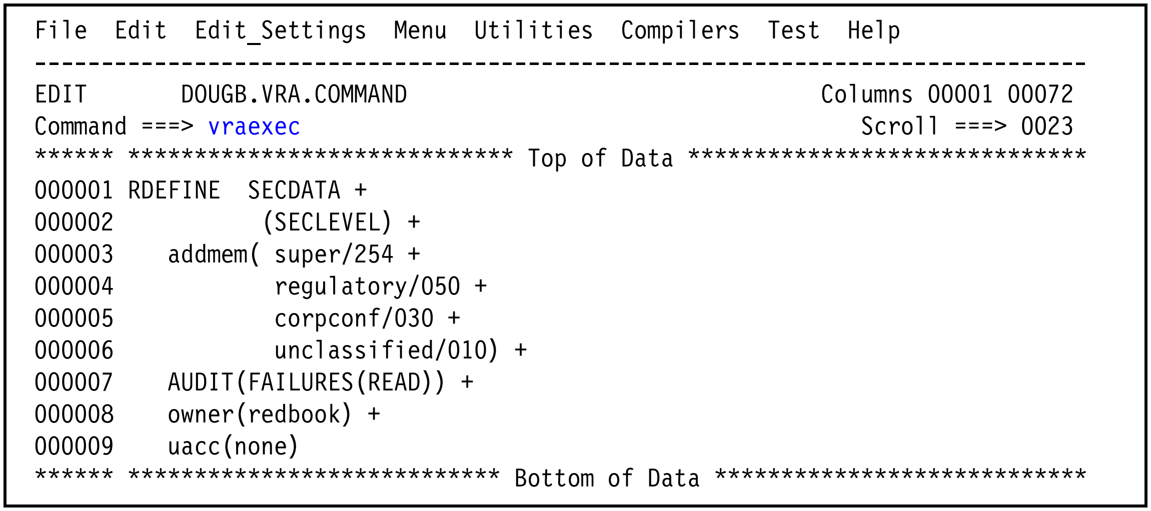
You are presented with a syntactically correct command string to create your profile. You have three options at this point, as described in the MSG lines on the panel. We select the VRAEXEC method and create our profile immediately (Figure 4-6).



```
File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
sssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
EDIT       SYS06345.T105036.RA000.DOUGB.R0101429          Columns 00001 00072
Command ===> vraexec                                          Scroll ===> PAGE
****** **************************** Top of Data ******************************
==MSG>    TO EXECUTE COMMANDS ENTER ONE OF THE FOLLOWING:
==MSG>      VRAEXEC  - EXECUTE COMMANDS REALTIME
==MSG>      VRABATCH - GENERATE JCL FOR YOU TO SUBMIT
==MSG>      VRASCHED - SCHEDULE COMMANDS FOR FUTURE DATE AND TIME
000001 RDEFINE SECDATA  +
000002           (CATEGORY) +
000003   addmem ( pay  +
000004            card  +
000005            pgms  +
000006            genl  +
000007            fin ) +
000008   audit(failures(READ)) +
000009   owner(redbook) +
000010   uaac(none)
****** *************************** Bottom of Data ***************************
```

Figure 4-6   Command execution

We then repeat this process (Figure 4-2 on page 63 through Figure 4-6 on page 67) for the SECLEVEL profile, resulting in the commands in Figure 4-7.

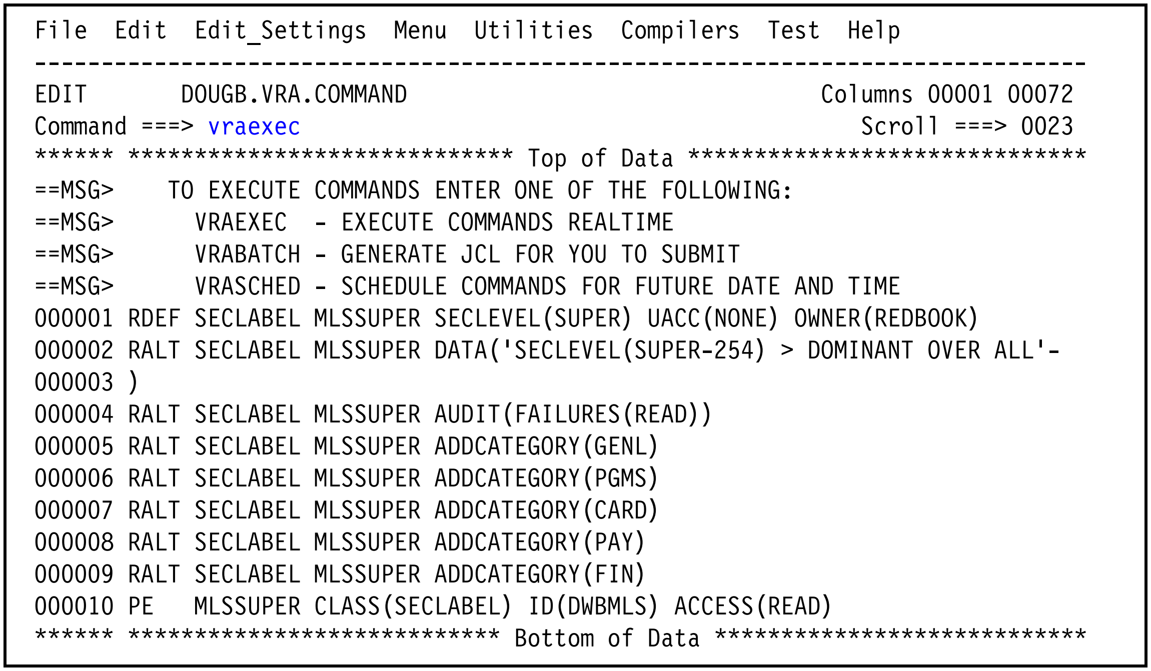


```
File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       DOUGB.VRA.COMMAND                              Columns 00001 00072
Command ===> vraexec                                         Scroll ===> 0023
****** ***************************** Top of Data ******************************
000001 RDEFINE  SECDATA +
000002           (SECLEVEL) +
000003    addmem( super/254 +
000004            regulatory/050 +
000005            corpconf/030 +
000006            unclassified/010) +
000007    AUDIT(FAILURES(READ)) +
000008    owner(redbook) +
000009    uacc(none)
****** **************************** Bottom of Data ****************************
```

Figure 4-7   Commands for SECLEVEL profile

We use this process (Figure 4-2 on page 63 through Figure 4-6 on page 67) for the various SECLABEL profiles we previously determined we need.

Finally, we generate a “super user” SECLABEL (Figure 4-8). This label is designed as a safety net during the initial implementation. It has dominance over all levels and categories and must be deleted after completing the implementation.



```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ---------------------------------------------------------------------------
 EDIT       DOUGB.VRA.COMMAND                              Columns 00001 00072
 Command ===> vraexec                                         Scroll ===> 0023
 ****** ***************************** Top of Data ******************************
 ==MSG>    TO EXECUTE COMMANDS ENTER ONE OF THE FOLLOWING:
 ==MSG>      VRAEXEC  - EXECUTE COMMANDS REALTIME
 ==MSG>      VRABATCH - GENERATE JCL FOR YOU TO SUBMIT
 ==MSG>      VRASCHED - SCHEDULE COMMANDS FOR FUTURE DATE AND TIME
 000001 RDEF SECLABEL MLSSUPER SECLEVEL(SUPER) UACC(NONE) OWNER(REDBOOK)
 000002 RALT SECLABEL MLSSUPER DATA('SECLEVEL(SUPER-254) > DOMINANT OVER ALL'-
 000003 )
 000004 RALT SECLABEL MLSSUPER AUDIT(FAILURES(READ))
 000005 RALT SECLABEL MLSSUPER ADDCATEGORY(GENL)
 000006 RALT SECLABEL MLSSUPER ADDCATEGORY(PGMS)
 000007 RALT SECLABEL MLSSUPER ADDCATEGORY(CARD)
 000008 RALT SECLABEL MLSSUPER ADDCATEGORY(PAY)
 000009 RALT SECLABEL MLSSUPER ADDCATEGORY(FIN)
 000010 PE   MLSSUPER CLASS(SECLABEL) ID(DWBMLS) ACCESS(READ)
 ****** **************************** Bottom of Data ****************************
```

Figure 4-8 Defining a SECLABEL “super user”

4.4 Assigning default security labels to users

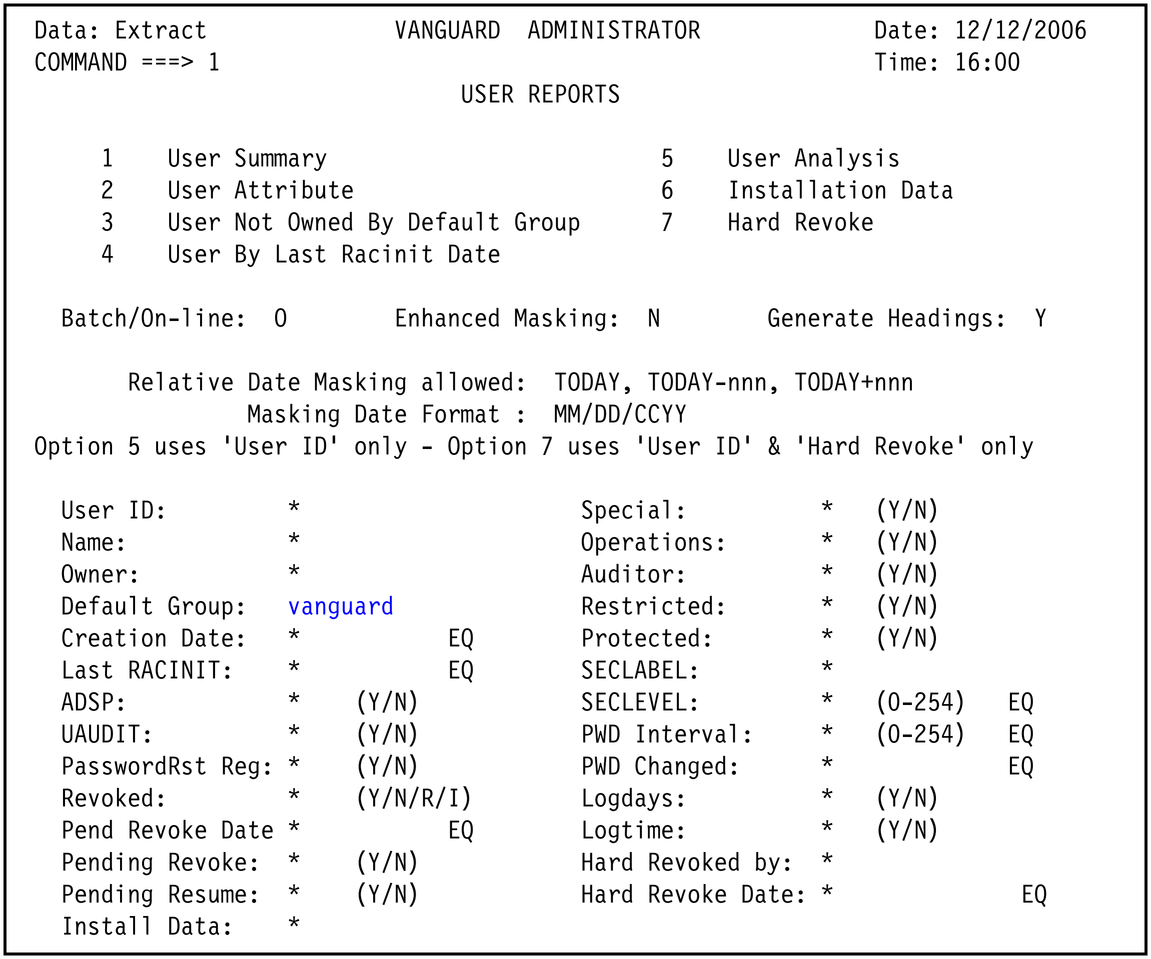
Now that the initial SECLABEL profiles are defined, we assign default SECLABELs to the user base.

In this example, we select a subset of our user base and generate the required commands to permit the use of a label and set it as the default.

We use option 3.1 from the main menu.The panel shown in Figure 4-9 on page 70 opens.

We begin with those users that have a default group of VANGUARD. In this instance, we can use standard masking and simply enter `VANGUARD` in the appropriate field and press Enter.

As a side note, we can use generic characters here to select multiple groups. For example, if we wanted to select all users who had a default group that started with VAN, we would specify VAN* in the field.

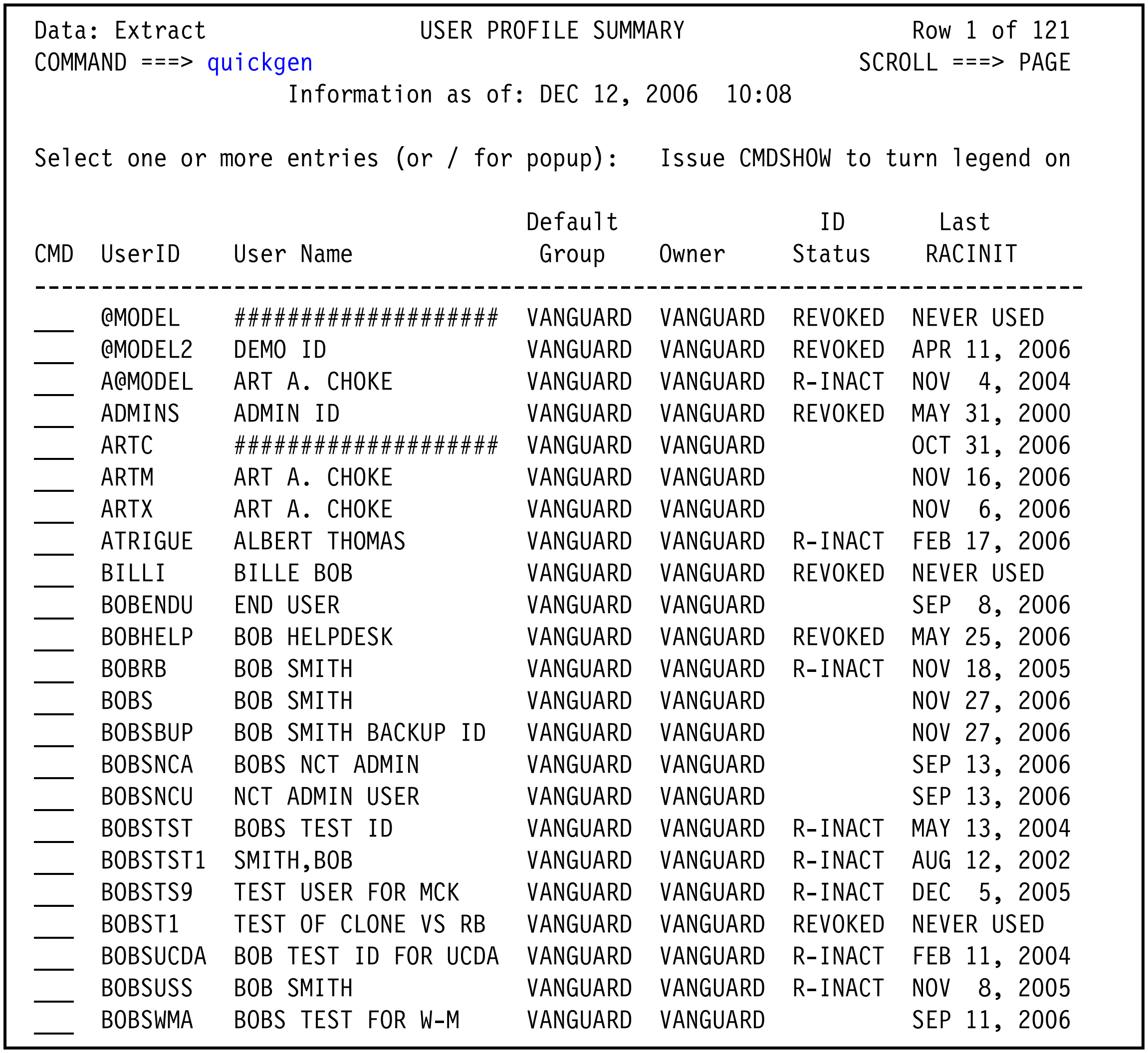


```
Data: Extract              VANGUARD  ADMINISTRATOR             Date: 12/12/2006
COMMAND ===> 1                                                 Time: 16:00
                                USER REPORTS

     1    User Summary                         5    User Analysis
     2    User Attribute                       6    Installation Data
     3    User Not Owned By Default Group      7    Hard Revoke
     4    User By Last Racinit Date

  Batch/On-line:  O        Enhanced Masking:  N        Generate Headings:  Y

       Relative Date Masking allowed:  TODAY, TODAY-nnn, TODAY+nnn
                Masking Date Format :  MM/DD/CCYY
Option 5 uses 'User ID' only - Option 7 uses 'User ID' & 'Hard Revoke' only

  User ID:         *                     Special:          *   (Y/N)
  Name:            *                     Operations:       *   (Y/N)
  Owner:           *                     Auditor:          *   (Y/N)
  Default Group:   vanguard              Restricted:       *   (Y/N)
  Creation Date:   *           EQ        Protected:        *   (Y/N)
  Last RACINIT:    *           EQ        SECLABEL:         *
  ADSP:            *    (Y/N)            SECLEVEL:         *   (0-254)   EQ
  UAUDIT:          *    (Y/N)            PWD Interval:     *   (0-254)   EQ
  PasswordRst Reg: *    (Y/N)            PWD Changed:      *             EQ
  Revoked:         *    (Y/N/R/I)        Logdays:          *   (Y/N)
  Pend Revoke Date *           EQ        Logtime:          *   (Y/N)
  Pending Revoke:  *    (Y/N)            Hard Revoked by:  *
  Pending Resume:  *    (Y/N)            Hard Revoke Date: *              EQ
  Install Data:    *
```

Figure 4-9 Select USER profiles

After completing the specification of our selection criteria, we press Enter. The report (Figure 4-10) listing all users that matched our selection criteria opens. In the upper-right corner, you will see that our criteria returned 121 users.

We invoke the QuickGen function by typing `QUICKGEN` (or `QG`) on the command line and pressing Enter (Figure 4-10).



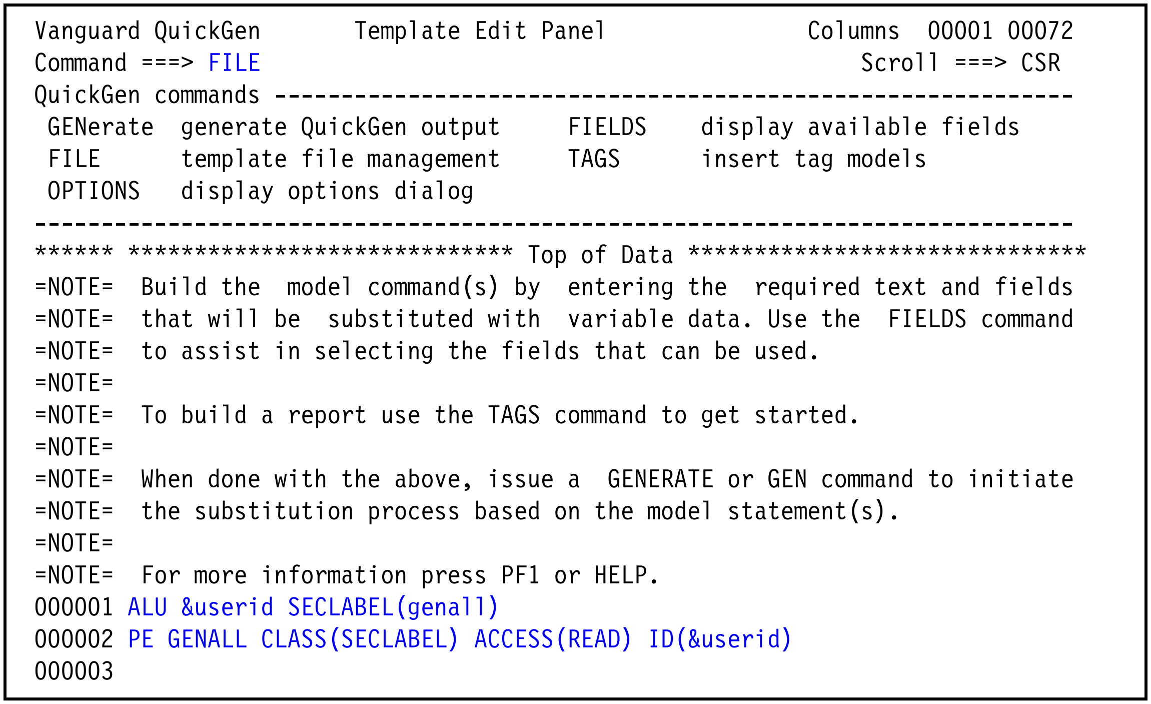
```
Data: Extract             USER PROFILE SUMMARY                Row 1 of 121
COMMAND ===> quickgen                                     SCROLL ===> PAGE
                   Information as of: DEC 12, 2006  10:08

Select one or more entries (or / for popup):   Issue CMDSHOW to turn legend on

                                  Default               ID       Last
CMD  UserID    User Name           Group    Owner     Status    RACINIT
------------------------------------------------------------------------------
___  @MODEL    ####################  VANGUARD  VANGUARD  REVOKED  NEVER USED
___  @MODEL2   DEMO ID               VANGUARD  VANGUARD  REVOKED  APR 11, 2006
___  A@MODEL   ART A. CHOKE          VANGUARD  VANGUARD  R-INACT  NOV  4, 2004
___  ADMINS    ADMIN ID              VANGUARD  VANGUARD  REVOKED  MAY 31, 2000
___  ARTC      ####################  VANGUARD  VANGUARD           OCT 31, 2006
___  ARTM      ART A. CHOKE          VANGUARD  VANGUARD           NOV 16, 2006
___  ARTX      ART A. CHOKE          VANGUARD  VANGUARD           NOV  6, 2006
___  ATRIGUE   ALBERT THOMAS         VANGUARD  VANGUARD  R-INACT  FEB 17, 2006
___  BILLI     BILLE BOB             VANGUARD  VANGUARD  REVOKED  NEVER USED
___  BOBENDU   END USER              VANGUARD  VANGUARD           SEP  8, 2006
___  BOBHELP   BOB HELPDESK          VANGUARD  VANGUARD  REVOKED  MAY 25, 2006
___  BOBRB     BOB SMITH             VANGUARD  VANGUARD  R-INACT  NOV 18, 2005
___  BOBS      BOB SMITH             VANGUARD  VANGUARD           NOV 27, 2006
___  BOBSBUP   BOB SMITH BACKUP ID   VANGUARD  VANGUARD           NOV 27, 2006
___  BOBSNCA   BOBS NCT ADMIN        VANGUARD  VANGUARD           SEP 13, 2006
___  BOBSNCU   NCT ADMIN USER        VANGUARD  VANGUARD           SEP 13, 2006
___  BOBSTST   BOBS TEST ID          VANGUARD  VANGUARD  R-INACT  MAY 13, 2004
___  BOBSTST1  SMITH,BOB             VANGUARD  VANGUARD  R-INACT  AUG 12, 2002
___  BOBSTS9   TEST USER FOR MCK     VANGUARD  VANGUARD  R-INACT  DEC  5, 2005
___  BOBST1    TEST OF CLONE VS RB   VANGUARD  VANGUARD  REVOKED  NEVER USED
___  BOBSUCDA  BOB TEST ID FOR UCDA  VANGUARD  VANGUARD  R-INACT  FEB 11, 2004
___  BOBSUSS   BOB SMITH             VANGUARD  VANGUARD  R-INACT  NOV  8, 2005
___  BOBSWMA   BOBS TEST FOR W-M     VANGUARD  VANGUARD           SEP 11, 2006
```

Figure 4-10   Invoke QuickGen

Now we proceed to lay out the templates for the commands we want to create. The QuickGen feature use variables (denoted by a leading &) to replace that value for each line in the report from which we just came (in this case, 121).

You can specify multiple lines in the template, enabling you to both assign the default label and the permission to use that SECLABEL resource.

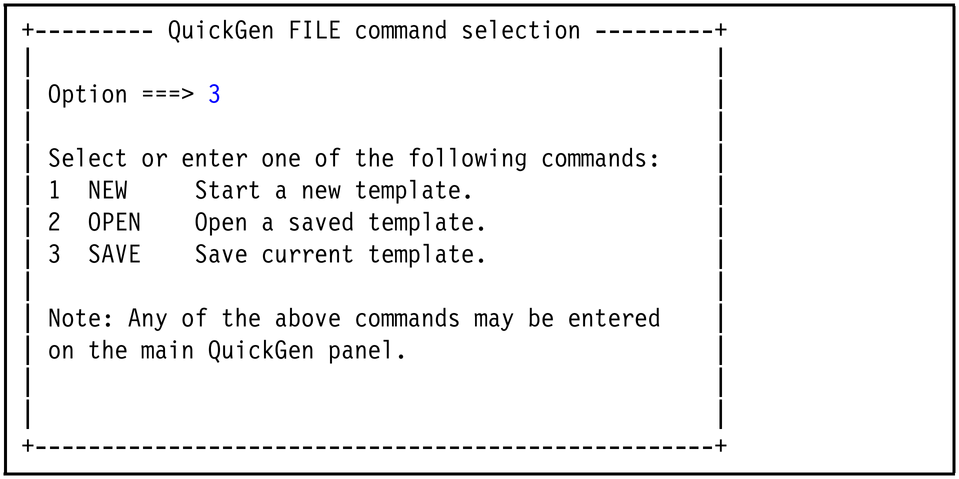
Because we perform this process a number of times for our various user groups, we save this template so that we do not need to re-type it numerous times. To do this, type `FILE` at the command line (Figure 4-11).



```
Vanguard QuickGen       Template Edit Panel            Columns  00001 00072
Command ===> FILE                                          Scroll ===> CSR
QuickGen commands -------------------------------------------------------------
 GENerate  generate QuickGen output     FIELDS    display available fields
 FILE      template file management     TAGS      insert tag models
 OPTIONS   display options dialog
------------------------------------------------------------------------------
****** ***************************** Top of Data *****************************
=NOTE=  Build the  model command(s) by  entering the  required text and fields
=NOTE=  that will be  substituted with  variable data. Use the  FIELDS command
=NOTE=  to assist in selecting the fields that can be used.
=NOTE=
=NOTE=  To build a report use the TAGS command to get started.
=NOTE=
=NOTE=  When done with the above, issue a  GENERATE or GEN command to initiate
=NOTE=  the substitution process based on the model statement(s).
=NOTE=
=NOTE=  For more information press PF1 or HELP.
000001 ALU &userid SECLABEL(genall)
000002 PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(&userid)
000003
```

Figure 4-11 Create QuickGen command template

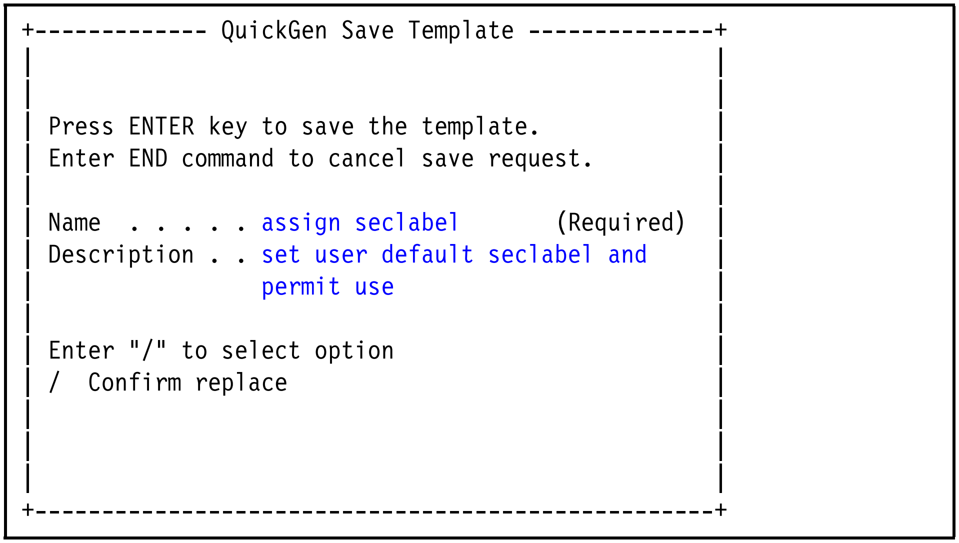
Specify option 3 to save this template for future use (Figure 4-12).



```
+--------- QuickGen FILE command selection ---------+
|                                                   |
| Option ===> 3                                     |
|                                                   |
| Select or enter one of the following commands:    |
| 1  NEW     Start a new template.                  |
| 2  OPEN    Open a saved template.                 |
| 3  SAVE    Save current template.                 |
|                                                   |
| Note: Any of the above commands may be entered    |
| on the main QuickGen panel.                       |
|                                                   |
|                                                   |
+---------------------------------------------------+
```

Figure 4-12   Storing a template

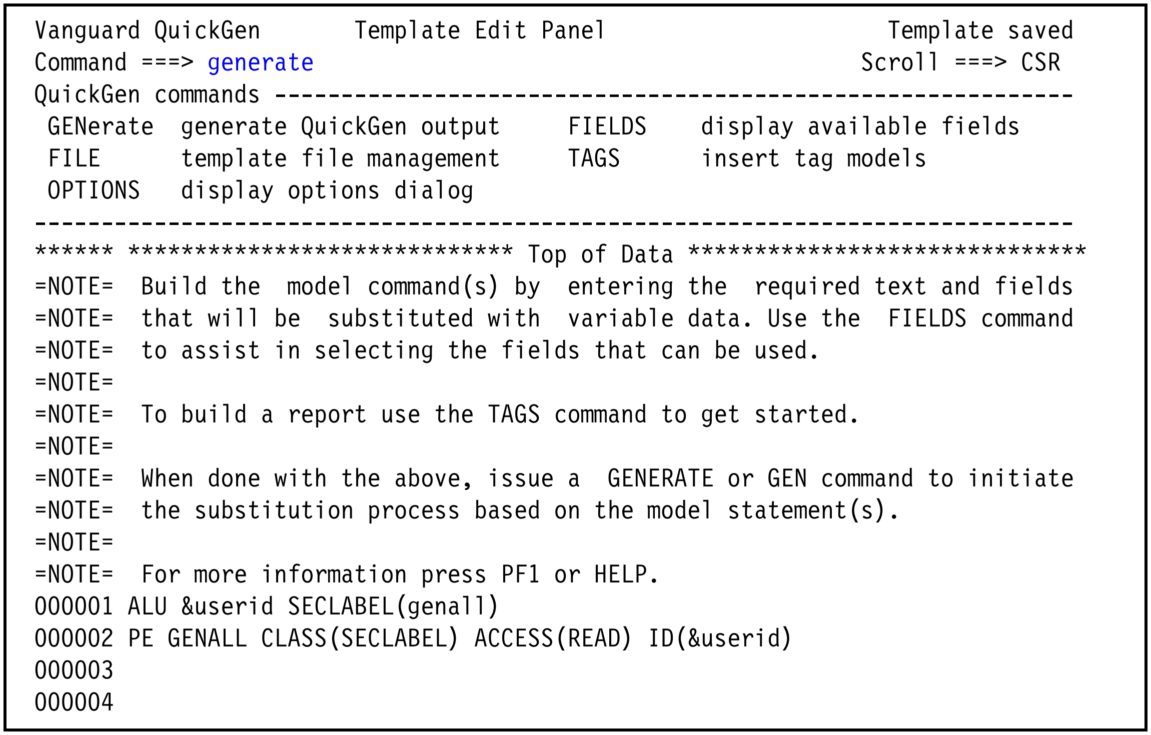
In the Save Template panel, we specify a template name and a brief description (Example 4-13). Press Enter to store the template in the QGVSAM file. This template is now available to all Administrator users in the future. Retrieve the template by specifying FILE and then OPEN at the Template Edit Panel.



```
+------------- QuickGen Save Template --------------+
|                                                   |
|                                                   |
| Press ENTER key to save the template.             |
| Enter END command to cancel save request.         |
|                                                   |
| Name  . . . . . assign seclabel       (Required)  |
| Description . . set user default seclabel and     |
|                 permit use                        |
|                                                   |
| Enter "/" to select option                        |
| /  Confirm replace                                |
|                                                   |
|                                                   |
|                                                   |
+---------------------------------------------------+
```

Figure 4-13   Label template

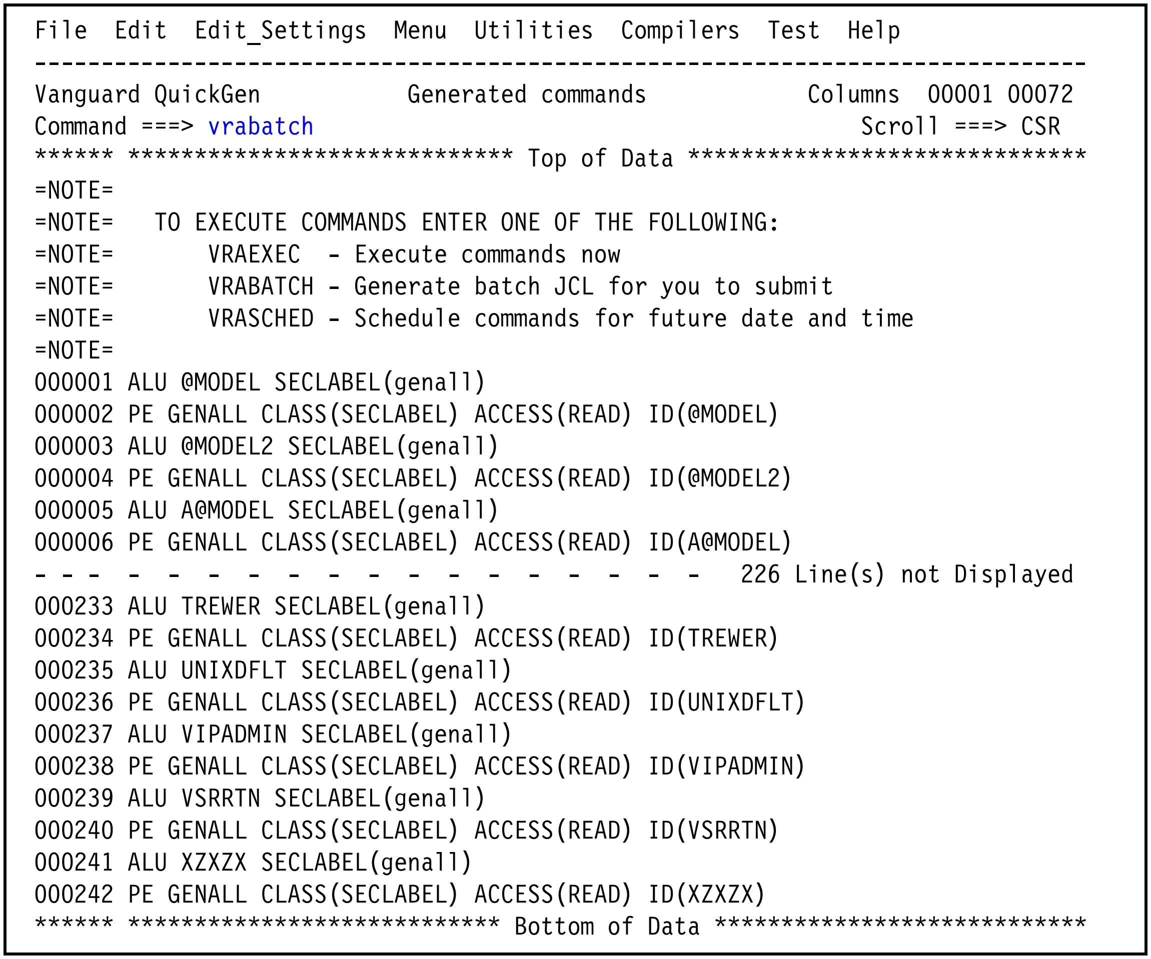
We return to the Template Edit Panel (Figure 4-14). Satisfied with our template, we type GENERATE (or GEN) at the command prompt. This takes each line from our report and applies the template we specified.



```
Vanguard QuickGen       Template Edit Panel                     Template saved
Command ===> generate                                          Scroll ===> CSR
QuickGen commands ------------------------------------------------------------
 GENerate  generate QuickGen output     FIELDS    display available fields
 FILE      template file management     TAGS      insert tag models
 OPTIONS   display options dialog
------------------------------------------------------------------------------
****** ***************************** Top of Data *****************************
=NOTE=  Build the  model command(s) by  entering the  required text and fields
=NOTE=  that will be  substituted with  variable data. Use the  FIELDS command
=NOTE=  to assist in selecting the fields that can be used.
=NOTE=
=NOTE=  To build a report use the TAGS command to get started.
=NOTE=
=NOTE=  When done with the above, issue a  GENERATE or GEN command to initiate
=NOTE=  the substitution process based on the model statement(s).
=NOTE=
=NOTE=  For more information press PF1 or HELP.
000001 ALU &userid SECLABEL(genall)
000002 PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(&userid)
000003
000004
```

Figure 4-14 Generate USER alter commands

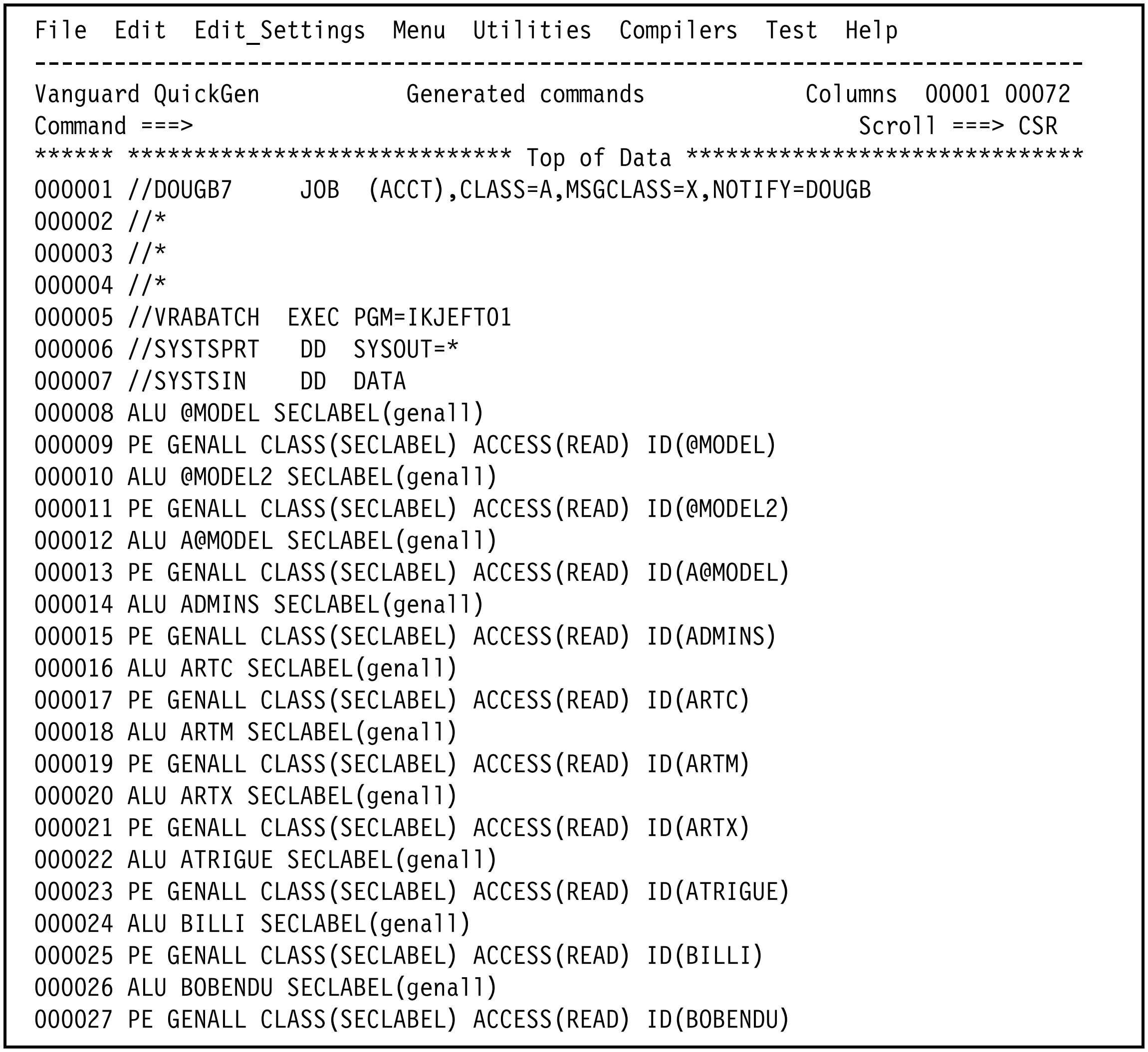
In a matter of seconds, we are presented with 242 lines of RACF commands (two for each line in the report). We can now elect one of the action options. In this case, because of the large number of commands generated, we select VRABATCH (Figure 4-15).



```
File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
Vanguard QuickGen           Generated commands           Columns  00001 00072
Command ===> vrabatch                                        Scroll ===> CSR
****** ***************************** Top of Data ******************************
=NOTE=
=NOTE=   TO EXECUTE COMMANDS ENTER ONE OF THE FOLLOWING:
=NOTE=       VRAEXEC  - Execute commands now
=NOTE=       VRABATCH - Generate batch JCL for you to submit
=NOTE=       VRASCHED - Schedule commands for future date and time
=NOTE=
000001 ALU @MODEL SECLABEL(genall)
000002 PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(@MODEL)
000003 ALU @MODEL2 SECLABEL(genall)
000004 PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(@MODEL2)
000005 ALU A@MODEL SECLABEL(genall)
000006 PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(A@MODEL)
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -   226 Line(s) not Displayed
000233 ALU TREWER SECLABEL(genall)
000234 PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(TREWER)
000235 ALU UNIXDFLT SECLABEL(genall)
000236 PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(UNIXDFLT)
000237 ALU VIPADMIN SECLABEL(genall)
000238 PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(VIPADMIN)
000239 ALU VSRRTN SECLABEL(genall)
000240 PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(VSRRTN)
000241 ALU XZXZX SECLABEL(genall)
000242 PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(XZXZX)
****** *************************** Bottom of Data ***************************
```

Figure 4-15 Execute commands in batch

Verify the JCL and type `submit` at the command prompt.



File Edit Edit_Settings Menu Utilities Compilers Test Help

```
Vanguard QuickGen          Generated commands          Columns  00001 00072
Command ===>                                                 Scroll ===> CSR
****** **************************** Top of Data ******************************
//DOUGB7     JOB  (ACCT),CLASS=A,MSGCLASS=X,NOTIFY=DOUGB
//*
//*
//*
//VRABATCH  EXEC PGM=IKJEFT01
//SYSTSPRT   DD  SYSOUT=*
//SYSTSIN    DD  DATA
ALU @MODEL SECLABEL(genall)
PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(@MODEL)
ALU @MODEL2 SECLABEL(genall)
PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(@MODEL2)
ALU A@MODEL SECLABEL(genall)
PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(A@MODEL)
ALU ADMINS SECLABEL(genall)
PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(ADMINS)
ALU ARTC SECLABEL(genall)
PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(ARTC)
ALU ARTM SECLABEL(genall)
PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(ARTM)
ALU ARTX SECLABEL(genall)
PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(ARTX)
ALU ATRIGUE SECLABEL(genall)
PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(ATRIGUE)
ALU BILLI SECLABEL(genall)
PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(BILLI)
ALU BOBENDU SECLABEL(genall)
PE GENALL CLASS(SECLABEL) ACCESS(READ) ID(BOBENDU)
```

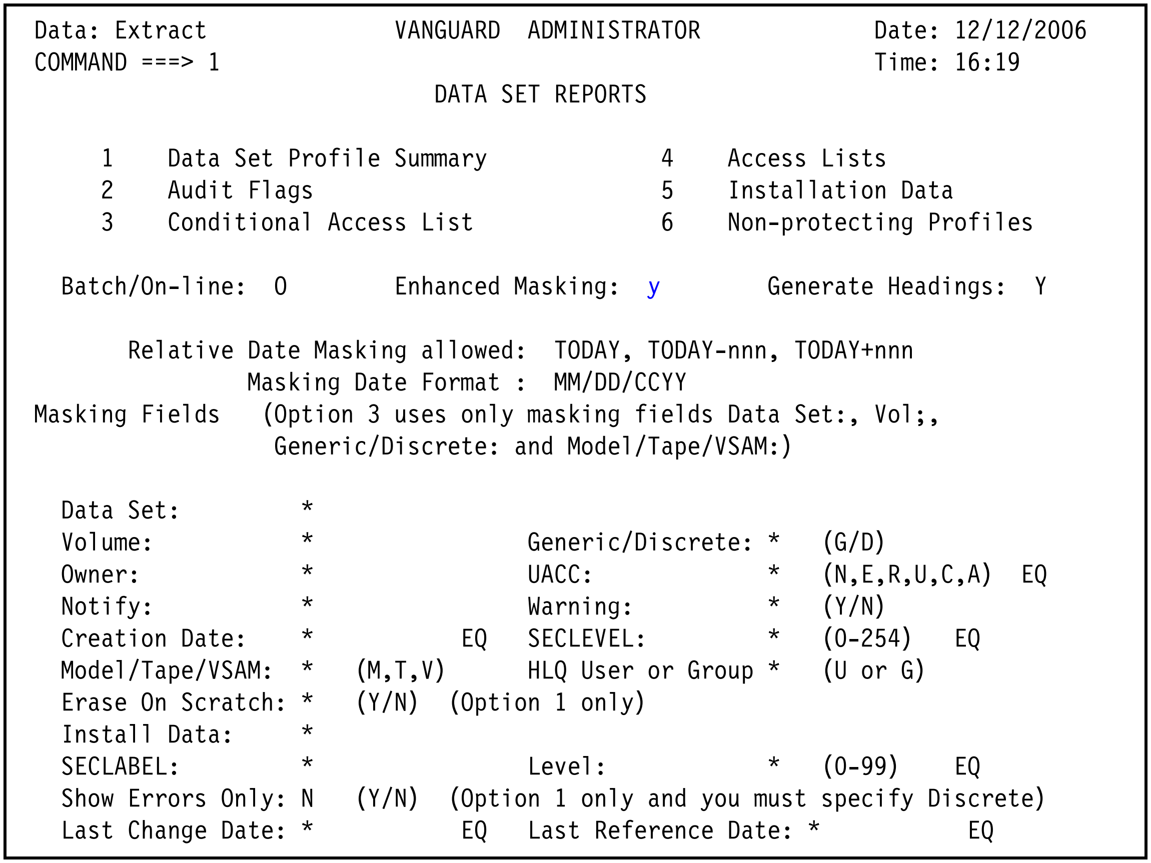
Figure 4-16 Submit commands for batch execution

Repeat this process to assign the appropriate SECLABEL to your user base as previously determined in your planning process. After finishing this process for all the users, we move on to the various resources.

4.5 Labeling existing resources

We now describe how you can easily apply SECLABELs to your existing resource profiles. For our example, we apply the REGFIN label to a series of data set profiles used to protect files containing data that falls under one of the regulatory guidelines for sensitivity. The first step is to determine what data set profiles need to be modified. We use the DATASET function in the Security Server Reports (option 3.3).

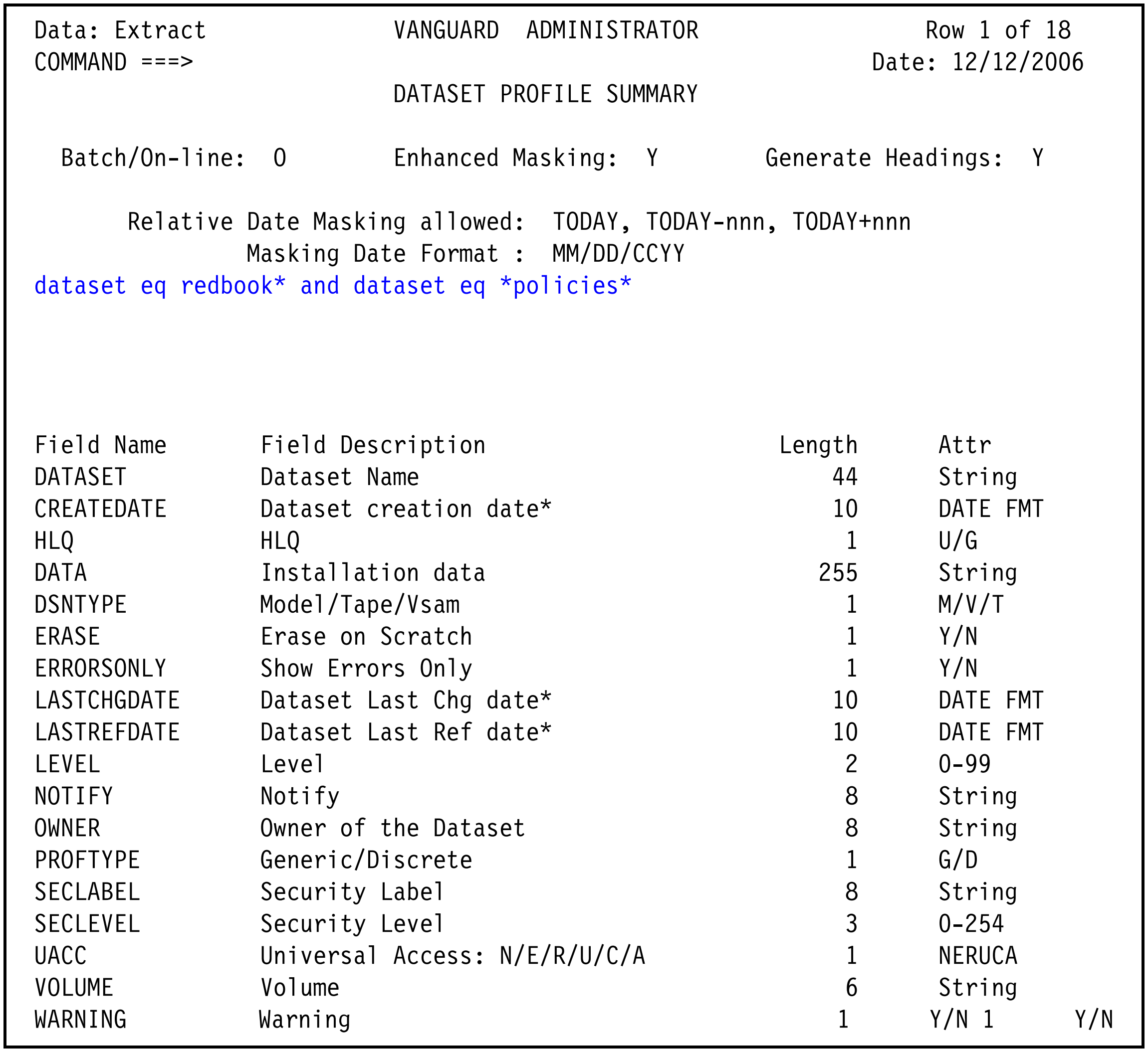
In this example, we use enhanced masking to select the profiles we want to modify. Specify `Y` for the Enhanced Masking option on the panel (Figure 4-17).



```
Data: Extract            VANGUARD  ADMINISTRATOR           Date: 12/12/2006
COMMAND ===> 1                                             Time: 16:19
                              DATA SET REPORTS

     1    Data Set Profile Summary            4    Access Lists
     2    Audit Flags                         5    Installation Data
     3    Conditional Access List             6    Non-protecting Profiles

  Batch/On-line:  O       Enhanced Masking:  y       Generate Headings:  Y

       Relative Date Masking allowed:  TODAY, TODAY-nnn, TODAY+nnn
                Masking Date Format :  MM/DD/CCYY
Masking Fields   (Option 3 uses only masking fields Data Set:, Vol;,
                  Generic/Discrete: and Model/Tape/VSAM:)

  Data Set:         *
  Volume:           *                Generic/Discrete: *   (G/D)
  Owner:            *                UACC:             *   (N,E,R,U,C,A)  EQ
  Notify:           *                Warning:          *   (Y/N)
  Creation Date:    *           EQ   SECLEVEL:         *   (0-254)   EQ
  Model/Tape/VSAM:  *   (M,T,V)      HLQ User or Group *   (U or G)
  Erase On Scratch: *   (Y/N)  (Option 1 only)
  Install Data:     *
  SECLABEL:         *                Level:            *   (0-99)    EQ
  Show Errors Only: N   (Y/N)  (Option 1 only and you must specify Discrete)
  Last Change Date: *           EQ   Last Reference Date: *          EQ
```

Figure 4-17 Select profiles using enhanced masking

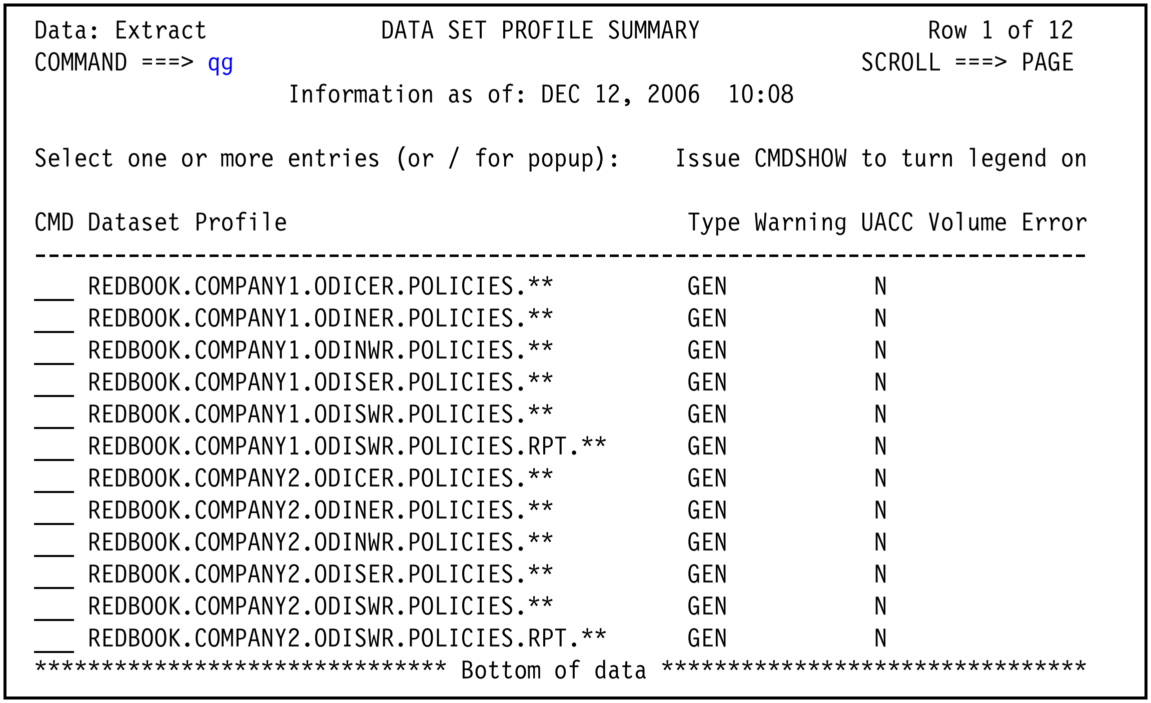
The panel in Figure 4-18 shows the field names and characteristics that we can use in our selection criteria. In this case, we want to select all data sets with a HLQ of REDBOOK and containing POLICIES elsewhere in the profile name. Using enhanced masking, we gain the full power of Boolean logic to set our selection criteria with a very high degree of granularity.



```
Data: Extract               VANGUARD  ADMINISTRATOR                Row 1 of 18
COMMAND ===>                                                   Date: 12/12/2006
                            DATASET PROFILE SUMMARY

  Batch/On-line:  O         Enhanced Masking:  Y       Generate Headings:  Y

       Relative Date Masking allowed:  TODAY, TODAY-nnn, TODAY+nnn
                Masking Date Format :  MM/DD/CCYY
dataset eq redbook* and dataset eq *policies*




Field Name        Field Description                       Length      Attr
DATASET           Dataset Name                                44      String
CREATEDATE        Dataset creation date*                      10      DATE FMT
HLQ               HLQ                                          1      U/G
DATA              Installation data                          255      String
DSNTYPE           Model/Tape/Vsam                              1      M/V/T
ERASE             Erase on Scratch                             1      Y/N
ERRORSONLY        Show Errors Only                             1      Y/N
LASTCHGDATE       Dataset Last Chg date*                      10      DATE FMT
LASTREFDATE       Dataset Last Ref date*                      10      DATE FMT
LEVEL             Level                                        2      0-99
NOTIFY            Notify                                       8      String
OWNER             Owner of the Dataset                         8      String
PROFTYPE          Generic/Discrete                             1      G/D
SECLABEL          Security Label                               8      String
SECLEVEL          Security Level                               3      0-254
UACC              Universal Access: N/E/R/U/C/A                1      NERUCA
VOLUME            Volume                                       6      String
WARNING           Warning                                     1      Y/N 1      Y/N
```

Figure 4-18 Specify enhanced masking criteria

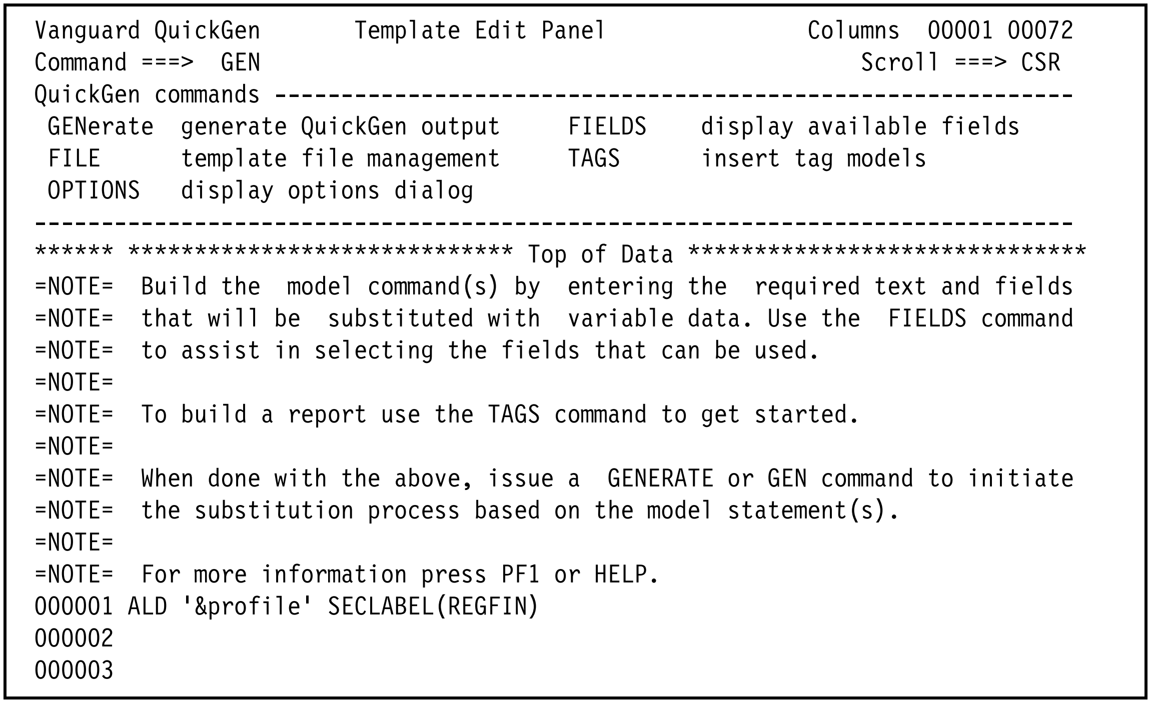
After confirming that the returned list of data set profiles is what we want, press Enter.



```
Data: Extract              DATA SET PROFILE SUMMARY                 Row 1 of 12
COMMAND ===> qg                                                SCROLL ===> PAGE
                   Information as of: DEC 12, 2006  10:08

Select one or more entries (or / for popup):    Issue CMDSHOW to turn legend on

CMD Dataset Profile                              Type Warning UACC Volume Error
-------------------------------------------------------------------------------
___ REDBOOK.COMPANY1.ODICER.POLICIES.**          GEN           N
___ REDBOOK.COMPANY1.ODINER.POLICIES.**          GEN           N
___ REDBOOK.COMPANY1.ODINWR.POLICIES.**          GEN           N
___ REDBOOK.COMPANY1.ODISER.POLICIES.**          GEN           N
___ REDBOOK.COMPANY1.ODISWR.POLICIES.**          GEN           N
___ REDBOOK.COMPANY1.ODISWR.POLICIES.RPT.**      GEN           N
___ REDBOOK.COMPANY2.ODICER.POLICIES.**          GEN           N
___ REDBOOK.COMPANY2.ODINER.POLICIES.**          GEN           N
___ REDBOOK.COMPANY2.ODINWR.POLICIES.**          GEN           N
___ REDBOOK.COMPANY2.ODISER.POLICIES.**          GEN           N
___ REDBOOK.COMPANY2.ODISWR.POLICIES.**          GEN           N
___ REDBOOK.COMPANY2.ODISWR.POLICIES.RPT.**      GEN           N
******************************* Bottom of data ********************************
```

Figure 4-19   Invoke QuickGen to create data set profile modification commands

Specify the template to modify the selected data set profiles (Figure 4-20). Our template command is an alter data set profile to add the SECLABEL parameter. Now, we enter GEN at the command prompt, which returns the requested commands for each profile that was shown in the previous report. At this point, we select a method to execute the generated commands.



```
Vanguard QuickGen       Template Edit Panel               Columns  00001 00072
Command ===>  GEN                                                 Scroll ===> CSR
QuickGen commands ---------------------------------------------------------------
 GENerate  generate QuickGen output     FIELDS    display available fields
 FILE      template file management     TAGS      insert tag models
 OPTIONS   display options dialog
------------------------------------------------------------------------------
****** ***************************** Top of Data ******************************
=NOTE=  Build the  model command(s) by  entering the  required text and fields
=NOTE=  that will be  substituted with  variable data. Use the  FIELDS command
=NOTE=  to assist in selecting the fields that can be used.
=NOTE=
=NOTE=  To build a report use the TAGS command to get started.
=NOTE=
=NOTE=  When done with the above, issue a  GENERATE or GEN command to initiate
=NOTE=  the substitution process based on the model statement(s).
=NOTE=
=NOTE=  For more information press PF1 or HELP.
000001 ALD '&profile' SECLABEL(REGFIN)
000002
000003
```

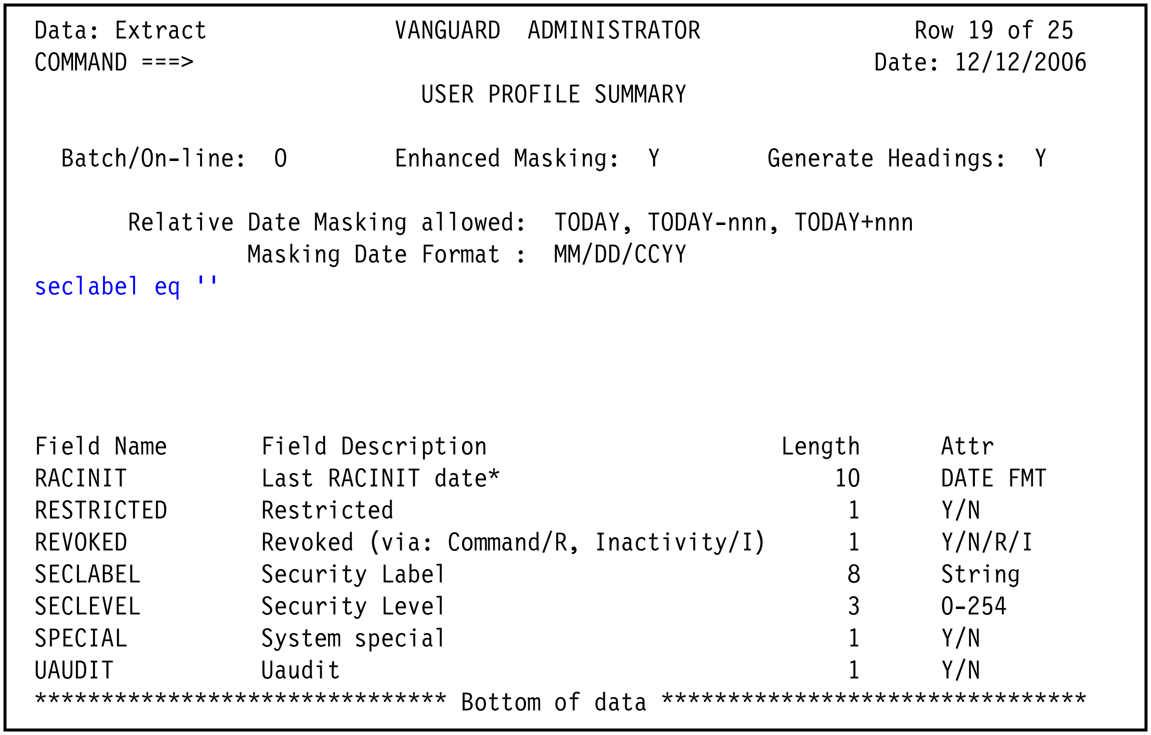
Figure 4-20   Generate ALD commands

Repeat this process for the various labels and profiles in your system.

4.6 Measuring our progress

As we progress through the process of defining our SECLABEL environment, we need to periodically check our progress. Using the power of enhanced masking, we can report on what profiles in our system have not yet been assigned a SECABEL value.

As an example, we generate a report showing which USER profiles do not have a SECLABEL assigned. We select option 3.1 from the main menu and use enhanced masking to specify our selection criteria (Figure 4-21).



```
Data: Extract              VANGUARD  ADMINISTRATOR                Row 19 of 25
COMMAND ===>                                                   Date: 12/12/2006
                             USER PROFILE SUMMARY

  Batch/On-line:  O         Enhanced Masking:  Y         Generate Headings:  Y

       Relative Date Masking allowed:  TODAY, TODAY-nnn, TODAY+nnn
                Masking Date Format :  MM/DD/CCYY
seclabel eq ''




Field Name       Field Description                      Length      Attr
RACINIT          Last RACINIT date*                         10      DATE FMT
RESTRICTED       Restricted                                  1      Y/N
REVOKED          Revoked (via: Command/R, Inactivity/I)      1      Y/N/R/I
SECLABEL         Security Label                              8      String
SECLEVEL         Security Level                              3      0-254
SPECIAL          System special                              1      Y/N
UAUDIT           Uaudit                                      1      Y/N
******************************* Bottom of data ********************************
```

Figure 4-21 Looking for USER profiles without SECLABEL

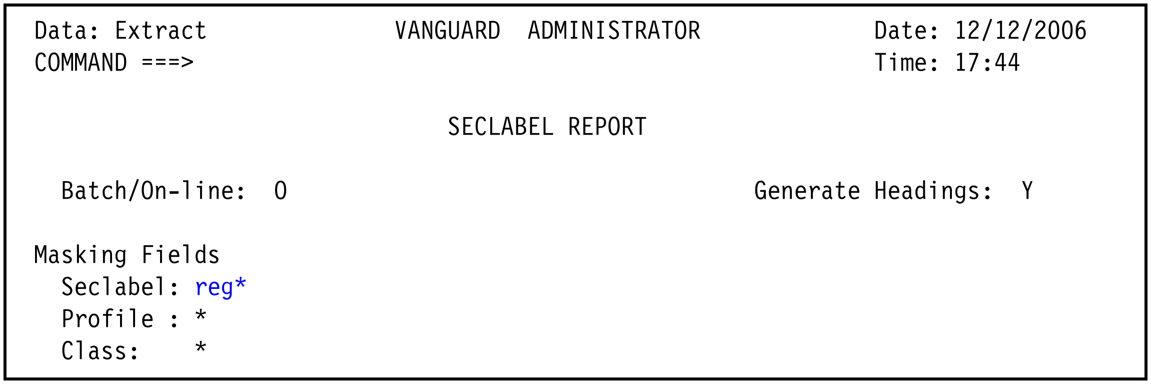
This returns a listing of all users that have not been assigned a default SECLABEL value. We can use this to ensure that we have all users properly defined. You can generate similar reports for the various data sets and general resource classes. This process provides us with the information we need to analyze as we move toward the ultimate goal of activating the SECLABEL class.

4.7 Reviewing SECLABEL use

Now that we defined our environment and activated SECLABEL protection, we need to periodically review what we have in place. In addition, we might need to provide information to our auditors to show how that we implemented MLS.

Vanguard Administrator has a simple report that provides you with a concise list of where the various SECLABELs are in use. Select option 3.18 from the main menu.

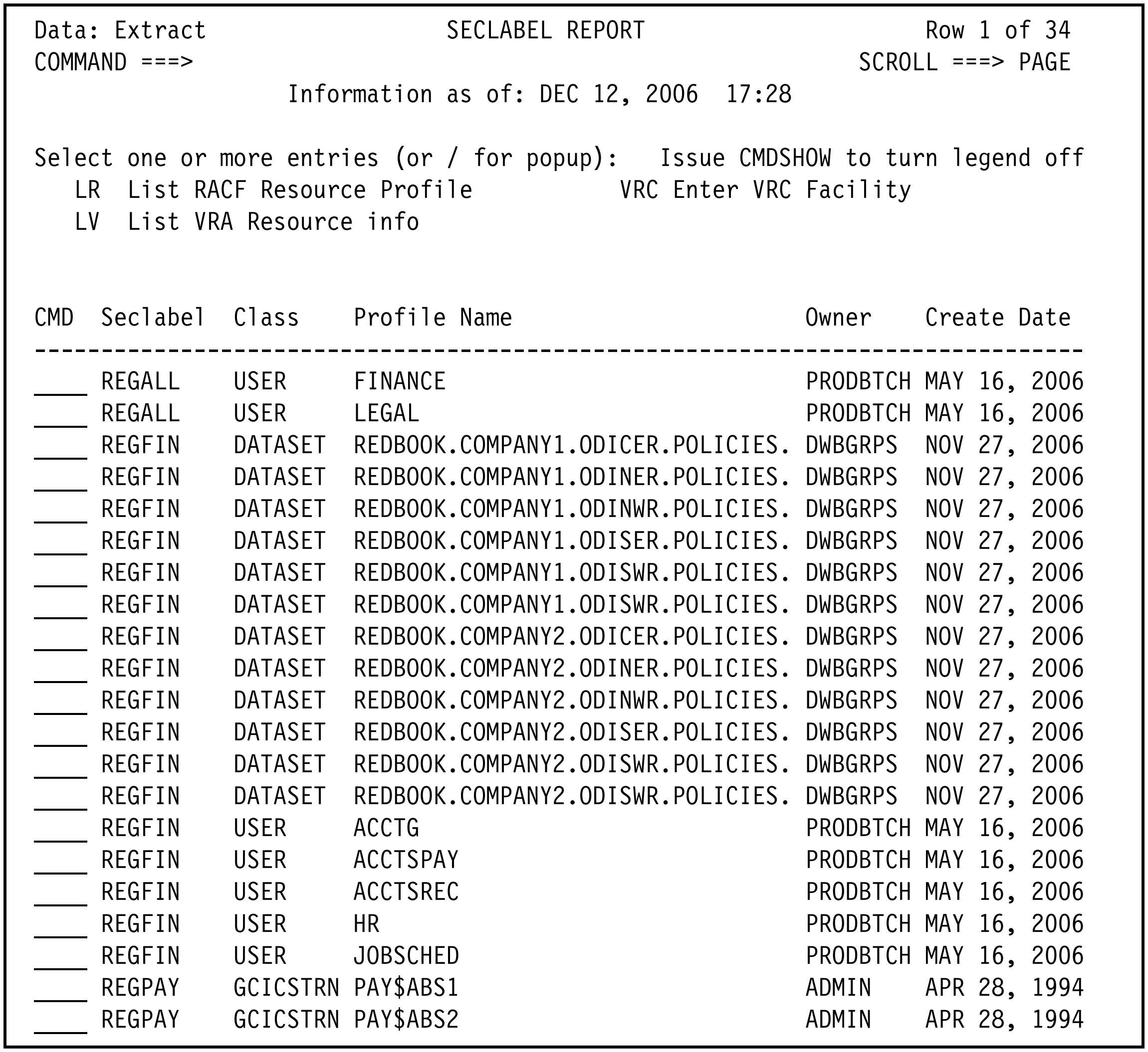
In our example, we ask for a report of all uses of those SECLABELs that we defined at a REGULATORY sensitivity level by specifying REG* as our selection criteria (Figure 4-22).



```
Data: Extract              VANGUARD  ADMINISTRATOR            Date: 12/12/2006
COMMAND ===>                                                  Time: 17:44

                                  SECLABEL REPORT

  Batch/On-line:  O                                   Generate Headings:  Y

Masking Fields
  Seclabel: reg*
  Profile : *
  Class:    *
```

Figure 4-22 Generate a SECLABEL report

Figure 4-23 shows the resulting report.



```
Data: Extract                  SECLABEL REPORT                     Row 1 of 34
COMMAND ===>                                                  SCROLL ===> PAGE
                   Information as of: DEC 12, 2006  17:28

Select one or more entries (or / for popup):   Issue CMDSHOW to turn legend off
   LR  List RACF Resource Profile           VRC Enter VRC Facility
   LV  List VRA Resource info


CMD  Seclabel  Class    Profile Name                      Owner    Create Date
-------------------------------------------------------------------------------
____ REGALL    USER     FINANCE                           PRODBTCH MAY 16, 2006
____ REGALL    USER     LEGAL                             PRODBTCH MAY 16, 2006
____ REGFIN    DATASET  REDBOOK.COMPANY1.ODICER.POLICIES. DWBGRPS  NOV 27, 2006
____ REGFIN    DATASET  REDBOOK.COMPANY1.ODINER.POLICIES. DWBGRPS  NOV 27, 2006
____ REGFIN    DATASET  REDBOOK.COMPANY1.ODINWR.POLICIES. DWBGRPS  NOV 27, 2006
____ REGFIN    DATASET  REDBOOK.COMPANY1.ODISER.POLICIES. DWBGRPS  NOV 27, 2006
____ REGFIN    DATASET  REDBOOK.COMPANY1.ODISWR.POLICIES. DWBGRPS  NOV 27, 2006
____ REGFIN    DATASET  REDBOOK.COMPANY1.ODISWR.POLICIES. DWBGRPS  NOV 27, 2006
____ REGFIN    DATASET  REDBOOK.COMPANY2.ODICER.POLICIES. DWBGRPS  NOV 27, 2006
____ REGFIN    DATASET  REDBOOK.COMPANY2.ODINER.POLICIES. DWBGRPS  NOV 27, 2006
____ REGFIN    DATASET  REDBOOK.COMPANY2.ODINWR.POLICIES. DWBGRPS  NOV 27, 2006
____ REGFIN    DATASET  REDBOOK.COMPANY2.ODISER.POLICIES. DWBGRPS  NOV 27, 2006
____ REGFIN    DATASET  REDBOOK.COMPANY2.ODISWR.POLICIES. DWBGRPS  NOV 27, 2006
____ REGFIN    DATASET  REDBOOK.COMPANY2.ODISWR.POLICIES. DWBGRPS  NOV 27, 2006
____ REGFIN    USER     ACCTG                             PRODBTCH MAY 16, 2006
____ REGFIN    USER     ACCTSPAY                          PRODBTCH MAY 16, 2006
____ REGFIN    USER     ACCTSREC                          PRODBTCH MAY 16, 2006
____ REGFIN    USER     HR                                PRODBTCH MAY 16, 2006
____ REGFIN    USER     JOBSCHED                          PRODBTCH MAY 16, 2006
____ REGPAY    GCICSTRN PAY$ABS1                          ADMIN    APR 28, 1994
____ REGPAY    GCICSTRN PAY$ABS2                          ADMIN    APR 28, 1994
```

Figure 4-23 Sample SECLABEL report

4.8 Summary

This chapter described how using the Vanguard Administrator can help you efficiently define the components you need to implement multilevel security in your organization. With these tools, you can concentrate your efforts in the planning and design of an effective implementation and leverage the functionality to perform the repetitive command generation task.

For you more information about the Vanguard suite of tools, go to the Web site:

http://www.go2vanguard.com

Or contact Vanguard at 702-734-0014.

5

MLS as applied to TCP/IP communications

This chapter provides a simple example of how to set up and use MLS to control access to TCP/IP resources and the data flowing through these resources.

Use the following guides as references:

- *z/OS Communications Server: IP Configuration Guide*, SC31-8775
- *z/OS Communications Server: IP Configuration Reference*, SC31-8776

The environment definitions to achieve MLS networking in z/OS are mainly based on the use of RACF profiles in the SERVAUTH class. These profiles were already available before extending the MLS support to TCP/IP resources at z/OS V1.5. z/OS V1.5 brought the capability of associating a security label and a port of entry to these resources.

5.1 z/OS TCP/IP and the SERVAUTH class

Protect TCP/IP resources such as stacks, ports, and networks by using profiles in the RACF SERVAUTH class. TCP/IP then acts as the resource manager, which uses the SAF interface to have the requestor's permission to the resource evaluated by RACF.

With MLS, we are interested in the profiles in the SERVAUTH class that protect access from, and optionally to, a network and to a local z/OS TCP/IP stack. They have not been implemented for MLS only, and we look first, as an introduction, at their initial intended use. Then we see how they can be used to contribute to the establishment of a multilevel security environment for z/OS TCP/IP communications.

Use of SERVAUTH profiles in a non-MLS environment

The SERVAUTH class of profiles was introduced in OS/390 V2.6 as a means of protecting resources managed by the OS/390 TCP/IP stack. The range of resources protected in this class has been extended with the further releases of OS/390 and z/OS, and includes, as of z/OS V1.8 today, the following resources:

- The TCP/IP stacks running in the z/OS image
- The networks to which the z/OS image has connectivity
- The ports reserved for permitted applications
- The **`Netstat`** command and its options
- The policy agent **`pasearch`** command
- The FTP SITE DUMP AND DEBUG commands
- Usage of SNMP subagents that connect to the TCP/IP SNMP agent
- The MODDVIPA utility
- The Fast Response Cache Accelerator (FRCA)
- TCP connection information service intended for network management applications and the selection of SMF records
- The TCP/IP packet trace service access control
- The HFS as accessed by FTP

Again, we focus here on the protection of TCP/IP stacks and networks.

5.1.1 Stack access control

Access to a given TCP/IP stack (remember that z/OS supports up to eight concurrent stacks) is controlled with a profile in the SERVAUTH class. The profile's name has the following form:

```
EZB.STACKACCESS.sysname.tcpproc
```

Where EZB.STACKACCESS is a fixed value, sysname is the value of the system static symbol &SYSNAME, and tcpproc is the name of the protected stack started procedure. See Figure 5-1 on page 88 for an example. In this example, two stacks are being protected with STACKACCESS profiles: The stack started with an STC called TCPIPA and the stack started with STC TCPIPB. Both profiles are defined with a UACC none, and users with the RACF user ID USER1 and USER2 are eventually given READ access to these resources. In this example, the application running with the user ID USER2 is denied access to the stack TCPIPA, because only USER1 is granted access to this stack.

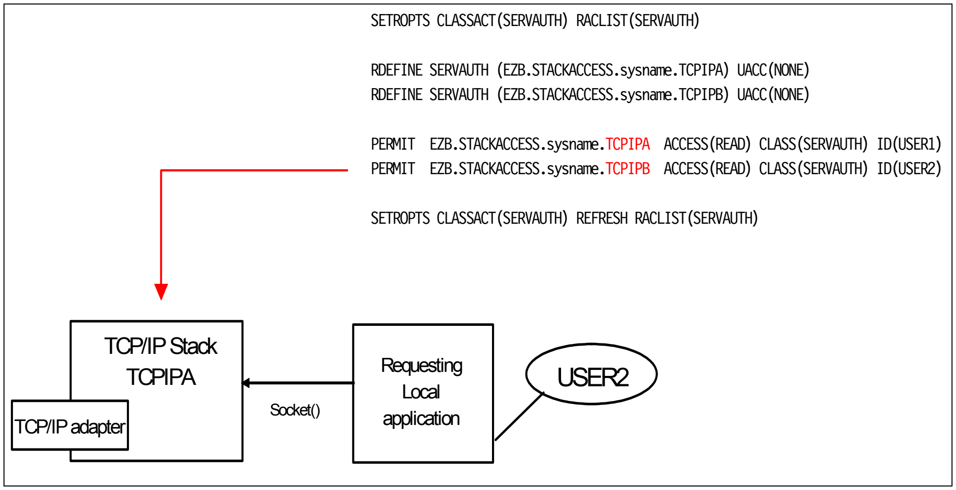
TCP/IP acts as a resource manager and requests SAF, provided that the SERVAUTH class is active, to check for permission of an application to issue the socket() call to the stack. The user ID considered for access control checking is:

- For a TSO user: The TSO user ID.
- For an OMVS shell user: The effective user ID of the process (this is normally the same user ID as the logged-on user, but can also have been changed by the setuid() or seteuid() functions.
- For a batch job: The user ID associated with the batch job through USER= in the job JCL or by RACF.
- For processes started from the MVS console: The user ID associated with the procedure by the STARTED class of profiles in the RACF database.

A RACF message on the system console shows a failed authorization. The application's behavior remains application dependant.

Note: TCP/IP internal tasks (except TN3270 with NACUSERID configured), VTAM, and the UNIX System Services kernel are exempt, by design, from STACKACCESS checking.

Note: All programs that need to access the stack (including utilities such as **`Netstat`** and **`ping`**) must have a user ID permitted to the STACKACCESS profile, if one has been defined.



```
SETROPTS CLASSACT(SERVAUTH) RACLIST(SERVAUTH)

RDEFINE SERVAUTH (EZB.STACKACCESS.sysname.TCPIPA) UACC(NONE)
RDEFINE SERVAUTH (EZB.STACKACCESS.sysname.TCPIPB) UACC(NONE)

PERMIT  EZB.STACKACCESS.sysname.TCPIPA  ACCESS(READ) CLASS(SERVAUTH) ID(USER1)
PERMIT  EZB.STACKACCESS.sysname.TCPIPB  ACCESS(READ) CLASS(SERVAUTH) ID(USER2)

SETROPTS CLASSACT(SERVAUTH) REFRESH RACLIST(SERVAUTH)
```

Figure 5-1 Example of STACKACCESS setup

MLS support

What we just described is the DAC piece of stack access control. For MAC to be enforced by the TCP/IP stack, give the RACF STACKACCESS profile a security label using the following RACF command:

```
RDEFINE SERVAUTH (EZB.STACKACCESS.sysname.tcpproc) +
SECLABEL(seclabel_name) UACC(READ)
```

The user ID the stack is running under must have this SECLABEL defined as its default SECLABEL and must be permitted to READ this SECLABEL profile.

Note that the universal access granted here in the STACKACCESS profile is READ, meaning that only MAC is to be enforced. It can be any other value if DAC enforcement is also needed to be subsequently in effect.

5.1.2 Network access control

Network access control implements the concept of *security zones*. A network security zone is an administrative name for a collection of systems that require the same access control policy. IP addresses are used to map systems into security zones.

The capability of sending packets to, or receiving packets from, a network security zone is controlled with NETACCESS profiles in the RACF SERVAUTH class. The TCP/IP stack checks that local applications are authorized to exchange information with these networks on the basis of the permission of their user ID to these profiles.

Definition of the network security zones

A NETACCESS statement and an ENDNETACCESS statement specify which network accesses have to be protected by RACF in the TCPIP.PROFILE data set. Here is an example of such a specification:

```
NETACCESS INBOUND OUTBOUND
192.168.0.0/ 16 SUBNET1 ; Subnet address
192.168.113.19/ 32 HOST1 ; Specific host address
192.168. 113.0 255.255.255.0 SUBNET2 ; Subnet address
192.168. 112.0 255.255.248.0 SUBNET3 ; Subnet address
DEFAULT 0 DEFZONE ; Optional Default zone
ENDNETACCESS
```

In these definitions, network security zones are specified as IP addresses and subnet masks and are given a SAF resource_name: SUBNET1, HOST1, SUBNET2, and so on. The DEFAULT keyword specifies any other IP addresses not explicitly specified in these statements, and in this example, is linked to the SAF resource_name DEFZONE. The SAF resource_name is mapped to profiles in the SERVAUTH class with profile names of the form:

```
EZB. NETACCESS. sysname. tcpname. saf_resname
```

Tip: Do not confuse the NETACCESS *statements* in the PROFILE.TCPIP and the EZB.NETACCESS *profiles* in the RACF database.

TCP/IP checks that the application that is to access the network to send or to receive data has READ permission to the corresponding EZB.NETACCESS RACF profile.

Note: These statements can be dynamically established at a TCP/IP stack using an OBEYFILE. The TCPIP.PROFILE data set can then be edited for permanent statements.

Note: The IP address mapped to a security zone can be a physical adapter address or a VIPA.

You can use the INBOUND and OUTBOUND keywords in the NETACCESS statement to indicate whether the network IP address being controlled is to be considered as a destination or origin IP address, or both, the default being NOINBOUND OUTBOUND.

Access to networks specified with OUTBOUND is controlled when the application calls the following services:

- TCP connect, write
- UDP connect, write
- RAW connect, write

Access to networks specified with INBOUND is controlled when the application calls for:

- TCP accept, read
- UDP read
- RAW read

Some notes about the OUTBOUND and INBOUND conditions follow.

OUTBOUND

Some applications, such FTP, start a new instance running under the authenticated user ID of the requesting user. Network access control is then exercised using this user ID when the new FTP instance communicates with the client.

INBOUND

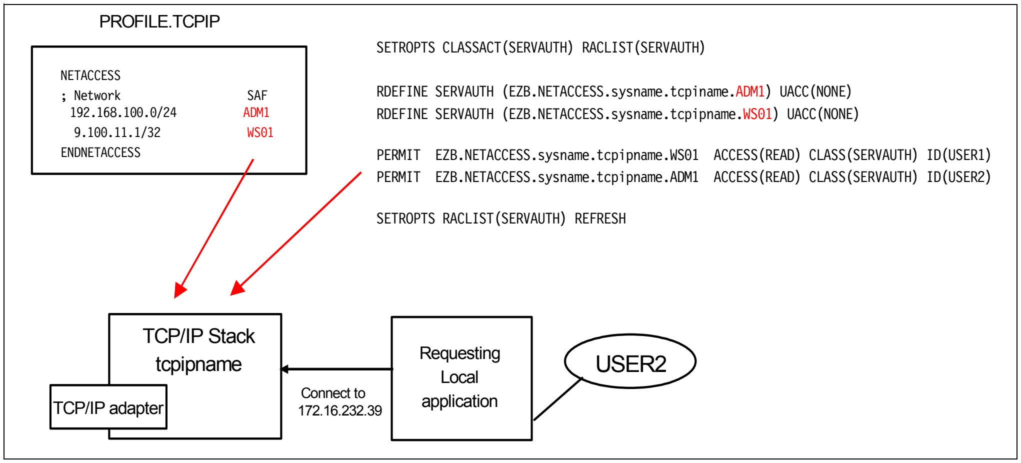
This is intended for applications that are to transfer the received data to another network, not actually sending data back to the original requestor. INBOUND sets control on which networks such an application can receive data from. It also controls an application's ability to bind() to a local IP address.

Pending TCP connections from unauthorized source addresses are silently reset and discarded during the accept processing. RAW and UDP datagrams from an unauthorized source address are silently discarded during the read processing.

The application, if not permitted to the network security zone, receives the following error code, which might show up in its log or trace:

```
ERRNO = 111 ñ 29452 (JRNetAccessDenied - Userid is not authorized to
access the network)
```

Figure 5-2 shows an example of NETACCESS setup.



```
NETACCESS
 ; Network              SAF
   192.168.100.0/24     ADM1
    9.100.11.1/32       WS01
ENDNETACCESS
```

```
SETROPTS CLASSACT(SERVAUTH) RACLIST(SERVAUTH)

RDEFINE SERVAUTH (EZB.NETACCESS.sysname.tcpiname.ADM1) UACC(NONE)
RDEFINE SERVAUTH (EZB.NETACCESS.sysname.tcpipname.WS01) UACC(NONE)

PERMIT  EZB.NETACCESS.sysname.tcpipname.WS01  ACCESS(READ) CLASS(SERVAUTH) ID(USER1)
PERMIT  EZB.NETACCESS.sysname.tcpipname.ADM1  ACCESS(READ) CLASS(SERVAUTH) ID(USER2)

SETROPTS RACLIST(SERVAUTH) REFRESH
```

Figure 5-2 Example of NETACCESS setup

Exempt applications

The network access control user ID is based on the application's address space information. TCP/IP is an exempt user and has access to all zones. Because Telnet runs in the TCP/IP address space, Telnet processes are also exempt from network access control based on address space user ID information, unless the TN3270 ports have been defined with NACUSERID.

MLS support

What has been described earlier is the DAC piece of network access. For MAC to be enforced by the TCP/IP stack, give the RACF NETACCESS profile a security label using the following RACF command:

```
RDEFINE SERVAUTH (EZB.NETACCESS.sysname.tcpproc) +
SECLABEL(seclabel_name) UACC(READ)
```

> **Important:** if no INBOUND and OUTBOUND keywords are stated, the default is NOINBOUND OUTBOUND. INBOUND OUTBOUND *must* be specified on MLS systems.

5.1.3 The notion of port of entry (POE)

As the name implies, the port of entry (POE) is an indication of how the request has entered the system, which RACF can exploit in its decision making by considering the POE as part of its permission criteria. A typical example of this is the TERMINAL class that can be used to identify a POE eventually considered for conditional access, as shown in the following example:

```
PERMIT SUBMIT.*.PAYROLL*.* CLASS(JESJOBS) ID(USER01) ACCESS(READ)
WHEN(TERMINAL(terminal-ID))
```

In this example, USER01 is permitted to submit the job only if USER01 entered the system through the terminal designated by terminal_ID.

As of z/OS V1.5, the POE can be the partner security zone, represented by the name of a corresponding NETACCESS profile in the SERVAUTH class. This can be taken into account for conditional access, such as:

```
PERMIT ‘MLS4.TEST.L3C’ CLASS(DATASET) ID(USER01) ACCESS(READ) +
WHEN(SERVAUTH(netaccess_profile))
```

The port of entry can also be used to associate a SECLABEL to the process or thread instantiated to serve a user request, such as when a user logs in to the z/OS FTP server. When FTP.DATA contains the following statement and the client’s IP address is mapped to a NETACCESS security zone, the NETACCESS profile is remembered as the POE:

```
PORTOFENTRY4 SERVAUTH
```

If MAC is enforced and the FTP user is permitted to the SECLABEL specified for the client’s security zone, this very SECLABEL is associated to the request. We demonstrate this in the example in 5.3.2, “Our test” on page 103.

Note: POE support is provided to applications or resource managers where the work entered the system with the following functions:

- __poe() or calls BPX1POE or BPX4POE prior to changing to client's login identity.
- SIOCGSOCKPOEATTRS ioctl and then use that information on RACROUTE VERIFY.

As of today, this is performed in FTPD, INETD (it covers all the daemons forked by INETD), and DB2 V8.

5.2 The MLS networking environment

In the networking environment, the information being protected is the data being read and written through sockets. Sockets are opened and used by applications running under user IDs. In a z/OS multilevel secure environment:

- Each user ID is permitted to use one or more security labels.
- Every job or login session is associated with a user ID.
- A user ID can use only one security label for each job or login session.
- The security label used must be limited by the port of entry (source type and location) of the job or login session. In other words, it is up to the application providing the service to ensure that, if running with the SYSMULTI security label, it properly associates the POE security label to any access done on behalf of the client.

The conceptual flow of information, as seen from each side of a TCP/IP communication, is depicted in Figure 5-3 on page 94. In order for the data to flow both ways (that is, in read and write mode), all the security labels encountered along the data path must be equivalent. These are the security labels of:

- The stored data, as defined in the RACF data set profile or the file security packet. Note that security label equivalency is not a requirement here, because you might precisely want to exploit the no read-up and no write-down properties implied by MLS dominance.
- The application reading or writing the data, which is running under the security label associated to its user ID or the POE.
- The TCP/IP stack hosting the local socket. z/OS introduces the concept of restricted and unrestricted stacks. We explain this concept later in this chapter.
- The packet’s origin and destination addresses, with an explicit packet tag as explained in “Security label of the origin and destination addresses” on page 94, or implicitly assumed through network security zones.
- The partner application.

Note that the exchanged packets can flow though intermediate networks through routing devices, which do not affect the origin and destination addresses in the packets. However, these intermediate devices might themselves generate packets intended for either end of the communication. These packets must also be associated with a security label in an MLS environment by defining security zones to map the subnetworks in which these devices are.

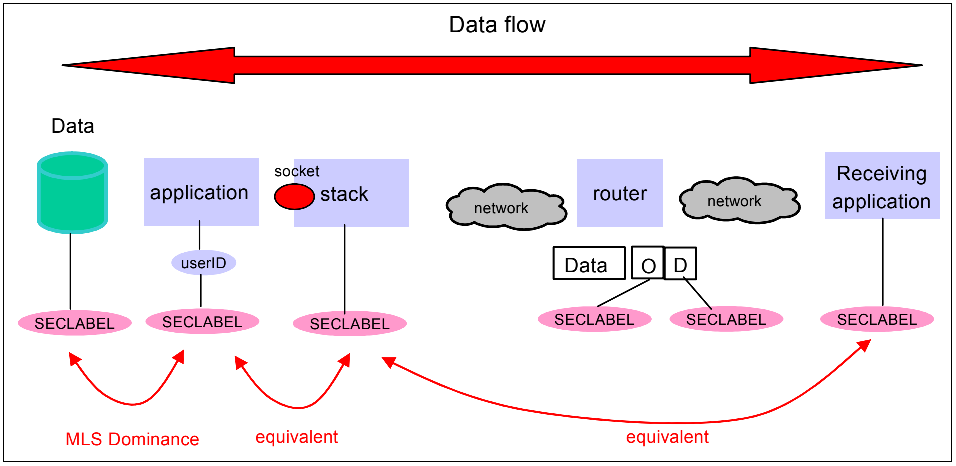


Figure 5-3 MLS network information flow

Security label of the origin and destination addresses

In this section, we discuss the security label of the origin and destination addresses.

Packet tagging

In order to indicate the security labels of an IP packet's origin and destination addresses, IBM developed a proprietary packet tag that is generated and exploited by the z/OS stacks in the following situations:

- The communicating stacks are located in the same z/OS image (environment variable IUTSAMEHOST).
- They are located in the same sysplex and communicate with IP over XCF, and the local and remote security zones have both the SYSMULTI security labels.

Note: During our trials, we experienced what seemed at first to be a problem, but was actually working as designed: The receiving MLS systems had both its local and remote security zones with the SYSMULTI security label. It was not answering to our client system and not located in the same sysplex or same z/OS image. The MLS system was expecting to receive packet tags along with IP packets, because of these SYSMULTI definitions, and was silently discarding the packets because there were not any.

Security zone security labels

When the conditions required to generate packet tags are not met, the security labels of the origin and destination addresses are the ones specified in the SERVAUTH NETACCESS profiles. Profiles are defined for the security zones corresponding to local IP addresses, and security zones corresponding to the remote partners addresses. The stack then assumes that these security labels pertain to origin and destination addresses. Figure 5-4 on page 96 shows this.

The POE security label is inherited from the security label of the security zone to which the partner's IP address is mapped, as specified in the RACF command:

```
RDEFINE SERVAUTH (EZB.NETACCESS.sysname.tcpiname.subnet_saf_name) +
SECLABEL(seclabel_name) UACC(READ)
```

In Figure 5-4 on page 96, the network, as seen by the z/OS TCP/IP stacks, has been defined as four security zones: A, B, C, and D (that is, sets of IP addresses covered by a common access policy). From the MLS standpoint, IP addresses in zone B are valid destinations for packets containing data with an equivalent security label to SECLABLB, or packets issued from these addresses have to contain data with an equivalent security label to SECLABLB. One z/OS stack, called a *restricted* stack, has its own IP addresses mapped to security zone D with SECLABLD, implying that data can only be exchanged with zones with an equivalent security label to SECLABLD. The other stack, the *unrestricted* stack, does not have any restriction with the zones with which it can communicate because its security label is SYSMULTI. We explain restricted and unrestricted stacks later in this chapter.

Security zone A is a special case because it designates a trusted internal subnet, properly protected from the other networks by firewalls, with a security label of SYSHIGH, implying that packets issued in this subnet can only be exploited through a highly trusted data path with the SYSHIGH security label (or SYSMULTI, but this is not recommended from a security standpoint).

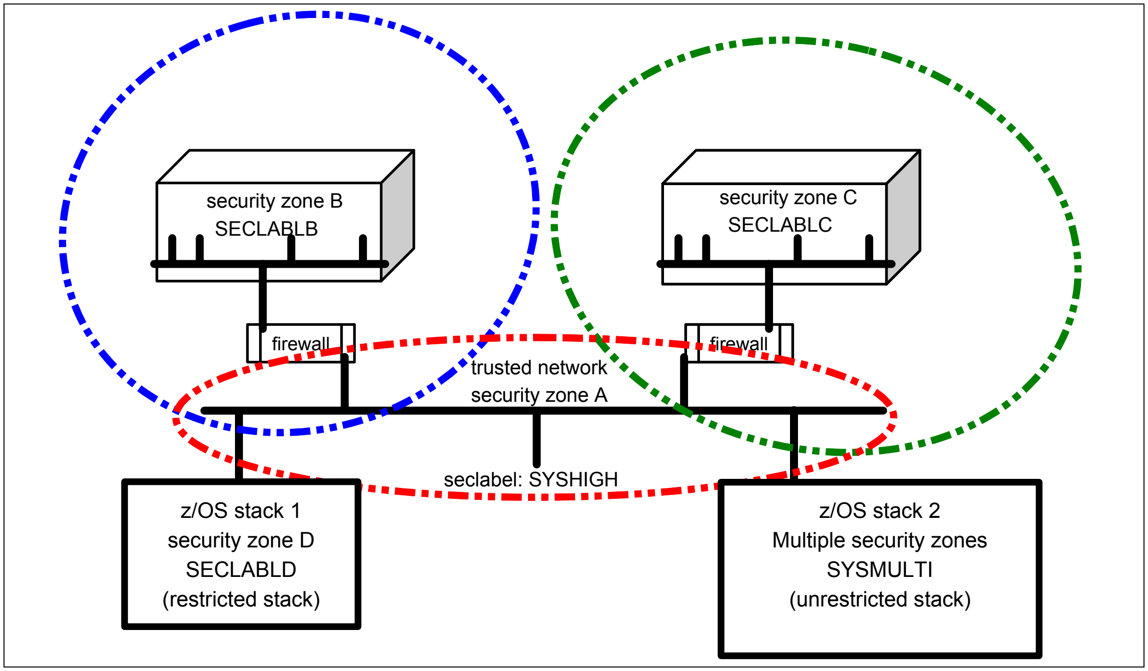


Figure 5-4 Assigning security zones with security labels

MLS enforcement and the SERVAUTH profiles

The MLS environment is enforced when the SECLABEL class and the MLACTIVE options are active in RACF. That is, applications and resources are given specific security labels and mandatory access control is applied accordingly. Note that:

- The profiles in the SERVAUTH class must have a SECLABEL when SETR MLACTIVE.
- The class is defined in the CDT with EQUALMAC (that is, MAC can be granted only for equivalent SECLABELs between the accessor and the resource).

A few more definitions

Now we review a few more definitions.

Nonsecure systems

Most systems do not support mandatory access control processing as z/OS does. The security label of data flowing from and to these systems must be assumed. If these systems are not physically managed, they normally must not participate in a multilevel secure network. Some installations might need to permit them to participate. In these cases, we recommend that they be assigned a security label, as seen from MAC supporting systems (using the proper network security zone with z/OS) with the lowest security level and a single security category that is not common with any other security label.

Managed systems or single-level security (SLS) systems

Systems that do not support mandatory access control processing can participate in a multilevel secure network if they are physically managed to guarantee that all information on the system has the same single security label and all users of the system are permitted to that security label. These systems are referred to as single-level security (SLS) or managed systems in this chapter. This management requires both physical control of the systems and careful management of the network. Managed systems must be prevented from communicating with other managed systems that do not have equivalent security labels. Systems that support mandatory access control and are configured to implicitly associate the correct security label with each managed system can also communicate with managed systems. The systems that perform mandatory access control are responsible for ensuring that only information from applications with an equivalent security label is sent to a managed system, and that information received from a managed system is given only to applications with an equivalent security label.

As mentioned, physical control must be extended to the networking environment so that:

- All interfaces on the subnet have the same security label.
- All packets generated within the subnet have a source address from the subnet.
- IP source routing options are not present.
- Communication is limited to other SLS subnets with equivalent security labels or to MLS systems.

Multilevel secure systems

Some systems in the network provide multilevel secure environments. These systems have mechanisms to associate security labels with information accessed through the system and with users logged into the system. In z/OS, this translates into the use of packet tagging or network security zones.

The system enforces mandatory access control policies to ensure proper separation of information. The packets being sent from a single IP address on the multilevel secure system might have originated from applications running under different security labels. Applications on a multilevel secure system can securely communicate with applications on a managed system.

Mandatory access control enforcement occurs only on the multilevel secure system. The multilevel secure system is responsible for ensuring that it sends only information from an application with an equivalent security label to any managed system. It also is responsible for ensuring that information received from a managed system is delivered only to an application with an equivalent security label. When two applications on multilevel secure systems communicate, the security label of the sending application must be communicated to the receiving system so that the receiving system can enforce mandatory access control prior to delivering the information to an application.

Restricted stack

In the case of a restricted stack:

- The stack runs under a specific security label (as opposed to a system-defined security label) it inherits from its started task user ID.
- It accepts only user IDs with an equivalent label to use the stack, because the stack security label is also specified in a corresponding SERVAUTH STACKACCESS profile.
- The stack accepts packets with the IBM tag only if they have an equivalent security label.
- In the case of a missing IBM tag on the received packets (that is, packets coming from another TCP/IP host not located in the same z/OS image and not using IP over XCF), the packet inherits the security label of the NETACESS profile that maps the origin subnet of the packet and must be of an equivalent security label to the stack security label.
- Packets are sent by this stack only to subnets with an equivalent security label to the security label in the NETACCESS profiles that map these subnets.

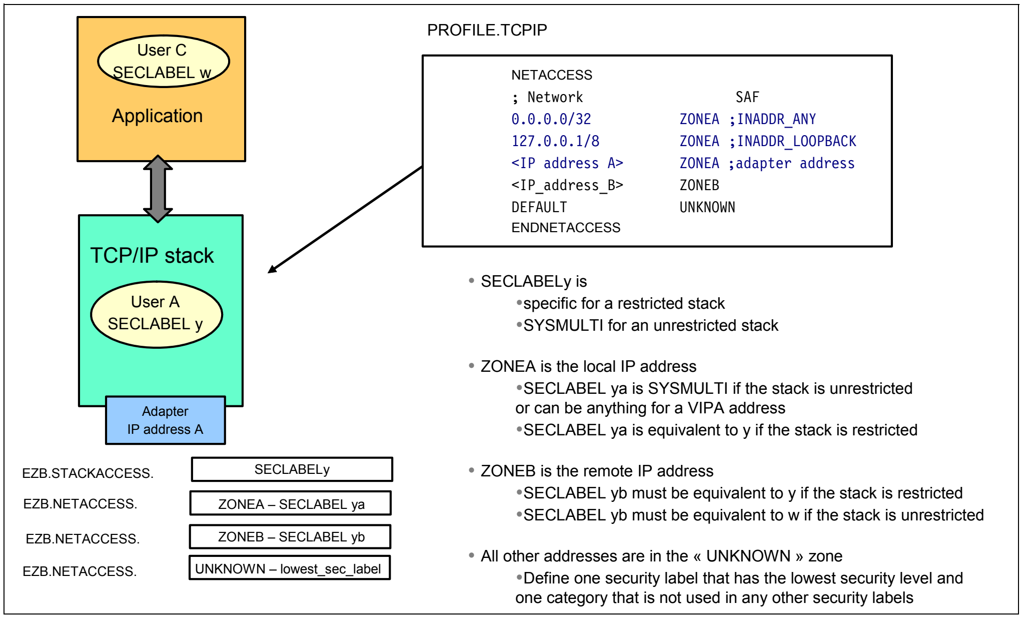
Because of all of this, a restricted stack behaves like a single-level security system to the other TCP/IP hosts in the MLS environment.

Unrestricted stack

For an unrestricted stack:

- The stack runs under a user ID with the SYSMULTI security label.
- The stack allows sockets to be opened by applications with any security label.
- Unrestricted stacks are permitted to define VIPAs in network security zones with security labels other than SYSMULTI. When one of these is used as either the source or destination IP address of a packet, it implicitly identifies the security label of the information.
- When both IP addresses in a packet are in security zones with the SYSMULTI security label, the application security label must be explicitly transmitted in the packet, as is the case with the IBM packet tag.

Figure 5-5 summarizes the setups of a restricted and an unrestricted stack. In this figure, the stack is running with the user ID A, which runs with security label y. The security zone A consists of the stack local IP addresses. We consider only one remote security zone, zone B.



```
NETACCESS
; Network                    SAF
0.0.0.0/32            ZONEA ;INADDR_ANY
127.0.0.1/8           ZONEA ;INADDR_LOOPBACK
<IP address A>        ZONEA ;adapter address
<IP_address_B>        ZONEB
DEFAULT               UNKNOWN
ENDNETACCESS
```

- SECLABELy is
  - specific for a restricted stack
  - SYSMULTI for an unrestricted stack
- ZONEA is the local IP address
  - SECLABEL ya is SYSMULTI if the stack is unrestricted or can be anything for a VIPA address
  - SECLABEL ya is equivalent to y if the stack is restricted
- ZONEB is the remote IP address
  - SECLABEL yb must be equivalent to y if the stack is restricted
  - SECLABEL yb must be equivalent to w if the stack is unrestricted
- All other addresses are in the « UNKNOWN » zone
  - Define one security label that has the lowest security level and one category that is not used in any other security labels

Figure 5-5 Restricted and unrestricted stacks

Note on the use of VIPAs

The SECLABELs specified for a given stack's IP addresses (including LOOPBACK and IN_ADDRANY) must be the same as the TCPIP job and STACKACCESS profile SECLABELs. The only exception is that VIPAs defined on a SYSMULTI stack can have any valid SECLABEL for that system image.

Exempt users

A SYSMULTI user with UPDATE authority to the EZB.STACKACCESS profile is exempt from the network access control restriction that all traffic must be with partners that are in security zones with security labels that are equivalent to the stack's security label or the security label associated with the local IP address.

We recommend that this authority be limited to the use of the programs that must be exempted. To do this, first specify UACC(READ) when defining the STACKACCESS profiles, and then grant conditional update access to each by specifying:

```
PERMIT stackaccess_profile_name CLASS(SERVAUTH) ID(*) ACCESS(UPDATE) -
WHEN(PROGRAM(ping,oping,tracert,otracert,omproute))
```

> **Note:** The WHEN(PROGRAM()) conditional access parameter is not supported on profiles in the SERVAUTH class, except where explicitly stated. PERMITs with WHEN(PROGRAM()) on other profiles might be ignored.

Stack recognition of a multilevel secure environment

What follows is fully available at z/OS V1.8, which was the level we used for this book. We learned that the z/OS V1.5 stack is much more restrictive about the sequence of events and does not tolerate some dynamic changes very well (such as cycling between SETROPTS NOMLACTIVE and SETROPTS MLACTIVE).

You can activate the SAF SECLABEL class and define security labels on SERVAUTH profiles. This causes the security server to enforce mandatory access control policies for those resources without fully activating a multilevel secure environment.

The z/OS Communications Server stack does not perform its extra mandatory access control policy enforcement until you issue the RACF command SETROPTS MLACTIVE. When a NETACCESS statement is encountered in TCPIP profile processing and MLACTIVE has been set, the stack activates extra mandatory access control policy enforcement in both restricted and unrestricted stacks, as follows:

- New sockets are allowed only if a STACKACCESS profile covers this stack.
- Network access is allowed only to IP addresses that are mapped into network security zones covered by NETACCESS profiles.
- Restricted stacks do not normally allow SYSMULTI tasks to have network access to security zones with security labels that are not equivalent to the stack’s security label.
- Unrestricted stacks transmit packet labels both internally and externally to enable an extra mandatory access control check between the sending task’s security label and the receiving task’s security label when both IP addresses are in security zones with a SYSMULTI security label.
- Distributing stacks consider security labels in choosing target applications. In-stack TN3270E servers consider security labels in mapping connections to LU names.

- Internal configuration consistency checks are performed whenever PROFILE.TCPIP or certain SERVAUTH class profile changes are made.

5.3 Setting up MLS for z/OS TCP/IP communications

In this section, we provide an example of an MLS network setup for TCP communications.

5.3.1 Our test configuration

Figure 5-6 on page 102 shows our test configuration. An MLS z/OS system provides FTP services to another z/OS host, configured as an SLS system, with IP address 10.6.6.81 and workstations located anywhere else in the 10.0.0.0 network. Figure 5-7 on page 102 shows the definitions of the security zone in the TCPIP.PROFILE data set. Note that the z/OS MLS system stack runs with the user ID TCPIP, which is associated with a security label SYSMULTI (an unrestricted stack).

The FTP server has been set up with PORTOFENTRY4 SERVAUTH and is invoked to transfer the data from and to the data set called MLS4.TEST.L3C. This data set is protected with the arbitrary L3C security label, the combination of security level 3 and category C. The FTP service is run, after proper user authentication, under the user ID MLS4, which is itself defined with the default security label L3C and is permitted to both security labels L3C and L4C. L4C, the combination of security level 4 and category C, dominates L3C.

The SERVAUTH profiles matching this configuration are defined as shown in Figure 5-8 on page 102. Requests sent from the client with IP address 10.6.6.81 are associated with the port of entry EZB.NETACCESS.TC7.TCPIP.SC60 with the security label L3C, while requests sent from other clients in the 10.0.0.0 network come through the port of entry EZB.NETACCESS.TC7.TCPIP.OTHERS with security label L4C.

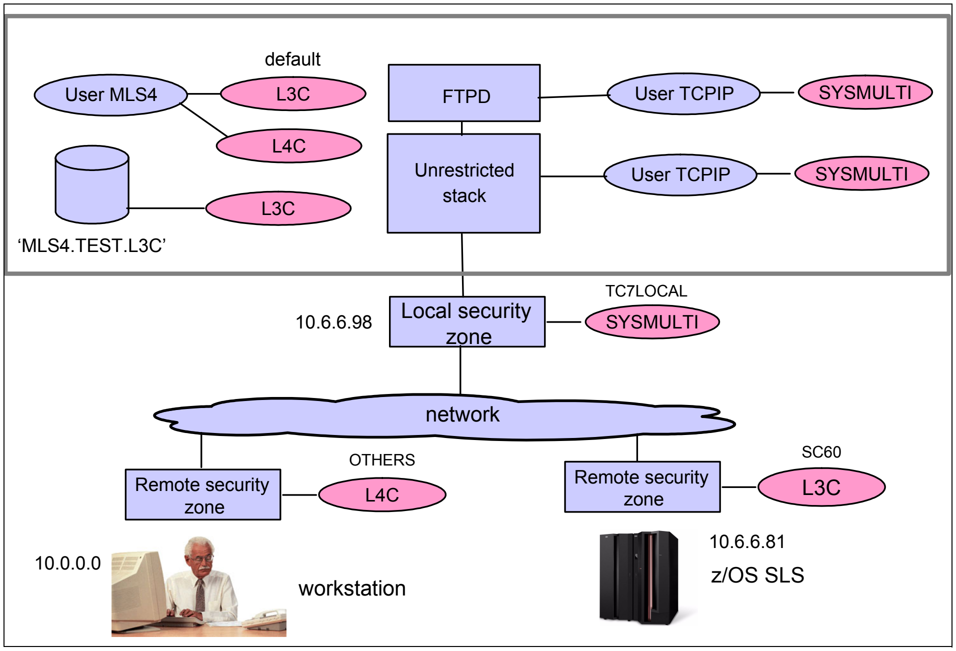


Figure 5-6 Our test configuration

```
NETACCESS INB OUTB
; Network                   SAF
 0.0.0.0/32                 STAKZONE ; INADDR_ANY
 127.0.0.1/8                STAKZONE ; INADDR_LOOPBACK
 10.6.6.98/32               TC7LOCAL ; system TC7local IP address
 10.6.6.81/32               SC60 ; system SC60 IP address
10.0.0.0/8 OTHERS ; other clients
 DEFAULT                    DEFAULT ; not 10.0.0.0 network
ENDNETACCESS
```

Figure 5-7 Security zones definition

```
RDEF SERVAUTH EZB.STACKACCESS.TC7.TCPIP UACC(READ) SECLABEL(SYSMULTI)
RDEF SERVAUTH EZB.NETACCESS.TC7.TCPIP.STAKZONE UACC(READ) SECLABEL(SYSMULTI)
RDEF SERVAUTH EZB.NETACCESS.TC7.TCPIP.TC7LOCAL UACC(READ) SECLABEL(SYSMULTI)
RDEF SERVAUTH EZB.NETACCESS.TC7.TCPIP.SC60 UACC(READ) SECLABEL(L3C)
RDEF SERVAUTH EZB.NETACCESS.TC7.TCPIP.OTHERS UACC(READ) SECLABEL(L4C)
RDEF SERVAUTH EZB.NETACCESS.TC7.TCPIP.DEFAULT UACC(READ) SECLABEL(SYSNONE)
```

Figure 5-8 SERVAUTH profiles

5.3.2 Our test

In this section, we discuss our test.

MLS enforcement

MLS enforcement was turned on with the following RACF commands:

```
SETR CLASSACT(SECLABEL)
SETR MLACTIVE(WARNING)
SETR MLS(FAILURE)
```

Stack recognition of the MLS environment

When running on an MLACTIVE system, the stack performs a multilevel security consistency check in any of the following situations:

- After initial profile processing
- After every VARY TCPIP,OBEYFILE command
- After SYSPLEX dynamic VIPA changes
- When it receives an ENF signal from RACF indicating that a RACLIST REFRESH was done for the SERVAUTH or SECLABEL class

Figure 5-9 on page 104 is an example of a failing MLS check. In this case, the NETACCESS statements and profiles setup were correct; however, the stack was started before we turned MLACTIVE on. The MLS check was triggered by a RACLIST REFRESH of SERVAUTH after we issued the SETR MLACTIVE. The stack is actually telling us that it did not perform the required MLS checking the last time it started.

```
EZD1215I MLSCHK STARTED STACK TCPIP USER TCPIP SECLABEL <NONE>
EZD1224I MLSCHK STACK SECLABEL <NONE> IS NOT VALID
EZD1221I MLSCHK STACKACCESS PROFILE HAS WRONG SECLABEL SYSMULTI 45
         RESNM EZB.STACKACCESS.TC7.TCPIP
         PRFNM EZB.STACKACCESS.TC7.TCPIP
EZD1223I MLSCHK LOCAL ZONE HAS INCORRECT SECLABEL SYSMULTI 457
         IPADR 10.6.6.98  IF OSA2CC0LNK
         ZONE  TC7LOCAL 10.6.6.98/32
         RESNM EZB.NETACCESS.TC7.TCPIP.TC7LOCAL
         PRFNM EZB.NETACCESS.TC7.TCPIP.TC7LOCAL
EZD1223I MLSCHK LOCAL ZONE HAS INCORRECT SECLABEL SYSMULTI 458
         IPADR 127.0.0.1  IF LOOPBACK
         ZONE  STAKZONE 127.0.0.0/8
         RESNM EZB.NETACCESS.TC7.TCPIP.STAKZONE
         PRFNM EZB.NETACCESS.TC7.TCPIP.STAKZONE
EZD1226I MLSCHK INADDR_ANY 0.0.0.0 HAS INCORRECT SECLABEL SYSMULTI
         ZONE  STAKZONE 0.0.0.0/32
RESNM EZB.NETACCESS.TC7.TCPIP.STAKZONE
         PRFNM EZB.NETACCESS.TC7.TCPIP.STAKZONE
EZD1217I MLSCHK FAILED STACK TCPIP - 5 MESSAGES WRITTEN TO JOBLOG
```

Figure 5-9 Failing the MLS check

Stopping and restarting the stack with MLACTIVE on showed that the MLS check passed successfully, as shown in Figure 5-10.

```
$HASP373 TCPIP    STARTED
IEE252I MEMBER CTIEZB00 FOUND IN SYS1.IBM.PARMLIB
IEE252I MEMBER CTIIDS00 FOUND IN SYS1.IBM.PARMLIB
EZZ7450I FFST SUBSYSTEM IS NOT INSTALLED
EZZ0300I OPENED PROFILE FILE DD:PROFILE
EZZ0309I PROFILE PROCESSING BEGINNING FOR DD:PROFILE
EZZ0316I PROFILE PROCESSING COMPLETE FOR FILE DD:PROFILE
EZD1215I MLSCHK STARTED STACK TCPIP USER TCPIP SECLABEL SYSMULTI
EZD1216I MLSCHK SUCCEEDED STACK TCPIP IS MULTILEVEL SECURE
EZZ0641I IP FORWARDING NOFWDMULTIPATH SUPPORT IS ENABLED
EZZ0350I SYSPLEX ROUTING SUPPORT IS ENABLED
EZZ0352I VARIABLE SUBNETTING SUPPORT IS ENABLED FOR OROUTED
...
```

Figure 5-10 Passing the MLS check

Reading and writing the data set from the OTHERS security zone

We had a workstation located in the 10.x.x.x subnetwork hosting the FTP client. The FTP requests were, therefore, identified at the MLS z/OS as coming from the

OTHERS security zone, and therefore, the POE was tagged with the L4C security label. We authenticated at the FTP server as user MLS4, who is permitted to SECLABEL L4C, and tried to read and write the ‘MLS4.TEST.L3C’ data set, which has SECLABEL L3C. Figure 5-11 shows the results as seen from the workstation. The read operation did succeed as the request was running with a security label (L4C) dominating the data set security label (L3C); however, for the same reason and because we had MLS(FAILURE) turned on, we failed the writing with the system console displaying the RACF message shown in Figure 5-12.

```
C:\Documents and Settings\kappeler>ftp 9.12.6.98
Connected to 10.6.6.98.
220-FTPD1 IBM FTP CS V1R6 at TC7.ITSO.IBM.COM, 18:20:
220 Connection will close if idle for more than 5 min
User (10.6.6.98:(none)): mls4
331 Send password please.
Password:
230 MLS4 is logged on.  Working directory is "MLS4.".
ftp> get 'mls4.test.l3c' c:\transfer\test.data
200 Port request OK.
125 Sending data set MLS4.TEST.L3C
250 Transfer completed successfully.
ftp: 25915 bytes received in 4,61Seconds 5,63Kbytes/s
ftp> put' c:\transfer\test.data 'mls4.test.l3c'
Invalid command.
ftp> put c:\transfer\test.data 'mls4.test.l3c'
200 Port request OK.
550 STOR fails: MLS4.TEST.L3C.  User not authorized.
```

Figure 5-11   FTP client in the OTHERS security zone

```
ICH408I USER(MLS4    ) GROUP(SYS1    ) NAME(MLS4
  MLS4.TEST.L3C CL(DATASET ) VOL(TC7TS1)
  INSUFFICIENT SECURITY LABEL AUTHORITY
  ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
```

Figure 5-12   No write-down RACF message

Note: We did not use the REPLYSECURITYLEVEL statement in FTP.DATA here. A REPLYSECURITYLEVEL 1 statement would have resulted in hiding IP addresses, host names, port numbers, or server operating system level information in FTP replies.

Reading and writing the data set from the SC60 security zone

When using the FTP client in the z/OS SLS system located at 10.6.6.81, the request is identified as coming from the SC60 security zone with SECLABEL L3C. After the proper authentication of the MLS4 user at the FTP server, reading and writing the 'MLS4.TEST.L3C' data set succeeds, because the POE is associated to SECLABEL L3C. See Figure 5-13.

```
EZA1450I IBM FTP CS V1R6
EZA1554I Connecting to:   10.6.6.98 port: 21.
220-FTPD1 IBM FTP CS V1R6 at TC7.ITSO.IBM.COM, 18:23:54 on 2005-02-13.
220 Connection will close if idle for more than 5 minutes.
 EZA1459I NAME (10.6.6.98:MLS6):
mls4
 EZA1701I >>> USER mls4
 331 Send password please.
 EZA1789I PASSWORD:


 EZA1701I >>> PASS
 230 MLS4 is logged on.  Working directory is "MLS4.".
 EZA1460I Command:
get 'mls4.test.l3c' 'mls4.test.sc60' (replace
 EZA1701I >>> PORT 9,12,6,81,4,41
 200 Port request OK.
 EZA1701I >>> RETR 'mls4.test.l3c'
 125 Sending data set MLS4.TEST.L3C
 250 Transfer completed successfully.
 EZA1617I 25915 bytes transferred in 0.060 seconds.  Transfer rate 431.92
Kbytes
         /sec.
 EZA1460I Command:
put 'mls4.test.sc60' 'mls4.test.l3c'
EZA1701I >>> SITE VARrecfm LRECL=256 RECFM=VB BLKSIZE=6233
200 SITE command was accepted
EZA1701I >>> PORT 9,12,6,81,4,42
200 Port request OK.
EZA1701I >>> STOR 'mls4.test.l3c'
125 Storing data set MLS4.TEST.L3C
250 Transfer completed successfully.
EZA1617I 25915 bytes transferred in 0.010 seconds.  Transfer rate 2591.50
Kbyte
        s/sec.
EZA1460I Command:
```

Figure 5-13 FTP client in the SC60 security zone

5.4 The big theoretical picture: TCP

In this section, we give a detailed explanation of the interactions that occur between a TCP client and server running on z/OS with MLS turned on. We do not develop the UDP or raw sockets communications in an MLS environment, because networking is not the primary topic of this book.

Reminder about TCP communications

Figure 5-14 describes the sequential interactions occurring between a client, on the left, and a server communicating through the TCP protocol.

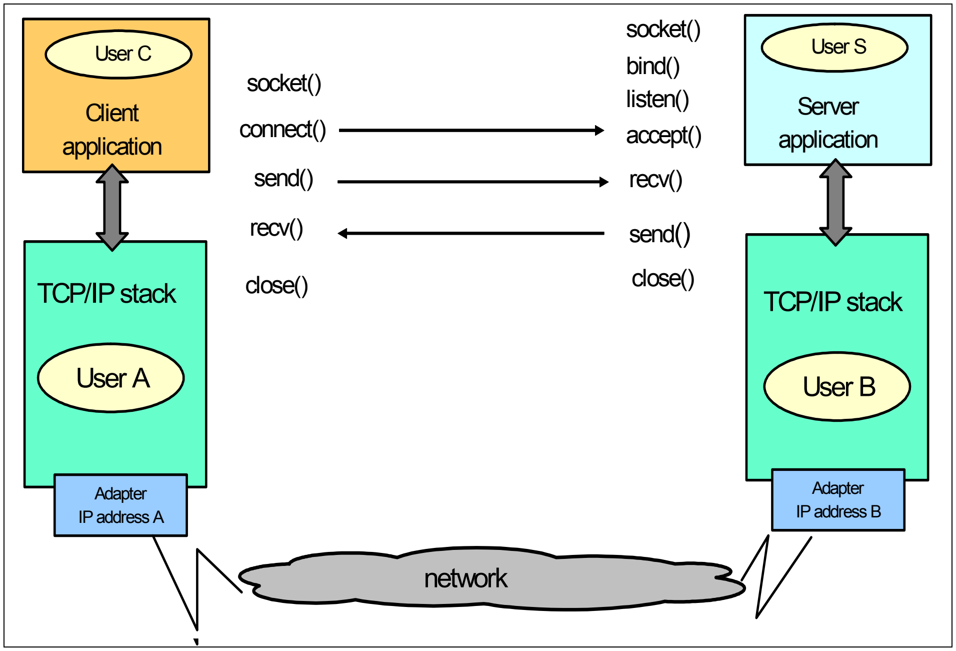


Figure 5-14 TCP/IP client-server interactions

Sequence of events

In this section, we discuss the sequence of events.

Socket() and explicit bind()

The first MAC-relevant step in this sequence is the socket() call. The application security label must be equivalent to the security label in the STACKACCESS SERVAUTH profile. Using an unrestricted stack with the SYSMULTI security label allows all applications to pass this MAC check. The second MAC event is the bind() call. We describe this in a generic manner in Figure 5-15, where the bind can succeed only if the stack SECLABEL is equivalent to the server application, and likewise the local IP address maps to a security zone with a SECLABEL equivalent to the application SECLABEL.

The server then performs a listen(), which does not have any MAC check, followed by an accept(). Return from the accept() occurs on receipt of a connect() by the client. Note that MAC is checked again on the accept, as explained later. DAC is enforced after successful MAC processing.

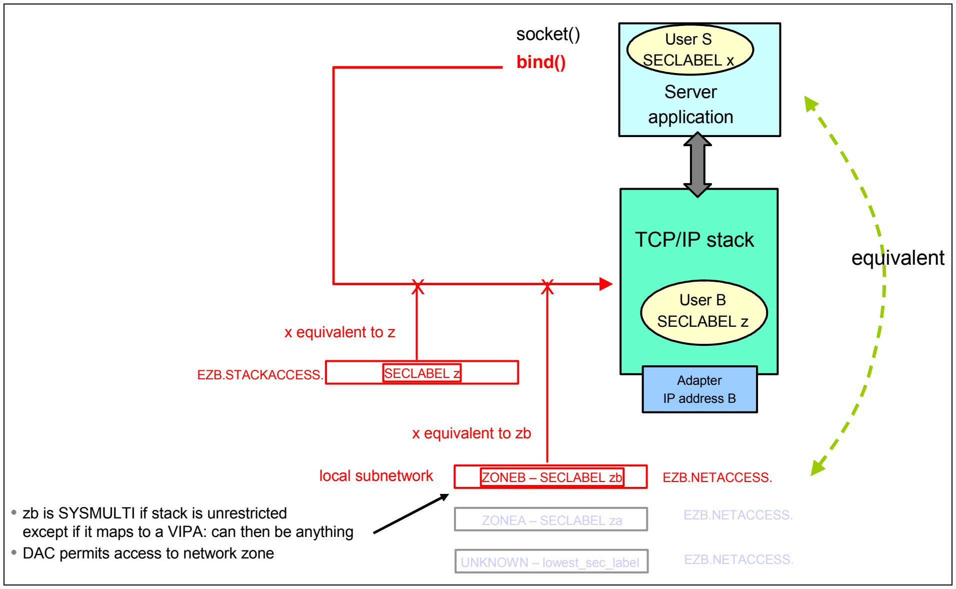


Figure 5-15 Explicit bind()

Implicit bind() and connect()

The next step is for the client, in zone A, to connect to the listening server. In Figure 5-16, we show how the connect() works in a z/OS MLS system. The connect() involves first an implicit bind() that can succeed only if the stack SECLABEL is equivalent to the client application SECLABEL, and likewise the local IP address maps to a security zone with an equivalent SECLABEL equivalent to the application SECLABEL.

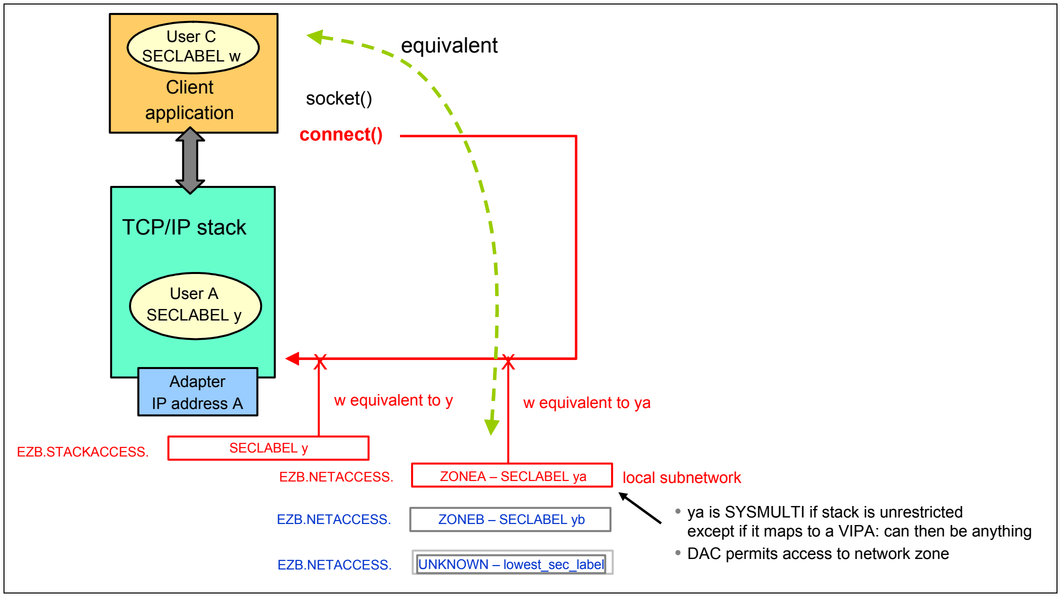


Figure 5-16 TCP connect() implicit bind() 1/2

The actual connect() is performed if the destination zone has a security label equivalent to the client application security label, as shown in Figure 5-17 on page 110. The datagrams to be exchanged must be associated to a security label so that MAC can be applied at the destination of the packet. We show the possible options to give the packet a security label in Figure 5-18 on page 110.

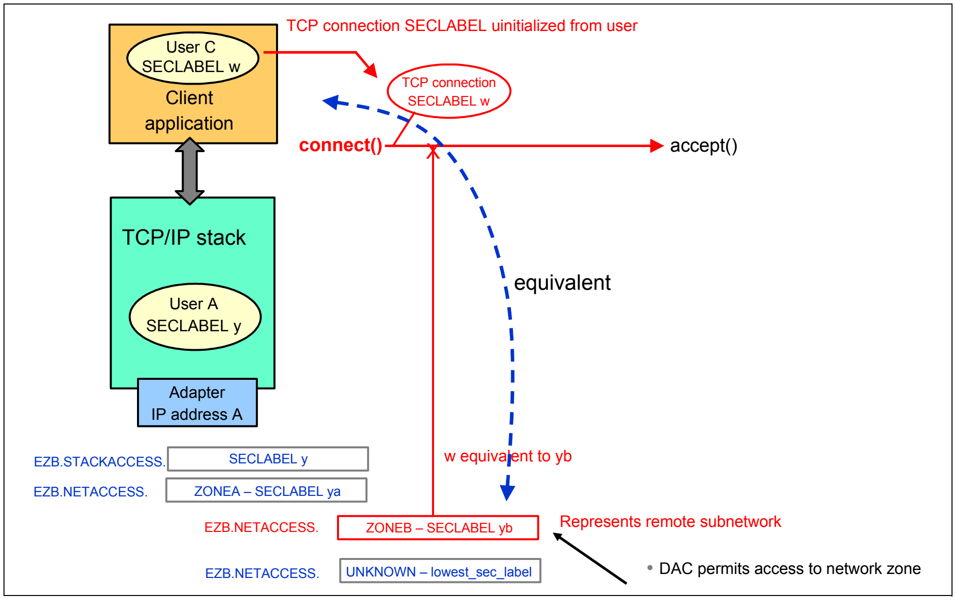


Figure 5-17 TCP connect()

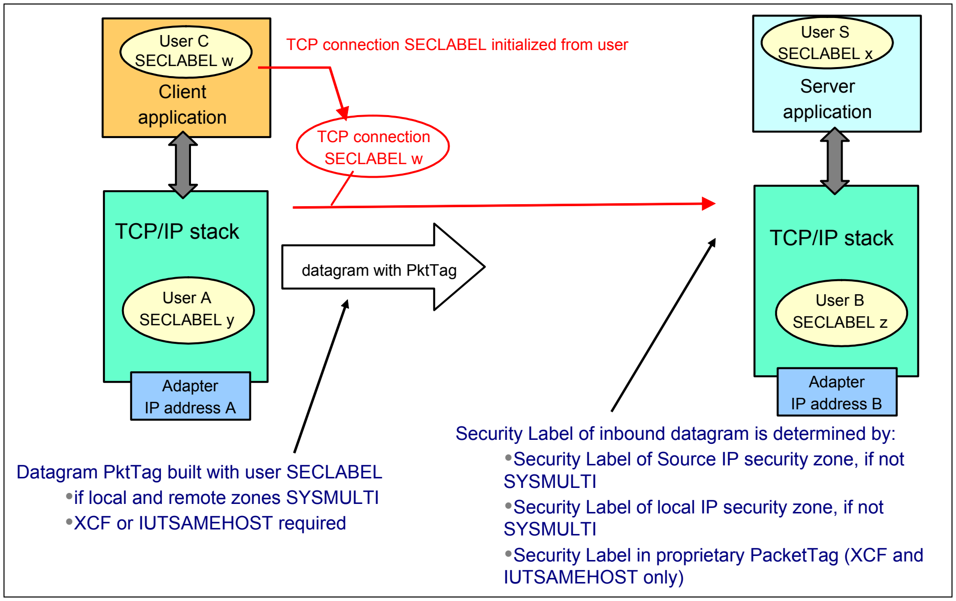

User C
SECLABEL w
Client
application
TCP connection SECLABEL initialized from user
TCP connection
SECLABEL w
User S
SECLABEL x
Server
application
TCP/IP stack
User A
SECLABEL y
Adapter
IP address A
datagram with PktTag
TCP/IP stack
User B
SECLABEL z
Adapter
IP address B
Datagram PktTag built with user SECLABEL
•if local and remote zones SYSMULTI
•XCF or IUTSAMEHOST required
Security Label of inbound datagram is determined by:
•Security Label of Source IP security zone, if not SYSMULTI
•Security Label of local IP security zone, if not SYSMULTI
•Security Label in proprietary PacketTag (XCF and IUTSAMEHOST only)

Figure 5-18 Packet tagging

Return from the accept() function at the server is performed if the connect() packet passes the MAC check, where its security label must be equivalent to the server's application security label, as shown in Figure 5-19. Note that the POE security label at the server can be initialized from:

- The packet tag, if present
- The source zone security label
- The destination zone security label

Accept()

During the accept() phase, the connection and POE security label are initialized from the source security zone, the destination security zone, or the packet tag. MAC is checked on accept(), and nonequivalent connections in the backlog are silently reset; the server gets the first equivalent one found. Every subsequent send and receive is checked for equivalence to the partner IP security zone. See Figure 5-19.

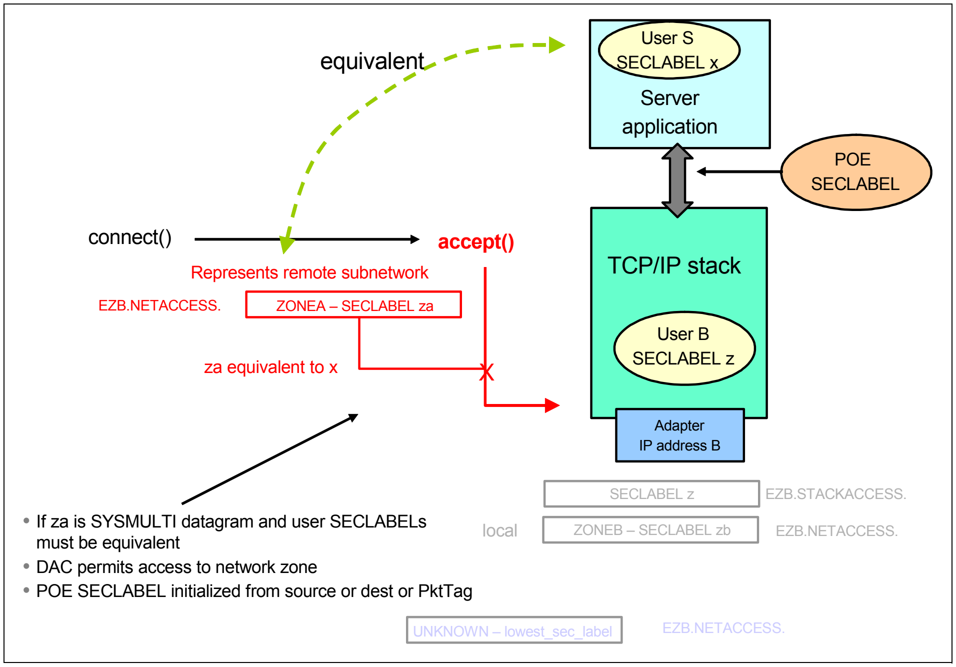


Figure 5-19 Accept()

Send() and receive()

Finally, the data are exchanged through send() and receive() functions, with proper checking of equivalency of security labels, as shown in Figure 5-20.

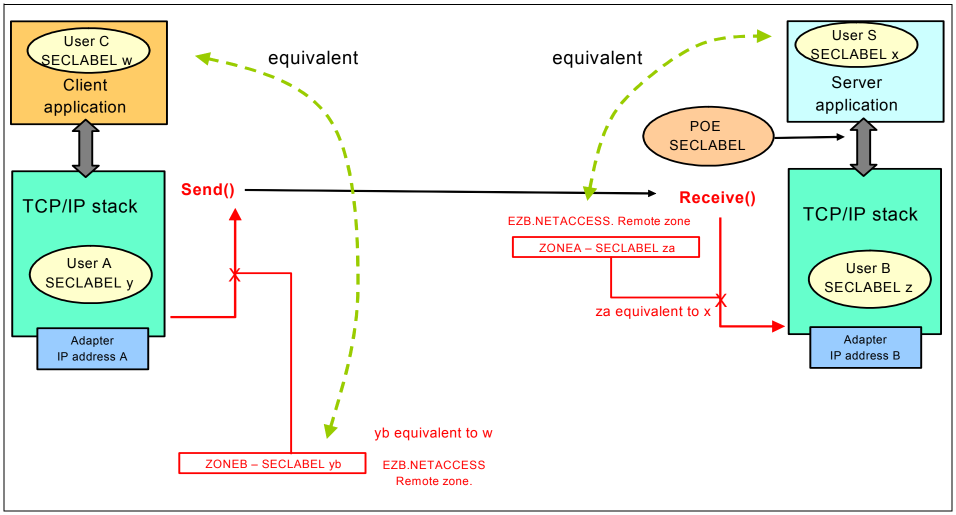


Figure 5-20 Send() and receive()

6

DB2 access control overview

This chapter provides an introduction to the two alternatives that DB2 for z/OS offers to protect its resources in general: DB2-managed security and RACF-managed security. Later chapters look at multilevel security for object-level access control and row-level security, and at the use of roles and trusted connections. These additional security mechanisms are independent of whether you use native DB2 authorization or RACF authorization as the basis for access control.

Native DB2 authorization uses the grant and revoke statements to keep the information in the DB2 catalog. Checking access means checking the DB2 catalog.

DB2 access control uses tight integration with DB2 and database integrity techniques to provide robust security. There will not be a security rule granted without a related object in most situations. For some customers and security needs, these techniques do not fit.

There are significant policy and people implications when using RACF access control. If you want the database administrators to manage security, integration with DB2 is very important. If you want security administrators to manage security, integration with the security server and the ability to have separate security and database administration are more important. The change to RACF access control causes roles to change and authorities to change.

Converting to RACF from DB2 security is not a completely compatible change. Authority based on secondary IDs, such as BINDAGENT, requires a new technique under RACF. There are some situations where DB2 access control must be used. V8 removed one situation where DB2 GRANT was needed for DB2 commands. If you want a security group to define authorization and a centralized security control point, RACF access control is a match.

Plan to use RACF facilities in a similar manner to groups and patterns. The implementation team requires both DB2 and RACF knowledge for implementation.

If you want a security group to define authorization and provide a centralized security control point, RACF access control is a match. As you implement RACF, plan to use security access patterns instead of access authorities on individual items. This can significantly reduce the number of rules needed compared to the number of grants in DB2. The implementation team needs both DB2 and RACF skills.

First, we give a short overview of how authorization IDs are assigned in DB2; this is independent of the type of access control used. Then, we describe the mechanisms for DB2-managed security, and in the last section, we discuss object access control through the use of RACF.

6.1 Authorization IDs for accessing data within DB2

Data access in a local DB2 system can be from a batch program, from a user on an interactive terminal session, or from a CICS or IMS™ transaction. For the purposes of security, DB2 uses the term *process* to represent all forms of access to data. Every process that connects to or signs on to DB2 is represented by a set of one or more authorization IDs. When authorization IDs are assigned, every process receives at least one primary authorization ID and one or more secondary IDs, and one of those IDs is designated as the current SQL ID. Also, from within trusted context, a role might be assigned. Note the following definitions:

- Primary authorization ID

  Generally, the primary authorization ID identifies a process. For example, statistics and performance trace records use a primary authorization ID to identify a process.

- Secondary authorization ID

  A secondary authorization ID is optional. It can hold additional privileges that are available to the process. For example, a secondary authorization ID can be a RACF group ID.

- SQL ID

  An SQL ID holds the privileges that are exercised when certain dynamic SQL statements are issued. The SQL ID can be set equal to the primary ID or any secondary ID.

- Role

  A role is available within a trusted context. You can define a role and assign it to authorization IDs in a trusted context. When associated with a role and using the trusted connection, an authorization ID inherits all the privileges granted to that role. The role, together with the trusted context, are new database objects introduced in DB2 V9. We describe in detail in Chapter 8, “Network trusted contexts and roles” on page 191.

Different local processes enter the DB2 access control procedure at different points, depending on the environment on which they originate.

Processes that go through *connection* processes only include:

- Requests that originate in the TSO foreground and background, including online utilities and requests through the call attachment facility
- JES-initiated jobs
- Requests through started task control address spaces as a result of the MVS START command

Processes that go through *connection* processes and that later on can go through the *sign-on* exit include:

- The overall connection between a CICS region and DB2, which is created by the CICS DB2 attachment facility
- The IMS control region
- DL/I batch
- Requests through the Resource Recovery Services attachment facility (RRSAF)

Processes that go through *sign-on* processing are:

- Requests from IMS-dependent regions, including MPP, BMP, and Fast Path
- CICS transactions that acquire a thread into DB2

During connection processing and sign-on processing, DB2 sets the primary and secondary authorization IDs for the process to use in the DB2 address space. By default, DB2 uses the authorization IDs that the process has provided. Next to default procedures, authorization IDs can also be assigned to those processes by user-written exit routines.

DB2 has two exit points for authorization routines, one in connection processing and one in sign-on processing. Both exit points perform crucial steps in the assignment of values to primary IDs, secondary IDs, and SQL IDs. DB2 has a default connection exit routine and a default sign-on exit routine. You can replace these with your own exit routines.

A sample connection exit routine and sign-on exit routine are supplied with DB2 to assist you with this. The name of the connection authorization exit routine is DSN3@ATH, and the name of the sign-on exit routine is DSN3@SGN. For a detailed description of the use of these exits, see Chapter 11, “Controlling access to a DB2 subsystem,” in *DB2 Universal Database for z/OS Administration Guide*, SC18-7413.

6.1.1 Processing connections

A connection request makes a new connection to DB2. It does not reuse an application plan that is already allocated. By default, DB2 uses the authorization IDs that the process has provided. For a TSO user, this is the TSO logon ID. For batch programs, CICS, and IMS control region, this is the user ID of the MVS region.

During connection processing, DB2 calls RACF to check whether the ID is authorized to use:

- The DB2 resource class DSNR
- The DB2 subsystem
- The requested connection type

If RACF is active and has verified the RACF user ID, DB2 runs the connection exit routine. Running with the default DB2-provided exit routine makes sure that the default user ID is the primary authorization ID, no secondary IDs exist, and the SQL ID is the same as the primary ID.

If you want to use DB2 secondary authorization IDs, you must replace the default connection exit routine. The connection authorization exit routine must be named DSN3@ATH. DB2 installation job DSNTIJEX in SDSNSAMP provides a step to replace the default connection exit with a sample connection exit routine, of which you can find the source in the SDSNSAMP library in member DSN3SATH. Basically, the sample connection exit routine sets the DB2 primary ID and the SQL ID in the same way as the default routine.

The setting of the secondary authorization ID depends on RACF options. There can be no secondary authorization ID at all, it can be one single ID, the default connected group name, or there can be a list of secondary authorization IDs equal to the list of group names to which the RACF user ID is connected. For more details about setting the different authorization IDs during connection processing, see Chapter 11, “Controlling access to a DB2 subsystem,” in *DB2 Universal Database for z/OS Administration Guide*, SC18-7413.

If the default connection exit routine and the sample connection exit routine do not provide the flexibility and features that your subsystem requires, you can write your own exit routine. For instructions about how to write your own exit routine, see Appendix B in *DB2 Universal Database for z/OS Administration Guide*, SC18-7413.

As a final step of connection processing, DB2 determines whether the connection is to be established as trusted. If a trusted context with a system authorization ID matching the primary authorization ID exists, and the attributes of the trusted context match those of the connection request, the connection is established as trusted. For more information about trusted contexts, see Chapter 8, “Network trusted contexts and roles” on page 191.

6.1.2 Processing sign-ons

For requests from IMS-dependent regions, CICS transaction subtasks, or RRS connections, the initial primary ID is not obtained until just before allocating a plan for a transaction. A new sign-on request can run the same plan without deallocating the plan and reallocating it. Nevertheless, the new sign-on request can change the primary ID. Unlike connection processing, sign-on processing does not check the RACF user ID of the address space.

First, DB2 determines the initial primary ID as follows:

- For CICS transactions, the ID that is used as the primary authorization ID is determined by attributes in the DB2CONN or DB2ENTRY definitions, depending on the thread type.
- For IMS sign-ons from message-driven regions, if the user has signed on, the initial primary authorization ID is the user's sign-on ID. IMS passes to DB2 the IMS sign-on ID and the associated RACF connected group name, if one exists. If the user has not signed on, the primary ID is the LTERM name, or if that is not available, the PSB name.
- For a batch-oriented region, the primary ID is the value of the USER parameter on the job statement, if that is available. If that is not available, the primary ID is the program's PSB name.

Then, DB2 runs the sign-on exit routine. Using the IBM-supplied default sign-on exit routine makes sure that the initial primary authorization ID remains the primary ID, the SQL ID is set equal to the primary ID, and no secondary IDs exist.

As for connection processing, if you want the primary authorization ID to be associated with DB2 secondary authorization IDs, you must replace the default sign-on exit routine. If you want to use RACF group names as DB2 secondary IDs, the easiest method is to use the IBM-supplied sample routine. The name of the sign-on exit routine must be DSN3@SGN. Another step in the DB2 installation job DSNTIJEX in SDSNSAMP enables you to replace the default sign-on exit with the sample provided exit, of which you can find the source in member DSN3SSGN, also in SDSNSAMP.

The sample sign-on routine sets the initial primary authorization ID unchanged as the DB2 primary ID, and the SQL ID is made equal to the DB2 primary ID. If RACF is not active, no secondary IDs exist. If RACF is active but its list of groups option is not active, one secondary ID exists and this is the name passed by CICS or by IMS. If RACF is active and you selected the option for a list of groups, the routine sets the list of DB2 secondary IDs to the list of group names to which the RACF user ID is connected. The list of group names includes the default connected group name.

If the default sign-on exit routine and the sample sign-on exit routine do not provide the flexibility and features that your subsystem requires, you can write your own exit routine. For instructions of how to write your own exit routine, see Appendix B in *DB2 Universal Database for z/OS Administration Guide*, SC18-7413.

As a final step of sign-on processing, DB2 checks to see whether the request is from a trusted connection performing a switch user, and if so, whether the primary authorization ID is permitted to switch. If the primary authorization ID is not allowed to switch, the connection is terminated. For more information about switching users in trusted connections, refer to Chapter 8, “Network trusted contexts and roles” on page 191.

6.2 DB2 managed security

Resource access control can be managed through a DB2 built-in mechanism. Two SQL statements are used to provide security control: GRANT and REVOKE. For example, grants control the use of buffer pools, utilities, storage groups, DB2 commands, and so on. Grants are recorded in the DB2 catalog.

Every process that connects to or signs on to DB2 is represented by one ID, called the primary authorization ID. All other IDs are secondary authorization IDs. One ID, either primary or secondary, is designated as the CURRENT SQLID. The CURRENT SQLID specifies the SQL authorization ID of the process. Additionally, a role can be assigned to a user if the connection is made through a trusted context.

The role is a new object introduced in DB2 V9 for z/OS. It is a database entity, available only in a trusted context, that groups together one or more privileges. A role can own database objects, which helps eliminate the need for individual users to own and control database objects. You can assign a role to an individual user or a group of users by defining a trusted context.

DB2 controls access to its objects by a set of privileges. Each privilege allows an action on some object. The GRANT statement grants privileges to authorization IDs or roles. Privileges can be explicit and implicit.

Explicit privileges have names and are held as the result of GRANT and REVOKE statements. There is a set of specific privileges for each type of DB2 object.

As an example, the privileges for a table are:

- ALTER: Change the table definition.
- DELETE: Delete rows in the table.
- INDEX: Create an index on the table.
- INSERT: Insert rows into the table.
- REFERENCES: Add or drop a referential constraint referring to the table.
- SELECT: Retrieve data from the table.
- TRIGGER: Define a trigger on the table.
- UPDATE: Change the contents of a specific column.

A privilege can be explicit, which means that it has a name and is held as the result of an SQL GRANT statement. Explicit privileges provide very detailed control. An example is:

```
GRANT SELECT ON TABLE SYSIBM.SYSDATABASE TO USER1
GRANT SELECT ON TABLE SYSIBM.SYSUSERAUTH TO PUBLIC
```

The first statement gives USER1 the authority to select from table SYSIBM.SYSDATABASE. The second statement allows all users to select rows from SYSIBM.SYSUSERAUTH. In the example, USER1 can be anything—a TSO user ID, a RACF group representing a list of users, or even just a short character string.

Sets of privileges are grouped into administrative authorities. These authorities form a hierarchy. Each hierarchy includes a specific group of privileges. The administrative authorities fall into the categories of system, database, and collection authorities. The highest ranking administrative authority is SYSADM. Each level of authority includes the privileges of all lower-ranking authorities.

The system authorities are:

SYSADM System administrator authority includes all DB2 privileges except a few that are reserved for installation SYSADM.

SYSCTRL System controller authority includes all DB2 privileges except those that read or modify user data.

SYSOPR System operator includes the privileges to issue most DB2 commands and to terminate utilities.

The database authorities are:

DBADM Database administrator authority includes the privileges required to control a database. Users with DBADM authority can access tables and alter or drop table spaces, tables, or indexes in that database.

DBCTRL Database controller authority includes the privileges required to control a database and run utilities against the database.

DBMAINT Database maintenance authority includes the privileges to work with certain objects and to issue certain utilities and commands on a specific database.

The package authority is:

PACKADM Package administrator has all package privileges on all packages in specific collections.

Examples of the usage of administrative authorities are:

```
GRANT SYSOPR TO USER1
GRANT DBADM ON DATABASE RACFDB2 TO USER1
```

There are two additional DB2 administrative authorities. One or two IDs are assigned the installation SYSADM authority, and one or two IDs are assigned the installation SYSOPR authority. These are similar to SYSADM and SYSOPR, respectively. However, DB2 does not record these authorities in the catalog, but they are defined in the subsystem initialization parameter module DSNZPARM. No other ID can revoke these installation authorities. These authorities are also allowed to perform some special actions such as running the CATMAINT utility, accessing DB2 when the subsystem is started with ACCESS(MAINT), or starting the directory and catalog databases when they are stopped or in restricted status and running all allowable utilities against these databases. For further details about the installation SYSADM and installation SYSOPR administrative authorities, see Chapter 9, “Controlling Access to DB2 objects,” in *DB2 Universal Database for z/OS Administration Guide*, SC18-7413.

Implicit privileges are related to the ownership of an object. Ownership is set at object creation time. When the user is the owner of the DB2 object, the user implicitly holds certain privileges over that object. The implicit privileges of ownership are different for each different object. As an example, for a table, these are the privileges:

- Alter or drop the table or any indexes on it
- Lock the table, comment on it, or label it
- Create an index or view for the table
- Select or update any row or column
- Insert or delete any row
- Use the LOAD utility for the table
- Define referential constraints on any table or set of columns
- Create a trigger on the table

In order for a user to have implicit authority over an object, the object owner needs to be either that user’s primary authorization ID, one of the user’s secondary authorization IDs, or the name of a role associated with the user in a trusted context.

> **Note:** When an object is created within a trusted context by an authorization ID with a role associated, and if this role is defined with ROLE AS OBJECT OWNER, the role is the owner of the object no matter whether the object is qualified or not.

We discuss roles and ownership in detail in Chapter 8, “Network trusted contexts and roles” on page 191.

DB2 default access is none. Until access is granted, nothing can be accessed. The DB2 models are:

- Direct: Check the user for authority to the data.
- Indirect: Check at bind time for the application process to the data, and at runtime, check for user to the application process.

Cascading revoke

When access is revoked from a certain DB2 authorization ID, all access granted by that authorization ID to the same resource is revoked as well. Assume the following scenario:

- User U1 grants a privilege P to user U2 on an object.
- User U2 grants the same privilege P to user U3 on the same object.

When U1 now revokes the privilege P from user U2, user U3 also loses its privilege on that object.

Time sequence

If the same privilege is granted to a user from two different authorization IDs and if that user in its turn grants that privilege to yet another user, the time in which these grants are given is important to determine what happens when one of the first users revokes the privilege. Consider the following scenario:

- User U1 grants privilege P to user U3 at time T1.
- User U2 grants privilege P to user U3 at time T2.
- User U3 grants privilege P to user U4 at time T3.

When user U1 now revokes the privilege P from user U3, the chronological order of T2 and T3 is important to determine if user U3 will keep its privilege or not:

- If T3 comes after T2, user U4 does not lose the privilege because at T3, user U3 might have granted the privilege on the basis of the privilege that was granted by U2.
- If T3 comes before T2, user U4 loses its privilege because user U3 might have granted the privilege only on the basis of the privilege that was granted by user U1.

6.3 RACF managed security

Even when only the internal DB2 mechanism is used to secure DB2 resources, RACF might already have been used to:

- Control connections to the DB2 subsystem.

  The ability of a user or address space to connect to DB2 is controlled through checks in the DSNR RACF resource class.

- Assign identities:
  - The DB2 primary authorization ID is a RACF identity.
  - Secondary authorization IDs are often derived by a sign-on exit routine from the RACF-generated list of groups.
- Protect the underlaying DB2 data store.

  The underlaying data sets of DB2 can be protected by RACF.

In addition to DB2 internal security, use RACF to control access to DB2 objects, authorities, commands, and utilities by making use of the RACF access control module. This RACF access control module is activated at the DB2 access control authorization exit point DSNX@XAC by replacing the default DB2-provided exit routine by a version that allows RACF checking.

The RACF access control module provides a mechanism to:

- Control and audit resources for multiple DB2 subsystems from a single point.
- Define security rules before a DB2 object is created.
- Preserve security rules for dropped DB2 objects.
- Protect multiple DB2 objects with a single security rule using a combination of RACF generic, grouping, and member profiles.
- Validate a user ID before giving it access to a DB2 object.
- Preserve DB2 privileges and administrative authorities.
- Provide flexibility for multiple DB2 subsystems with a single set of RACF profiles.
- Administer DB2 security with a minimum of DB2 skills.
- Eliminate the DB2 cascading revoke.

The RACF access control module is invoked:

- Once at DB2 subsystem startup to perform any required setup prior to authorization checking. Authorization profiles are loaded during this invocation.

- For each DB2 authorization request. This corresponds to the point when DB2 accesses security tables in the catalog to check authorization on privileges.
- Once at DB2 subsystem termination to perform its cleanup before DB2 stops.

DB2 provides a default exit, DSNX@XAC, that is available as a load module in the SDSNLOAD data set. Its source is in SDSNSAMP in member DSNXSXAC. This version of the exit routine returns a code to its invoker, indicating that a user-defined access control authorization exit is not available.

A sample replacement exit source is available in data set SDSNSAMP in member DSNXRXAC. This source can be compiled to replace the dummy-provided exit. Installation job DSNTIJEX provides a step to assemble and link-edit the routine and to place it in the APF-authorized library SDSNEXIT. The load module name or alias name of the access control authorization exit must be DSNX@XAC. The exit routine must have a CSECT name and an entry point with the same name DSNX@XAC.

> **Note:** Starting in DB2 V8, the RACF access control module DSNX@XAC is provided by DB2. In earlier versions, the exit was provided by RACF.
>
> DB2 provides two sources in SDSNSAMP:
>
> - DSNXSXAC: Dummy default version. A compiled version is present in SDSNLOAD for use when not using external security.
> - DSNXRXAC: Sample version that allows the use of RACF for external control of DB2 resources. A compilation is required to replace the default version.

RACF stores all information about users and resources in profiles. A profile is a record containing RACF information that has been defined by the security administrator.

Each RACF-defined user has a user profile containing information about his or her identity, user attributes, group, and password. Using information in its profiles, RACF authorizes access to certain resources. In addition to defining individual user profiles in RACF, you can define group profiles. A group profile defines a group of users. Users who are members of groups can share common access authorities.

A class is a collection of resources with similar characteristics. Resources are assigned to various classes. You can control the access to a resource in the class by defining profiles in the class. 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. RACF supplies a class descriptor table (CDT) that contains a list of default resource class names. You can also define your own class names and use these classes instead of or next to the default provided names.

There are nine DB2 object types. When using RACF for resource security, RACF provides a member class and a grouping class for each DB2 object. Member classes represent single objects or multiple similar named objects if generics are used. A member class name starts with the character M. An example is MDSNDTB, which is the DB2 class name for table object. Grouping classes can represent one or more objects that have similar security requirements. You can first define a group profile in the grouping class and then add individual members to that profile. A grouping class name starts with the character G. An example is GDSNTB.

Using the sample exit version as provided by DB2 allows the RACF access control module to use the default DB2 resource classes as defined in the RACF-supplied class descriptor table (CDT). However, the DB2 RACF access control module can be customized to choose a classification model and the format of the class names for DB2 objects depends on the classification used. Classification model 1 uses one set of classes for each different DB2 subsystem, and classification model 2 uses one set of classes for all DB2 subsystems.

The customization of the assembly of DSNXRXAC is obtained through specifying different values for the SET symbols at the beginning of the assembler source. Table 6-1 lists the options that define the class and resource profile names.

Table 6-1 RACF/DB2 external security module SET options

SET symbol	Default value	Description
&CLASSOPT	2	Specifies the classification option to be used: 1 = Single subsystem scope 2 = Multiple subsystem scope
&CLASSNMT	DSN	Specifies the DB2 system name format to be used
&CHAROPT	1	Specifies the one character name suffix to be used

&CLASSOPT allows you to choose for a single or multiple subsystem scope.

Characteristics of a multiple subsystem scope include:

- One set of general resource classes that protect multiple subsystems.
- General resource names are prefixed with the DB2 subsystem name.
- Classes provided in the IBM-supplied CDT are multisystem scope.
- Protect multiple subsystems with a single set of resources.
- Fewer classes overall.

Characteristics of a single subsystem scope include:

- One set of general resource classes dedicated to one subsystem.
- General resource names are not prefixed with DB2 subsystem name.
- Classes must be defined by the installation.
- Segregates resources by subsystem.
- Fewer profiles per class.

For example, running a DB2 subsystem with the name DB8L in a multiple subsystem scope, when a table is accessed, the RACF access control module generates a resource access check of the form of *DB8L.table-owner.table-name.privilege* in the class *MDSNTB*, which has the format of abbbbccd, where:

a M for member class or G for grouping class

bbbb &CLASSNMT with default = DSN

cc Type of object = TB

d &CHAROPT but ignored if &CLASSNMT = DSN

Running in a single subsystem scope environment, the resource name is *table-owner.table-name.privilege* with a class name of *MDB8LTB1*, and is installation defined. Again, the format is abbbbccd, where:

a M for member class or G for grouping class

bbbb &CLASSNMT = DB8L

cc Type of object = TB

d &CHAROPT with default = 1

The class names used in the DB2 access control module generated with the default values correspond to the default RACF-provided class names.

There are four more customization options. Table 6-2 shows their SET symbol name, default, and description.

Table 6-2 More customization SET options for DSNXRXAC

SET symbol	Default value	Description
&ERROROPT	1	Specifies the action to take in the event of an initialization or authorization error: 1 = Native DB2 authorization is used. 2= The DB2 subsystem is requested to stop.
&PCELLCT	50	Specifies the number of primary work area cells.
&SCELLCT	50	Specifies the number of secondary work area cells.
&SERVICELEVEL	HDRE810	Release/APAR number. This symbol is for IBM use only.

Next to specific DB2 privileges, there are also DB2 administrative authority privilege profiles. There is one additional class for the DB2 administrative authorities. The generated class name has the format of yyyyADMz, where:

yyyy &CLASSNMT with default = DSN

ADM Fixed character string designated for administrative authority classes

z &CHAROPT with default =1 but ignored if &CLASSNMT = DSN

The class name used in the sample DB2 access control module without changing any of the defaults is DSNADM. In the previous example for the single scope system, the administrative authority class name would be DB8LADM1.

For further details about customizing the RACF access control module, see Chapter 2, “Planning,” in *DB2 UDB for z/OS RACF Access Control Module Guide*, SC18-7433.

RACF classes must be activated through a RACF SETROPTS command. When additional classes have been set up and activated, DB2 must be restarted.

After the DB2 resource classes are active, RACF profiles are to be defined in these classes. The way DB2 builds the profile format is to use a combination of the resource name and the name of the privilege required to access that resource.

For further details about the use of the RACF DB2 profiles, see Appendix D, “RACF authorization checking reference,” in *DB2 UDB for z/OS RACF Access Control Module Guide*, SC18-7433.

The order in which the RACF access control module checks user authorization is to first do some special checking before any RACF checking is done. If applicable, the first test is to check if a user owns the resource or if the user's name matches the schema name. Then, RACF checks against specific DB2 privileges. If the check to access a specific resource does not allow access, additional DB2 privilege checks on administrative privileges are performed. Administrative authorities have the same meaning as in the DB2 internal security environment.

As an example, to SELECT from a table, an authorization ID must at least have one of the following:

- Ownership of the table
- SELECT privilege on the table
- DBADM authority for the database
- SYSCTRL (catalog tables only)
- SYSADM authority

When a user is the owner of a DB2 object, that user might have some implicit privileges, but not all privileges associated with the object. Table 6-3 shows how the RACF access control module supports specific DB2 objects with their associated implicit privileges of ownership.

Table 6-3 DB2 objects and associated implicit ownership privileges

DB2 object	**Implicit privileges**
Java™ archive (JAR)	USAGE
Package	BIND COMMENT ON COPY
Plan	BIND COMMENT ON
Schema	ALTERIN COMMENT ON DROPIN
Sequence	ALTER COMMENT ON USAGE
Stored procedure	DISPLAY EXECUTE START STOP

DB2 object	Implicit privileges

Table	All privileges except:
	CREATE SYNONYM
	DROP SYNONYM
	CREATE VIEW

User-defined distinct type	USAGE
User-defined function	DISPLAY EXECUTE START STOP
View	COMMENT ON DROP

Table 6-4 gives an overview of the DB2 administrative authority profiles in the RACF class DSNADM. The RACF access control module is bypassed for installation SYSADM and installation SYSOPR.

Table 6-4 Overview of RACF profiles in DSNADM class

RACF class	DB2 object	Db2 privilege	RACF profile
DSNADM	System	SYSADM	DB2-subsystem.SYSADM
		SYSCTRL	DB2-subsystem.SYSCTRL
		SYSOPR	DB2-subsystem.SYSOPR
	Database	DBADM	DB2-subsystem.database-name.DBADM
		DBCTRL	DB2-subsystem.database-name.DBCTRL
		DBMAINT	DB2-subsystem.database-name.DBMAINT
	All collections	PACKADM	DB2-subsystem.PACKADM
	Specific collection	PACKADM	DB2-subsystem.collection-ID.PACKADM

A profile can represent a specific object in a class, but can also be generic to represent a group of objects. An example of a discrete profile for a specific table is DB8L.DSN8810.EMP.SELECT. A generic profile for the same table that covers all table privileges is DB8L.DSN8810.EMP.*. The replacement of the specific privilege by an asterisk (*) makes the profile generic, and thus it represents all privileges for the EMP table accessed in the DB2 subsystem DB8L.

Tip: One thing to watch out for is if generics is off for the class. If it is, the DB8L.DSN8810.EMP.* profile will be discrete, and not generic.

A profile contains the universal access for the object for which it is defined, in other words, the access that any user has to access this resource. The RACF RDEFINE command creates profiles. The RACF command to create a discrete profile in the MDSNTB class so that nobody is to delete the EMP table is:

```
RDEF MDSNTB 'DB8L.DSN8810.EMP.DELETE' UACC(NONE)
```

The profile that is equivalent to granting PUBLIC to the EMP table on subsystem DB8L is:

```
RDEF MDSNTB 'DB8L.DSN8810.EMP.*' UACC(READ)
```

For PUBLIC AT ALL LOCATIONS, the command is:

```
RDEF MDSNTB '*.DSN8810.EMP.*' UACC(NONE)
```

The next step is then to add an individual user or groups of users to the access list of these resource profiles, thus giving them individual access to these profiles. To do this, use the RACF PERMIT command. The RACF commands to allow user USRT060 to be the only user to delete from the EMP table are:

```
RDEF MDSNTB 'DB8L.DSN8810.EMP.DELETE' UACC(NONE)
PE 'DB8L.DSN8810.EMP.DELETE' CLASS(MDSNTB) ID(USRT060) ACC(READ).
```

Presuming that we are working on an active DB2 system where the MDSNTB class is activated, after a profile has been added or modified, it is required to refresh the RACF profiles with the RACF SETROPTS REFRESH command, such as:

```
SETROPTS RACLIST(MDSNTB) REFRESH.
```

The DB2 access control module checks the corresponding RACF resource class profiles and attempts to make an authority decision. If the DB2 access control module cannot make a pass or fails decision, it defers the decision to native DB2 authority checking. This allows us to implement DB2 RACF security one object at a time. For more details about the actual decision tree the DB2 access control module makes, see *DB2 UDB for z/OS RACF Access Control Module Guide*, SC18-7433.

Sample scenario

As an example of how to use RACF external security, we first re-assembled the DB2-provided RACF access control module with the source that is provided in data set SDSNSAMP in member DSNXRXAC. We copied member DSNTIJEX from SDSNSAMP to our private data set and modified it to only execute the assembly of DSNXRXAC into SDSNEXIT.

With this new version of the RACF access control module, we stopped and started DB2. The RACF access control module generated the output shown in Example 6-1.

Example 6-1   RACF access control module output with inactive RACF classes

```
*IRR901A  RACF/DB2 EXTERNAL SECURITY MODULE FAILED TO INITIALIZE
          FOR DB2 SUBSYSTEM DB8L BECAUSE NO ACTIVE DB2 RELATED CLASSES
          WERE FOUND.
*IRR912I NATIVE DB2 AUTHORIZATION IS USED.
 IRR908I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM DB8L HAS
         A MODULE VERSION OF HDRE810  AND A MODULE LENGTH OF 00005BD0.
 IRR909I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM DB8L
         IS USING OPTIONS: &CLASSOPT=2
                           &CLASSNMT=DSN
                           &CHAROPT=1
                           &ERROROPT=1
                           &PCELLCT=50
                           &SCELLCT=50
 IRR910I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM DB8L
         INITIATED RACLIST FOR CLASSES:
          MDSNDB   MDSNPK   MDSNPN   MDSNBP   MDSNCL
          MDSNTS   MDSNSG   MDSNTB   MDSNSM   MDSNSC
          MDSNUT   MDSNUF   MDSNSP   MDSNJR   MDSNSQ
          DSNADM
 IRR911I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM DB8L
         SUCCESSFULLY RACLISTED CLASSES:
          * NONE *

 DSNX210I  -DB8L DSNXACAE - ACCESS CONTROL
 AUTHORIZATION EXIT ROUTINE (DSNX@XAC) HAS INDICATED THAT IT SHOULD
 NOT BE CALLED, HAS ABENDED OR HAS RETURNED AN INVALID RETURN CODE
 DURING INITIALIZATION. RETURN CODE=000C, REASON CODE=00000004.
 CUMULATIVE ABENDS DURING EXIT PROCESSING=0000.
 EXIT ROUTINE STATUS: STOPPED.
```

Message IRR901A tells us that initialization failed because the DB2 classes are not active. IRR912I tells us that native DB2 authorization will be used. We ended the example with the situation that USRT060 could read from the EMP table, where user USRT051 could not. In the actual situation, we tried to read the EMP with both user IDs and found out that was still the case.

The next step was to activate RACF resource classes for DB2. Example 6-2 shows the commands we used for this.

Example 6-2 RACF commands to activate DB2 resource classes

```
//RUNDB2   EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD   SYSOUT=*
//SYSPRINT DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=*
//SYSTSIN  DD   *
 SETROPTS CLASSACT(DSNADM)
 SETROPTS CLASSACT(MDSNPK) GENERIC(MDSNPK)
 SETROPTS CLASSACT(MDSNUT) GENERIC(MDSNUT)
 SETROPTS CLASSACT(MDSNSQ) GENERIC(MDSNSQ)
 SETROPTS CLASSACT(DSNADM) GENERIC(DSNADM)
 SETROPTS CLASSACT(MDSNBP) GENERIC(MDSNBP)
 SETROPTS CLASSACT(MDSNCL) GENERIC(MDSNBP)
 SETROPTS CLASSACT(MDSNDB) GENERIC(MDSNDB)
 SETROPTS CLASSACT(MDSNJR) GENERIC(MDSNJR)
 SETROPTS CLASSACT(MDSNPN) GENERIC(MDSNPN)
 SETROPTS CLASSACT(MDSNSC) GENERIC(MDSNSC)
 SETROPTS CLASSACT(MDSNSG) GENERIC(MDSNSG)
 SETROPTS CLASSACT(MDSNSM) GENERIC(MDSNSM)
 SETROPTS CLASSACT(MDSNSP) GENERIC(MDSNSP)
 SETROPTS CLASSACT(MDSNTB) GENERIC(MDSNTB)
 SETROPTS CLASSACT(MDSNTS) GENERIC(MDSNTS)
 SETROPTS CLASSACT(MDSNUF) GENERIC(MDSNUF)
/*
```

After activating the DB2 resource classes, we again stopped and started DB2. This time we received the output from the RACF access control module, as shown in Example 6-3.

Example 6-3 RACF access control module output with DB2 resource classes active

```
IRR908I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM DB8L HAS
         A MODULE VERSION OF HDRE810  AND A MODULE LENGTH OF 00005BD0.
 IRR909I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM DB8L
         IS USING OPTIONS: &CLASSOPT=2
                           &CLASSNMT=DSN
                           &CHAROPT=1
                           &ERROROPT=1
                           &PCELLCT=50
                           &SCELLCT=50
 IRR910I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM DB8L
         INITIATED RACLIST FOR CLASSES:
          MDSNDB   MDSNPK   MDSNPN   MDSNBP   MDSNCL
          MDSNTS   MDSNSG   MDSNTB   MDSNSM   MDSNSC
          MDSNUT   MDSNUF   MDSNSP   MDSNJR   MDSNSQ
          DSNADM
 IRR911I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM DB8L
         SUCCESSFULLY RACLISTED CLASSES:
          MDSNDB   MDSNPK   MDSNPN   MDSNBP   MDSNCL
          MDSNTS   MDSNSG   MDSNTB   MDSNSM   MDSNSC
          MDSNUT   MDSNUF   MDSNSP   MDSNJR   MDSNSQ
          DSNADM
```

Again, we tried to read the EMP table with both user IDs, and again found out that USRT060 is able to read from the EMP table, while USRT051 is not. If access was denied from RACF, we would have seen RACF message ICH408I for user ID USRT051. Because this was not the case, this proves that authority checking is still being deferred to DB2-managed authorization checking, and that the access was denied by managed DB2 security. This is because the DB2 access control module detected no object profile in the active MDSNTB class covering the read action, and no administrative profile in the active DSNADM class allowed access.

To make sure that resource control was done by RACF over all resources, we provided a “top” generic profile that protects all of the resources in the MDSNTB class by giving it a universal access of NONE. We obtained this by submitting the job shown in Example 6-4.

Example 6-4 Defining a “top” generic profile to protect all classes

```
//RUNDB2   EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD   SYSOUT=*
//SYSPRINT DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=*
//SYSTSIN  DD   *
 RDEF MDSNTB ** UACC(NONE)
 SETROPTS RACLIST(MDSNTB) REFRESH
/*
```

Note that after these commands we did not stop and restart the DB2 system. The SETR RACLIST REFRESH updated the RACF security information.

After this, neither of the two user IDs was able to read from the table. Example 6-5 shows that the access was denied by RACF.

Example 6-5 Access denied by RACF for USERT060 and USRT051

```
ICH408I USER(USRT060 ) GROUP(SYS1    ) NAME(MLS TEST USER       )
  DB8L.DSN8810.EMP.SELECT CL(MDSNTB  )
  INSUFFICIENT ACCESS AUTHORITY
  FROM ** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
ICH408I USER(USRT051 ) GROUP(SYS1    ) NAME(MLS TEST USER       )
  DB8L.DSN8810.EMP.SELECT CL(MDSNTB  )
  INSUFFICIENT ACCESS AUTHORITY
  FROM ** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
```

To allow USRT060 to select again from the EMP table, we now add to the MDSNTB class a discrete profile with the SELECT privilege for EMP and that has universal access NONE. Then, we add a profile with READ access for USRT060 to the access list of this newly created discrete profile. We do this by submitting the job shown in Example 6-6.

Example 6-6 Giving USRT060 SELECT access on table EMP

```
//RUNDB2   EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD   SYSOUT=*
//SYSPRINT DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=*
//SYSTSIN  DD   *
 RDEF MDSNTB DB8L.DSN8810.EMP.SELECT UACC(NONE)
 PERMIT DB8L.DSN8810.EMP.SELECT CLASS(MDSNTB) ID(USRT060) ACC(READ)
 SETROPTS RACLIST(MDSNTB) REFRESH
/*
```

To give SELECT access to all resources owned by DSN8810 on system DB8L, we could have defined a generic profile DB8L.DSN8810.*.SELECT with universal access READ. However, we prefer to give access to USRT051 to only a selected number of tables owned by DSN8810. To demonstrate the use of a grouping class, we define the list of the selected tables in the GDSNTB class. There is no need to a define a generic ** profile in the grouping class GDSNTB, because there is already a generic covering profile in the MDSNTB member class. We first define the profile DSN8810_TABGROUP, and with the ADDMEM option, we add the discrete profiles of the different members. Other tables can later be added or deleted using the RALTER command with ADDMEM or DELMEM options, respectively. Then, we use the PERMIT command to add user USRT051 to the access list of the grouping profile. Example 6-7 on page 135 shows the commands that we issued to obtain this. Note that the refresh is done on the table member class after we updated the table grouping class.

Example 6-7 RACF commands to populate the GDSNTB class

```
//RUNDB2   EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD   SYSOUT=*
//SYSPRINT DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=*
//SYSTSIN  DD   *
 RDEF GDSNTB DSN8810_TABGROUP UACC(NONE) +
           ADDMEM (DB8L.DSN8810.ACT.SELECT, +
                   DB8L.DSN8810.DEPT.SELECT)
 RALT GDSNTB DSN8810_TABGROUP ADDMEM(DB8L.DSN8810.PROJ.SELECT)
 PERMIT DSN8810_TABGROUP CLASS(GDSNTB) ID(USRT051) ACC(READ)
 SETROPTS RACLIST(MDSNTB) REFRESH
/*
```

USRT051 can now perform selects on the three tables. When USRT060 tries to select on the PROJ table, that user is denied, as shown in Example 6-8. Note that it is the member class MDSNTB that is referenced in message ICH408I.

Example 6-8 USRT060 denied from a select on the PROJ table

```
ICH408I USER(USRT060 ) GROUP(SYS1    ) NAME(MLS TEST USER       )
   DB8L.DSN8810.PROJ.SELECT CL(MDSNTB  )
   INSUFFICIENT ACCESS AUTHORITY
   ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
 ***
```

To assign DBMAINT authority to user USRT052 on the two databases used in the sample application, DSN8D81A and DSN8D81P, we submitted the RACF commands shown in Example 6-9.

Example 6-9 Giving DBMAINT authority to USRT052

```
//RUNDB2   EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD   SYSOUT=*
//SYSPRINT DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=*
//SYSTSIN  DD   *
 RDEF DSNADM ** UACC(NONE)
 RDEF DSNADM DB8L.DSN8D81*.DBMAINT UACC(NONE)
 PERMIT DB8L.DSN8D81*.DBMAINT CLASS(DSNADM) ID(USRT052) ACC(READ)
 SETROPTS RACLIST(DSNADM) REFRESH
/*
```

We first issued a DISPLAY DATABASE(DSN8*) command with user ID USRT060. Example 6-10 shows the resulting output.

*Example 6-10 USRT060 is denied from DISPLAY DATABASE on DSN8**

```
DSNT300I  -DB8L AUTH-ID USRT060 NOT AUTHORIZED TO PERFORM DISPLAY ON
 DATA BASE DSN8D81A. REQUEST REJECTED
 DSNT300I  -DB8L AUTH-ID USRT060 NOT AUTHORIZED TO PERFORM DISPLAY ON
 DATA BASE DSN8D81P. REQUEST REJECTED
 DSNT365I  -DB8L    NO DATABASES FOUND
 DSN9022I  -DB8L DSNTDDIS 'DISPLAY DATABASE' NORMAL COMPLETION
 ***
```

The console and DB2 job log show the RACF command, as shown in Example 6-11.

*Example 6-11 Commands stating USRT060 has no DBMAIN authority on DSN8D81**

```
ICH408I USER(USRT060 ) GROUP(SYS1    ) NAME(MLS TEST USER       )
   DB8L.DSN8D81A.DBMAINT CL(DSNADM  )
   INSUFFICIENT ACCESS AUTHORITY
   FROM DB8L.DSN8D81*.DBMAINT (G)
   ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
 ICH408I USER(USRT060 ) GROUP(SYS1    ) NAME(MLS TEST USER       )
   DB8L.DSN8D81P.DBMAINT CL(DSNADM  )
   INSUFFICIENT ACCESS AUTHORITY
   FROM DB8L.DSN8D81*.DBMAINT (G)
   ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
```

Example 6-12 shows the output from USRT052 from the DISPLAY DATABASE command.

Example 6-12 USRT052 DISPLAY DATABASE output

```
DSNT360I  -DB8L ***********************************
 DSNT361I  -DB8L *  DISPLAY DATABASE SUMMARY
                 *    GLOBAL
 DSNT360I  -DB8L ***********************************
 DSNT362I  -DB8L     DATABASE = DSN8D81A  STATUS = RW
                    DBD LENGTH = 24218
 DSNT397I  -DB8L
 NAME     TYPE PART  STATUS            PHYERRLO PHYERRHI CATALOG  PIECE
 -------- ---- ----- ----------------- -------- -------- -------- -----
 DSN8S81D TS         RW
 DSN8S81E TS    0001 RW
     -THRU      0005
 DSN8S81P TS         RW
 DSN8S81R TS         RW,AREO*
 DSN8S81S TS         RW,AREO*
```

```
XACT1    IX         RW
XACT2    IX         RW
XDEPT1   IX         RW
XDEPT2   IX         RW
XDEPT3   IX         RW
XEMP1    IX    0001 RW
    -THRU      0005
XEMP1SHE IX         RW
XEMP2    IX    L*   RW
XEMPPROJ IX         RW
XPARTS   IX         RW,AREO*
XPROJ1   IX         RW
XPROJ2   IX         RW
XPROJAC1 IX         RW
******* DISPLAY OF DATABASE DSN8D81A ENDED      **********************
DSNT360I  -DB8L ***********************************
DSNT362I  -DB8L     DATABASE = DSN8D81P  STATUS = RW
                   DBD LENGTH = 8066
DSNT397I  -DB8L
NAME     TYPE PART  STATUS            PHYERRLO PHYERRHI CATALOG  PIECE
-------- ---- ----- ----------------- -------- -------- -------- -----
DSN8S81C TS         RW
DSN8S81Q TS         RW
XCONA1   IX         RW
XDSPTXT1 IX         RW
XMAPRTBL IX         RW
XOPTVAL1 IX         RW
******* DISPLAY OF DATABASE DSN8D81P ENDED      **********************
DSN9022I  -DB8L DSNTDDIS 'DISPLAY DATABASE' NORMAL COMPLETION
***
```

7

DB2 and multilevel security

In this chapter, we first discuss the two ways how multilevel security can be implemented in DB2. These are at object level and at row level. We give a short overview of the object-level implementation, and then we show the different steps we went through to implement row-level security in DB2 V8. We start with describing row-level security as a subset of multilevel security, with an emphasis on how row-level security is implemented by DB2. Then we go through sample scenarios to demonstrate how row-level security applies to the SQL operations SELECT, INSERT, UPDATE, and DELETE.

With Version 9, DB2 provides two significant new database entities: network trusted context and roles. At a high level, roles are only available through trusted contexts and roles cannot be associated with a SECLABEL. A trusted context can exist without multilevel security. However, using them together allows for a user to be automatically switched to one of their defined SECLABELS for the duration of their work within the trusted connection. It also allows a default SECLABEL to be specified for the trusted context, thus associating a default SECLABEL to users that do not have one directly assigned (must also be a valid SECLABEL for those users). Improved access control with network trusted context and roles allows more precise control of security. For more information, see Chapter 8, “Network trusted contexts and roles” on page 191.

7.1 Multilevel security in DB2

In multilevel security, the relationship between DB2 users and DB2 objects is important. In the context of multilevel security, a user is any entity that requires access to system resources. The term user includes not only human users, but also stored procedures or batch jobs.

In the context of multilevel security, an object is any system resource to which access must be controlled. Examples of objects include:

- Data sets
- Tables
- Rows
- Commands

Using multilevel security, you can define security for DB2 objects. By assigning security labels to your DB2 objects, you can define a hierarchy between those objects. Multilevel security then restricts access to an object based on the security level of that object.

DB2 also has an implementation of multilevel security at the row level. This implementation enables us to perform row-level security checks, which enable us to control which users have authorization to view, modify, or perform other actions on specific rows of data.

7.2 Row-level security as a subset of multilevel security

In this section, we review row-level security as a subset of multilevel security.

7.2.1 The need for row-level security

In today's complex world, organizations can have considerable needs to restrict access to data in their database applications. Privacy and data protection legislation, anti-trust legislation, and considerations of national security are just a few of the reasons why organizations might need to make sure that people in one part of the business do not know and cannot find out what is going on in another part.

Companies might lose business if they cannot demonstrate to security-conscious potential customers that data relating to them is strictly protected from unauthorized access. Regulatory authorities might require that computer systems be separately maintained if it cannot be shown that the different divisions of a company are prevented from accessing details of the other divisions' operations.

In an increasingly interconnected era, organizations might want to offer limited access to their operational systems to the clients, suppliers, and trading partners, but not want to give those people freedom to roam through data pertaining to their competitors.

We summarize the requirement as follows, using a customer order database as an example. It must include:

- A means of marking a customer as being in a set of protected customers
- A means of propagating such markings to related data after the customer marking is made
- A means of marking new data for such customers and transient data relating to such customers with the security markings of the customer
- The restriction of access to data based on such markings
- A means of ensuring that the security markings of the data are not changed other than by authorized users and processes

7.2.2 DB2 solutions

The traditional response to these sorts of requirements for DB2 applications has been to use views. However, views have a number of disadvantages:

- Complex security requirements can entail a huge proliferation of views, which imposes a significant maintenance burden.
- Views might involve joins to authorization tables, which can render the views read-only, thus requiring awkward application coding to perform updates.
- The data structures might already be extremely complex, and providing an efficient method of controlling access might involve propagation of large amounts of essentially redundant data into tables that have no need for it other than the access control requirement.
- The worst disadvantage of views, though, and the one that can sink a security scheme based on them in the eyes of auditors, regulatory authorities, and those involved in law enforcement and national security, is that they are SQL constructs and only have effect in an SQL context: They do not prevent access through processes that do not use the SQL interface, such as unload utilities. Views only work *within* an application, and do little or nothing to protect data access outside the application.

Another approach that some DB2 customers have adopted is to use exits, using fieldprocs or editprocs. These overcome several of the difficulties with views, because they are invoked when the data is accessed outside the application as well as within it; but they bring problems of their own:

- They usually need to be coded in Assembler, and Assembler skills might not be readily available.
- They can impose unacceptable restrictions on how the application can process data.
- They can easily entail a significant performance processing time.
- Auditors often lack the skills to satisfy themselves that the resultant security scheme is valid, making it difficult for the organization to acquire the certifications they might need.

DB2 V6 introduced database triggers, which can be used for security purposes, among other things. While not subject to many of the same constraints as fieldprocs and editprocs with regard to coding language, they incur most of the other disadvantages of the exit approach.

7.2.3 RACF requirements for basic SECLABEL processing

To use security labels, all that is required is the RACF definition of the range of security labels to be used and activation of the RACF SECLABEL class.

This is completely separate from the use of RACF to control access to DB2 resources, which is an alternative to using the DB2 GRANT and REVOKE mechanism to control such access.

> **Important:** You do not have to enable the DB2 RACF exit DSNX@XAC in order to use SECLABELs for row-level security. You can continue to use native DB2 GRANTs and REVOKEs to control all other DB2 access, but you will not have SECLABELs for object-level security.

7.2.4 Write-down in DB2

Enabling the write-down option for DB2 controls whether a user can change the classification of data. This is the same as its effect on other products running on the same z/OS system under RACF control. The security administrator should carefully think through the use of this option because a user possessing this privilege can inadvertently or intentionally desclassify potentially sensitive information. It is important to note the write-down option is system-wide and applies not only to DB2, but to the entire z/OS system.

We explain the RACF requirements for enabling write-down control in detail in Chapter 3, “MLS” on page 27.

In a multilevel security environment, the RACF SETR MLS(FAILURES) command enables the write-down option and the RACF SETR NOMLS command disables it. At a system level, the security administrator controls whether write-down is allowed by activating and deactivating the RACF MLS option.

In an environment where write-down control is not enabled by the RACF MLS option as a result of the DB2 implementation of multilevel security, *all* DB2 users with valid security labels have the same ability to change the classification of data that users having the write-down privilege possess (when SETR MLS is in effect).

Note: In the “no write-down” environment, all DB2 users can operate as though they have write-down authority, but this does not apply or extend to users outside of DB2.

Here is an example showing how declassification of data in DB2 can be done by individuals with the ability to write down:

- Read a row from a lower level and write data to that row.

  Suppose that a user has a SECLABEL of L2C and that a row has a SECLABEL of L1C. If the user has the write-down authority, the user is allowed to update a column in the row and specify the same SECLABEL value L1C for the row. In this manner, the user can save higher classified L2 data in a lower classified L1 row, and thus declassify this L2 data.

- Change the SECLABEL in a DB2 table row.

  Suppose that both a user and a row have a SECLABEL of L2C. If the user has the write-down authority, that user can change the SECLABEL value to a valid security label value that is not disjoint in relation to that user’s own security label. If user changes the SECLABEL value from L2C to L1C, this declassifies all of the data in the row because now a user with a security label of L1C can access it.

To run in a DB2 row-level security environment, it is sufficient to have:

- The RACF SECLABEL class active
- SECLABELs for users
- SECLABELs on DB2 table rows

With this setup, *all DB2 users* have write-down authority. If this is not acceptable for your installation and you do not have your own mechanism to prevent write-down being implemented, you can enable write-down control with the

RACF MLS option. No write-down is established by issuing the RACF command SETROPTS MLS(FAILURES). In this case, *no user* has the write-down privilege. As mentioned before, remember that MLS(FAILURES) is a system-wide option and that it controls more than just DB2.

However, there can be cases where you want to allow write-down control by selected individuals. You now can assign the write-down privilege to a selected number of users by performing following steps:

1. Define a resource profile IRR.WRITEDOWN.BYUSER in the FACILITY class with universal access NONE:

   ```
   RDEFINE FACILITY IRR.WRITEDOWN.BYUSER UACC(NONE)
   ```

2. Authorize users for the write-down privilege by adding those users, individually or by group, to the access list with UPDATE authority:

   ```
   PERMIT IRR.WRITEDOWN.BYUSER CLASS(FACILITY) ID(user) ACCESS(UPDATE)
   ```

   UPDATE access is required because RACF is designed so that both access authorities READ and UPDATE allow users to query, set, and reset the write-down mode of their address spaces with the RACPRIV RACF command. A user with READ authority will not have the write-down privilege by default, and needs to execute the RACPRIV command to obtain the write-down privilege. A user with UPDATE authority will be in write-down mode by default. Because it is not possible to execute the RACPRIV command in a DB2 environment, it is required to specify access UPDATE in the PERMIT command.

Here we have to warn you that granting a user ID access to the IRR.WRITEDOWN.BYUSER profile has a global impact. If a user is given authority to this profile, not only can the user change the classification of data in DB2, but on the rest of the system as well.

Note also that implementing write-down privileges by activating the RACF MLS option (FAILURES) does not require having MLACTIVE in effect.

When working in a DB2 environment where MLACTIVE and MLS are not activated, assigning a SECLABEL to a DB2 user does not have that much impact on the rest of the system. When setting SETR MLS as active, the situation becomes more complicated because the SETR MLS option can have major implications to the multilevel security environment. It can require a user with a SECLABEL to be only able to access certain resources when it is also assigned the appropriate SECLABEL. This has a serious impact on the entire system. A user who could enter SPUFI and access a table with a security column will now require TSO and all data sets accessed under TSO to have security labels as well.

7.2.5 DB2 row-level security implementation

When you CREATE a table, you can decide to implement row-level security by including a column that specifies the AS SECURITY LABEL attribute. You can also execute an ALTER or add a column with this attribute to an existing table. You can assign any name to the security label column, but a column with this attribute cannot be specified more than once in the table. The security label column must be data type single-byte character (SBCS), CHAR(8), NOT NULL WITH DEFAULT.

Defining a table with a column with the AS SECURITY LABEL attribute allows DB2 to verify access at the row level, using multilevel security with row level granularity. The special column contains the security label (SECLABEL) of the row. Each row is then assigned a SECLABEL when the table is populated.

Security labels are defined and provided by RACF. When connecting to DB2, the user's SECLABEL is retrieved from RACF. When rows are accessed, DB2 checks this SECLABEL against the SECLABEL value in each row. If access is allowed, you can access the row. If access is not allowed, the data is not returned and you are even not even aware that the row exists.

Normally, when a user performs an INSERT or UPDATE operation, or LOADs a row, this user's SECLABEL value is stored in the table's column that is defined with the AS SECURITY LABEL attribute. However, if write-down is not in effect, or write-down is in effect and the user has write-down privilege, the user can specify any valid SECLABEL for that row.

After the column with the AS SECURITY LABEL attribute is present in a table, the only technique to disable row-level security is to drop the table, table space, or database.

This column cannot have field procedures, edit procedures, or check constraints.

When the audit trace (class 3) is active, an audit record IFCID 0142 is created. A table with a security label is treated like an audited table.

7.2.6 Accessing data in a table defined with row-level security

Let us now have a look at how the different DB2 operations that you normally perform against data rows in a table are affected by having multilevel security with row granularity for that table.

SELECT

For a SELECT statement, the user's SECLABEL is compared to the SECLABEL of the each row in the table. If user’s SECLABEL dominates the row’s SECLABEL, the row is returned. If user’s SECLABEL does not dominate the row’s SECLABEL, the row is not included in the data returned, but no error is reported.

The user must be identified to RACF with a valid SECLABEL. If not, an authorization error and audit record (IFCID 140) are produced, provided the audit trace is active.

INSERT

If a user does not have the write-down privilege and write-down control is enabled, the value of the SECLABEL in the inserted row is set to the value of the user’s SECLABEL. If the user does have the write-down privilege or write-down checking is not in effect, the user can set the value of the SECLABEL column to any defined but not disjoint SECLABEL. This means the row can be moved to higher or lower security classification. An important effect of having write-down authority is that a user can potentially insert a row with a higher security label that the user cannot SELECT later.

UPDATE

For accessing rows to be updated, the rules for UPDATE are similar to those for SELECT. For setting the SECLABEL during the UPDATE, the rules are similar to those for INSERT. UPDATE requires SECLABEL equivalence for users who are not allowed to write down. If the user has write-down authority, rows with a lower-level SECLABEL can be accessed and updated to a SECLABEL that can be higher or lower than the user’s SECLABEL.

The user's SECLABEL is compared to the SECLABEL of the row to be updated. The update proceeds according to the following rules:

- If the security label of the user and the security label of the row are equivalent, the row is updated and the value of the security label is determined by whether the user has write-down privilege:
  - If the user has write-down privilege or write-down control is not enabled, the user can set the security label of the row to a valid security label that is not disjoint in relation to the user’s security. However, if no value is specified for the security label, the security label of the row is set to the value of the security label of the user.
  - If the user does not have write-down privilege and write-down control is enabled, the security label of the row is set to the value of the security label of the user.

- If the security label of the user is greater than the security label of the row, the result of the UPDATE statement is determined by whether the user has write-down privilege:
  - If the user has write-down privilege or write-down control is not enabled, the down-level row is updated and the user can set the security label of the row to a valid security label that is not disjoint in relation to the user's security. However, if no value is specified for the security label, the security label of the row is set to the value of the security label of the user.
  - If the user does not have write-down privilege and write-down control is enabled, the row is not updated.
- If the user's security label is less than the row security label, the row will not be updated.

The user must be identified to RACF with a valid SECLABEL. If not, an authorization error and an audit record are produced.

DELETE

Delete operations in a multilevel security with row granularity environment proceed according to the following rules:

- If the security label of the user and the security label of the row are equivalent, the row is deleted.
- If the security label of the user is greater than the security label of the row, the user's write-down privilege determines the result of the DELETE statement:
  - If the user has write-down privilege or write-down control is not enabled, the down-level row is deleted.
  - If the user does not have write-down privilege and write-down control is enabled, the down-level row is not deleted.
- If the user's security label is less than the row security label, the row will not be deleted.

7.2.7 DDL for CLONE table

A clone table is a table that is structurally identical to a base table. A clone table is created from a base table by using the ALTER TABLE statement that includes an ADD CLONE clause that specifies the name of the clone table. The base table and the clone table each have separate underlying VSAM data sets (identified by their data set instance numbers) that contain separate rows of data.

When the base table contains a column defined with the AS SECURITY LABEL clause, a clone created from this base table contains an identical column with the AS SECURITY LABEL clause.

7.2.8 Summary

DB2 V8 introduced a new facility for securing data based on the RACF security label facility. Use of security labels provides some significant improvements over the methods previously described:

- Users no longer have to develop their own security code.
- There is much less need to propagate redundant data.
- The performance processing time is slight.
- Security auditors are usually familiar with, and satisfied by, RACF’s lack of vulnerability and will therefore grant certification more readily.

In outline, implementing security labels for DB2 involves the following steps:

1. Identify which users and groups require what access to which rows of which tables.
2. Design a set of security labels for users and table rows that reflects the result of step 1.
3. Get the RACF administrators to define that set in RACF, and then activate the RACF SECLABEL class.
4. Add a security label column to each table requiring row-level security. This process assigns an initial default value to every row.
5. Update the security labels of the rows to appropriate values.

After you have done this, you have controlled row-level access to your data. Data that users are not allowed to see is now invisible to them. They cannot update or delete such data. Aggregation functions such as SUM or AVG disregard rows that are not authorized to the user. From the point of view of the unauthorized user, the data has disappeared, whether they use SQL applications or low-level access methods such as utilities.

However, if you have followed the previous implementation outline, all users will have write-down authority. There are basically two approaches to preventing users from reclassifying data. You can have write-down control enforced by the system by enabling the RACF option MLS(FAILURES), otherwise known as enabling write-down control, or you can code your own mechanisms.

7.3 Additional considerations about row-level security

In this section, we discuss additional considerations about row-level security.

7.3.1 DB2 utilities and multilevel security

Some of the utilities behave slightly differently when you implement row-level security. Here, we describe how they work.

Why utilities are affected

There is little point in going to all the trouble of protecting your sensitive data from unauthorized access and manipulation through applications and query tools if people can use DB2 utilities to bypass the controls you put in place. You do not want people using UNLOAD or REORG UNLOAD to extract the sensitive data to data sets that they can browse at their leisure; you do not want them to delete it by means of REORG DISCARD or LOAD REPLACE; and you do not want them adding data with inappropriate security labels by means of LOAD.

The IBM LOAD, UNLOAD, and REORG utilities have been enhanced to cope with row-level security, and we discuss these changes in this chapter. It is beyond the scope of this book to examine the various equivalent products provided by other vendors. If you are implementing row-level security and have such third-party products, check that they offer similar protections in order to avoid compromising your carefully designed security scheme.

Some of the stand-alone utilities can also represent a security exposure.

Only the utilities that insert and delete data have the new multilevel security access controls, but other utilities are not changed. An administrator who has access to all of the data will generally be running these utilities. If a user who does not have access to all of the data is running the utilities, the user will only have access to the rows whose security labels her security label dominates.

Executing the following utilities against a table space containing tables with multilevel security with row granularity requires that the user be identified to RACF and have a valid security label. If the user does not have a valid SECLABEL, an authorization error message and an audit trace record (IFCID 140) is produced, provided audit trace class 1 is active.

UNLOAD and REORG UNLOAD EXTERNAL

The UNLOAD and REORG UNLOAD EXTERNAL utilities use rules similar to SELECT. These utilities read information, such as the SELECT statement, so the authorization is similar to SELECT rules. Rows can only be unloaded if the user security label dominates the data security label. Rows for which this is not true are not unloaded and no error is returned.

UNLOAD and REORG UNLOAD enable you to extract data from a DB2 table and put the results into sequential files quickly and easily. You can use a WHEN clause on the control statement to extract a subset of rows rather than the whole table. REORG UNLOAD only operates at a complete row level and does not allow you to reformat the data (other than translating from internal formats understood only by DB2 to external character format). UNLOAD has a richer range of options, allowing you to extract a subset of columns, reorder columns, and reformat the output data.

If the table from which you are extracting has row-level security, UNLOAD and REORG UNLOAD only extract the rows that would be accessible to you in an SQL SELECT statement. If your SECLABEL dominates a row's SECLABEL, the row is unloaded; otherwise, the row is not unloaded and no error is returned. There is no indication that rows have not been extracted because the user was not authorized to access them.

The UNLOAD utility has a SAMPLE option, allowing you to specify a percentage of rows to extract. You can specify the percentage to a precision of up to four decimal places, for example, 25, 2.5, or 0.0025. The sampling filter operates only on the rows the user is allowed to access, so it is not possible for an unauthorized user to calculate the total size of the table.

We created a table with 1,000,000 rows and ran UNLOAD with a SAMPLE of 1. With a SECLABEL of SYSHIGH, the sample was 10,000 rows. With a SECLABEL of L2C, only 140,000 rows should have been accessible, and the UNLOAD correctly unloaded only 1400 rows.

UNLOAD (but not REORG UNLOAD) can also use an image copy as input. It behaves just the same on an image copy, correctly honoring SECLABELs. The test using a user ID with an L2C SECLABEL again unloaded 1400 rows as a 1% sample.

LOAD RESUME

The LOAD RESUME utility adheres to the same rules as INSERT. Without write-down authorization, the SECLABEL in the rows of the table are set to the SECLABEL associated with the user ID executing the LOAD RESUME utility. With write-down, a valid SECLABEL that is not disjoint in relation to the user's security label can be specified.

LOAD REPLACE

The LOAD REPLACE utility deletes all rows in a table space. Therefore, write-down authority is required. If the user ID associated with the job that is running the LOAD REPLACE utility does not have write-down privilege, and write-down is in effect, an error message is issued.

REORG DISCARD

The REORG utility with the DISCARD option adheres to the same rules as the SQL DELETE statement. For each row unloaded from those tables, if the row qualifies to be discarded, the user's SECLABEL is compared to the data's SECLABEL:

- If the SECLABELs are the same, the row is discarded.
- If the SECLABELs are not the same, equivalence is checked:
  - If the SECLABELs are equivalent, the row is discarded.
  - If not, a check is done to see if write-down privilege is in effect:
    - If write-down privilege is in effect, and the user has write-down, rows that are dominated are discarded.
    - If write-down privilege is in effect, and the user does not have write-down, the row is not considered to be a match and the row is not discarded.

DSN1COPY, DSN1PRNT, and DSN1LOGP

DSN1* utilities operate directly on data sets without going through the DB2 subsystem. They are unable to detect that data in a table space is secured.

This represents a serious security exposure if you do not take steps to prevent unauthorized activity.

Suppose you have a table PROD.CUSTOMERS that has a column CLASSIFICATION that is defined as a security label column. Someone with the general SELECT privilege on the table, even though his own SECLABEL would only allow him access to a subset of the rows in it, could issue the following SQL:

```
CREATE TABLE MY.CUSTOMERS LIKE PROD.CUSTOMERS                            IN
MYDBASE.MYTSPACE
```

When you use CREATE TABLE LIKE, the column definitions are copied from the source table, but not the security attributes. So although PROD.CUSTOMERS.CLASSIFICATION is defined as CHAR(8) NOT NULL WITH DEFAULT AS SECURITY LABEL, the new column MY.CUSTOMERS.CLASSIFICATION is set up as a plain CHAR(8) NOT NULL WITH DEFAULT.

Having created this duplicate table, our user can potentially use DSN1COPY with OBID translation to directly copy the table space data set containing PROD.CUSTOMERS to the duplicate, giving himself an unsecured copy of the table to which he would have complete access.

Users do not even have to set up a duplicate table, though. If users are simply going through a listing, they can use DSN1COPY or DSN1PRNT to print out the contents of the table space data set, and thereby get to see everything in the table, not just the rows that they are authorized to see.

And the risk is not only from the database data sets. DSN1COPY and DSN1PRNT can also use image copies as input.

DSN1LOGP prints out the contents of the DB2 log, so a determined user could go through the log and examine the activity on any secured tables without going through SECLABEL checking. By default, DB2 only logs the changed portion of a row, which would make it difficult, though not impossible, to work out whether an update was on sensitive data. However, there is an option, DATA CAPTURE CHANGES, that can be set for individual tables and has the effect of making DB2 log the entire row image, which makes examination of the log for sensitive transactions much easier.

Data sets for copies, work files, and log files need to be protected. DSN1* utilities require access control for the data sets. Use RACF controls for the data sets at the highest level of data within the data set. For better control of the other utilities, use RACF access control.

RUNSTATS and COLCARDF for SECLABEL column

There is no special handling of a security label column, so COLCARDF will contain the number of distinct security label values.

7.3.2 Security labels and indexes

The performance of accessing tables that contain a security label can be impacted if the SECLABEL column is not included in the indexes. The security label column is used whenever a table with multilevel security enabled is accessed. Therefore, it is a good idea to include the SECLABEL column in your existing indexes, especially when your queries are using index-only access today.

However, adding a column to a unique index can affect applications and data, so a review of data and application design will be required.

7.3.3 Restrictions when using multilevel security with row granularity

In the following paragraphs, we describe some of the restrictions that you need to be aware of when planning to use multilevel security with row granularity.

Sysplex query parallelism

Sysplex query parallelism cannot be used for queries that access tables that have a SECLABEL column defined.

Global temporary tables

For a declared temporary table with a column definition, no syntax exists to specify a security label on a DECLARE GLOBAL TEMPORARY TABLE statement. An attempt to specify a security label results in an error. If a DECLARE GLOBAL TEMPORARY TABLE statement uses a fullselect or a LIKE predicate, or a CREATE GLOBAL TEMPORARY TABLE statement uses a LIKE predicate, the resulting temporary table can inherit the security label column from the referenced table or view.

However, the temporary table does not inherit any security attributes on that column. That means that the inherited column in the temporary table is not defined AS SECURITY LABEL. The column in the temporary table is defined as NOT NULL, with no default. Therefore, any statements that insert data in the temporary table must provide a value for the inherited column.

Materialized query tables

A materialized query table (MQT) is a table that contains information that is derived and summarized from other tables. An MQT is user-created and specifies a fullselect on the CREATE TABLE statement. As a result, an MQT can contain the results of queries with expensive join and aggregation operations. Depending on how the table is defined, it is user-maintained or system-maintained.

If one or more of the source tables for a materialized query table has multilevel security with row-level granularity enabled, some additional rules apply to working with the materialized query table and the source tables:

- When creating a materialized query table, if any of the source tables in the fullselect of the materialized query table definition contains a security label column:
  - If only one source table contains a security label column and the materialized query table is defined with the DEFINITION ONLY clause, the materialized query table inherits the values in the security label column from the source table. However, the inherited column is not a security label column.
  - If only one source table contains a security label column, the security label column must be included in the materialized query table definition with the AS SECURITY LABEL clause. The materialized query table will inherit the security label column from the source table. The MAINTAINED BY USER option is allowed.
  - If more than one source table contains a security label column, DB2 returns an error code and the materialized query table is not created.

- Using an ALTER TABLE statement to add a security label column to a table will fail if the table is a source table for a materialized query table.
- When using the REFRESH TABLE statement to delete the data currently in a materialized query table and then to repopulate the materialized query table by executing the fullselect, DB2 does not check for multilevel security with row-level granularity. The row-level granularity check is enforced when using the materialized query table, either by exploiting the MQT or by using the MQT directly.

Constraints

A unique constraint is allowed on a security label column.

A referential constraint is not allowed on a security label column. Although a referential constraint is not allowed for the security label column, DB2 enforces referential constraints for other columns in the table that are not defined with a security label.

A check constraint is not allowed on a security label column.

Row-level security checking is not enforced for referential constraints.

Field procedures, edit procedures, validation procedures

You cannot define a field procedure on a SECLABEL column or an editproc on a table that has a SECLABEL column.

Validation procedures are allowed on a table that is defined with a security label column. When an authorized user with write-down privilege makes an INSERT or UPDATE request for a row, the validation procedure passes the new row with the security label of the user. If the authorized user does not have write-down privilege, the security label of the row remains the same.

Triggers

When a transition table is generated as the result of a trigger, the security label of rows from the original table are not inherited by the transition table. Therefore, multilevel security with row-level checking is not enforced for transition tables and transition values.

If an ALTER TABLE statement is used to add a security label column to a table with a trigger on it, the same rules apply to the new security label column that would apply to any column that is added to the table with the trigger on it.

When a BEFORE trigger is activated, the value of the NEW transition variable that corresponds to the security label column is set to the security label of the user if either of the following criteria are met:

- Write-down control is in effect and the user does not have the write-down privilege.
- The value of the security label column is not specified.

Common Criteria PTF UK11425

DB2 V8 will be certified for the Common Criteria at level EAL 3+ in 2007. If you have the Common Criteria PTF UK11425 applied to your DB2 V8 system and you have the new system parameter COMCRIT set equal to YES, the operation of the CREATE TABLE, ALTER TABLE, CREATE TEMPORARY TABLE, and DECLARE TEMPORARY TABLE statements and materialized query tables are affected as described here.

Note: DB2 V9 has not undergone a Common Criteria evaluation as of this writing.

This PTF provides support for COMCRIT, a new system parameter. COMCRIT is online, updateable, and can be set to NO or YES. For this setting:

- NO is the default value. A value of NO results in compatible behavior and does not change the current operation of DB2.
- YES activates the Common Criteria environment and requires that every new table that is created has a security label column, which enables multilevel security. If the AS SECURITY LABEL clause is missing from a CREATE TABLE statement, DB2 issues an error and the table is not created. Existing tables are not affected. Use the same value of COMCRIT for all members of a DB2 data sharing system.

SQL restrictions

When DB2 is started in a Common Criteria environment, DB2 issues the new SQLCODE -4708 under the following circumstances:

- Whenever a CREATE TABLE statement does not include a column with the AS SECURITY LABEL clause. Every normal base table must include a security label column in a Common Criteria environment.
- Whenever a CREATE or ALTER TABLE statement attempts to define a materialized query table. You cannot define materialized query tables in a Common Criteria environment.

- Whenever the LIKE or AS (fullselect) clauses are specified as part of a CREATE TABLE or DECLARE GLOBAL TEMPORARY TABLE statement. These clauses are not supported in a Common Criteria environment.

7.3.4 DB2 session variable

The security label of the current process is available using the GETVARIABLE function, specifying SYSIBM.SECLABEL. For instance, if the table column you defined AS SECURITY LABEL is called CLASSIFICATION, a program can retrieve two columns, c1 and c2, from all the rows that have the same SECLABEL as the user with SQL, such as this:

```
SELECT c1, c2 FROM table WHERE CLASSIFICATION =
GETVARIABLE(‘SYSIBM.SECLABEL’)
```

7.3.5 Using views to restrict access

If you do not want users to see a security label column, you can create views that do not include the column. To do so, you can use the SQL statements as shown in Example 7-1.

Example 7-1 CREATE VIEW SQL to create a view without security column

```
CREATE VIEW view AS
  SELECT columns but excluding the security column
    FROM table
```

Alternatively, you can create views that give each user access only to the rows that include that user’s security label column. To do so, retrieve the value of the SYSIBM.SECLABEL session variable, and create a view that includes only the rows that match the session variable value. To allow access only to the rows that match the user’s security label, use the CREATE statement (Example 7-2).

Example 7-2 CREATE VIEW SQL to access only rows with user’s SECLABEL

```
CREATE VIEW view AS
  SELECT * FROM table
     WHERE SECURITY = GETVARIABLE(‘SYSIBM.SECLABEL’)
```

7.4 DB2 multilevel security implementation at the object level

When implementing multilevel security at the object level, you are responsible for ensuring a proper hierarchy of security labels. In general, the security label of an object that is higher in the object hierarchy dominates the security labels of objects that are lower in the hierarchy. RACF and DB2 do not enforce the hierarchy, but they enforce the dominance rules that you establish.

You can use RACF to define security labels for the DB2 objects in the following object hierarchy:

- Subsystem or data sharing group
  - Database
    Table space
    - Table
    - Column
    - Row
  - View
  - Storage group
  - Buffer pool
  - Plan
  - Collection
    - Package
  - Schema
    - Stored procedure or user-defined function
    - Java archive (JAR)
    - Distinct type
    - Sequence

When the hierarchy is set up and which objects will require mandatory access checking is decided, define security labels in RACF by using the RDEFINE command. The corresponding RACF resource classes in which you have to define a SECLABEL in order to implement multilevel security at the object level are:

- DSNADM for administrative authorities
- DSNR for access to DB2 subsystems
- MDSNBP and GSNBP for buffer pools
- MDSNCL and GDSNCL for collections
- MDSNJR and MDSNJR for Java archive
- MDSNPN and GDSNPN for plans
- MDSNSC and GDSNSC for schema
- MDSNSG and GDSNSG for storage groups
- MDSNSM and GDSNSM for system privileges

- MDSNSP and GDSNSP for stored procedures
- MDSNSQ and GDSNSQ for sequences
- MDSNTB and GDSNTB for tables, views, and indexes
- MDSNTS and GDSNTS for table spaces
- MDSNUF and GDSNUF for user-defined functions

When using multilevel security at the object level, mandatory access checking evaluates dominance and determines whether to allow certain actions, based on the following rules:

- If the security label of the user dominates the security label of the object, the user can read from the object.
- If the security label of a user and the security label of the object are equivalent, the user can read from and write to the object.
- If the security label of the object dominates the security label of the user or the security labels are incompatible, the user cannot read or write from the object.

The required DB2 environment to perform multilevel security at the object level is that the RACF SECLABEL class must be active and that DB2 must be running with the RACF external security active, which means running with the RACF access control module DSNX@XAC. Note that it is not required to have MLACTIVE on for multilevel security at the object level.

In this environment, if specified, object SECLABELs and user SECLABELs are compared to perform mandatory access checking. However, there is one exception to this general rule. User IDs with Install SYSADM authority bypass mandatory access checking at the DB2 object level because actions by Install SYSADM do not invoke the external access control exit routine DSNX@XAC. Note that this exception does not apply to the DB2 row-level security implementation where multilevel security with row-level granularity is also enforced for user IDs with the Install SYSADM authority.

Write-down rules also apply to the object-level implementation. Write-down rules are governed by RACF. Refer to 7.2.4, “Write-down in DB2” on page 142 for more information.

We go through a sample scenario, showing the required commands for implementing and performing multilevel security checking at the object level. See Chapter 10, “RACF access control module” on page 279.

7.5 Sample scenario

In this section, we review the sample scenario that exercises multilevel security at the row-level, first with write-down control not enabled and then with write-down control enabled.

7.5.1 Preparation steps

To set up the basic environment, we first need to create SECLABELs on the user IDs that we are using for this chapter. For further information about SECLABELs, see Chapter 2, “Security labels” on page 11.

We first show how we created a security label L2C, and then how we assigned user ID USRT070 to use this security label.

A security label consists of two parts: one security level and zero or more categories. The different possible levels and categories are defined in the RACF SECDATA class. Our RACF administrator used the commands shown in Example 7-3. The first command defines the SECLEVEL profile with universal access NONE in the SECDATA class. Then, four possible name/value levels are added to this profile with the ADDMEM option. Similar commands are executed to add a CATEGORY profile to the SECDATA class that contains five possible categories that we decided to use.

Example 7-3 Define levels and categories for SECLABELs

```
************************************************************************
*  Define security levels                                              *
************************************************************************
 RDEFINE SECDATA SECLEVEL UACC(NONE)
 RALTER  SECDATA SECLEVEL ADDMEM(L1/10 L2/20 L3/30 L4/40)
************************************************************************
*  Define security categories                                          *
************************************************************************
 RDEFINE SECDATA CATEGORY UACC(NONE)
 RALTER  SECDATA CATEGORY ADDMEM(A B C D E)
************************************************************************
```

The next step was to define the SECLABEL itself. Example 7-4 shows the RDEF RACF command to do so. It defines that the SECLABEL profile L2C will have L2 access for one category, C, and that universal access is NONE.

Example 7-4 Define a SECLABEL with level and one category

```
************************************************************************
*  Define security labels with level and categories                    *
************************************************************************
.
.
RDEF SECLABEL L2C SECLEVEL(L2) ADDCATEGORY(C) U(NONE)
.
.
```

To be able to use a SECLABEL, a user must have READ access on the SECLABEL resource profile. When a profile is added, a REFRESH of the class is also required to make it active. Example 7-5 shows the commands to give user USRT070 READ access to the L2C SECLABEL and make it active. If handling many users in combination with many different SECLABELs, there are better ways to manage access lists in RACF. For more of an explanation and tips about managing groups of users, see Chapter 2, “Security labels” on page 11.

Example 7-5 Permit USRT070 to use the L2C SECLABEL and refresh

```
************************************************************************
*  Permit USRT070 on the L2C profile and refresh                       *
************************************************************************
PERMIT L2C CLASS(SECLABEL) ID(USRT070) ACC(READ)
SETROPTS RACLIST(SECLABEL) REFRESH
```

The sample EMP table, as provided by DB2, is created with an editproc. The scenario that follows is based on the EMP table, and the intention is to add a SECLABEL column to the EMP table. Therefore, we removed the EDITPROC DSN8EAE1 statement from the DSNTEJ1 job before creating the EMP table.

After the EMP table was loaded, we used user ID MLS4 to alter the EMP table to add a security label column. This failed because the RACF SECLABEL class was not activated yet. The result of this is shown in Example 7-6.

Example 7-6 Alter table with SECLABEL class inactive

```
ALTER TABLE DSN8810.EMP
      ADD CLASSIFICATION    CHAR(8) FOR SBCS DATA
                            NOT NULL WITH DEFAULT
                            AS SECURITY LABEL;
---------+---------+---------+---------+---------+---------+---------+--
DSNT408I SQLCODE = -20265, ERROR:  SECURITY LABEL IS BLANK FOR MLS4
DSNT418I SQLSTATE   = 42501 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXISB5 SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = 285 0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'0000011D'  X'00000000'  X'00000000'  X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
---------+---------+---------+---------+---------+---------+---------+--
DSNE618I ROLLBACK PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

To activate the SECLABEL class, we submitted the job shown in Example 7-7.

Example 7-7 Making the SECLABEL class active

```
//RUNDB2   EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD   SYSOUT=*
//SYSPRINT DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=*
//SYSTSIN  DD   *
 SETROPTS CLASSACT(SECLABEL)
 SETROPTS RACLIST(SECLABEL) REFRESH
/*
```

A display of the RACF options using SETROPTS LIST shows that our current environment now has the following characteristics:

- SECLABEL class active.
- MULTI-LEVEL ACTIVE not in effect.

  This means that we are not running in a full MLACTIVE environment.
- NO WRITE-DOWN checking is in effect.

  If “no write-down is not in effect” is displayed in a SETROTS LIST, all DB2 users with valid security labels are equivalent to users having the write-down privilege when write-down is enabled.

We must mention that we were running DB2 without that RACF access control module so that native DB2 security was active for resource access control.

Now that the SECLABELs are active, before continuing in this environment, we stopped and restarted DB2 and logged off and back on again with our TSO sessions. We did this to assign the security labels to all applications.

We re-submitted the alter job, and this time we got the result shown in Example 7-8. The SECLABEL of the TSO user ID used for SPUFI was SYSHIGH. The first SQL statement executed added a column named CLASSIFICATION to contain the security labels. We then listed the EMP table to verify that all columns received the default security label value of SYSHIGH. Then, we ran different SQL UPDATE requests to change the security labels to our needs:

- Only employee number 000010 receives a SECLABEL L4A.
- All managers receive L3. The security label name is built by concatenating the first letter of the WORKDEPT value to the security level value.
- All operators receive L2N.
- All clerks receive L1. Here also the first letter of WORKDEPT is used as a category.
- All others receive L2; again the WORKDEPT is used to make up the SECLABEL name.

Finally, we listed the EMP table again to check our changes to the CLASSIFICATION column. Note that assigning different SECLABELs was only possible because we, just like any other DB2 user, were running as though we had the write-down privilege.

Example 7-8 Successfully added the SECLABEL column

```
ALTER TABLE DSN8810.EMP
      ADD CLASSIFICATION    CHAR(8) FOR SBCS DATA
                            NOT NULL WITH DEFAULT
                            AS SECURITY LABEL;
---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+
 COMMIT;
---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+

SELECT EMPNO,LASTNAME,JOB,WORKDEPT,CLASSIFICATION
   FROM DSN8810.EMP;
---------+---------+---------+---------+---------+---------+
EMPNO   LASTNAME         JOB       WORKDEPT  CLASSIFICATION
---------+---------+---------+---------+---------+---------+
000010  HAAS             PRES      A00       SYSHIGH
000020  THOMPSON         MANAGER   B01       SYSHIGH
000030  KWAN             MANAGER   C01       SYSHIGH
000050  GEYER            MANAGER   E01       SYSHIGH
000060  STERN            MANAGER   D11       SYSHIGH
000070  PULASKI          MANAGER   D21       SYSHIGH
000090  HENDERSON        MANAGER   E11       SYSHIGH
000100  SPENSER          MANAGER   E21       SYSHIGH
000110  LUCCHESI         SALESREP  A00       SYSHIGH
000120  O'CONNELL        CLERK     A00       SYSHIGH
... (rows omitted)
200240  MONTEVERDE       CLERK     D21       SYSHIGH
200280  SCHWARTZ         OPERATOR  E11       SYSHIGH
200310  SPRINGER         OPERATOR  E11       SYSHIGH
200330  WONG             FIELDREP  E21       SYSHIGH
200340  ALONZO           FIELDREP  E21       SYSHIGH
DSNE610I NUMBER OF ROWS DISPLAYED IS 42
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+

UPDATE DSN8810.EMP
 SET CLASSIFICATION = 'L4A' WHERE EMPNO = '000010';
---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+

UPDATE DSN8810.EMP
 SET CLASSIFICATION = 'L3'||SUBSTR(WORKDEPT,1,1)
 WHERE JOB = 'MANAGER';
---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 7
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+

UPDATE DSN8810.EMP
 SET CLASSIFICATION = 'L2N'
 WHERE JOB = 'OPERATOR';
---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 6
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+

UPDATE DSN8810.EMP
 SET CLASSIFICATION = 'L2'||SUBSTR(WORKDEPT,1,1)
 WHERE JOB NOT IN ('PRES', 'MANAGER', 'CLERK', 'OPERATOR');
---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 20
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+

UPDATE DSN8810.EMP
 SET CLASSIFICATION = 'L1'||SUBSTR(WORKDEPT,1,1)
 WHERE JOB = 'CLERK';
---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 8
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+

SELECT EMPNO,LASTNAME,JOB,WORKDEPT,CLASSIFICATION
   FROM DSN8810.EMP;
---------+---------+---------+---------+---------+---------+
EMPNO   LASTNAME         JOB       WORKDEPT  CLASSIFICATION
---------+---------+---------+---------+---------+---------+
000010  HAAS             PRES      A00       L4A
000020  THOMPSON         MANAGER   B01       L3B
000030  KWAN             MANAGER   C01       L3C
000050  GEYER            MANAGER   E01       L3E
000060  STERN            MANAGER   D11       L3D
000070  PULASKI          MANAGER   D21       L3D
000090  HENDERSON        MANAGER   E11       L3E
000100  SPENSER          MANAGER   E21       L3E
000110  LUCCHESI         SALESREP  A00       L2A
000120  O'CONNELL        CLERK     A00       L1A
000130  QUINTANA         ANALYST   C01       L2C
000140  NICHOLLS         ANALYST   C01       L2C
000150  ADAMSON          DESIGNER  D11       L2D
000160  PIANKA           DESIGNER  D11       L2D
000170  YOSHIMURA        DESIGNER  D11       L2D
000180  SCOUTTEN         DESIGNER  D11       L2D
000190  WALKER           DESIGNER  D11       L2D
000200  BROWN            DESIGNER  D11       L2D
000210  JONES            DESIGNER  D11       L2D
000220  LUTZ             DESIGNER  D11       L2D
000230  JEFFERSON        CLERK     D21       L1D
000240  MARINO           CLERK     D21       L1D
000250  SMITH            CLERK     D21       L1D
000260  JOHNSON          CLERK     D21       L1D
000270  PEREZ            CLERK     D21       L1D
000280  SCHNEIDER        OPERATOR  E11       L2N
000290  PARKER           OPERATOR  E11       L2N
000300  SMITH            OPERATOR  E11       L2N
000310  SETRIGHT         OPERATOR  E11       L2N
000320  MEHTA            FIELDREP  E21       L2E
000330  LEE              FIELDREP  E21       L2E
000340  GOUNOT           FIELDREP  E21       L2E
```

```
200010  HEMMINGER        SALESREP  A00       L2A
200120  ORLANDO          CLERK     A00       L1A
200140  NATZ             ANALYST   C01       L2C
200170  YAMAMOTO         DESIGNER  D11       L2D
200220  JOHN             DESIGNER  D11       L2D
200240  MONTEVERDE       CLERK     D21       L1D
200280  SCHWARTZ         OPERATOR  E11       L2N
200310  SPRINGER         OPERATOR  E11       L2N
200330  WONG             FIELDREP  E21       L2E
200340  ALONZO           FIELDREP  E21       L2E
DSNE610I NUMBER OF ROWS DISPLAYED IS 42
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
COMMIT;
---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+
---------+---------+---------+---------+---------+---------+
DSNE617I COMMIT PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

Important: The sample DB2 application tables are very small. For the sake of speed and simplicity of illustration, we did not go through all the steps that a normal production system requires. We describe a more rigorous plan of action in 7.6, “Real-world implementation” on page 186.

7.5.2 Row-level security applied on SELECT

The security rule for SELECT is that the user's SECLABEL is compared to the data’s SECLABEL of the row to be selected. If user’s SECLABEL dominates the data’s SECLABEL, the row is returned. If user’s SECLABEL does not dominate the data’s SECLABEL, the row is not included in the returned data and no error is reported.

We started by doing a TSO logon with user ID USRT045 running with security label L4ABCDE. This is the highest possible security level and it is able to read all categories. Example 7-9 on page 166 shows the result of a sample SELECT query that we will repeat here with different security labels. The output shows that L4ABCDE dominates the security levels of all data rows in the table so that the returned result contains the 42 rows that we updated in Example 7-8 on page 162. For this exercise, we added the column defined with the AS SECURITY LABEL attribute to the query. In a real-life situation, this is probably not the intention. Users or applications must not be aware of this column. The column can be defined as IMPLICITLY HIDDEN. This is a new DB2 V9 clause.

Example 7-9 SELECT result with SECLABEL L4ABCDE

```
SELECT EMPNO,
       LASTNAME,
       WORKDEPT,
       SALARY,
       CLASSIFICATION
  FROM DSN8810.EMP
---------+---------+---------+---------+---------+---------+--
EMPNO   LASTNAME         WORKDEPT       SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+--
000010  HAAS             A00          52750.00  L4A
000020  THOMPSON         B01          41250.00  L3B
000030  KWAN             C01          38250.00  L3C
000050  GEYER            E01          40175.00  L3E
000060  STERN            D11          32250.00  L3D
000070  PULASKI          D21          36170.00  L3D
000090  HENDERSON        E11          29750.00  L3E
000100  SPENSER          E21          26150.00  L3E
000110  LUCCHESI         A00          46500.00  L2A
000120  O'CONNELL        A00          29250.00  L1A
000130  QUINTANA         C01          23800.00  L2C
000140  NICHOLLS         C01          28420.00  L2C
000150  ADAMSON          D11          25280.00  L2D
000160  PIANKA           D11          22250.00  L2D
000170  YOSHIMURA        D11          24680.00  L2D
000180  SCOUTTEN         D11          21340.00  L2D
000190  WALKER           D11          20450.00  L2D
000200  BROWN            D11          27740.00  L2D
000210  JONES            D11          18270.00  L2D
000220  LUTZ             D11          29840.00  L2D
000230  JEFFERSON        D21          22180.00  L1D
000240  MARINO           D21          28760.00  L1D
000250  SMITH            D21          19180.00  L1D
000260  JOHNSON          D21          17250.00  L1D
000270  PEREZ            D21          27380.00  L1D
000280  SCHNEIDER        E11          26250.00  L2N
000290  PARKER           E11          15340.00  L2N
000300  SMITH            E11          17750.00  L2N
000310  SETRIGHT         E11          15900.00  L2N
000320  MEHTA            E21          19950.00  L2E
000330  LEE              E21          25370.00  L2E
000340  GOUNOT           E21          23840.00  L2E
200010  HEMMINGER        A00          46500.00  L2A
200120  ORLANDO          A00          29250.00  L1A
200140  NATZ             C01          28420.00  L2C
200170  YAMAMOTO         D11          24680.00  L2D
200220  JOHN             D11          29840.00  L2D
200240  MONTEVERDE       D21          28760.00  L1D
```

```
200280  SCHWARTZ        E11         26250.00  L2N
200310  SPRINGER        E11         15900.00  L2N
200330  WONG            E21         25370.00  L2E
200340  ALONZO          E21         23840.00  L2E
DSNE610I NUMBER OF ROWS DISPLAYED IS 42
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

We logged on with user ID USRT053 with security label L3BCD assigned. Example 7-10 shows the result of the same SELECT query, but this time we have only columns with L1, L2, and L3, and this for the categories B, C, and D. We also received the rows with the security label L2N that was defined with RACF command RDEF SECLABEL L2N SECLEVEL(L2) UACC(NONE). This security label contains the mandatory level specification but no category. To be dominant, a security label has to include the categories of the second security label. Because the second secondary label has no categories at all, it is normal that we have the rows with uncategorized data in our result.

Rows with security level L4 or belonging to the categories A and E, or both, are not shown because the security label L3BCD is not dominant to these rows.

Example 7-10 SELECT result with SECLABEL L3BCD

```
SELECT EMPNO,
       LASTNAME,
       WORKDEPT,
       SALARY,
       CLASSIFICATION
  FROM DSN8810.EMP
---------+---------+---------+---------+---------+---------+--
EMPNO   LASTNAME        WORKDEPT       SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+--
000020  THOMPSON        B01         41250.00  L3B
000030  KWAN            C01         38250.00  L3C
000060  STERN           D11         32250.00  L3D
000070  PULASKI         D21         36170.00  L3D
000130  QUINTANA        C01         23800.00  L2C
000140  NICHOLLS        C01         28420.00  L2C
000150  ADAMSON         D11         25280.00  L2D
000160  PIANKA          D11         22250.00  L2D
000170  YOSHIMURA       D11         24680.00  L2D
000180  SCOUTTEN        D11         21340.00  L2D
000190  WALKER          D11         20450.00  L2D
000200  BROWN           D11         27740.00  L2D
000210  JONES           D11         18270.00  L2D
000220  LUTZ            D11         29840.00  L2D
000230  JEFFERSON       D21         22180.00  L1D
000240  MARINO          D21         28760.00  L1D
000250  SMITH           D21         19180.00  L1D
```

```
000260  JOHNSON          D21          17250.00  L1D
000270  PEREZ            D21          27380.00  L1D
000280  SCHNEIDER        E11          26250.00  L2N
000290  PARKER           E11          15340.00  L2N
000300  SMITH            E11          17750.00  L2N
000310  SETRIGHT         E11          15900.00  L2N
200140  NATZ             C01          28420.00  L2C
200170  YAMAMOTO         D11          24680.00  L2D
200220  JOHN             D11          29840.00  L2D
200240  MONTEVERDE       D21          28760.00  L1D
200280  SCHWARTZ         E11          26250.00  L2N
200310  SPRINGER         E11          15900.00  L2N
DSNE610I NUMBER OF ROWS DISPLAYED IS 29
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

We now logged on with user ID USRT041 running with security label L1B. Example 7-11 shows the result of our SELECT. Because the EMP table does not contain any L1B security level, no data rows are returned, and the user receives a normal SQLCODE 100. The result of the select depends on the security label that the user has and is totally transparent to the user. We receive the same result when we were executing this query with user ID USRT044 running with security label SYSLOW, which means that its security level is equal to the lowest security level defined in the system and has no categories assigned.

Example 7-11 SELECT result with SECLABEL L1B

```
SELECT EMPNO,
       LASTNAME,
       WORKDEPT,
       SALARY,
       CLASSIFICATION
  FROM DSN8810.EMP;
---------+---------+---------+---------+---------+---------+--
EMPNO   LASTNAME         WORKDEPT       SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+--
DSNE610I NUMBER OF ROWS DISPLAYED IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

Not only the result of selecting rows, but also report functions, will give different results depending on the security label that a user has to read a table. In Example 7-12 on page 169, you see the result of a SUMMARY function on the EMP table when read by user ID USRT045 who has security label L4ABCDE. These are totals for the whole organization as seen by the company's president. The result shows all different departments in the company and includes all data, including the uncategorized data.

Example 7-12 SUMMARY report results when read with L4ABCDE

```
SELECT WORKDEPT,
       SUM(SALARY)
   FROM DSN8810.EMP
   GROUP BY WORKDEPT;
---------+---------+---------+---------+---------+---------
WORKDEPT
---------+---------+---------+---------+---------+---------
A00               204250.00
B01                41250.00
C01               118890.00
D11               276620.00
D21               179680.00
E01                40175.00
E11               147140.00
E21               144520.00
DSNE610I NUMBER OF ROWS DISPLAYED IS 8
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

In our example, the security labels are set up so that they are related to the different work departments. Presume that a manager at level L3 wants to calculate the totals of the salaries for the departments B, C, and D. We expect that user ID USRT053 with security label L3BCD would return the required result. Example 7-13 shows the result of reading the EMP table with security label L3BCD.

Example 7-13 SUMMARY report results when read with L3BCD

```
SELECT WORKDEPT,
       SUM(SALARY)
   FROM DSN8810.EMP
   GROUP BY WORKDEPT;
---------+---------+---------+---------+---------+---------+
WORKDEPT
---------+---------+---------+---------+---------+---------+
B01                41250.00
C01               118890.00
D11               276620.00
D21               179680.00
E11               117390.00
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

Although we expected only output for departments B, C, and D, the result contains results for the department E11—because rows with SECLABEL L2N are included in the result (in other words, department E is included because of the uncategorized data at level 2). This proves that uncategorized data must be handled very carefully. When querying directly on the base table, the column that contains the security labels can be used in the query, and this is one of the possibilities of excluding uncategorized data, as shown in Example 7-14.

Example 7-14 Excluding uncategorized data by using the SECLABEL value

```
SELECT WORKDEPT,
       SUM(SALARY)
   FROM DSN8810.EMP
   WHERE CLASSIFICATION <> 'L2N'
   GROUP BY WORKDEPT;
---------+---------+---------+---------+---------+---------
WORKDEPT
---------+---------+---------+---------+---------+---------
B01                 41250.00
C01                118890.00
D11                276620.00
D21                179680.00
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

It is probably not the intention that users can see the security column or know the values of the security labels assigned to a row. In this case, a view can be used, as shown in Example 7-15. However, the safest possibility is not to use uncategorized data at all.

Example 7-15 Hiding uncategorized data by means of a VIEW

```
CREATE VIEW DEPTSAL
  AS
  SELECT WORKDEPT, SALARY
    FROM DSN8810.EMP
    WHERE CLASSIFICATION <> 'L2N';
---------+---------+---------+---------+---------+---------
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------
SELECT WORKDEPT,
       SUM(SALARY)
   FROM DEPTSAL
   GROUP BY WORKDEPT;
---------+---------+---------+---------+---------+---------
WORKDEPT
---------+---------+---------+---------+---------+---------
B01                 41250.00
C01                118890.00
D11                276620.00
D21                179680.00
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

7.5.3 Row-level security applied on INSERT

We logged on with user ID USRT070 with security label L2C. In the first instance, we inserted two rows. Note that for the first row, we explicitly specified a value of L1C for the security label, which is lower than the user's SECLABEL. For the second row, we explicitly defined L4C, which is higher than L2C. Because write-down checking is not in effect, USRT070 can act as though she possesses the write-down privilege, and so she can change the security label values of the inserted rows to a value of any existing, but not disjoint SECLABEL from her own SECLABEL value. We then select the rows to show the results. After that, we inserted a third row to show how the option DEFAULT can be used instead of explicitly specifying a SECLABEL value. This is useful when users or programs do not know their own SECLABEL value. Example 7-16 shows the result.

The two rows are inserted, but when we then do a query on the table, we see that only the L1C record is returned. This is normal, because, not having write-down control active, we can insert the record, and because of not having a dominant SECLABEL, we are not allowed to read it. For the third row, we see that the DEFAULT option resulted in the user's SECLABEL being inserted.

Example 7-16 Insert of rows with different SECLABEL values

```
INSERT INTO DSN8810.EMP
 VALUES ('000301','NORBERT','W','VERBESTEL','C01','3350','2004-10-19',
         'SOFTCE',14,'M','1952-05-23',11111.00,1000.00,111.00,'L1C');
---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+
INSERT INTO DSN8810.EMP
 VALUES ('000302','CHRIS','J','RAYNS','C01','1234','1999-10-19',
         'BOSS',14,'M','1962-12-06',85500.00,8000.00,888.00,'L4C');
---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+
SELECT EMPNO,LASTNAME,SALARY,CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------+---------+
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+---------+
000130  QUINTANA            23800.00  L2C
000140  NICHOLLS            28420.00  L2C
000301  VERBESTEL           11111.00  L1C
200140  NATZ                28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

```
---------+---------+---------+---------+---------+---------+---------+-
INSERT INTO DSN8810.EMP
 VALUES ('000303','MIKE','D','HOLMANS','C01','3456','2004-10-19',
         'HACKER',14,'M','1958-07-24',60000.00,5000.00,333.00,DEFAULT);
---------+---------+---------+---------+---------+---------+---------+-
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+-
SELECT EMPNO,LASTNAME,SALARY,CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------+---------+-
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+---------+-
000130  QUINTANA            23800.00  L2C
000140  NICHOLLS            28420.00  L2C
000303  HOLMANS             60000.00  L2C
000301  VERBESTEL           11111.00  L1C
200140  NATZ                28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

To verify that the row with SECLABEL L4C was really inserted, we logged on with a user ID having SYSHIGH and executed the same query. The result is in Example 7-17 and shows that EMPNO 000302 indeed received SECLABEL L4C. It also shows that employee 000030 with SECLABEL L3C is displayed.

Example 7-17 Query with SYSHIGH shows row with L4C

```
SELECT EMPNO,
       LASTNAME,
       SALARY,
       CLASSIFICATION
  FROM DSN8810.EMP
  WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------+--
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+--
000030  KWAN                38250.00  L3C
000130  QUINTANA            23800.00  L2C
000140  NICHOLLS            28420.00  L2C
000303  HOLMANS             60000.00  L2C
000302  RAYNS               85500.00  L4C
000301  VERBESTEL           11111.00  L1C
200140  NATZ                28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 7
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

After we finished the scenario with write-down control not being enabled, we enabled write-down control by executing RACF command SETROPTS MLS(WARNING). A display of the RACF options shows that our current RACF environment now has the following MLS characteristics:

```
NO WRITE-DOWN IS IN EFFECT. CURRENT OPTIONS:
"MLS WARNING" OPTION IS IN EFFECT
```

We did not define any user on the access list of the IRR.WRITDEOWN.BUYUSER profile. We then ran the same scenario again. We prefer to merge the results of the two scenarios, because we think this is the easiest way to compare the results. From now on, we always explicitly repeat in which environment we were running.

Example 7-18 shows the result of the same SQL as executed in Example 7-16 on page 171, but this time with MLS active. No user has the write-down privilege at this time. The result shows that all three inserts are allowed and, as we can see for employee numbers 000301 and 000302, that they always receive the value of the SECLABEL of the user. For employee number 000303, there is no difference. An additional query with SYSHIGH was not required to show all rows, because there was no write-up. The main conclusion, however, is that there is no declassification of data possible in this case.

Example 7-18 Insert with write-down control active

```
INSERT INTO DSN8810.EMP
 VALUES ('000301','NORBERT','W','VERBESTEL','C01','3350','2004-10-19',
         'SOFTCE',14,'M','1952-05-23',11111.00,1000.00,111.00,'L1C');
---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+
INSERT INTO DSN8810.EMP
 VALUES ('000302','CHRIS','J','RAYNS','C01','1234','1999-10-19',
         'BOSS',14,'M','1962-12-06',85500.00,8000.00,888.00,'L4C');
---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+
SELECT EMPNO,LASTNAME,SALARY,CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------+---------+
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+---------+
000130  QUINTANA            23800.00  L2C
000140  NICHOLLS            28420.00  L2C
000301  VERBESTEL           11111.00  L2C
```

```
000302  RAYNS               85500.00  L2C
200140  NATZ                28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+-
INSERT INTO DSN8810.EMP
 VALUES ('000303','MIKE','D','HOLMANS','C01','3456','2004-10-19',
         'HACKER',14,'M','1958-07-24',60000.00,5000.00,333.00,DEFAULT);
---------+---------+---------+---------+---------+---------+---------+-
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+-
SELECT EMPNO,LASTNAME,SALARY,CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------+---------+-
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+---------+-
000130  QUINTANA            23800.00  L2C
000140  NICHOLLS            28420.00  L2C
000301  VERBESTEL           11111.00  L2C
000302  RAYNS               85500.00  L2C
000303  HOLMANS             60000.00  L2C
200140  NATZ                28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 6
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

We now switch back to the NOMLS environment for another insert test. Running under user ID USRT070 with SECLABEL L2C, we try to insert two additional users, one assigning the corresponding row with a non-existing security label value L2BC and one with a disjoint or incompatible security label value L2A. To make sure that we are in the NOMLS environment and also to show the actual values of the rows, we first issue a query on the EMP table.

Important: The results of the following test with disjoint security labels assumes that the recent 2006 Consolidated Service Test (CST) level is used. For more information about CST availability, review the information at:

http://www.ibm.com/servers/eserver/zseries/zos/servicetst/mission.html

Example 7-19 shows that we receive an SQLCODE of -695 when trying to assign a non-existing SECLABEL to a row. Because of being equivalent to having write-down privilege, the specified value has been inserted in the security label column of the EMP table. However, before doing the insert, the SECLABEL value is checked with RACF and rejected because its value is not defined in the SECLABEL class.

Example 7-19 Insert of a row with non-existing SECLABEL

```
SELECT EMPNO,LASTNAME,SALARY,CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------+---------+-----
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+---------+-----
000130  QUINTANA            23800.00  L2C
000140  NICHOLLS            28420.00  L2C
000303  HOLMANS             60000.00  L2C
000301  VERBESTEL           11111.00  L1C
200140  NATZ                28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+-----
INSERT INTO DSN8810.EMP
 VALUES ('000304','ROLAND','H','GRANT','C01','7448','1978-07-03',
         'WRITER',14,'M','1955-10-01',45000.00,700.00,2500.00,'L2BC');
---------+---------+---------+---------+---------+---------+---------+-----
DSNT408I SQLCODE = -695, ERROR:  INVALID VALUE L2BC SPECIFIED FOR SECURITY
         LABEL COLUMN OF TABLE DSN8810.EMP
DSNT418I SQLSTATE   = 23523 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXRINS SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = -110 13174288  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'FFFFFF92'  X'00C90610'  X'00000000'  X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
---------+---------+---------+---------+---------+---------+---------+-----
DSNE618I ROLLBACK PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

The two SQL insert statements were in the same SPUFI input member. Because of the negative SQL code, the second insert was not executed. We commented out the first request and ran the second insert again where we tried to insert another employee number and assign its row an existing but disjoint or noncompatible security label value. The result is in Example 7-20 on page 176, and it shows that this time, we again cannot insert the row, but receive SQLCODE -20264. The supplied security label value for the row is checked to ensure that it either dominates the user's security label or the user's security label dominates the supplied security label for the row. If this condition is not met, the supplied security label for the row is disjoint with respect to the user's security label. For an INSERT or UPDATE statement, SQLCODE -20264 is issued.

Example 7-20 Insert of a row with noncompatible SECLABEL in NOMLS environment

```
SELECT EMPNO,LASTNAME,SALARY,CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------+---------+--
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+---------+--
000130  QUINTANA            23800.00  L2C
000140  NICHOLLS            28420.00  L2C
000303  HOLMANS             60000.00  L2C
000301  VERBESTEL           11111.00  L1C
200140  NATZ                28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+--
--INSERT INTO DSN8810.EMP
-- VALUES ('000304','ROLAND','H','GRANT','C01','7448','1978-07-03',
--         'WRITER',14,'M','1955-10-01',45000.00,700.00,2500.00,'L2BC');
INSERT INTO DSN8810.EMP
 VALUES ('000305','JUN','S','OGATA','C01','7660','1989-06-28',
         'TESTER',14,'M','1967-09-29',35000.00,600.00,1888.00,'L2A');
---------+---------+---------+---------+---------+---------+---------+--------
DSNT408I SQLCODE = -20264, ERROR:  FOR TABLE DSN8810.EMP, USRT070 WITH
         SECURITY LABEL L2C ID NOT AUTHORIZED TO          PERFORM MLS WRITE ON
         A ROW WITH SECURITY LABEL L2A, THE RECORD IDENTIFIER (RID) OF THIS
         ROW IS *N
DSNT418I SQLSTATE   = 42512 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXRINS SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = -110 13174296  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'FFFFFF92'  X'00C90618'  X'00000000'  X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
---------+---------+---------+---------+---------+---------+---------+--------
DSNE618I ROLLBACK PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

Let us now show the result of the same inserts, but in an MLS-enabled environment. Example 7-21 shows that the big difference in this case is that both rows are inserted. Not having the write-down privilege, the rows are inserted and the user SECLABEL is assigned to the rows without checking the specified security label value. The subsequent query shows the two inserted rows with SECLABEL L2C.

Example 7-21 Insert of non-existing and noncompatible SECLABELs

```
SELECT EMPNO,LASTNAME,SALARY,CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------+---------+
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+---------+
000130  QUINTANA            23800.00  L2C
000140  NICHOLLS            28420.00  L2C
000301  VERBESTEL           11111.00  L2C
000302  RAYNS               85500.00  L2C
000303  HOLMANS             60000.00  L2C
200140  NATZ                28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 6
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+
INSERT INTO DSN8810.EMP
 VALUES ('000304','ROLAND','H','GRANT','C01','7448','1978-07-03',
         'WRITER',14,'M','1955-10-01',45000.00,700.00,2500.00,'L2BC');
---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+
INSERT INTO DSN8810.EMP
 VALUES ('000305','JUN','S','OGATA','C01','7660','1989-06-28',
         'TESTER',14,'M','1967-09-29',35000.00,600.00,1888.00,'L2A');
---------+---------+---------+---------+---------+---------+---------
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------
SELECT EMPNO,LASTNAME,SALARY,CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------+---------
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+---------
000130  QUINTANA            23800.00  L2C
000140  NICHOLLS            28420.00  L2C
000301  VERBESTEL           11111.00  L2C
000302  RAYNS               85500.00  L2C
000303  HOLMANS             60000.00  L2C
```

```
000304  GRANT             45000.00  L2C
000305  OGATA             35000.00  L2C
200140  NATZ              28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 8
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

7.5.4 Row-level security applied on UPDATE

For an update operation, when SECLABELs are equivalent, a row is always updated, and the SECLABEL is unchanged.

If the SECLABELs are not equivalent for an update, the rules are as follows:

- If the user’s SECLABEL is greater than the security label of the row, the result of the UPDATE operation is determined by the write-down option. If the user does not have write-down, the row is not updated. If the user has write-down privilege or write-down control is not enabled, the row is updated and the SECLABEL can be set to a valid value that is not disjoint in relation to the user’s security label, or is set to the user’s SECLABEL if no value is specified.
- If the security label of the row is greater than the security label of the user, the row is not updated.

In other words, we can say that a user, to be able to update a row in a table, must be able to read that row from that table. If security labels are equivalent, the row is updated. To update rows with a lower security label, the user requires write-down authority. In both cases the write-down capability determines the setting of the value of the SECLABEL.

Let us start in a NOMLS environment. We logged on with user ID USRT053 having SECLABEL L3BCD. We decided in the first instance to give a salary increase of 3% to all employees in the C01 department. The starting SELECT in Example 7-22 on page 179 shows us the rows that are read with a security label L3BCD. There are no rows with equivalent SECLABELs. Write-down control is not in effect. This user will be able to update the rows and set the SECLABEL to any valid value. When a SECLABEL value is not specified for the update, the security label column will be set to the SECLABEL value of the user. To avoid this user’s SECLABEL propagation, we added the statement CLASSIFICATION = CLASSIFICATION to our update SQL statement. This way, we force the SECLABEL to be the same as the actual value of the security label. The result of the UPDATE SAL statement is that all rows are updated and the SECLABEL values are unchanged. This can be seen in the SELECT output that comes after the UPDATE statement.

Example 7-22 Update where user dominates row and no change of security label value

```
SELECT EMPNO, LASTNAME, SALARY, CLASSIFICATION
     FROM DSN8810.EMP WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------
000030  KWAN                38250.00  L3C
000130  QUINTANA            23800.00  L2C
000140  NICHOLLS            28420.00  L2C
000303  HOLMANS             60000.00  L2C
000301  VERBESTEL           11111.00  L1C
200140  NATZ                28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 6
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------
UPDATE DSN8810.EMP
     SET SALARY = SALARY * 1.03,
     CLASSIFICATION = CLASSIFICATION
     WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------
DSNE615I NUMBER OF ROWS AFFECTED IS 6
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------
SELECT EMPNO, LASTNAME, SALARY, CLASSIFICATION
     FROM DSN8810.EMP WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------
000030  KWAN                39397.50  L3C
000130  QUINTANA            24514.00  L2C
000140  NICHOLLS            29272.60  L2C
000303  HOLMANS             61800.00  L2C
000301  VERBESTEL           11444.33  L1C
200140  NATZ                29272.60  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 6
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

To show the user SECLABEL value propagation, we decided to run a similar SQL statement, but with the CLASSIFICATION = CLASSIFICATION statement removed, and this only for employee number 000030. The result is shown in Example 7-23. Only the row with employee number 000030 updated, and the security label value became L3BCD.

Example 7-23 Update where user dominates row and SECLABEL propagation

```
SELECT EMPNO, LASTNAME, SALARY, CLASSIFICATION
     FROM DSN8810.EMP WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------
000030  KWAN                39397.50  L3C
000130  QUINTANA            24514.00  L2C
000140  NICHOLLS            29272.60  L2C
000303  HOLMANS             61800.00  L2C
000301  VERBESTEL           11444.33  L1C
200140  NATZ                29272.60  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 6
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------
UPDATE DSN8810.EMP
     SET SALARY = SALARY * 1.03
     WHERE EMPNO = '000030';
---------+---------+---------+---------+---------+---------
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------
SELECT EMPNO, LASTNAME, SALARY, CLASSIFICATION
     FROM DSN8810.EMP WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------
000030  KWAN                40579.42  L3BCD
000130  QUINTANA            24514.00  L2C
000140  NICHOLLS            29272.60  L2C
000303  HOLMANS             61800.00  L2C
000301  VERBESTEL           11444.33  L1C
200140  NATZ                29272.60  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 6
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

Example 7-24 shows the result of running both UPDATE statements in an MLS-enabled environment. Not having write-down capabilities, the result is that no row is updated, because this requires equivalent security labels. Note that there is no message DSNE615I after the UPDATE SQL statement, because there are no rows updated.

Example 7-24 Same UPDATE SQL in an MLS environment

```
SELECT EMPNO, LASTNAME, SALARY, CLASSIFICATION
     FROM DSN8810.EMP WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------
EMPNO   LASTNAME             SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------
000030  KWAN               38250.00  L3C
000130  QUINTANA           23800.00  L2C
000140  NICHOLLS           28420.00  L2C
000301  VERBESTEL          11111.00  L2C
000302  RAYNS              85500.00  L2C
000303  HOLMANS            60000.00  L2C
000304  GRANT              45000.00  L2C
000305  OGATA              35000.00  L2C
200140  NATZ               28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 9
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------
UPDATE DSN8810.EMP
     SET SALARY = SALARY * 1.03,
     CLASSIFICATION = CLASSIFICATION
     WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------
UPDATE DSN8810.EMP
     SET SALARY = SALARY * 1.03
     WHERE EMPNO = '000030';
---------+---------+---------+---------+---------+---------
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------
SELECT EMPNO, LASTNAME, SALARY, CLASSIFICATION
     FROM DSN8810.EMP WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------
EMPNO   LASTNAME             SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------
000030  KWAN               38250.00  L3C
000130  QUINTANA           23800.00  L2C
000140  NICHOLLS           28420.00  L2C
000301  VERBESTEL          11111.00  L2C
000302  RAYNS              85500.00  L2C
000303  HOLMANS            60000.00  L2C
```

```
000304  GRANT               45000.00  L2C
000305  OGATA               35000.00  L2C
200140  NATZ                28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 9
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

7.5.5 Row-level security applied on DELETE

For a DELETE operation, a user is only able to delete a row from a table on the condition that he can read this row. For equivalent security labels, a row is always deleted. If the user's security label dominates a row's security label, he needs the write-down option to be able to delete the row.

Again, we start in the NOMLS environment. We are still logged on with user ID USRT053, who has SECLABEL L3BCD. The intention is to delete the five user IDs that we tried to add during this sample scenario. The user IDs that we inserted were all inserted in work department C01, but with different security labels. From the INSERT test in 7.5.3, “Row-level security applied on INSERT” on page 171, we know that in the NOMLS environment, only three rows were inserted, all with different security labels (L1C, L2C, and L4C). Example 7-25 shows that the user ID USRT053 first listed the employees that he is able to see with his security label L3BCD. From the inserted records, he only could read the rows with security levels L1C and L2C. Then, he issued a DELETE SQL request for the five inserted rows. In this NOMLS environment, with write-down control not being enabled, user ID USRT053 can delete the two employees 000301 and 000303. This can be seen in the message DSNE615I that tells us that only two rows were affected. The query that follows after the DELETE statement now shows four rows instead of six. In this environment, any user ID having a security label that has access to L2A and L4C can delete the other rows.

Example 7-25 Delete of added user IDs in NOMLS environment with L3BCD

```
SELECT EMPNO, LASTNAME, SALARY, CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------+---
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+---
000030  KWAN                40579.42  L3BCD
000130  QUINTANA            24514.00  L2C
000140  NICHOLLS            29272.60  L2C
000303  HOLMANS             61800.00  L2C
000301  VERBESTEL           11444.33  L1C
200140  NATZ                29272.60  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 6
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

```
---------+---------+---------+---------+---------+---------+---
DELETE FROM DSN8810.EMP
 WHERE EMPNO IN ('000301','000302','000303','000304','000305');
---------+---------+---------+---------+---------+---------+---
DSNE615I NUMBER OF ROWS AFFECTED IS 2
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---
SELECT EMPNO, LASTNAME, SALARY, CLASSIFICATION
FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------
000030  KWAN                40579.42  L3BCD
000130  QUINTANA            24514.00  L2C
000140  NICHOLLS            29272.60  L2C
200140  NATZ                29272.60  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

We now show the result of the same delete statements in an MLS-enabled environment. Remember that in the MLS environment, we were able to insert all five rows and they all have security label L2C. Example 7-26 shows that we now can read all five inserted records. However, not having write-down authority, user ID USRT053 is not able to delete the rows.

Example 7-26 Same DELETE statements in an MLS-controlled environment

```
SELECT EMPNO, LASTNAME, SALARY, CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------+---
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+---
000030  KWAN                38250.00  L3C
000130  QUINTANA            23800.00  L2C
000140  NICHOLLS            28420.00  L2C
000301  VERBESTEL           11111.00  L2C
000302  RAYNS               85500.00  L2C
000303  HOLMANS             60000.00  L2C
000304  GRANT               45000.00  L2C
000305  OGATA               35000.00  L2C
200140  NATZ                28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 9
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---
DELETE FROM DSN8810.EMP
 WHERE EMPNO IN ('000301','000302','000303','000304','000305');
```

```
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------
SELECT EMPNO, LASTNAME, SALARY, CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------
000030  KWAN                38250.00  L3C
000130  QUINTANA            23800.00  L2C
000140  NICHOLLS            28420.00  L2C
000301  VERBESTEL           11111.00  L2C
000302  RAYNS               85500.00  L2C
000303  HOLMANS             60000.00  L2C
000304  GRANT               45000.00  L2C
000305  OGATA               35000.00  L2C
200140  NATZ                28420.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 9
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

7.5.6 RACF-controlled write-down

In the previous section, we saw that when running in a NOMLS environment, all users are equivalent to having write-down authority. This allows them to declassify data. After migration to an MLS-controlled environment, by default no user has write-down authority. In this environment, you can select some individual users and assign them the write-down privilege. Example 7-27 shows the commands that we executed to create the IRR.WRITEDOWN.BYUSER profile in the RACF FACILITY general resource class. We defined the profile with UACC(NONE) so that we now can put individual users on its access list. The command shows that we added the user ID USRT053 and gave it UPDATE access to the profile. The last command is to refresh the FACILITY class.

Example 7-27 RACF commands to assign write-down privilege to USRT053

```
//RUNDB2   EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD   SYSOUT=*
//SYSPRINT DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=*
//SYSTSIN  DD   *
 RDEFINE FACILITY IRR.WRITEDOWN.BYUSER UACC(NONE)
 PERMIT IRR.WRITEDOWN.BYUSER ID(USRT053) ACCESS(UPDATE) CLASS(FACILITY)

 SETR RACLIST(FACILITY) REFRESH
```

USRT053 is now the only user ID to have the write-down privilege. To prove this, we re-submitted the SQL statements from Example 7-26 on page 183. USRT053 has SECLABEL L3BCD. Therefore, this time, USRT053 is able to delete the five records that were inserted earlier.

Example 7-28 DELETE rows is now possible USRT053

```
SELECT EMPNO, LASTNAME, SALARY, CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------+---
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------+---
000030  KWAN                40579.42  L3BCD
000130  QUINTANA            24514.00  L2C
000140  NICHOLLS            29272.60  L2C
000301  VERBESTEL           11444.33  L2C
000302  RAYNS               88065.00  L2C
000303  HOLMANS             61800.00  L2C
000304  GRANT               46350.00  L2C
000305  OGATA               36050.00  L2C
200140  NATZ                29272.60  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 9
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---
DELETE FROM DSN8810.EMP
 WHERE EMPNO IN ('000301','000302','000303','000304','000305');
---------+---------+---------+---------+---------+---------+---
DSNE615I NUMBER OF ROWS AFFECTED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------
SELECT EMPNO, LASTNAME, SALARY, CLASSIFICATION
 FROM DSN8810.EMP
 WHERE WORKDEPT = 'C01';
---------+---------+---------+---------+---------+---------
EMPNO   LASTNAME              SALARY  CLASSIFICATION
---------+---------+---------+---------+---------+---------
000030  KWAN                40579.42  L3BCD
000130  QUINTANA            24514.00  L2C
000140  NICHOLLS            29272.60  L2C
200140  NATZ                29272.60  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

7.6 Real-world implementation

In this section, we discuss real-world implementation.

7.6.1 Introduction

In the real world, the process of implementing row-level security is likely to be more complicated. In our illustration, we did not add the CLASSIFICATION column to the indexes on the EMP table, nor did we reorganize the data. Because we used SPUFI rather than application programs, we did not go through the process of rebinding plans and packages that referenced the table either.

You will almost certainly want to add the SECLABEL column to all the indexes on tables that have security labels. If you do not, DB2 will have to go to the table whenever you access it. It can no longer use index-only access because it does not know whether the row referenced by an index entry is available to the user until it attempts to retrieve the row.

7.6.2 Preparation suggestions

Note the following preparation suggestions:

- For ease of administration, you need to architect a sound, flexible, and extensible naming convention for SECLABELs before you begin. Do not just "dive in" and implement anything.
- Do not have too many SECLABELs in your tables. Too many security labels cannot only become difficult to manage, but there are also some performance implications. SECLABEL values are cached after they have been checked, so subsequent checking of already checked values is more efficient. For more details about Version 8 performance and row-level security, see *DB2 UDB for z/OS Version 8 Performance Topics*, SG24-6465. As of this writing, this book has not been updated for DB2 Version 9.
- You need to include row-level security in your application at the design level. Preferably, do not implement it into existing applications as an add-on.
- You need to plan and design how you will classify your data. This will impact the results of your INSERT and UPDATE SQL statements. The whole issue of managing how write-down will be used has to be planned, as well as designing how the data cannot be reclassified by just anybody.

- You might have already implemented some sort of home-grown row-level security mechanism in your applications, which is probably not compatible with the DB2 implementation of row-level security. Therefore, you might need to dramatically change your application anyway, at least to address the previous issues with write-down and protect against inappropriately reclassifying of data. The DB2 implementation of row-level security is transparent for accessing data but is not as transparent when you need to update the data. You might need to change your SQL and processes to protect the classification of data.
- There might be some “design” considerations in using column functions such as SUM and AVERAGE. These functions only take into account the data you are allowed to see, which might be different from how your applications work today.
- Finally, there might be issues involved in moving some work or responsibility for security administration from the DBA team to the security team. There might be cultural issues you need to address.

7.6.3 A suggested procedure

We suggest that you take these steps to add row-level security:

1. Consider the sizing implications of adding 8 bytes to every row of each affected table and every index key entry. In most cases, you want to increase the values specified for PRIQTY and SECQTY for the underlying table spaces and the index spaces.
2. Decide how you will set the values of the security label columns. When you execute the ALTER TABLE statement, this column is populated with the security label associated with the user ID under which the ALTER TABLE executes. This is unlikely to be the value you want in every single row.
3. Shut down all application processes that can access the affected tables and indexes.
4. Stop the table spaces and index spaces.
5. For each table:
   a. ALTER TABLE to DROP the primary key, any foreign key referential constraints, and any unique constraints.

   > **Note:** This puts any table spaces dependent on the referential constraints into check pending (CHKP) state.

b. If the table is in a partitioned table space *and* the partitioning is index-controlled, as it well might be if you have recently migrated from V7, DROP the partitioning index. This automatically converts the table to table-controlled partitioning, so you do not have to go through the old procedure of unloading the data, dropping the table space, and then re-creating.

c. ALTER TABLE ADD column-name CHAR(8) FOR SBCS DATA NOT NULL WITH DEFAULT AS SECURITY LABEL.

d. ALTER TABLESPACE to set the new PRIQTY and SECQTY values.

e. For each remaining index, ALTER INDEX to ADD COLUMN security label and update PRIQTY and SECQTY.

> **Note:** Make sure that you execute steps c, d, and e in the same unit of work. If you do so, the indexes are only put into advisory reorg pending (AREO*) state. If you COMMIT after step c or d, the indexes will be put into rebuild pending (RBDP) state and you will have to rebuild them all.

f. If you dropped the partitioning index, re-create it with the addition of the new column and updated PRIQTY and SECQTY values. Use the DEFER option. In some rare cases, where the V7 or prior partitioning index was only ever used to assign rows to partitions and was never used by queries or WHERE clauses of UPDATE and DELETE statements, you will be able to omit this step.

g. ALTER TABLE to re-create the PRIMARY KEY and any referential and other unique constraints that you previously dropped. You will probably want to add the security label column to these constraints, although it is not strictly necessary. If you are adding the security label column as the last column of indexes used to support these constraints, DB2 is clever enough to realize that the earlier columns are sufficient to support the constraints, which you will not have violated by adding the column.

6. START the table spaces and associated index spaces.
7. Rebuild any indexes you had to re-create with the DEFER option.
8. Run CHECK TABLESPACE SCOPE PENDING on any table spaces that were set to CHKP by step 5.a.
9. Run the SECLABEL population process.
10. REORG the table spaces and indexes, taking an inline copy and performing statistics collection.
11. Rebind all affected plans and packages.
12. Restart the application processes.

7.6.4 Designing the population process

You are going to need a process to set the security labels to useful values when you convert to using row-level security. When you add the SECLABEL column, DB2 sets the initial value to the SECLABEL associated with the process, which executes the ALTER TABLE. Because it is likely that this will be done by a DBADM or SYSADM with a SECLABEL of SYSHIGH, or possibly SYSLOW, it is unlikely to leave the tables in a useful state.

You need to work out the correct logical order in which to convert the tables. For instance, to protect the tables containing data relevant to a set of customers, you will probably first secure the rows in the CUSTOMER table, and then propagate the values to the CUSTOMER_ORDER table, and from there to the ORDER_ITEM table.

To enable write-down control using MLS(FAILURES), ensure that the user ID associated with the process has UPDATE access to IRR.WRITEDOWN.BYUSER and operates in write-down mode.

When designing the initial population process, remember that you might have an ongoing need to undertake this sort of SECLABEL propagation, and think about how you might be able to reuse the logic, because it will save you much time and effort later.

8

Network trusted contexts and roles

DB2 V9 provides new options for tighter security and allows for more granularity and additional flexibility. You now are able to create two new database entities:

- Trusted context
- Role

A trusted context establishes a trusted relationship between DB2 and an external entity, such as a middleware server or another DB2 subsystem. At connect time, a series of trust attributes are evaluated to determine if a specific context can be trusted. After a trusted connection is established, one of the new abilities is to acquire, through a role, a special set of privileges for a DB2 authorization ID within the specific connection that are not available to it outside the trusted connection.

When defined, connections from specific users through defined attachments and source servers allow trusted connections to DB2. The users in this context can also be defined to obtain a database role.

A role is a database entity that groups together one or more privileges and can be assigned to users. A role can provide privileges that are in addition to the current set of privileges granted to the user’s primary and secondary authorization IDs.

Users must be allowed to use a trusted context.

A trusted context can exist without a role. A role is usable only within an established trusted connection based on a predefined trusted context.

A default role can be assigned to a trusted context. Similarly, a default SECLABEL can be required to use a trusted context. It is also possible to explicitly specify a role and SECLABEL for an individual user of the trusted context.

Within a trusted connection, DB2 allows one and only one role to be associated with a thread at any point in time.

Now, we describe some existing challenges.

8.1 Existing challenges

We now provide a summary of existing challenges that led to providing the new DB2 V9 trusted context and role capabilities.

Trust all connection requests?

DB2 receives requests from many external entities. Currently, you have the option to set a system parameter that indicates to DB2 that all connections are to be trusted. Trusted means that users do not have to be authenticated to RACF with credentials. It is unlikely that all connection types, such as DRDA, RRS, TSO and batch, from all sources will fit into this category.

Application server user ID/password

Most existing application servers connect to DB2 using a single user ID to initiate requests to DB2. This ID must have all administrative privileges and the privileges required to execute the business transactions. If someone steals that user ID and password, that person can exercise those privileges from another place in the network. This is a significant exposure.

It also means diminished accountability. For example, there is no ability to separate access performed by the middleware server for its own purposes from those performed on behalf of users.

Characteristics of the common, three-tier architecture include:

- The middle layer (sometimes called the middleware layer) authenticates users running client applications.
- The middle layer also manages interactions with the database server (DB2).
- The middle layer's user ID and password are used for authentication purposes.
- The database privileges associated with that authorization ID are checked when accessing the database, including all access on behalf of all users.
- The middle layer's user ID must be granted privileges on all the resources that might be accessed by user requests.

Figure 8-1 illustrates the three-tier architecture.

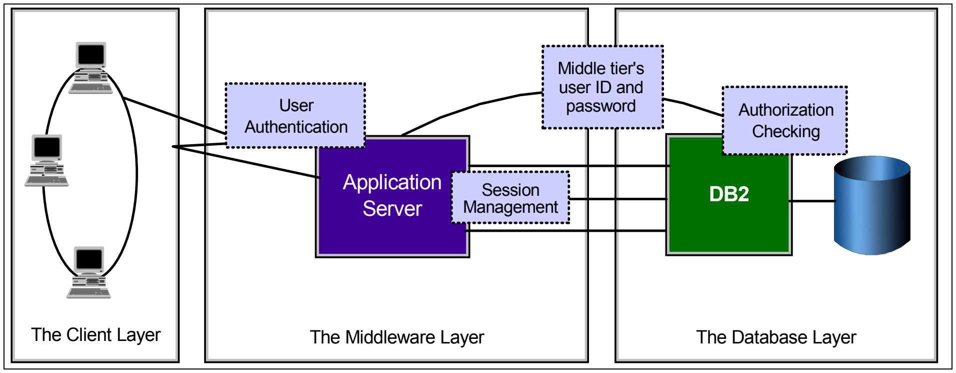


Figure 8-1 Three-tier architecture

Dynamic SQL auditability

Looking at Figure 8-1, now we discuss the reasons that led to this configuration and the resulting weak auditing capability.

The middleware layer's authid is used to access DB2 so that customers do not have to grant dynamic SQL privileges directly to their users. DB2 only authenticates and knows about a single authid, which make all the requests, when, in fact, there are multiple users initiating requests through the application. A consequence of this approach is diminished accountability; there is no ability in DB2 to separate access performed by the middleware layer's authid server for its own purposes from those performed on behalf of users.

Business user authentication is performed by the middleware component and not in DB2, which means a loss of control over user access to the database.

Another reason for using a three-tiered approach is connection pooling in the application server. There is a performance processing time associated with simply creating a new connection to the database server and re-authenticating the user at the database server. This can be a problem for middleware servers that do not have access to user credentials, such as IBM Lotus® Domino®.

Already verified DRDA

Many customers want to migrate from SNA and PRIVATE protocol to a solution based on TCP/IP and DRDA protocol. Prior to DB2 V9, the lack of already verified support in TCP/IP prevented many customers from migrating to DRDA.

The main reason people did not move to TCP/IP is because of the difficulty of managing RACF user IDs and passwords on the middle layer. They needed a way to authenticate the connection and flow user IDs without the processing time of managing the passwords or credentials.

Currently, a system-wide zparm determines whether all TCP/IP can come in as already verified or none can come in as already verified.

Securing DBA privileges

We describe some challenges and new opportunities for securing DBA privileges.

Shared SYSADM ID

In some shops, certain DBAs have powerful user IDs that hold administrative privileges such as SYSADM or DBADM. Although they might only use certain privileges infrequently, when acquired, they typically hold these privileges all the time.This increases the risk of adversely affecting DB2 objects or table data when an error or mistake occurs. Such a situation is potentially more open to malicious activity or fraud.

Suppose a team of DBAs share a generic user ID that holds SYSADM or DBADM privileges. When a DBA leaves the team for a new internal or external job, there is no cascading effect. However, by using a shared ID there is no individual accountability, which is a Sarbanes-Oxley requirement.

Additionally, many customers are concerned about DBA access to sensitive data.

Dual responsibilities

A DBA might have a dual job function, where he or she maintains databases in both development and production using the same user ID in both environments. In this scenario, a mistake can occur when making changes if a DBA thinks she is connected to development, when she is really connected to production.

Full-time access to sensitive/private data by DBA

When DBAs create tables, they have full access to the contents of the tables all of the time.

DBADM can create view for another ID, but cannot drop/alter

An authid, with DBADM privilege, under certain circumstances can create views for others. However, this same authid is not able to drop or grant privileges on these views to others. Because views do not have an underlying database, the DBADM privilege cannot be recognized for views.

Without reserving a RACF group, a table can be dropped

When the creator of a table is not a user ID, and a RACF group has not been set up with same name as the creator, it is possible for someone else to create the group, connect themselves to it, and drop the table. Another user ID can do this with basic access to DB2 and without any grants received on that table.

Privileges granted can be exercised from anywhere

Before trusted contexts and roles, privileges held by an authid were universal, irrespective of context. This can weaken security because a privilege can be used for purposes other than those originally intended. For example, if an authid is granted SELECT on a payroll table, the authid can exercise that privilege regardless of how it gains access to the table.

8.2 Network trusted context

It is likely that only a subset of connection requests for any type and source are trusted or that you want to restrict trusted connections to a specific server.

DB2 V9 introduces a new database entity called a trusted context. It provides a technique to work with other environments more easily than before, improving flexibility and security.

Trusted context addresses the problem of establishing a trusted relationship between DB2 and an external entity, such as a middleware server, for example:

- IBM WebSphere® Application Server
- IBM Lotus Domino
- SAP NetWeaver®
- Oracle® PeopleSoft®
- Siebel® Optimizer

More granular flexibility will allow for the definition of trusted connection objects. When defined, connections from specific users through defined attachments (DDF, RRS Attach, DSN) and source servers allow trusted connections to DB2.

The relationship between a connection and a trusted context is established when a connection to the server is first created and remains for the life of that connection.

Characteristics of a trusted context

A trusted context is an independent database entity that is based on a system authorization ID and connection trust attributes.

The system authorization ID is a DB2 primary authorization ID that is used to establish the trusted connection. A SYSTEM AUTHID for a connection can only be associated with a single trusted context. This ensures that there is always a clear mapping between a specific connection and a specific trusted context.

The connection trust attributes are ADDRESS, SERVAUTH, ENCRYPTION, and JOBNAME. They identify specific connections to consider as part of the trusted context.

Any defined ADDRESS, SERVAUTH, or JOBNAME must also be unique in a trusted context.

A trusted connection can be established for a local or a remote application. The attributes used to establish a trusted context are different for remote versus local applications. We describe these next:

- For a remote trusted connection, the SYSTEM AUTHID is derived from the system user ID provided by an external entity, for example, a middleware server, when initiating the connection.
- For a remote trusted connection, the attributes considered are:
  - ADDRESS: IP address or domain name. The protocol is restricted to TCP/IP only.
  - SERVAUTH: A resource in the RACF SERVAUTH class.

    TCP/IP network resources (IP addresses) can be mapped to RACF SERVAUTH security zones. Using the RACF SERVAUTH class, it is possible to protect access to those network resources.
  - ENCRYPTION: Minimum level of encryption for the connection:
    - NONE: No encryption. This is the default.
    - LOW: DRDA data stream encryption.
    - HIGH: Secure Sockets Layer (SSL) encryption.
- For a local trusted connection, the SYSTEM AUTHID is derived as follows:
  - Started task (RRSAF): JOB statement USER or RACF USER
  - TSO: TSO logon ID
  - BATCH: JOB statement USER
- For a local trusted connection, the attribute considered is JOBNAME, which is derived as follows:
  - Started task (RRSAF): JOB or started class name
  - TSO: TSO logon ID

- BATCH: JOB name

Trusted contexts cannot be defined for CICS and IMS. DB2 online utilities can run within a trusted connection.

Trusted contexts can be altered, for example, they can be enabled or disabled, and users can be added, replaced, or dropped.

Within an established trusted connection, it is possible to switch to a different user without authenticating the new user.

Later in this chapter, we provide some examples.

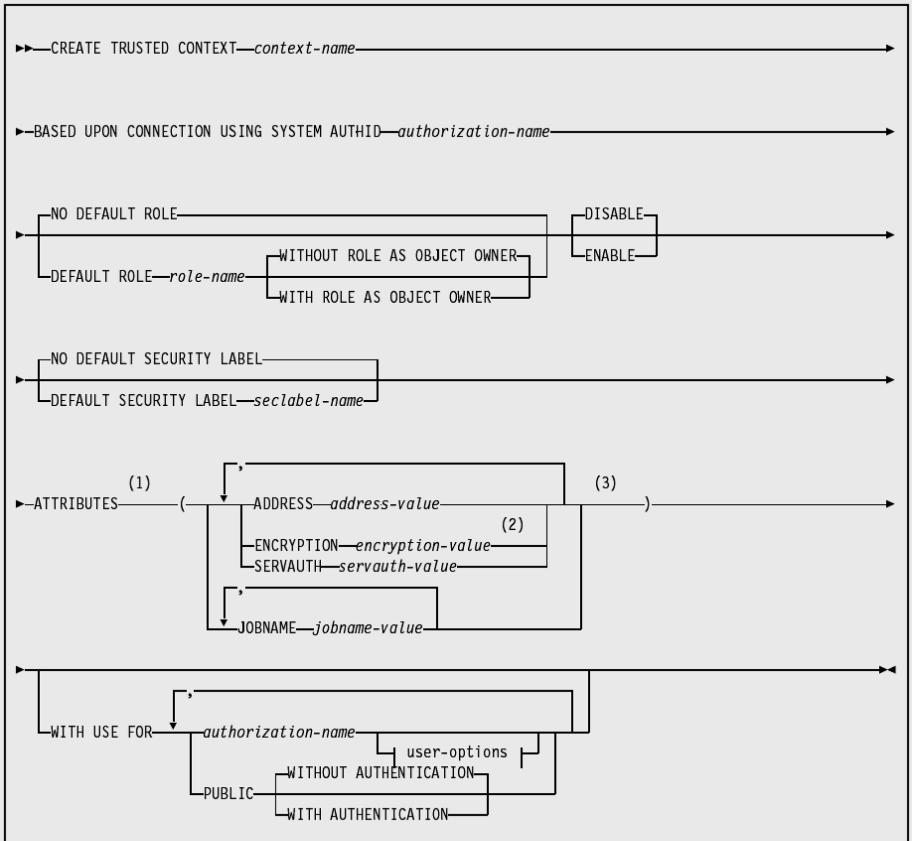
8.2.1 Trusted context DDL, catalog tables

New DDL statements are added to allow an administrator to define, alter, or drop a trusted context. The privilege set required to create a trusted context must include SYSADM authority.

DDL

Here we describe the DDL for creating a trusted context.

The CREATE TRUSTED CONTEXT statement defines a trusted context at the current server (Figure 8-2).



Notes:

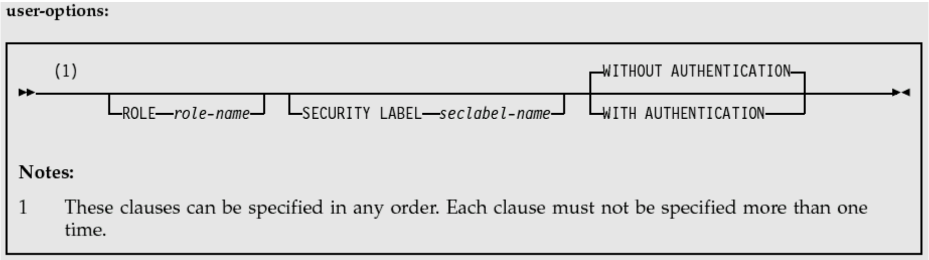
1 This clause and the clauses that follow can be specified in any order. Each clause must not be specified more than one time.

2 ENCRYPTION must not be specified more than one time.

3 Each pair of attribute name and corresponding value must be unique.

Figure 8-2 Create trusted context syntax

Figure 8-3 shows the details of user-options.



Notes:

1 These clauses can be specified in any order. Each clause must not be specified more than one time.

Figure 8-3 Create trusted context syntax continued

For complete details about the syntax, see Appendix A, “Trusted context syntax” on page 303.

New and modified SQL statements

We describe the new and modified SQL statements that apply to the trusted context:

- ALTER TRUSTED CONTEXT
  - Can alter, add, or drop attributes, for example, SYSTEM AUTHID.
  - Can add, replace, or drop USE FOR options for an authorization name or for PUBLIC.
- COMMENT ON TRUSTED CONTEXT
- CREATE TRUSTED CONTEXT
- DROP TRUSTED CONTEXT
- GET DIAGNOSTICS
  A new value for DB2_AUTHENTICATION_TYPE. T indicates trusted context authentication.

New catalog tables

DB2 V9 introduces new catalog tables to store a trusted context:

- SYSIBM.SYSCONTEXT contains the context name, context ID, and other context related information.
- SYSIBM.SYSCTXTTRUSTATTRS holds the attributes for a given context.
- SYSIBM.SYSCONTEXTAUTHIDS stores the authorization IDs that can be “switched to” in a trusted connection.

Dropping a trusted context

When a trusted context is dropped, all the attributes and users associated with the context are dropped. If the trusted context is dropped while the trusted connections for this context are active, the connections remain trusted until they terminate or until the next reuse attempt. An error is returned if an attempt is made to switch the user on these trusted connections (SQLSTATE = 42704, SQLCODE = -204) and the connection is returned to an unconnected state.

The dropped trusted context takes effect after the DROP TRUSTED CONTEXT statement is committed. If the DROP TRUSTED CONTEXT request results in an error or is rolled back, there is no change.

8.2.2 How a trusted connection comes alive and ends

A trusted connection is a database connection that is established when the incoming connection attributes match the attributes of a unique, enabled trusted context defined at the server. The trust attributes identify a set of characteristics about the specific connection that are required for the connection to be considered a trusted connection. The relationship between a connection and a trusted context is established when the connection to the server is first created and that relationship remains for the life of that connection.

If a trusted context definition is altered, the changed attributes of the trusted context take effect when the next new connection request comes in, or a switch user request is issued within an existing trusted connection.

With DB2 as a *server*, on receipt of the connection request:

1. DB2 performs standard authorization checking.
2. DB2 invokes the connection exit to optionally change the authorization ID or to associate the primary authid, any secondary authids, and an SQL ID with the request.
3. DB2 searches for a trusted context by matching the primary authorization ID with a trusted context SYSTEM AUTHID.
4. This step differs for local and remote requests:
   - For a local request, DB2 checks that the JOB name matches the JOBNAME attribute for the identified trusted context.
   - For a remote request:
     - If a SERVAUTH attribute is defined for the trusted context and a RACF SERVAUTH profile name for the TCP/IP resource, DB2 matches the two.

- If no SERVAUTH attribute is defined, or the SERVAUTH name provided does not match that of the trusted context, DB2 matches the remote client address with the trusted context ADDRESS attribute.
- DB2 verifies the encryption used matches that in the ENCRYPTION attribute.

5. For both local and remote requests, if the SYSTEM AUTHID is associated with a SECURITY LABEL from the trusted context definition, DB2 validates it with RACF (used for multilevel security).
6. If validation is successful the connection is established as trusted. Otherwise, the connection is established as a normal “untrusted” connection (SQLCODE +20360).

A DB2 for z/OS system can establish a trusted connection as a *requester* to a remote DB2 subsystem. The information required is obtained from:

1. The new SYSIBM.LOCATIONS column TRUSTED (Y indicates that access to the remote location requires a trusted context defined at the remote location).
2. Existing SYSIBM.IPNAMES column SECURITY_OUT (must be E, P, or R) and modified SYSIBM.IPNAMES column USERNAMES (new value S indicates that a corresponding row in the SYSIBM.USERNAMES table is used to obtain the SYSTEM AUTHID for the trusted context).

   Modified value O now allows the corresponding row in SYSIBM.USERNAMES to be used to obtain the SYSTEM AUTHID for the trusted context, but the ID is subject to outbound ID translation.

8.2.3 Authid switching within a trusted connection

DB2 allows an established trusted connection to be used under a different user. To allow this, the trusted context must be defined with use for the specific user. If PUBLIC is specified for the user, it allows the trusted connection to be used by any authorization ID.

When a trusted connection is established, DB2 enables the trusted connection to be reused under a different user on a transaction boundary.

When a switch user event takes place, the new user does not inherit any privileges or resources from the previous user.

The trusted connection can be used by a different user with or without authentication. This is specified by the WITH AUTHENTICATION clause.

New catalog table SYSIBM.SYSCONTEXTAUTHIDS is introduced to store the authorization IDs that can be “switched to” in a trusted connection.

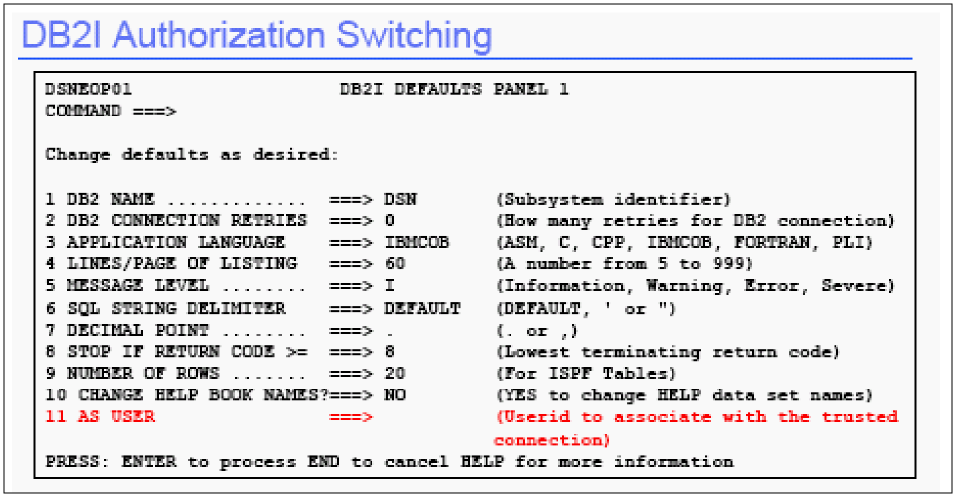
Switch in a local connection

A trusted connection can be reused at a local DB2 subsystem by using RRSAF, the DSN command processor under TSO and DB2I, and the SQL CONNECT statement with the USER and USING clauses:

- RRSAF: The SIGNON function in CALL DSNRLI now allows authid switching (it is the primary authid that is checked):
  a. DB2 calls the sign-on exit.
  b. DB2 performs SECURITY LABEL verification for the new user ID, if applicable.
- The DSN command processor: The new ASUSER option allows switching to the user specified by ASUSER. Again, the primary authid is checked for access to the trusted context.
  a. DB2 calls the connection exit.
  b. DB2 performs SECURITY LABEL verification for the new user ID, if applicable.
- SQL CONNECT supports authid switching through the USER and USING clauses locally only, and the primary authid is checked to see if it has access to the trusted context:
  a. DB2 calls the connection exit.
  b. DB2 performs SECURITY LABEL verification for the new user, if applicable.

In all cases, if the primary authid does not have access to the trusted context, the connection request fails and returns to an unconnected state.

In Figure 8-4, we show the new option to switch authids in the DB2I Defaults Panel.



DB2I Authorization Switching

```
DSNEOP01                    DB2I DEFAULTS PANEL 1
COMMAND ===>

Change defaults as desired:

1 DB2 NAME .............   ===> DSN        (Subsystem identifier)
2 DB2 CONNECTION RETRIES   ===> 0          (How many retries for DB2 connection)
3 APPLICATION LANGUAGE     ===> IBMCOB     (ASM, C, CPP, IBMCOB, FORTRAN, PLI)
4 LINES/PAGE OF LISTING    ===> 60         (A number from 5 to 999)
5 MESSAGE LEVEL ........   ===> I          (Information, Warning, Error, Severe)
6 SQL STRING DELIMITER     ===> DEFAULT    (DEFAULT, ' or ")
7 DECIMAL POINT ........   ===> .          (. or ,)
8 STOP IF RETURN CODE >=   ===> 8          (Lowest terminating return code)
9 NUMBER OF ROWS .......   ===> 20         (For ISPF Tables)
10 CHANGE HELP BOOK NAMES?===> NO          (YES to change HELP data set names)
11 AS USER                 ===>            (Userid to associate with the trusted
                                           connection)
PRESS: ENTER to process END to cancel HELP for more information
```

Figure 8-4 New DB2I AS USER option

Switch in a remote connection

As a requester, DB2 automatically switches the user on a trusted connection to the primary authid when:

- The SYSTEM AUTHID differs from the primary authid associated with the application user.
- Outbound translation is required for the primary AUTHID and SYSTEM AUTHID row is defined in the USERNAMES table.
- The SYSTEM AUTHID differs from the authid in the SQL CONNECT statement with the USER and USING clauses.

When DB2, as a server, receives a request to switch users:

1. DB2 calls the connection exit routine, which associates the primary authid, any secondary authids, and an SQL ID with the remote request, replacing the previous IDs.
2. DB2 then determines if the primary authid is allowed to use the trusted connection: If WITH AUTHENTICATION is specified, an authentication token (a password, RACF passticket, or Kerberos ticket) is required.
3. DB2 performs SECURITY LABEL verification for the new user ID.
4. DB2 initializes the connection, ensuring that the environment is as though a new user had established the connection, for example, open cursors are closed and temporary table information is dropped.

5. If the primary authid is not allowed to use the trusted connection or if SECURITY LABEL verification fails, the connection is returned to an unconnected state.

8.3 Roles

DB2 extends the trusted context concept to optionally assign a default role to a trusted context and optionally assign a role to a user of the context.

Roles provide a more flexible technique than groups or users in assigning and controlling authorization, while improving consistency with the industry and improving security.

A database role is a virtual authorization ID that is assigned to an authid through an established trusted connection.

Within a trusted connection, DB2 allows one and only one role to be associated with a thread at any point in time.

A role is a database entity to which one or more DB2 privileges can be granted to or revoked from. Roles provide a means to acquire context-specific privileges.

A trusted context definition specifies one or more authids that are allowed to use it. As part of “allowing” an authid to use the context, you can also assign a role for each authid. A trusted context can be defined with or without a default role and with or without any assigned roles.

Any authorization ID using the trusted context can inherit a role assigned to its authid as part of the definition of the trusted context. If there was no role associated with that authid within this context, this authid inherits the privileges of the default role if one was defined for this context. Otherwise, the authid can only exercise all privileges that it had prior to initiating the connection (privileges granted in the usual manner and not through a role). An example of the benefits of a trusted context without roles is in 8.10.1, “Already verified DRDA requests into a DB2 server” on page 218.

If there is a default role and an assigned role, the authid’s assigned role takes precedence and the default role does not apply.

Multiple users can be allowed to use a particular trusted context. A user can be allowed to use multiple trusted contexts.

To support roles in a trusted context, DB2 extends the GRANT and REVOKE statements to add roles to the list of authorization names to which privileges are granted and revoked. Privileges can be granted and revoked by a role within an established trusted connection.

As mentioned earlier, role privileges are in addition to other sets of privileges granted to the primary and secondary authids, except for object ownership. Roles can create and own objects. If specified in the trusted context definition (ROLE AS OBJECT OWNER), a role becomes the owner of objects created in a trusted connection. Roles must have all the privileges necessary to create the objects. If a role owns a created object, the user requires a GRANT to access it outside the context.

You can revoke all privileges that are assigned to a role by simply dropping the role itself or using the REVOKE statement. When you attempt to drop a role, make sure that the role does not own any objects. If the role owns objects, the DROP statement is rolled back. If the role does not own any objects, the role is dropped. As a result, all privileges held by this role are revoked and the revocation is cascaded.

You can simplify the administration of authorization by having roles as object owners. For example, if the owner of an object is an authorization ID and if you need to transfer the ownership to another ID, you have to drop the object first and re-create it with the new authorization ID as the owner. You do not have to do the same if the owner is a role because all users that are associated with that role have the same owner privilege. This capability allows a DBA to have privileges to create objects and manage them for a time, even though ownership is to be another ID. A role is not managed through RACF.

The role can be assigned and removed from individuals through the trusted context as needed. This allows a DBA to perform object maintenance during a change control window and then lose the role privileges when the window is shut. Auditing trails of the work completed during the maintenance window are available for verification by a security administrator or auditor. The role approach is better than sharing a user ID/password that has administrative authority (SYSADM privilege), which is potentially contrary to a segregation of duties policy and is not auditable.

Roles are a way to allow multiple DBA authids have ownership of an object at the same time or at different times, rather than only one DBA authid having the ownership all the time.

There are many catalog tables that also reflect that ownership can be held by a role. For example, SYSIBM.SYSTABLES has a new column OWNERTYPE, where blank is for an AUTHID and L is for role.

When compared with RACF group secondary authids, trusted contexts and roles provide more dynamic controls for where and when and how DB2 privileges can be exercised. In addition, the DB2 log also contains information about creating and changes to the new database entities, roles, trusted contexts, and also about changes to their allowed users.

Install SYSADM and Install SYSOPR cannot be assigned to a role.

Roles provide:

- The capability for a DBA to acquire the privileges to create objects even though ownership is to be another ID.
- The ability to more finely control when and from where a privilege can be exercised. The role can be assigned and removed from individuals through the trusted context as needed. This allows a DBA to perform object maintenance during a change control window on a Saturday night, for example. But when Monday arrives, the role is removed, and the DBA no longer holds the privileges required to do this same work.
- The ability to audit the work completed during the maintenance window, even though the role made this possible. These audit trails are available for verification by a security administrator or auditor. This makes it possible to conclusively establish the identity of those performing the actions rather than the weaker scenario of a shared ID/password with high privileges.
- A mechanism other than authorization IDs through which you can assign privileges and authorities.

Characteristics of a role

A role is only relevant as part of one or more trusted contexts. It is only usable within an established trusted connection.

Prior to DB2 V9, if you used RACF to manage external access to DB2, you could use secondary authorization IDs to define user groups and associate primary authorization IDs with those user groups. Using this approach, primary authorization IDs are the RACF user IDs, the secondary authorization IDs are the names of the groups with which the primary IDs are associated.

Now in DB2 V9, the new database entity called role is recognized and used to control access to data. See Figure 8-5.

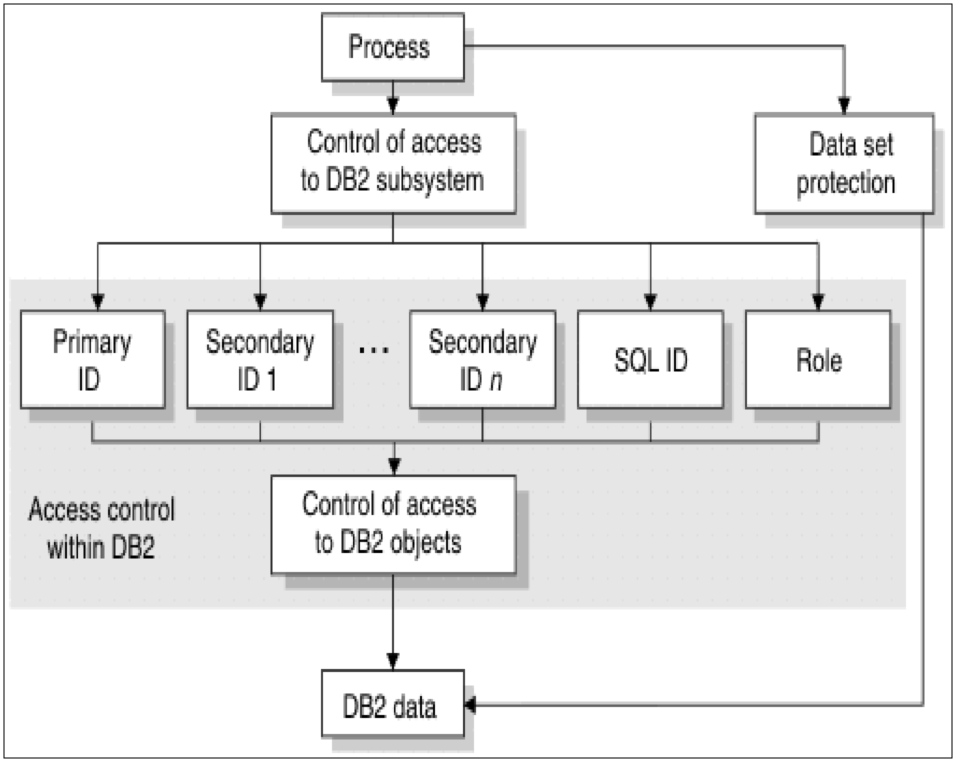

Process
Control of access to DB2 subsystem
Data set protection
Primary ID
Secondary ID 1
...
Secondary ID *n*
SQL ID
Role
Access control within DB2
Control of access to DB2 objects
DB2 data

Figure 8-5 Access control through a role

One of the ways that DB2 controls access to data is by using authorization IDs or roles. DB2 relies on IDs or roles to determine whether to allow or prohibit certain processes. DB2 assigns privileges and authorities to IDs or roles so that the owning users can take actions on objects. In this sense, it is an ID or a role, not a user, that owns an object. In other words, DB2 does not base access control on a specific user or person who need access. For example, if you allow other users to use your IDs, DB2 recognizes only the IDs, not the people or programs that use them.

Security and auditing

A role is a database entity that groups together one or more privileges. By associating a role with a user, the user inherits all the privileges held by the role, in addition to the set of privileges held by the user outside of the established trusted connection. A role is associated with a user only within one or more trusted contexts.

Therefore, when an SQL statement is processed within a trusted context, the DB2 authorization model considers the following permissions:

- The privileges granted to the primary authorization ID
- The privileges granted to the secondary authorization ID, if applicable
- The privileges granted to the role associated with the authorization ID of the statement
- The privileges granted to PUBLIC

All DB2 privileges and authorities that are granted can be granted to a role. For example, a role can be granted any of the following privileges:

- DBADM, DBCTRL, LOAD, CREATETAB database privileges
- BIND, EXECUTE plan and package privileges
- Any database object privilege

Role DDL, catalog tables, and grants

In this section, we describe role DDL, catalog tables, and grants.

DDL

Here, we describe the DDL for creating a role. Figure 8-6 shows the CREATE ROLE syntax.

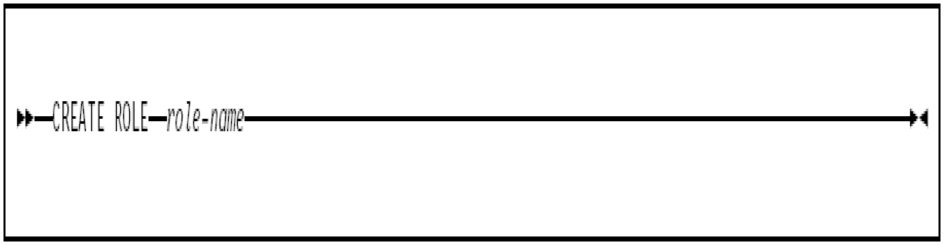


Figure 8-6 CREATE ROLE syntax

Role-name identifies that name of the role that is created. The name must not identify a role that exists at the current server.

New and modified SQL statements include:

- COMMENT ON ROLE
- CREATE ROLE
- DROP ROLE
- All GRANT/REVOKE statements are modified for ROLES. They are:
  - GRANT/REVOKE (collection privileges)
  - GRANT/REVOKE (database privileges)
  - GRANT/REVOKE (distinct type or JAR privileges)
  - GRANT/REVOKE (function or procedure privileges)

  - GRANT/REVOKE (package privileges)
  - GRANT/REVOKE (plan privileges)
  - GRANT/REVOKE (schema privileges)
  - GRANT/REVOKE (sequence privileges)
  - GRANT/REVOKE (system privileges)
  - GRANT/REVOKE (table or view privileges)
  - GRANT/REVOKE (use privileges)

Catalog tables

There are two new catalog tables to support roles:

- SYSIBM.SYSROLES contains one row for each role.
- SYSIBM.SYSOBJROLEDEP lists the dependent objects for each role.

Dropping a role

When a role is dropped, all privileges and authorities that have been previously granted to that role are revoked.

If the role is the owner of any of the following items, the action taken is as follows:

- Plans or packages are made inoperative.
- Cached dynamic statements are invalidated.

The role is not dropped if any REVOKE restrictions are encountered. REVOKE restrictions include those that are encountered during the cascading of the revocation of a role's privileges.

If RESTRICT is specified, the role is not dropped if any of the following dependencies exist:

- The role is associated with any trusted context or any user in a trusted context.
- The role is associated with a currently running thread.
- The role is the owner of any of the following:
  - Alias
  - Database
  - Distinct type
  - Index
  - JAR
  - Materialized query table
  - Plans
  - Packages
  - Role
  - Sequence
  - Storage group

- Stored procedure
- Table
- Table space
- Trigger
- Trusted context
- User-defined function
- View

8.4 Trusted contexts, roles, and MLS

A trusted context can exist without multilevel security. However, using them together allows for a user to be automatically switched to one of their defined SECLABELS for the duration of their work within the trusted connection. It also allows a default SECLABEL to be specified for the trusted context, thus associating a default SECLABEL to users that do not have one directly assigned (must also be a valid SECLABEL for those users).

Combining the network trusted context, roles, and MLS allows more precise control of security.

8.5 Challenges addressed by roles and trusted contexts

In this section, we describe how the new role and trusted context database entities overcome the challenges highlighted at the beginning of the chapter.

Trust all connection requests?

It is likely that only a subset of requests for any type and source are considered trusted. In addition, you might want to restrict trusted connections to one or more specific servers. You can control the source of incoming requests by creating trusted contexts that allow trusted connections to be controlled.

Application server user ID/password

Use trusted context and roles to provide added security for your network-attached application servers and thus limit exposure. These new capabilities allow the DBA to make GRANTs to a role, for example, SAP_ROLE, which can only be used from a specific list of IP addresses. If someone steals the application server's user ID/password, they will not be able to access the database from another place in the network unless they are also able to execute the SQL statement on one of the approved application servers. This protection requires no change to the code in the application server.

Dynamic SQL auditability

The new roles and trusted context database entities also enable customers to improve DB2 system auditing without compromising performance.

The user’s authid can be used to run database transactions, so the DB2 audit is able to identify the users individually (an important capability for meeting some regulatory compliance requirements).

User passwords can be optional.

The trusted context retains many of the performance benefits of connection pooling with auditability. A key element is the ability to reuse the same physical connection without the need to re-authenticate the user in DB2.

Already verified DRDA

With trusted context and changes to the communications database (CDB) in DB2 V9, customers are now able to identify DB2 servers that are trusted to send already verified connection requests.

This new functionality can be used to establish already verified TCP/IP connections and improves the ability to replace SNA connections with TCP/IP. The communications database is used to identify trusted connections and specify a SYSTEM AUTHID for the trusted context. This makes it possible to automatically propagate the user identity from one DB2 system to another.

Customers can now identify DB2 servers that are trusted to send already verified DRDA requests.

Shared SYSADM ID

DB2 V9 can help by enabling an auditable process. You can use the new trusted context and role support to implement DBA privileges that can easily be disconnected and reconnected to individual employees.

Privileges can be granted to a role. When the DBA needs to perform a system change, a trusted context is used to assign the role to the DBA temporarily and from a specific batch job or location. A trace is started, the DBA is given the request and executes the database changes, the trace is stopped and the trusted context is disabled, and lastly another DBA reviews the audit trace.

This kind of approach provides abilities similar to a shared SYSADM or DBADM user ID but avoids the known audit compliance problems associated with shared user IDs. The roles can own DB2 objects, so disconnecting a DBA from a role does not cause the objects to be cascade deleted.

With these capabilities, customers can create DBA procedures that can be audited and protected so that one individual cannot violate the established rules without being detected during the audit review.

Dual responsibilities

Although a total segregation of duties eliminates this risk, the new trusted context and role capabilities can be used to reduce this risk by limiting how and when the production environment can be accessed for this DBA, while allowing full privileges and full time access to the development environment.

Full-time access to sensitive/private data

By having roles own objects, even tighter controls can be implemented. Using roles allows customers to move to a model where DBAs have no access to data in production except when they are performing approved scheduled DBA activities. The ability to do those activities can be controlled with a trusted context and a temporary role.

DBADM can create view for another ID, but cannot drop/alter

A trusted context can allow an authid with DBADM or any permitted user to assume the identity of another user, such as the view owner, and then perform the desired actions.

Without reserving a RACF group, a table can be dropped

To further protect your tables from this known gap, grant the create table or DBADM privilege solely to a role with role ownership and use the role to create all the tables.

Privileges granted can be exercised from anywhere

When granting the UPDATE privilege on a payroll table to an authid, it is a better situation if this privilege is available to the authid only when it is connected to a computer located inside the company offices. The new trusted context and role capabilities allow the DBA to GRANT privileges, for example, UPDATE, that can only be used from a specified list of IP addresses.

8.6 Role ownership of objects

Outside trusted contexts and roles, object ownership is tied to a user. When a user creates an object, the user becomes the owner of the object. If that user changes jobs within the company or leaves the company, in order to remove the privileges of that user on the object, the object has to be dropped, and as a result, all grants associated with it are revoked. The object then has to be re-created and the privileges re-granted.

However, if the object owner is a role, removing user privileges does not require the object to be dropped and re-created.

When a role is defined for a trusted context, the role becomes the owner of the objects created in a trusted context if the ROLE AS OBJECT OWNER clause is specified. If a role is defined to be the owner, the role must have all the privileges necessary to create the objects.

The role exists as an object independent of its creator, so creating the role does not produce a dependency on its creator.

If a role owns a created object, the user inheriting the privileges of the role through a trusted context requires a GRANT to access it outside the trusted context.

Role ownership allows more tightly controlled security where DBAs only exercise privileges when they perform approved activities through a trusted context and its role.

If ROLE AS OBJECT OWNER is not specified, object ownership is determined as usual.

8.6.1 Plan and package ownership

When BIND or REBIND are issued in a trusted context specified with ROLE AS OBJECT OWNER, ownership is determined as follows:

- If OWNER is not specified, the role associated with the binder becomes the owner.
- If OWNER is specified, the role specified in the OWNER option becomes the owner (in a trusted context, the OWNER specified must be a role).

Note the following plan and package ownership considerations:

- For a package to be bound remotely with a role as the owner of the package at the remote DB2, the trusted context at the remote DB2 must be specified as ROLE AS OBJECT OWNER.
- BIND/REBIND on the package must be performed in a trusted context.
- If OWNER is specified for a remote BIND across a trusted connection, OWNER can be a role or an authid. Outbound authid translation is not performed for the OWNER.

- If the plan owner is a role and the application uses a package bound at a remote DB2 server, the plan owner privilege to execute the package is not considered at the remote DB2 server.

  The package owner/the process runner (as determined by DYNAMICRULES) at the DB2 server must have the EXECUTE privilege on the package at the server.

8.6.2 Ownership of other objects

Here we describe role ownership rules for objects other than plans and packages:

- If CREATE is issued by static SQL, for the role to become the owner of the objects created by executing the plan or package, the bind of that plan or package must have been performed in a trusted connection with ROLE AS OBJECT OWNER specified.

  Otherwise, normal object ownership rules apply.

- If CREATE is issued by dynamic SQL in a trusted context with ROLE AS OBJECT OWNER specified, the role becomes the owner of the objects.

  A limitation is that it is not possible to specify the owner of an object created in a trusted context through SET CURRENT SQLID. If specified, SET CURRENT SQLID is ignored.

  Otherwise, normal object ownership rules apply.

8.7 Communicating with other systems

In this section, we describe changes to the SYSIBM.LOCATIONS table in the communications database and provide an example of how to use location aliases to identify secure and unprotected connections.

Trusted

The communications database changed to allow a DB2 requester to establish a trusted context. A new column, TRUSTED, is added to the SYSIBM.LOCATIONS table. This column is supported only for connections using TCP/IP.

```
TRUSTED CHAR(1) NOT NULL WITH DEFAULT 'N'
```

This identifies if the connection to the remote location can be trusted. This is restricted to TCP/IP only. The following values apply:

- Y: Location is trusted. Access to the remote location requires a trusted context defined at the remote location.
- N: Location is not trusted.

Secure

A new column, SECURE, is added to the SYSIBM.LOCATIONS table to indicate whether a secure connection using SSL is required for the outbound DRDA connection:

```
SECURE CHAR(1) NOT NULL WITH DEFAULT 'N'
```

This indicates the use of the Secure Sockets Layer (SSL) protocol for outbound DRDA connections when local DB2 applications connect to the remote database server using TCP/IP. The following values apply:

- Y: Indicates that a secure connection using SSL is required for the outbound DRDA connection.
- N: Indicates that a secure connection is not required for the outbound DRDA connection.

LOCATION ALIAS considerations

The use of a LOCATION alias allows some DB2 applications to benefit from SSL protection, and others to be satisfied with unprotected connections.

To use location aliases to identify secure or unprotected connections:

- At the requester, define a row in SYSIBM.LOCATIONS specifying the location name to be used for non-secure communications, with the server’s DRDA port.
- Define a second row with a different location name to be used for secure communications, with the server’s SECPORT, and SECURE specified as Y.

8.8 Roles and secondary authids

Having described some of the ways roles and contexts can be used, you can see that these features are very powerful and can be used in many different ways. But how do they compare with secondary authids?

For the purpose of managing authorizations, roles are more flexible than groups or users. It is possible for RACF groups and roles to complement each other. Consider the following scenario.

Suppose there is a team of eight production DBAs and a team of five who runs the operational environment and take cares of the nighttime batch runs.

The DBAs can acquire basic DB2 privileges through a RACF group called PROD_BASIC, which provides access to the catalog, some display capabilities, but no access to data or utilities.

Then, roles (associated to an application) are created, for example, PROD_LOAN_ROLE, PROD_CREDIT_CARD_ROLE, or PROD_FX_ROLE.

A role is granted DBADM against an application's databases. For example, the PROD_LOAN_ROLE role has DBADM privilege on the 10 databases that belong to the LOAN application.

Privileges to make structural changes to the LOAN databases can be granted solely to PROD_LOAN_ROLE.

Each DBA has a context defined with their user ID and no default role. When one of the LOAN databases needs to be enhanced and there is an implementation scheduled, one of the eight DBAs has his or her context altered to set the DEFAULT ROLE to PROD_LOAN_ROLE with ROLE as OBJECT OWNER. Then, on the implementation weekend, the trusted context is enabled and a trusted connection is established. The DBA acquires the required privileges that are additional to what is available through the PROD_BASIC secondary authid. Using a trusted connection, the DBA makes changes to the LOAN database. When the implementation is complete, the DBA's trusted context is altered to remove the defaut role. The role is defined on a permanent basis and is made available only during implementation events. With this approach, the objects in the LOAN databases are always structurally changed through the role. The LOAN databases can be maintained by one or more DBAs. The role owns all the objects in the 10 LOAN databases.

A similar approach can be used for the operational team. The team also has basic privileges through a PROD_BASIC secondary authid. Set up the role PROD_UTIL permanently to allow this team of five to run DB2 utilities against production databases. The PROD_UTIL role can be the standard DEFAULT ROLE used when defining a trusted context for each member of this team. Or define a "utility" role for each application, providing more granularity and segregation.

8.9 IFCID support for trusted contexts and roles

Note the following instrumentation facility component identifier (IFCID) changes in support of trusted contexts and roles:

- Adds trusted context name, role name, original application user, and security token fields to the correlation header.
- Adds new statement and object types to IFCID 062 for trusted contexts and roles.
- Adds a new authid checked type field to IFCID 140.
- Adds a new grantor/revoker type field to IFCID 141.
- Adds a new table owner type field to IFCID 142.
- Changes IFCID 169 (audit trace) to include the new identifier type S for trusted context SYSTEM AUTHID translation.
- Adds a new IFCID 269 audit trace record to trace when a trusted connection is established or reused at a DB2 server.
- Adds a new IFCID 270 audit trace record to trace when a trusted context is created or altered at a DB2 server.
- Adds IFCIDs 269 and 270 to audit trace class 10.
- Adds a new role name field to IFCID 314.

8.10 Examples of roles and trusted contexts

For an extensive description of applying some of the new security techniques to a distributed three-tiered application using WebSphere, see Chapter 9, “A WebSphere implementation” on page 245.

We provide detailed examples of how trusted contexts and roles can be used within the realm of database administration and also to secure connections with other DB2 subsystems.

8.10.1 Already verified DRDA requests into a DB2 server

Many customers want to migrate from SNA and PRIVATE protocol to a solution based on TCP/IP and DRDA protocol. Prior to DB2 V9, lack of already verified support in TCP/IP prevented many customers from migrating to DRDA.

With trusted context and changes to the communications database (CDB) in DB2 V9, customers now can identify DB2 servers that are trusted to send already verified connection requests.

See Figure 8-7.

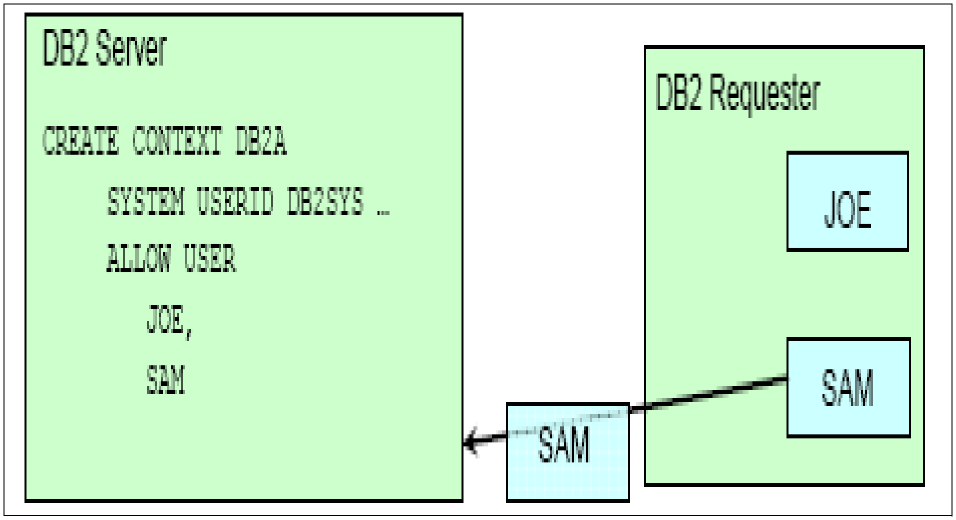


Figure 8-7 Already verified TCP/IP

Two DB2 subsystems, one requester, one server

In this scenario, V91A is a DB2 subsystem and a requester, and V91B is a DB2 subsystem and a server.

We show how USRT025 (as a requester from V91A) can remotely select data from the DSN8910.NEWPHONE table in subsystem V91B (server).

The setup we describe next, where incoming requests from anywhere are considered already verified, is somewhat of a security risk.

Server allows already verified incoming requests

The following steps illustrate this scenario:

1. The server subsystem (V91B) has the TCPALVER zparm set to YES.

   TCP/IP already verified set as YES means that a connection request is accepted with a user ID only, no password is required, and the requests are treated as already verified. This applies to all TCP/IP incoming requests. Regardless of location, the value must be the same for all members of a data sharing group.

2. Here are the contents of CDB tables in the requester subsystem V91A:
   - SYSIBM.LOCATIONS:

     ```
     LOCATION=STLEC1B, LINKNAME=TYEC1B, PORT=447, TRUSTED=N ,SECURE=N
     ```
   - SYSIBM.IPNAMES:

     ```
     LINKNAME=TYEC1B, SECURITY_OUT=A (already verified), USERNAMES =
     (no translation),IPADDRESS=9.30.114.22
     ```
   - SYSIBM.USERNAMES is empty, with no outbound translations defined.

     Note that STLEC1B identifies subsystem V91B.
3. USRT025 initiates a SELECT statement from SPUFI in DB2 V91A:

   ```
   SELECT LASTNAME,FIRSTNAME, PHONENUMBER FROM STLEC1B.DSN8910.NEWPHONE
   WHERE FIRSTNAME IN ('THEODORE','VINCENZO','WING','HELENA');
   ---------+---------+---------+---------+---------+---------
   LASTNAME         FIRSTNAME     PHONENUMBER
   ---------+---------+---------+---------+---------+---------
   SPENSER          THEODORE      0972
   LEE              WING          2103
   WONG             HELENA        2103
   DSNE610I NUMBER OF ROWS DISPLAYED IS 3 .
   ```

   This is successful because of two things. First, the SECURITY_OUT column in the IPNAMES table is set to A, which means “already verified.” Second, the system-wide zparm TCPALVER is set to YES.
4. While the previous query is running, a DISPLAY LOCATION is issued at the server V91B.

   The result of the command shows 9.30.114.22 as a requester:

   ```
   DSNL200I  & DISPLAY LOCATION REPORT FOLLOWS-
   LOCATION                           PRDID    REQSTR SERVER CONVS
   ::FFFF:9.30.114.22                               0      1      1
   ::FFFF:9.30.28.254                               0      1      1
   DISPLAY LOCATION REPORT COMPLETE
   ***
   ```
5. Now a DISPLAY THREAD(*) is executed at the server V91B:

   ```
   DSNV401I  ) DISPLAY THREAD REPORT FOLLOWS -
   DSNV402I  ) ACTIVE THREADS -
   NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
   TSO      T  *     3 ADMF001      ADMF001           0051   218
   TSO      TR      31 USRT025      USRT025  DSNESPCS 004C   216
    V444-USIBMSY.SYEC1DB2.BFD1949148B6=216 ACCESSING DATA AT
     STLEC1B-::FFFF:9.30.114.22..447
   DISPLAY ACTIVE REPORT COMPLETE
   ```

Server disallows already verified, context controls incoming requests

We show the effect of changing the TCPALVER zparm to NO in the server subsystem V91B. This blocks any already verified TCP/IP requesters from connecting to the V19B DB2 server. No incoming TCP/IP requests can connect to V91B without providing a password. The following steps illustrate this scenario:

1. The TCPALVER zparm is set to NO in module CHGZPARM.
2. In V91B, the server subsystem, the following command is executed:

   ```
   -SET SYSPARM LOAD(CHGZPARM)
   ```

3. USRT025 initiates the same SELECT statement from SPUFI in DB2 V91A:

   ```
   ---------+---------+---------+---------+---------+---------+-----
   SELECT LASTNAME,FIRSTNAME, PHONENUMBER
            FROM STLEC1B.DSN8910.NEWPHONE WHERE FIRSTNAME IN
     ('THEODORE','VINCENZO','WING','HELENA');
   ---------+---------+---------+---------+---------+---------+-----
   DSNT408I SQLCODE = -30082, ERROR:  CONNECTION FAILED FOR SECURITY
   REASON 17
            (UNSUPPORTED FUNCTION)
   ```

Now we show how we can define a trusted context on the server subsystem, V91B. This allows a trusted connection to be established with DB2 requester subsystem V91A.

After a trusted connection is established, V91B can accept an already verified authid from V91A and only from V91A:

1. At the server(V91B), we create a trusted context as follows:

   ```
   CREATE TRUSTED CONTEXT TRUSTED_SERVERS
    BASED UPON CONNECTION USING SYSTEM AUTHID USRT060
    ATTRIBUTES (ADDRESS '9.30.114.22')
    ENABLE
    WITH USE FOR USRT025 WITHOUT AUTHENTICATION ;
    DSNT400I SQLCODE = 000
   ```

2. At the requester(V91A), we make the following changes:

   ```
   UPDATE SYSIBM.LOCATIONS SET TRUSTED = 'Y' WHERE LOCATION =
   'STLEC1B';
   UPDATE SYSIBM.IPNAMES SET SECURITY_OUT = 'P', USERNAMES = 'S' WHERE
   LINKNAME = 'TYEC1B';
   INSERT INTO SYSIBM.USERNAMES VALUES('S', 'USRT060', 'TYEC1B', ' ',
   'ISITSAFE', 'N') ;
   ```

   Note that STLEC1B identifies subsystem V91B.

3. At the requester subsystem (V91A), we STOP and START DDF (to refresh the contents of the tables in the communications database (CDB).
4. At the requester (V91A), USRT025 exercises the same SELECT in SPUFI:

   ```
   SELECT LASTNAME,FIRSTNAME, PHONENUMBER
   FROM STLEC1B.DSN8910.NEWPHONE WHERE
   FIRSTNAME IN
   ('THEODORE','VINCENZO','WING','HELENA');
   ---------+---------+---------+---------+---------+---------+
   LASTNAME FIRSTNAME PHONENUMBER
   ---------+---------+---------+---------+---------+---------+
   SPENSER  THEODORE  0972
   LEE      WING      2103
   WONG     HELENA    2103
   DSNE610I NUMBER OF ROWS DISPLAYED IS 3
   DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
   ```

   The request is succesful, under the covers, with the proper CDB definitions. USRT025 comes in as already verified under the “umbrella” of USRT060, the system authorization ID associated with the trusted context.

5. Here are the results of a -DISPLAY LOCATION command at the server:

   ```
   DSNL200I  & DISPLAY LOCATION REPORT FOLLOWS-
   LOCATION                          PRDID    REQSTR SERVER CONVS
   ::FFFF:9.30.114.22                             0      1      1
     TRUSTED = 'Y'
   DISPLAY LOCATION REPORT COMPLETE
   ***
   ```

   Note the new information indicating that this is a trusted connection.

6. Here are results of a -DISPLAY LOCATION command at the requester:

   ```
   DSNL200I  ) DISPLAY LOCATION REPORT FOLLOWS-
   LOCATION                         PRDID    REQSTR SERVER CONVS
   ::FFFF:9.30.30.222                            0      1      1
   STLEC1B                         DSN09015      1      0      1
    ::FFFF:9.30.114.22
     TRUSTED = 'Y'
   DISPLAY LOCATION REPORT COMPLETE
   ***
   ```

   Note the new information indicating that this is a trusted connection.

7. Here are results of a -DISPLAY THREAD(*) command at the server:

```
DSNV401I  & DISPLAY THREAD REPORT FOLLOWS -
 DSNV402I  & ACTIVE THREADS -
 NAME     ST A  REQ ID          AUTHID   PLAN     ASID TOKEN
 TSO      T  *    3 ADMF001     ADMF001           0039    39
 TSO      RA *    3 USRT025     USRT025  DSNESPCS 003F    36
 V485-TRUSTED CONTEXT=TRUSTED_SERVERS,
      SYSTEM AUTHID=USRT060,
      ROLE=*
  V445-USIBMSY.SYEC1DB2.BFD1F1649087=36 ACCESSING DATA FOR
   ::FFFF:9.30.114.22
 DISPLAY ACTIVE REPORT COMPLETE
 DSN9022I  & DSNVDT '-DISPLAY THREAD' NORMAL COMPLETION
 ***
```

Note the new information indicating that this is a trusted connection, and the identity of the corresponding SYSTEM AUTHID and the ROLE. In this case, no ROLE is used in the definition of the trusted context.

Summary

This illustrates how customers are now able to identify DB2 servers that are trusted to send already verified connection requests, thus improving their ability to migrate from SNA and PRIVATE protocol to a solution based on TCP/IP and PRIVATE protocol. This is done through changes in the communications database and allows for automatic propagation of a user's identity from one DB2 subsystem to another.

You can use the same approach to control any incoming TCP/IP connection requests.

Note that you can also set up the requester location definition with SECURE=Y to require SSL encryption.

8.10.2 View maintenance on behalf of another user

In DB2 Version 8, a user ID holding the DBADM privilege can create a view for someone else. However, this same user ID is not able to drop or alter the view, or grant privileges on the view.

We describe a way to allow a DBA holding the DBADM privilege to perform maintenance on a view whose schema is the same as the user ID. In this example, USRT005 creates the table and the index. USRT001, the DBA holding the DBADM privilege, creates the view. The following steps illustrate this scenario:

1. With a user ID holding the SYSADM privilege, the following DDL is executed:

```
CREATE STOGROUP SGRB0101
    VOLUMES(XTRA01)
    VCAT DSNCAT;
CREATE DATABASE DBRB0101
    BUFFERPOOL BP1
    STOGROUP SGRB0101;

CREATE TABLESPACE TSRB0101 IN DBRB0101;

GRANT DBADM ON DATABASE DBRB0101 TO USRT001;
GRANT CREATETAB ON DATABASE DBRB0101 TO USRT005;
GRANT USE OF STOGROUP SGRB0101 TO USRT005;
GRANT USE OF TABLESPACE DBRB0101.TSRB0101 TO USRT005;
```

2. USRT005 creates a table and an index and populates the table:

```
CREATE TABLE USRT005.CUST(
    CUSTID         CHAR(10) NOT NULL PRIMARY KEY,
    MARITAL_STATUS CHAR(1),
    INCOME_RANGE   INT NOT NULL,
    ZIPCODE        INT,
    RESIDENCE      VARCHAR(5),
    EDUCATION      CHAR(8) NOT NULL WITH DEFAULT
    )
 IN DBRB0101.TSRB0101;

CREATE UNIQUE INDEX IUSO_CUSTID ON USRT005.CUST(CUSTID);

INSERT INTO USRT005.CUST VALUES
('1         ','M',64000,95030,'Own','COLLEGE ');
INSERT INTO USRT005.CUST VALUES
('2         ','M',41000,95030,'Own','HS      ');
INSERT INTO USRT005.CUST VALUES
('3         ','M',95000,95048,'Own','UNIV    ');
```

3. USRT001 creates a view:

   ```
   CREATE VIEW   USRT005.VCUST AS SELECT
       CUSTID         ,
       MARITAL_STATUS ,
       ZIPCODE        ,
       RESIDENCE      ,
       EDUCATION
     FROM USRT005.CUST ;
   ```

   This view omits the INCOME_RANGE column.

4. The DBA has made a mistake and needs to re-create the view. USRT001 runs:

   ```
   DROP VIEW  USRT005.VCUST ;
   ```

   This statement fails on a -551.

   ```
   USRT001 DOES NOT HAVE THE PRIVILEGE TO PERFORM OPERATION operation
   ON OBJECT USRT005.VCUST
   ```

5. With a user ID holding the SYSADM privilege, the following DDL is executed:

   ```
   CREATE TRUSTED CONTEXT CTXT_VIEW_MAINT
     BASED UPON CONNECTION USING SYSTEM AUTHID USRT001
     ATTRIBUTES (JOBNAME 'DROPVW',JOBNAME 'GRANTVW',JOBNAME 'ALTERVW')
     ENABLE
     WITH USE FOR USRT005 ;
   ```

   Note that, for this and the first example, a role is not defined for the trusted context, just to emphasize that you have a powerful construct even without using a role.

6. USRT001 submits the following BATCH job:

   ```
   //DROPVW JOB 'USER=$$USER','<USERNAME:JOBNAME>',CLASS=A
   //      ,MSGCLASS=A,MSGLEVEL=(1,1),USER=USRT001,REGION=0M,
   //         PASSWORD=LOSGATOS
   /*ROUTE PRINT STLVM14.YUKI
   //******************************************************
   //* DROPVW  JOB: - Drop view USRT005.VCUST to show
   //*       USRT001 can do this in a trusted context.
   //* - Must use DROPVW as the job name and run job with
   //*       USER=USRT001 and ASUSER(USRT005).
   //****************************************************
   //* DSNTEP3 STEP
   //****************************************************
   //CREATE1  EXEC TSOBATCH,DB2LEV=DB2A
   //SYSTSIN  DD *
   DSN SYSTEM(SSTR) ASUSER(USRT005)
   ```

```
RUN PROGRAM(DSNTEP3)
//SYSIN    DD *
 DROP VIEW  USRT005.VCUST ;
/*
//
```

The DROP is now successful because DB2 found a match for the SYSTEM AUTHID and the attributes defined for the trusted context. DB2 established a trusted connection and allowed USRT001 to “take over’ the identity of USRT005.

In this way, a database administrator can assume the identity of other users and perform actions on their behalf.

8.10.3 View maintenance on a view whose schema is not a user ID

We describe a way to perform maintenance on a view whose schema is not the same as a user ID.

In this example, a role holding the DBADM privilege creates and maintains the view. The following steps illustrate this scenario:

1. With a user ID holding the SYSADM privilege, the following DDL is executed:

```
CREATE STOGROUP SGRB0101
    VOLUMES(XTRA01)
    VCAT DSNCAT;
CREATE DATABASE DBRB0101
    BUFFERPOOL BP1 STOGROUP SGRB0101;

CREATE TABLESPACE TSRB0101 IN DBRB0101;

GRANT CREATETAB ON DATABASE DBRB0101 TO USRT005;
GRANT USE OF STOGROUP SGRB0101 TO USRT005;
GRANT USE OF TABLESPACE DBRB0101.TSRB0101 TO USRT005;
```

2. USRT005 now creates a table and an index and populates the table:

```
CREATE TABLE USRT005.CUST(
    CUSTID         CHAR(10) NOT NULL PRIMARY KEY,
    MARITAL_STATUS CHAR(1),
    INCOME_RANGE   INT NOT NULL,
    ZIPCODE        INT,
    RESIDENCE      VARCHAR(5),
    EDUCATION      CHAR(8) NOT NULL WITH DEFAULT
    )
 IN DBRB0101.TSRB0101;

CREATE UNIQUE INDEX IUSO_CUSTID ON USRT005.CUST(CUSTID);

INSERT INTO USRT005.CUST VALUES
('1         ','M',64000,95030,'Own','COLLEGE ');
INSERT INTO USRT005.CUST VALUES
('2         ','M',41000,95030,'Own','HS      ');
INSERT INTO USRT005.CUST VALUES
('3         ','M',95000,95048,'Own','UNIV    ');
```

3. With a user ID holding the SYSADM privilege, the following DDL is executed:

```
CREATE ROLE ROLE_CTXT_VIEW_MAINT_ACME;

CREATE TRUSTED CONTEXT CTXT_VIEW_MAINT_ACME
  BASED UPON CONNECTION USING SYSTEM AUTHID USRT001
  ATTRIBUTES (JOBNAME 'DROPVW',JOBNAME 'CRVIEW',JOBNAME 'ALTERVW')
  DEFAULT ROLE ROLE_CTXT_VIEW_MAINT_ACME
    WITH ROLE AS OBJECT OWNER
  ENABLE;

GRANT DBADM ON DATABASE DBRB0101 TO ROLE ROLE_CTXT_VIEW_MAINT_ACME;
```

4. USRT001 submits the following batch job to create the view:

```
//CRVIEW JOB 'USER=$$USER','',CLASS=A,
// MSGCLASS=A,MSGLEVEL=(1,1),USER=USRT001,REGION=0M,
// PASSWORD=LOSGATOS
/*ROUTE PRINT STLVM14.YUKI
//******************************************************
//*
//* CRVIEW JOB : - Create a view based upon USRT005.CUST
//* - Must use USRT001 for this statement.
//*
//******************************************************
//* DSNTEP3 STEP
//******************************************************
```

```
//CREATE1 EXEC TSOBATCH,DB2LEV=DB2A
//SYSTSIN DD *
DSN SYSTEM(V91A)
RUN PROGRAM(DSNTEP3)
//SYSIN DD *
CREATE VIEW ACME.VCUST AS SELECT
   CUSTID ,
   MARITAL_STATUS ,
   ZIPCODE ,
   RESIDENCE
 FROM USRT005.CUST ;
```

Note: The job name is important and must appear in the trusted context definition.

USRT001 does not have DBADM or SELECT on the USRT005.CUST table. The context enables the creation of the view.

In SYSIBM.SYSTABLES, we see the following information for the view ACME.VCUST:

```
CREATOR=ACME
 NAME=VCUST
 CREATEDBY=USRT001
 OWNER=ROLE_CTXT_VIEW_MAINT_ACME
 OWNER_TYPE=L  (indicates role)
```

5. USRT001 submits the following batch job to drop the view:

```
//DROPVW JOB 'USER=$$USER','',CLASS=A,
// MSGCLASS=A,MSGLEVEL=(1,1),USER=USRT001,REGION=0M,
// PASSWORD=LOSGATOS
/*ROUTE PRINT STLVM14.YUKI
//*****************************************************
//* DROPVW JOB: - Drop view ACME.VCUST to show
//* USRT001 can do this in a trusted context.
//* - Must use DROPVW as the job name and run job with
//* USER=USRT001.
//****************************************************
//* DSNTEP3 STEP
//****************************************************
//CREATE1 EXEC TSOBATCH,DB2LEV=DB2A
//SYSTSIN DD *
DSN SYSTEM(SSTR)
RUN PROGRAM(DSNTEP3)
//SYSIN DD *
 DROP VIEW ACME.VCUST ;
```

The DROP is succesful.

Note: The job name is important and must appear in the trusted context definition.

As we have seen, this kind of view can be maintained by a role.

8.10.4 Backing up a DBA, assuming the identity of another user ID

In this example, there are three application databases in a production environment. Each database is supported by a different DBA, each holding the DBADM privilege over their respective database. Work needs to be done on a particular database and the DBA is not available. We show, using a trusted context, how one DBA can act as another DBA in the event that the other DBA is on vacation or unavailable.

The three databases are:

- DLOAN
- DCUST
- DCARD

The following steps illustrate this scenario:

1. A user ID with SYSADM executes the following statements:

```
CREATE STOGROUP SGRB0201
    VOLUMES(XTRA01)
    VCAT DSNCAT;

CREATE DATABASE DLOAN
    BUFFERPOOL BP1
    STOGROUP SGRB0201;

CREATE DATABASE DMTG
    BUFFERPOOL BP1
    STOGROUP SGRB0201;

CREATE DATABASE DCARD
    BUFFERPOOL BP1
    STOGROUP SGRB0201;
GRANT DBADM ON DATABASE DLOAN TO USRT020;
GRANT DBADM ON DATABASE DMTG  TO USRT030;
GRANT DBADM ON DATABASE DCARD TO USRT040;
GRANT USE OF STOGROUP SGRB0201 TO USRT020;
GRANT USE OF STOGROUP SGRB0201 TO USRT030;
GRANT USE OF STOGROUP SGRB0201 TO USRT040
```

2. A new table needs to be created in the DCARD database.

   DBA USRT040 is not available.

3. USRT030 executes the following DDL:

```
CREATE TABLE  CARD.CUSTOMER(
    CUSTID         CHAR(10) NOT NULL PRIMARY KEY,
    CARD_NUMBER    CHAR(16),
    BALANCE        DEC(10,2),
    LIMIT          DEC(10,2)
    )
 IN DATABASE DCARD;
DSNT408I SQLCODE = -551, ERROR:  CARD DOES NOT HAVE THE PRIVILEGE TO
PERFORM
         OPERATION CREATE TABLE ON OBJECT DCARD
```

4. USRT030 is connected to RACF group CARD.

5. A user ID with SYSADM creates the following context to enable USRT030 to do this urgent implementation:

```
CREATE TRUSTED CONTEXT CTXT_BACKUP_PROD_DBA
  BASED UPON CONNECTION USING SYSTEM AUTHID USRT030
  ATTRIBUTES (JOBNAME 'USRT030')
  WITH USE FOR USRT040
  ENABLE
  ;
```

   This trusted context allows USRT030 to connect and switch to user ID USRT040 in order to implement the new table.

6. USRT030 logs on to TSO.

   This establishes a trusted connection because the SYSTEM AUTHID and job name match the attributes of a defined trusted context.

7. USRT030 goes to the DB2I defaults panel and sets ASUSER=USRT040:

```
DB2I DEFAULTS PANEL 1
COMMAND ===>

Change defaults as desired:

 1  DB2 NAME ............. ===> V91A      (Subsystem identifier)
 2  DB2 CONNECTION RETRIES ===> 0         (How many retries for DB2
                                           connection)
 3  APPLICATION LANGUAGE   ===> IBMCOB    (ASM, C, CPP, IBMCOB,
                                           FORTRAN, PLI)
 4  LINES/PAGE OF LISTING  ===> 60        (A number from 5 to 999)
 5  MESSAGE LEVEL ........ ===> I         (Information, Warning,Error,
                                           Severe)
 6  SQL STRING DELIMITER   ===> DEFAULT   (DEFAULT, ' or ")
 7  DECIMAL POINT ........ ===> .         (. or ,)
 8  STOP IF RETURN CODE >= ===> 8         (Lowest terminating return
                                           code)
 9  NUMBER OF ROWS ....... ===> 20        (For ISPF Tables)
10  AS USER                ===> USRT040   (Userid to associate with the
                                           trusted connection)
```

Within the established trusted connection, USRT030 is allowed to proceed as USRT040. If IFCID 269 is activated, DB2 records the user switch.

USRT030 as user USRT040 now holds the DBADM privilege on the DCARD database.

8. USRT030 as USRT040 executes the following statements from SPUFI:

```
CREATE TABLE CARD.CUSTOMER(
   CUSTID CHAR(10) NOT NULL PRIMARY KEY,
   CARD_NUMBER CHAR(16),
   BALANCE DEC(10,2),
   LIMIT DEC(10,2) )
 IN DATABASE DCARD;
---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+
```

The statement is successful because the DBADM privilege held by USRT040 is used.

In SYSIBM.SYSTABLES, we see that:

```
CREATOR=CARD,
NAME=CUSTOMER,
DBNAME=DCARD,
CREATEDBY=USRT040
OWNER=CARD.
```

9. The user ID with SYSADM disables the trusted context:

```
ALTER TRUSTED CONTEXT CTXT_BACKUP_PROD_DBA ALTER DISABLE ;
```

10. USRT030 tries to switch to USRT040 on the DB2I defaults panel. The result is:

```
USRT030 tries to access SPUFI ASUSER USRT040 and this fails with
YOU ARE NOT AUTHORIZED TO CONNECT TO DB2 ASUSER USRT040
***
```

Summary

We have shown that in a situation where an unexpected request occurred, it is possible, using a trusted context, for one DBA to assume the identity of another DBA and create the table that was urgently needed in the production environment. In our example, USRT030 is allowed to run under the authorization ID of USRT040. This is a powerful capability.

8.10.5 Securing DBA activities

Many customers are concerned about DBA access to sensitive customer data. Another common problem is the use of a shared SYSADM user ID or a shared DBADM user ID.

We provide an example of implementing a new application and database that contain private customer information. The implementation adds many tables to a new database called DCUST. Because of the sensitive nature of the data to be stored in this database, the DBAs only have access to the database during the implementation weekend. Afterward, they will have no access at all to the DCUST database.

The following steps illustrate this scenario:

1. A user ID with SYSADM authority executes the following command:

```
CREATE STOGROUP SGRB0401
    VOLUMES(XTRA01)
    VCAT DSNCAT;

CREATE DATABASE DCUST
    BUFFERPOOL BP1
    STOGROUP SGRB0401;

CREATE ROLE PROD_CUST_DBA_ROLE ;
GRANT DBADM ON DATABASE DCUST TO ROLE PROD_CUST_DBA_ROLE ;
GRANT USE OF STOGROUP SGRB0401 TO ROLE PROD_CUST_DBA_ROLE ;

 CREATE TRUSTED CONTEXT CTXT_PROD_CUST_DBA_ROLE_USRT029
   BASED UPON CONNECTION USING SYSTEM AUTHID USRT029
  DEFAULT ROLE PROD_CUST_DBA_ROLE WITH ROLE AS OBJECT OWNER
   ATTRIBUTES (JOBNAME 'USRT029')
   DISABLE
   ;

 CREATE TRUSTED CONTEXT CTXT_PROD_CUST_DBA_ROLE_USRT039
   BASED UPON CONNECTION USING SYSTEM AUTHID USRT039
  DEFAULT ROLE PROD_CUST_DBA_ROLE WITH ROLE AS OBJECT OWNER
   ATTRIBUTES (JOBNAME 'USRT039')
   DISABLE
   ;
```

This shows how two DBAs can make changes to the DCUST database in parallel by sharing a role that has the DBADM privilege on the DCUST database.

Outside of an established trusted connection, USRT029 and USRT039 do not have access to the DCUST database or the data in its tables.

Both contexts are created in a disabled state. The plan is to enable the contexts Saturday morning and disable them Sunday evening after the implementation is over.

2. Friday afternoon, USRT029 having some spare time, executes the following command:

```
CREATE TABLE CUST.CUSTOMER(
    CUSTID     CHAR(10) NOT NULL PRIMARY KEY,
    FIRST_NAME CHAR(32),
    LAST_NAME  CHAR(32),
    CUST_SINCE DATE,
    CREDIT_RATING INTEGER )
  IN DATABASE DCUST;
```

This results in:

```
DSNT408I SQLCODE = -551, ERROR:  USRT029 DOES NOT HAVE THE PRIVILEGE
TO
         PERFORM OPERATION CREATE TABLE FOR USER CUST ON OBJECT
CUST.CUSTOMER
```

A similar attempt is made by USRT039 with the same result.

USRT029 and USRT039 do not have the appropriate secondary authid to create a table with CUST as the creator.

3. A RACF administrator connects USRT029 and USRT039 to RACF group CUST.

4. Later that day, USRT029 tries again:

```
---------+---------+---------+---------+---------+---------+--------
 CREATE TABLE  CUST.CUSTOMER(
     CUSTID        CHAR(10) NOT NULL PRIMARY KEY,
     FIRST_NAME    CHAR(32),
     LAST_NAME     CHAR(32),
     CUST_SINCE    DATE,
     CREDIT_RATING  INTEGER
     )
```

```
  IN DATABASE DCUST;
---------+---------+---------+---------+---------+---------+--------
DSNT408I SQLCODE = -551, ERROR:  CUST DOES NOT HAVE THE PRIVILEGE TO
PERFORM
         OPERATION CREATE TABLE ON OBJECT DCUST
```

It fails again because USRT029 holds no privileges on the DCUST database because the context is not enabled.

5. Now, the weekend is here, so the user ID with SYSADM enables the trusted contexts:

```
ALTER  TRUSTED CONTEXT CTXT_PROD_CUST_DBA_ROLE_USRT029
  ALTER ENABLE ;

ALTER  TRUSTED CONTEXT CTXT_PROD_CUST_DBA_ROLE_USRT039
  ALTER ENABLE ;
```

   Both contexts are now enabled.

6. The user ID with SYSADM starts a trace on the role using the following command:

```
-START TRACE(ACCTG) DEST(SMF) ROLE(PROD_CUST_DBA_ROLE)
```

   The result is as follows:

```
DSNW130I  ) ACCTG TRACE STARTED, ASSIGNED TRACE NUMBER 04
DSN9022I  ) DSNWVCM1 '-START TRACE' NORMAL COMPLETION
```

> **Note:** Also consider starting the AUDIT trace:
>
> ```
> START TRACE(AUDIT) DEST(SMF)
> ```

7. USRT029 logs on to TSO.

   The trusted connection is established because both the SYSTEM AUTHID and the TSO JOB attributes match the attributes for trusted context CTXT_PROD_CUST_DBA_ROLE_USRT029. This trusted connection exists for the duration of the logon session.

8. Similarly, USRT039 logs on to TSO.

   The second trusted connection is established.

   Both DBAs now have acquired the DBADM privilege on the DCUST database because they inherit the privileges of the DEFAULT ROLE within their trusted connections.

9. USRT029 takes the following action:

```
CREATE TABLE  CUST.CUSTOMER(
     CUSTID        CHAR(10) NOT NULL PRIMARY KEY,
     FIRST_NAME    CHAR(32),
     LAST_NAME     CHAR(32),
     CUST_SINCE    DATE,
     CREDIT_RATING INTEGER
     )
  IN DATABASE DCUST;
---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+
 INSERT INTO CUST.CUSTOMER VALUES(
   '0001','TOM','THUMB',CURRENT_DATE,100);
---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

   USRT029 can now create tables in the DCUST database.

   You might have noticed an interesting feature in V9. The table has been defined with a primary key and the corresponding unique index has been automatically created. This is why the insert statement is successful.

   Also, we did not specify a table space. DB2 automatically created a table space in database DCUST.

10. USRT039 logs on and executes the following command:

```
CREATE TABLE  CUST.PROFILE(
    CUSTID        CHAR(10) NOT NULL PRIMARY KEY,
    ASSETS        DEC(10,2),
    LIABILITIES   DEC(10,2)
    )
 IN DATABASE DCUST;
INSERT INTO CUST.PROFILE VALUES
  ('0001',201267.10,85719.20);
```

   This implementation is actually a migration from another platform, and there are many tables to be created and populated. We only show the beginning of the activities that are to be done in parallel by both DBAs.

11. The person having the user ID with SYSADM privilege is curious to see whether the DBAs have started their work and issues the following command:

```
-DISPLAY THREAD(*)

DSNV401I  ) DISPLAY THREAD REPORT FOLLOWS -
DSNV402I  ) ACTIVE THREADS -
```

```
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
TSO      T  *     3 SYSADM       SYSADM            003D    95
TSO      N       36 USRT039      USRT039           003C     0
V485-TRUSTED CONTEXT=CTXT_PROD_CUST_DBA_ROLE_USRT039,
     SYSTEM AUTHID=USRT039,
     ROLE=PROD_CUST_DBA_ROLE
TSO      N       27 USRT029      USRT029           0023     0
V485-TRUSTED CONTEXT=CTXT_PROD_CUST_DBA_ROLE_USRT029,
     SYSTEM AUTHID=USRT029,
     ROLE=PROD_CUST_DBA_ROLE
DISPLAY ACTIVE REPORT COMPLETE
DSN9022I  ) DSNVDT '-DISPLAY THD' NORMAL COMPLETION
***
```

We see three connections, and two are trusted connections. This shows that DBAs can work in parallel under the same role within two different established trusted connections.

Note also that the PROD_CUST_DBA_ROLE role is the owner of all the objects created during the established trusted connections because both trusted contexts have the same default role and both context definitions have the WITH ROLE AS OBJECT OWNER clause.

12. It is now Sunday evening, and while USRT029 is still logged on, the user ID with SYSADM issues the following statement:

```
ALTER  TRUSTED CONTEXT CTXT_PROD_CUST_DBA_ROLE_USRT029
  ALTER DISABLE ;
```

13. USRT029 issues the following statement:

```
SELECT * FROM CUST.CUSTOMER;
00038000
---------+---------+---------+---------+---------+---------+
CUSTID      FIRST_NAME                        LAST_NAME
---------+---------+---------+---------+---------+---------+
0001        TOM                               THUMB
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
```

This is successful because the trusted connection is still active even though the trusted context has been altered.

14. USRT029 logs off and then logs back on, and reissues same statement:

```
SELECT * FROM CUST.CUSTOMER;
---------+---------+---------+---------+---------+---------+-------
DSNT408I SQLCODE = -551, ERROR:  USRT029 DOES NOT HAVE THE PRIVILEGE
 TO PERFORM OPERATION SELECT ON OBJECT CUST.CUSTOMER
```

When USRT029 logs on this time, there is no match possible with the trusted context because it is in a disabled state. So, be aware that when disabling a trusted context, an existing established connection is allowed to continue until the end of the connection.

At this point, USRT039 is still logged on and can run this SELECT query successfully.

15. The user ID holding SYSADM executes the following command:

```
ALTER  TRUSTED CONTEXT CTXT_PROD_CUST_DBA_ROLE_USRT039
   ALTER DISABLE ;
```

Again, even though the trusted context is disabled, USRT039 retains the DBADM privilege within the established connection.

USRT039 logs off, and both USRT029 and USRT039 no longer hold any privileges on the DCUST database.

16. The trace is stopped:

```
-STOP  TRACE(ACCTG) DEST(SMF) ROLE(PROD_CUST_DBA_ROLE)
```

17. We produce an audit report.

We review all the activities done by the role PROD_CUST_DBA_ROLE. The trace shows that all the activities performed under the role are attributable to USRT029 and USRT039.

You can leave the trace on to continually monitor the access to the DCUST database by the role.

Summary

We have shown how a role can be used in a manner similar to a shared user ID with DBADM.

This method:

- Avoids the audit compliance problems associated with shared user IDs and is auditable.
- Enables the objects created to be owned by the role, removing the complications associated with a user ID owning the objects. Disabling a person’s access to the role does not cause the objects to be cascade deleted.
- Provides additional protection for the data, because the DBAs only have access to the objects during an implementation and not all the time.

You can use trusted context and role support to implement DBA privileges that can easily be disconnected and reconnected to individual employees. You can use a similar approach to implement a PROD_SYSADM_ROLE that is enabled only when required. Another possible use is to define a context and role that allows a DBA to exercise the DB2 runtime client only from a specific remote IP address, for example, at their vacation location during an “emergency.”

With these capabilities, customers are able to create DBA procedures that can be audited and protected so that one individual cannot violate the established rules without being detected during the audit review.

8.10.6 Using a temporary SYSADM role during major ERP release implementation

We have an existing ERP application in production. A new release will be implemented over the weekend. The ERP application currently has thousands of tables and indexes. The application’s databases are all within a single data sharing group and this application is the only one residing in that group.

The new release will make substantial changes, including dropping 50 tables, altering many table spaces, and creating new databases, table spaces, tables, and indexes.

The implementation events include a user demo portion during which business users will exercise old and new functionality. Because of a lack of documentation and the magnitude of the functionality available within the application and the high number of tables, it is not clear exactly which tables might be affected during the user demo activity.

If a “no go” decision is reached after the user demo, not only do the object structures need to be reversed, the data in the tables changed during the demo must also be restored to the original content. Not knowing exactly which tables could have been affected makes documenting the fallback procedures very challenging.

Because this ERP application resides by itself in a data sharing group, the decision is made to capture the complete current state of the application with a BACKUP SYSTEM step at the very beginning of the implementation. (Note that testing of system-level backup and restore has been done and it is known how long a system backup and restore is expected to take for this ERP application.)

The production DBA (USRT009) executing the implementation holds the DBADM privilege on all the databases within this ERP application in this data sharing group. The BACKUP SYSTEM command requires SYSADM authority.

Because there are many activities during this implementation weekend, the person who normally holds the SYSADM privilege within this data sharing group decides to temporarily allow the DBA to perform the BACKUP SYSTEM command when the time is right, thus freeing herself from being on call to perform this activity. She will also start the appropriate traces for the duration of the entire weekend in order to be able to review exactly what was done by USRT009 while holding the SYSADM privilege. On Monday morning, a full review of the audit info will be done.

The following steps illustrate this scenario:

1. The person holding SYSADM in production runs the following command:

   ```
   CREATE ROLE PROD_ERP_TEMP_SYSADM;
   GRANT SYSADM TO ROLE PROD_ERP_TEMP_SYSADM;

   CREATE TRUSTED CONTEXT CTXT_PROD_ERP_TEMP_SYSADM_USRT009
      BASED UPON CONNECTION USING SYSTEM AUTHID USRT009
      DEFAULT ROLE PROD_ERP_TEMP_SYSADM
      ATTRIBUTES (JOBNAME 'BKSYSERP')
      DISABLE
      ;
   ```

   It is created ahead of the weekend and in a disabled state.

2. On the Saturday morning, the person holding SYSADM runs the following command:

   ```
   -START TRACE(ACCTG) DEST(SMF) ROLE(PROD_ERP_TEMP_SYSADM)

   ALTER  TRUSTED CTXT_PROD_ERP_TEMP_SYSADM_USRT009
     ALTER ENABLE;
   ```

   > **Note:** Also consider starting the AUDIT trace:
   >
   > `START TRACE(AUDIT) DEST(SMF)`

3. When the implementation begins, USRT009 runs a utility job named BKSYSERP, which includes the following statement:

   ```
   BACKUP SYSTEM
   ```

   Because this is submitted by URT009 with a job name of BKSYSERP, a trusted connection is established and USRT009 acquires the SYSADM privilege through the PROD_ERP_TEMP_SYSADM role and is able to successfully run the utility.

4. USRT009 proceeds with the remaining activities.
5. The user demo takes place and a “go” decision is made.

6. On Monday morning, the user ID holding SYSADM issues the following command:

```
ALTER  TRUSTED CTXT_PROD_ERP_TEMP_SYSADM_USRT009
  ALTER DISABLE;
-STOP TRACE(ACCTG) DEST(SMF) ROLE(PROD_ERP_TEMP_SYSADM)
```

7. The same person produces an audit report.

   The Monday review will show all activities done while the trusted connection existed.

Summary

We demonstrated an auditable process that allowed a DBA to perform an activity that is not possible within the privileges they normally hold. This has allowed one person to do the work, rather than involving two people. The SYSADM privilege was only available within a trusted connection, and the trusted connection was only possible while the trusted context for that DBA was enabled. Afterward, because the trace is at the role level, it was possible to audit everything that was done while the SYSADM privilege was available to USRT009.

8.10.7 Reducing risk of a table being dropped by another person

We describe how some sloppy procedures can lead to someone else dropping a table you created. The following steps illustrate this scenario:

1. USRT011 has DBADM privilege on the DXYZ database and runs the following command:

```
CREATE TABLE    XYZ.SALE_INFO
   (CUST_ID     CHAR(9)         NOT NULL             ,
    PRODUCT_NBR INTEGER         NOT NULL             ,
    PRICE       DECIMAL (10,2)  NOT NULL             ,
    SOLD_DATE   DATE            NOT NULL )
       IN DXYZ.SXYZSLI
       AUDIT NONE
       CCSID EBCDIC;
---------+---------+---------+---------+---------+--------
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

2. USRT044 is disgruntled and wants to cause some trouble and runs the following command:

```
SELECT * FROM XYZ.SALE_INFO;
---------+---------+---------+---------+---------+---------+--------
DSNT408I SQLCODE = -551, ERROR:  USRT044 DOES NOT HAVE THE PRIVILEGE
TO PERFORM
         OPERATION SELECT ON OBJECT XYZ.SALE_INFO
```

USRT044 has access to the DB2 subsystem but holds no privileges on the new table.

3. USRT044 is also the RACF administrator on this z/OS image and executes the following commands within RACF:

```
ADDGROUP XYZ SUPGROUP(HLEVEL)
CONNECT USRT044 GROUP(XYZ)
```

4. Then from SPUFI, USRT044 executes:

```
---------+---------+---------+---------+---------+---------
SELECT * FROM XYZ.SALE_INFO;
---------+---------+---------+---------+---------+---------
CUST_ID    PRODUCT_NBR         PRICE  SOLD_DATE
---------+---------+---------+---------+---------+---------
DSNE610I NUMBER OF ROWS DISPLAYED IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------
DROP  TABLE   XYZ.SALE_INFO;
---------+---------+---------+---------+---------+---------
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------
```

This is now successful because:

- USRT044 has XYZ as a secondary authid.
- XYZ is the authorization ID of the table owner.
- USRT011 did not protect the XYZ.* tables by arranging for the RACF group XYZ to be created and properly controlled.

USRT011 has DBADM on DXYZ and had only one created table. To further protect the XYZ.* tables, a role will be used to create the objects.

5. USRT011 requests a person with SYSADM to do the following action:

```
REVOKE DBADM ON DATABASE DXYZ FROM USRT011;

CREATE ROLE DEV_XYZ_DBA_ROLE;
GRANT DBADM ON DATABASE DXYZ TO ROLE DEV_XYZ_DBA_ROLE;
GRANT USE OF STOGROUP SGRB0401 TO ROLE DEV_XYZ_DBA_ROLE;

 CREATE TRUSTED CONTEXT CTXT_DEV_XYZ_DBA_ROLE_USRT011
   BASED UPON CONNECTION USING SYSTEM AUTHID USRT011
  DEFAULT ROLE DEV_XYZ_DBA_ROLE WITH ROLE AS OBJECT OWNER
   ATTRIBUTES (JOBNAME 'USRT011')
   ENABLE
   ;
```

Now all tables created in this database will be created through the role, and even though USRT044 can create RACF group XYZ and connect USERT044 to it, no malicious activity can take place.

Summary

When no RACF group was reserved and controlled for XYZ.* tables, the door was open to malicious activity. By using a role as the object owner, that risk is eliminated.

8.10.8 Limiting salary updates from a single source

To allow salary updates to be done only from a specific user and a specific IP address within a trusted connection, execute:

```
CREATE ROLE USRROLE1;
CREATE TRUSTED CONTEXT CTX1
BASED UPON CONNECTION USING SYSTEM AUTHID PAUL
ATTRIBUTES (ADDRESS '2.65.102.200')
ENABLE;
ALTER TRUSTED CONTEXT CTX1
ADD USE FOR SAM ROLE USRROLE1;

GRANT UPDATE (EMPNO, EMPSALARY) ON TABLE EMPPAYROLL TO ROLE USRROLE1;
```

SAM can use the privileges associated with role USRROLE1 to update the EMPNO and EMPSALARY columns in the EMPPAYROLL table, which are not available to SAM outside this context.

8.11 Summary

The enhancements described in this chapter strengthen DB2 security in the z/OS environment.

Roles and trusted contexts provide an opportunity for system administrators to control access to enterprise objects at a level of abstraction that is close to the structure of their enterprise.

These new database entities and the concept of trusted connections provide considerable benefits:

- Manageability:
  - Allows database administrators to manage and control access from network-attached middleware servers.

  - Reusing a connection without the need for DB2 to authenticate the new user provides the ability to switch without a password. This eliminates the need to manage multiple passwords at the client.
  - The role object ownership in a trusted context eliminates the object dependency on authorization IDs.
  - Authorization IDs already acquire the necessary database privileges that are associated with either an assigned role or a default role specific to the trusted context, simplifying management of dynamic SQL privileges.
  - Allows an authid to acquire a special set of privileges that are not available to it outside the trusted context.
- User accountability without compromising performance:
  - Knowing the user's identity provides improved data access auditing capability. Therefore, the middleware server can distinguish the transactions performed by the middleware server for its own purpose from those performed on behalf of some user.
  - The role object ownership in a trusted context gives the ability to customers to eliminate shared user IDs and have each person audited and accountable with their own authorization IDs.
- Improved security:
  - With trusted context, administrators can configure their system in such a way that the SYSTEM AUTHID has all the privileges only when used from the trusted context. This eliminates the concern about the theft of the middle tier SYSTEM AUTHID. Even if someone learns the SYSTEM AUTHID credentials, they would not be able to penetrate the DB2 server unless they also had the ability to penetrate the trusted middle tier server as well.
  - Roles holding privileges can only be exercised within a trusted connection.
  - With roles in a trusted context, privileges granted to authorization IDs can be restricted, which improves security, allowing the authorization IDs to perform only the approved activities.
  - You now have the ability to control where, when, and how local and remote requesters communicate with DB2.

9

A WebSphere implementation

In this chapter, we create an application applying some of the security mechanisms and techniques outlined in this book to a distributed application using IBM WebSphere Application Server Version 6.1 with DB2 V9 for z/OS.

We implement:

- A trusted context using roles.
- The propagation of user IDs from the client to DB2.
- Enterprise Identity Mapping (EIM) for mapping user IDs with authorization IDs on DB2. For more information about EIM, see Appendix C, “Enterprise Identity Mapping” on page 327.
- Secure Sockets Layer (SSL) encryption instead of DRDA encryption.

The goals are to:

- Limit the access to the DB2 server by the authorization IDs belonging to the application to a single IP address.
- Improve auditing of users by being able to pinpoint which user did what.
- Ensure the highest available encryption on distributed connections to DB2.

Let us first look at the overall application architecture. We then describe the basic setup at the application server layer and, finally, implement some of the features available to enhance security and auditing.

9.1 The architecture behind the application

The application runs in a typical three-tier environment. We have DB2 V9 for z/OS acting as a database server running on z/OS V1.7. The application is a Java servlet running in WebSphere Application Server V6.1 on a Microsoft® Windows® server, connecting to DB2 using a type 4 Universal Java driver. The application is invoked through any Internet browser. See Figure 9-1.

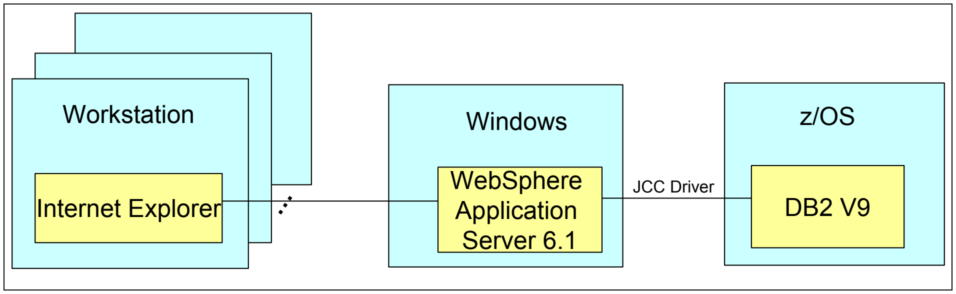


Figure 9-1 Three-tiered application model

The architecture is typical of a three-tiered application model using DB2 for z/OS as a database server. With respect to security, authentication, and auditing, it represents a number of distributed applications that up until DB2 V9 had the same challenges.

Among these distributed applications are SAP NetWeaver, PeopleSoft, Siebel, CRM, Lotus Domino, and, of course, WebSphere Application Server.

The challenges faced by these applications relate to the use of a single authorization ID to connect and run all the application logic. When using a single authorization ID for everything, regardless of the application invoked or the user invoking it, you are faced with the following problems:

- The authorization ID must hold all the privileges necessary to invoke any part of the application, no matter whether they are related.
- The application alone must control which parts of the application each user is permitted to use.
- If the authorization ID is compromised, the impact might be very large.
- It is impossible to monitor the access by users.

We propose solving these problems by using some of the new features available in DB2 V9 for z/OS.

9.2 The implementation

There are a number of steps to go through to implement the security enhancements available for distributed applications connecting to DB2 V9 for z/OS. Some are naturally linked and only work together, such as the propagation of user IDs and the use of EIM. Others are independent.

We first look at the basic setup needed for the application server, and then go through the different enhancements step by step.

We develop the application using IBM Rational® Application Developer V6.0.1. Rational Application Developer is a development platform for developing J2CE and J2EE™ applications with a focus on applications to be deployed to WebSphere Application Server and WebSphere Portal.

In this instance, we develop Java servlets using J2EE with WebSphere Application Server.

9.2.1 Setting up the application

Rational Application Developer provides creation wizards for different types of projects. We chose to create a dynamic Web project that contains resources needed for Web applications, such as JSPs, Java servlets, HTML, and other files.

The dynamic Web project wizard provides the capability to configure the version of the Java servlet specification, target server, EAR file name, and context root.

We do not be going through the entire setup because this is beyond the scope of this book, but only point out where something special has to be done in relation to the security features.

For further information about Rational Application Developer, see *Rational Application Developer V6 Programming Guide*, SG24-6449.

We update the Web deployment descriptor, which contains elements that describe how to deploy the Web application and its contents to the servlet container within the Web server, including the security settings to be used in the runtime environment.

Specifically, we add a security role named *eimsecurity* and a security constraint *GetMethodConstraint*. A security role is a logical grouping of principals; access to operations is controlled by granting access to a role.

The constraints define the policies for access to Web resources and which roles are authorized to use them. In our case, the resources are defined as the **GET** method of the servlet and **/*** for the URL pattern, meaning that the constraint applies to the get method with any URL pattern. See Figure 9-2.

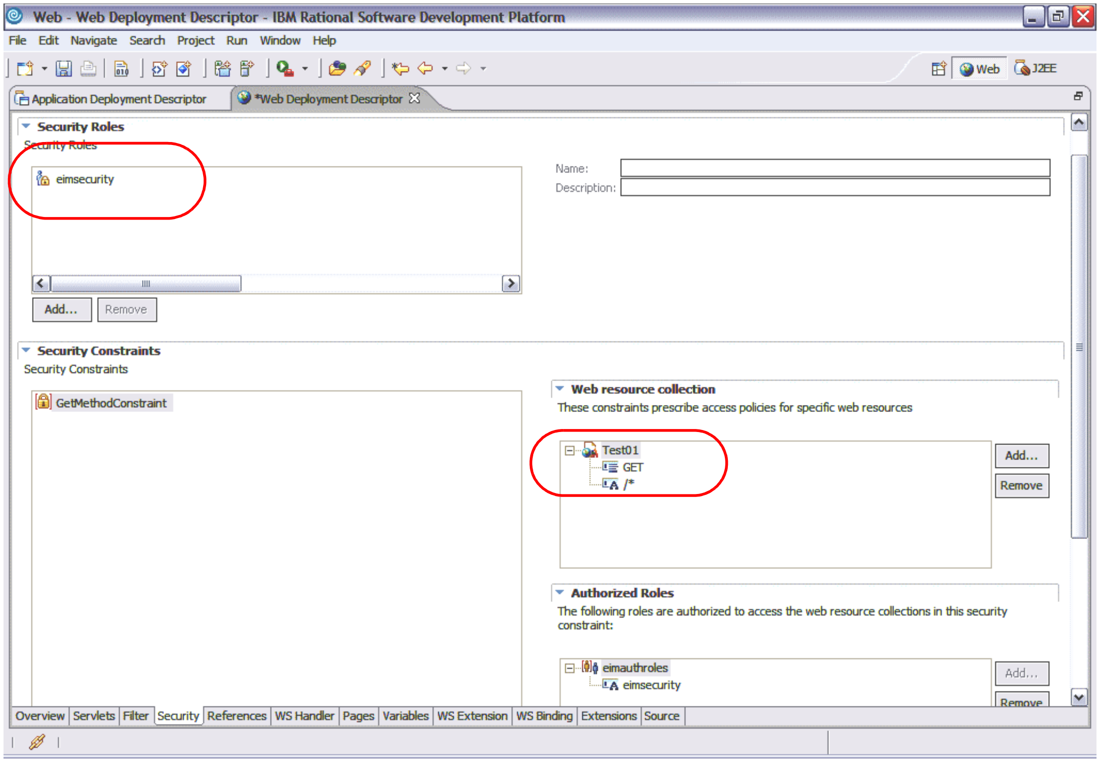

Web - Web Deployment Descriptor - IBM Rational Software Development Platform
File Edit Navigate Search Project Run Window Help
Web J2EE
Application Deployment Descriptor
*Web Deployment Descriptor
Security Roles
Security Roles
eimsecurity
Name:
Description:
Add... Remove
Security Constraints
Security Constraints
GetMethodConstraint
Web resource collection
These constraints prescribe access policies for specific web resources
Test01
GET
/*
Add... Remove
Authorized Roles
The following roles are authorized to access the web resource collections in this security constraint:
eimauthroles
eimsecurity
Add... Remove
Overview Servlets Filter Security References WS Handler Pages Variables WS Extension WS Binding Extensions Source

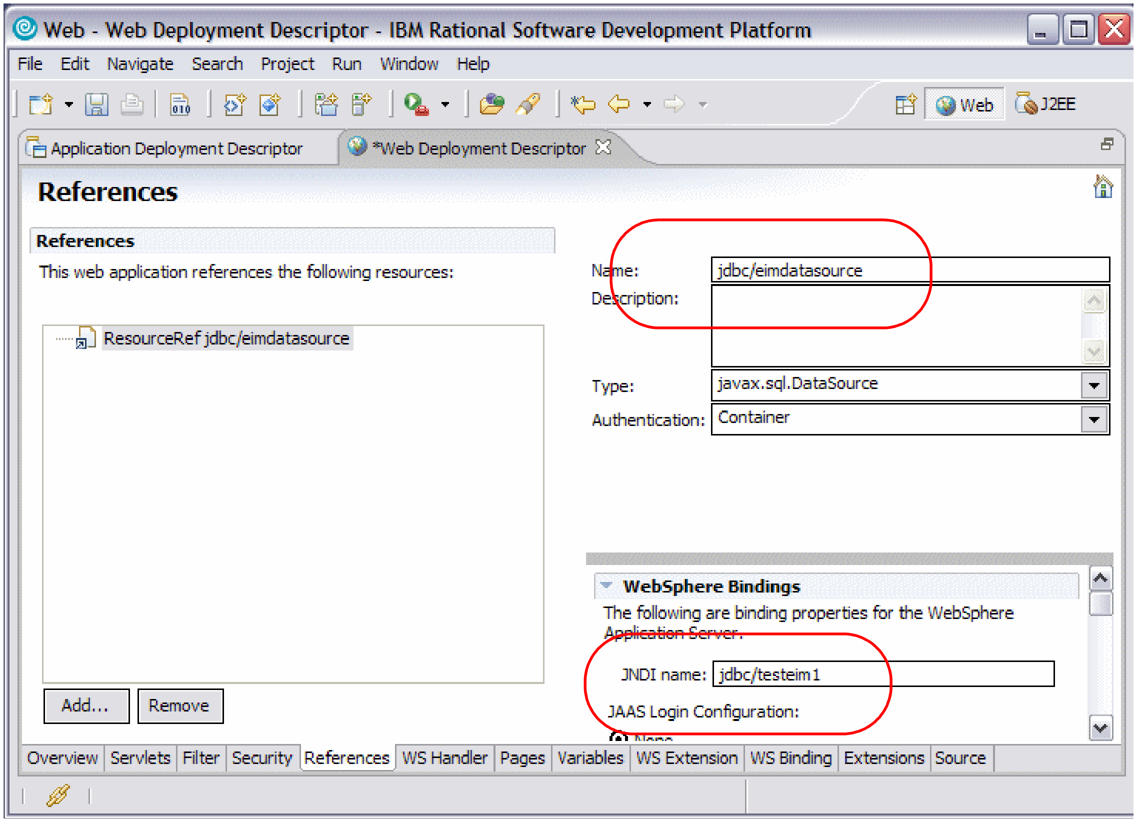
Figure 9-2 Web Deployment Descriptor (1 of 2)

Also in the Web deployment descriptor, we create a resource reference. The resource reference maps the name used in the application to the actual JNDI resource specified in WebSphere.

The Java Naming and Directory Interface™ (JNDI) is part of the Java platform, providing applications based on Java technology with a unified interface to multiple naming and directory services.

The reference should be defined with **Container** authentication.

We name the resource jdbc/eimdatasource. This is the name we will use in the Java servlets. The JNDI name is jdbc/testeim1. This is the name we will use in WebSphere Application Server. See Figure 9-3.



References

References

This web application references the following resources:

ResourceRef jdbc/eimdatasource

Name: jdbc/eimdatasource

Description:

Type: javax.sql.DataSource

Authentication: Container

WebSphere Bindings

The following are binding properties for the WebSphere Application Server.

JNDI name: jdbc/testeim1

JAAS Login Configuration:

Figure 9-3 Web Deployment Descriptor (2 of 2)

Finally, we create the Web application enterprise archive (EAR) file. EAR files represent a J2EE application that can be deployed to WebSphere Application Server. The EAR file format is based on the Java archive (JAR) file format. The EAR file contains the deployment descriptor.

9.2.2 Setting up the application server

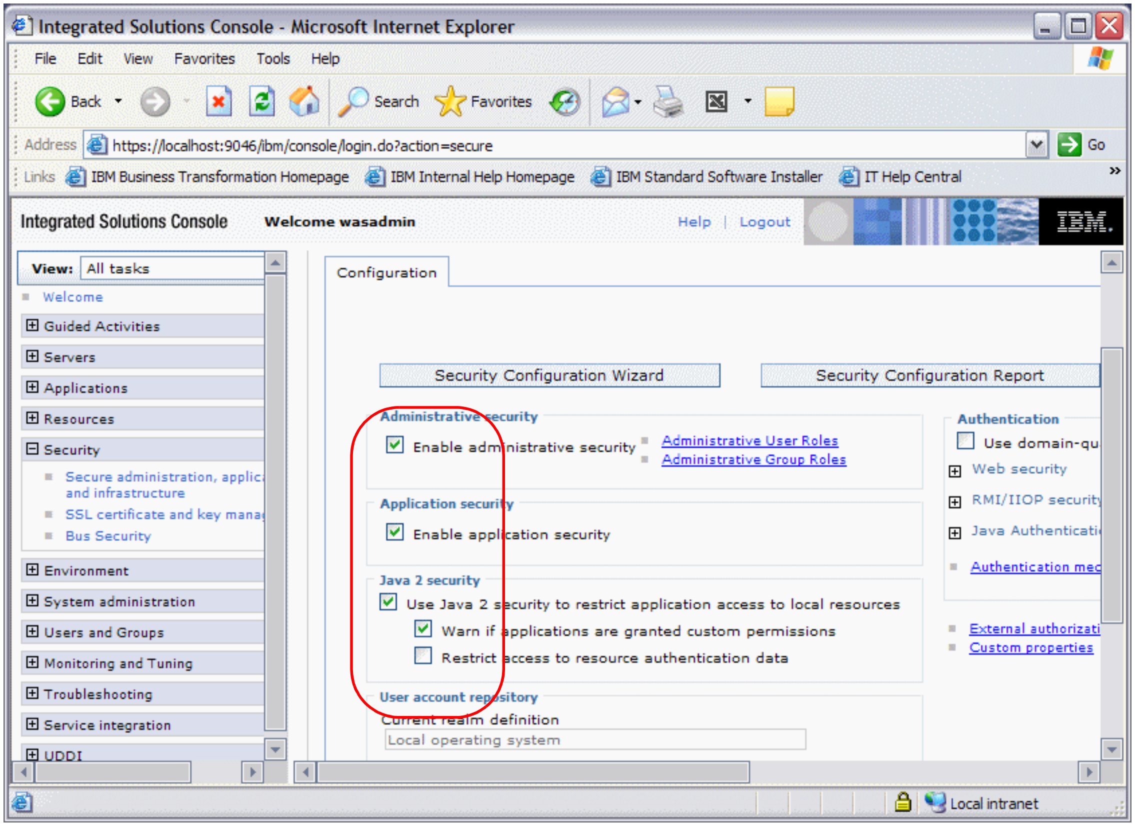
For the application server, we first need to create Windows user accounts for the users of the application. We define users USRT001, USRT002, and USRT003.

We then need to configure security, users, connections, and so on in WebSphere Application Server. We do all this using the WebSphere Application Server administrative console.

Again, it is beyond the scope of this book to describe all the setup needed for the application server. We focus on the things needed for the security features. For further information, refer to *WebSphere Application Server V6.1: System Management and Configuration*, SG24-7304.

We set up general security information and user account repository by invoking the security configuration wizard. From the overall panel on the left, choose **Security → Secure administration, application and infrastructure**.

Select both **Administrative security** and **Application security**. Set the user account repository to **Local operating system**, and use **Java 2 security**, as shown in Figure 9-4.


Integrated Solutions Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://localhost:9046/ibm/console/login.do?action=secure
Go
Links IBM Business Transformation Homepage IBM Internal Help Homepage IBM Standard Software Installer IT Help Central
Integrated Solutions Console
Welcome wasadmin
Help | Logout
IBM.
View: All tasks
Welcome
Guided Activities
Servers
Applications
Resources
Security
Secure administration, applic
and infrastructure
SSL certificate and key mana
Bus Security
Environment
System administration
Users and Groups
Monitoring and Tuning
Troubleshooting
Service integration
UDDI
Configuration
Security Configuration Wizard
Security Configuration Report
Administrative User Roles
Administrative Group Roles
Authentication
Web security
RMI/IIOP security
Java Authenticati
Authentication mec
External authorizati
Custom properties
Local intranet

Administrative security

☑ Enable administrative security

Application security

☑ Enable application security

Java 2 security

☑ Use Java 2 security to restrict application access to local resources

☑ Warn if applications are granted custom permissions

☐ Restrict access to resource authentication data

User account repository

Current realm definition

Local operating system

☐ Use domain-qu

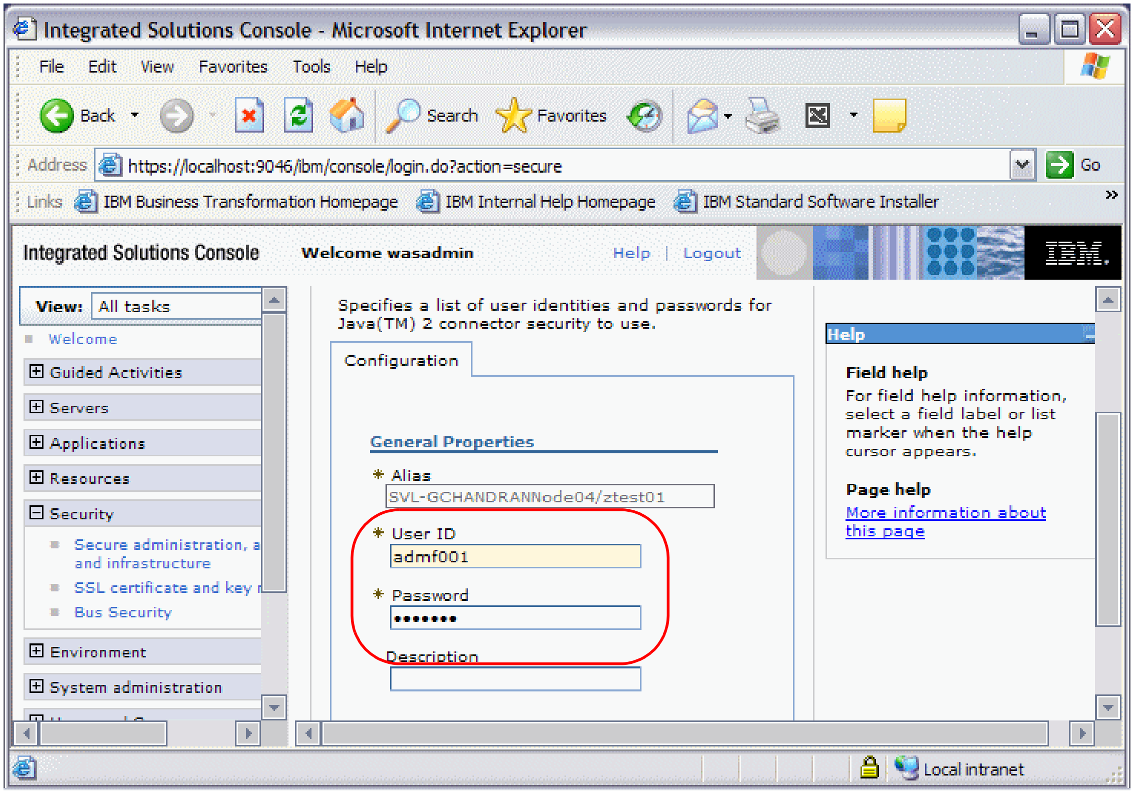
Figure 9-4 Security Configuration Wizard (1 of 2)

We then set up the Java authentication parameters. We define an alias and the user ID and password for the Java connector to use to connect to DB2 on behalf of the WebSphere application.

This user ID and password must be a valid RACF user ID and password on z/OS; it is to be used as the system authorization ID for the trusted context.

From the same window, we select **Java Authentication and Authorization Service** → **J2C authentication data** → **New**.

We enter the user ID `admf001` and corresponding password, as shown in Figure 9-5.


Integrated Solutions Console - Microsoft Internet Explorer
File
Edit
View
Favorites
Tools
Help
Back
Search
Favorites
Address
https://localhost:9046/ibm/console/login.do?action=secure
Go
Links
IBM Business Transformation Homepage
IBM Internal Help Homepage
IBM Standard Software Installer
Integrated Solutions Console
Welcome wasadmin
Help | Logout
IBM.
View: All tasks
Welcome
Guided Activities
Servers
Applications
Resources
Security
Secure administration, and infrastructure
SSL certificate and key
Bus Security
Environment
System administration
Configuration
Local intranet

Specifies a list of user identities and passwords for Java(TM) 2 connector security to use.

General Properties

* Alias: SVL-GCHANDRANNode04/ztest01

* User ID: admf001

* Password: •••••••

Description:

Help

Field help

For field help information, select a field label or list marker when the help cursor appears.

Page help

More information about this page

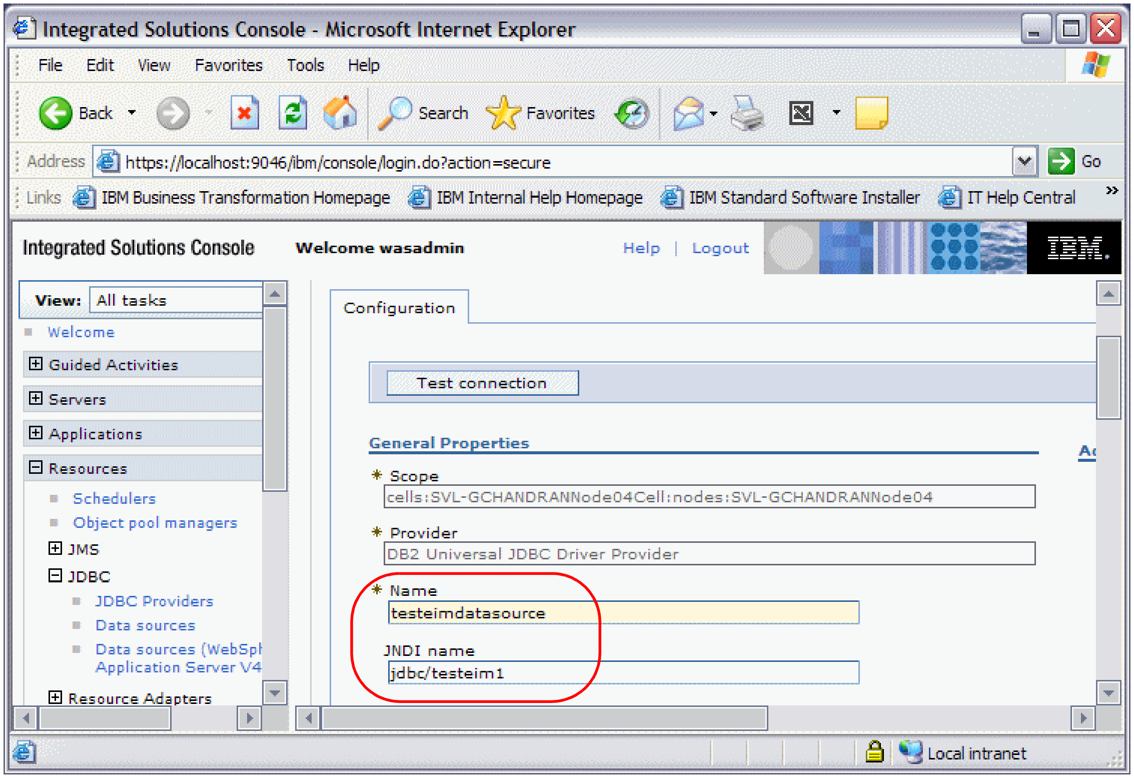
Figure 9-5 Security Configuration Wizard (2 of 2)

We now turn to the configuration of the data source and JDBC™ provider. We require DB2 JDBC Universal Driver 3.1.6 or later.

From the Resources panel, we select **JDBC** → **Data Sources** → **New**.

We give the data source a name and a JNDI name. Note that this must match the JNDI name entered in Rational Application Developer.

We enter `testeimdatasource` for the name and `jdbc/testeim1` for the JNDI name, as shown in Figure 9-6.



Address: https://localhost:9046/ibm/console/login.do?action=secure

* Scope: cells:SVL-GCHANDRANNode04Cell:nodes:SVL-GCHANDRANNode04

* Provider: DB2 Universal JDBC Driver Provider

* Name: testeimdatasource

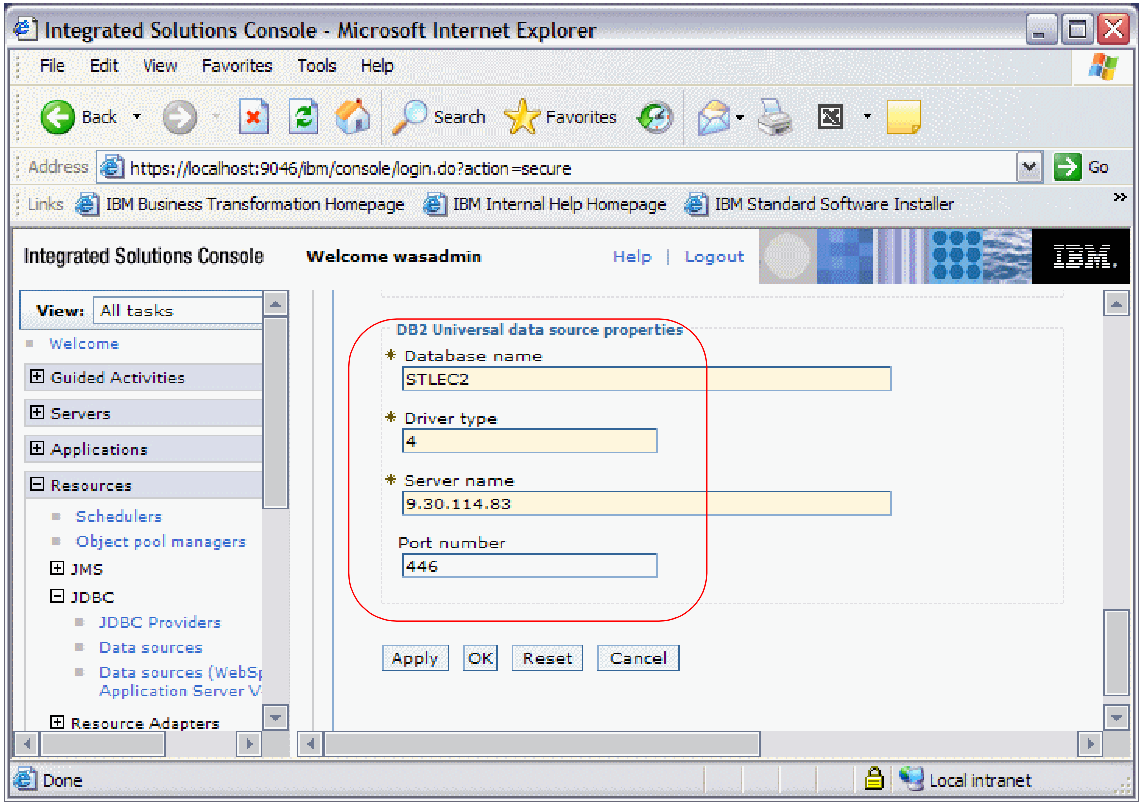
JNDI name: jdbc/testeim1

Figure 9-6 Configuring a data source (1 of 2)

Further down in the data source definition, we enter the information needed to connect to DB2.

This consists of the database name, which is the location name for the DB2 subsystem, the driver type, the IP address or server name of the z/OS, and the port number used for TCP/IP connections to the DDF. All this information, except for driver type, is available in the output of the DISPLAY DDF command.

We enter `STLEC2` for the database name, `9.30.114.83` for server name, and `446`, the default, for the port number, as shown in Figure 9-7. The driver type in this case is `4`, a type 2 driver would require the use of DB2 Connect™.


Integrated Solutions Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://localhost:9046/ibm/console/login.do?action=secure Go
Links IBM Business Transformation Homepage IBM Internal Help Homepage IBM Standard Software Installer
Integrated Solutions Console
Welcome wasadmin
Help | Logout
IBM.
View: All tasks
Welcome
Guided Activities
Servers
Applications
Resources
Schedulers
Object pool managers
JMS
JDBC
JDBC Providers
Data sources
Data sources (WebSp Application Server V
Resource Adapters
Apply OK Reset Cancel
Done
Local intranet

DB2 Universal data source properties

* Database name: STLEC2

* Driver type: 4

* Server name: 9.30.114.83

Port number: 446

Figure 9-7 Configuring a data source (2 of 2)

We are now ready to install the EAR file, created in Rational Application Developer. This was the file that contained the Web deployment descriptor with security definitions used in the application.

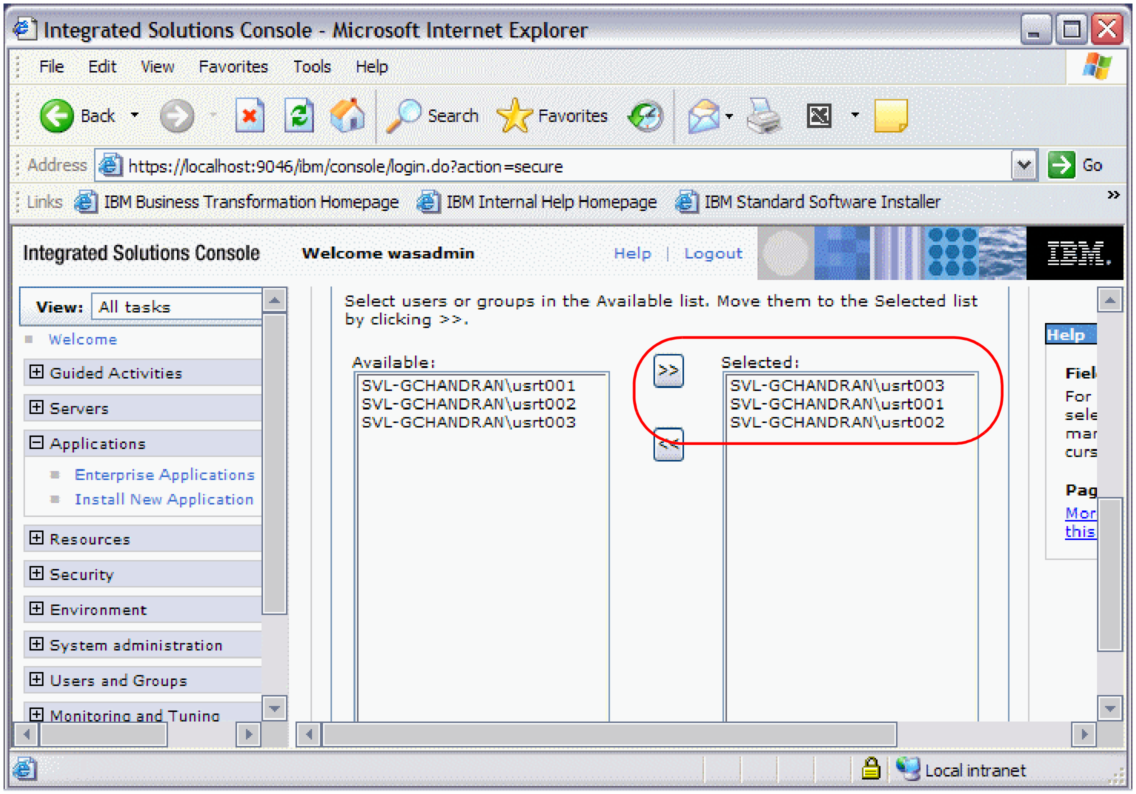
Do this from the Applications panel by choosing the **Install New Application** option and supplying the name of the EAR file.

Finally, we set up the mapping of users to the security role that was previously defined in Rational Application Developer and now installed. These are now the users authorized to use the application.

To do this, from the Applications panel, select **Enterprise Applications** → **Test** (the name of application we just installed) → **Security role to user/group mapping**.

The users are picked from the Windows user registry.

We select the users **usrt001**, **usrt002**, and **usrt003**, as shown in Example 9-8.



Select users or groups in the Available list. Move them to the Selected list by clicking >>.

Available:
SVL-GCHANDRAN\usrt001
SVL-GCHANDRAN\usrt002
SVL-GCHANDRAN\usrt003

Selected:
SVL-GCHANDRAN\usrt003
SVL-GCHANDRAN\usrt001
SVL-GCHANDRAN\usrt002

Figure 9-8 Mapping user to security role

This completes the setup of the application and application server. We now have an application that is characterized by all the challenges with respect to security and audibility mentioned earlier.

We are ready to start implementing enhancements.

9.3 Adding security features

We start with the most simple enhancement and gradually build on that, introducing more and more security features as we go along.

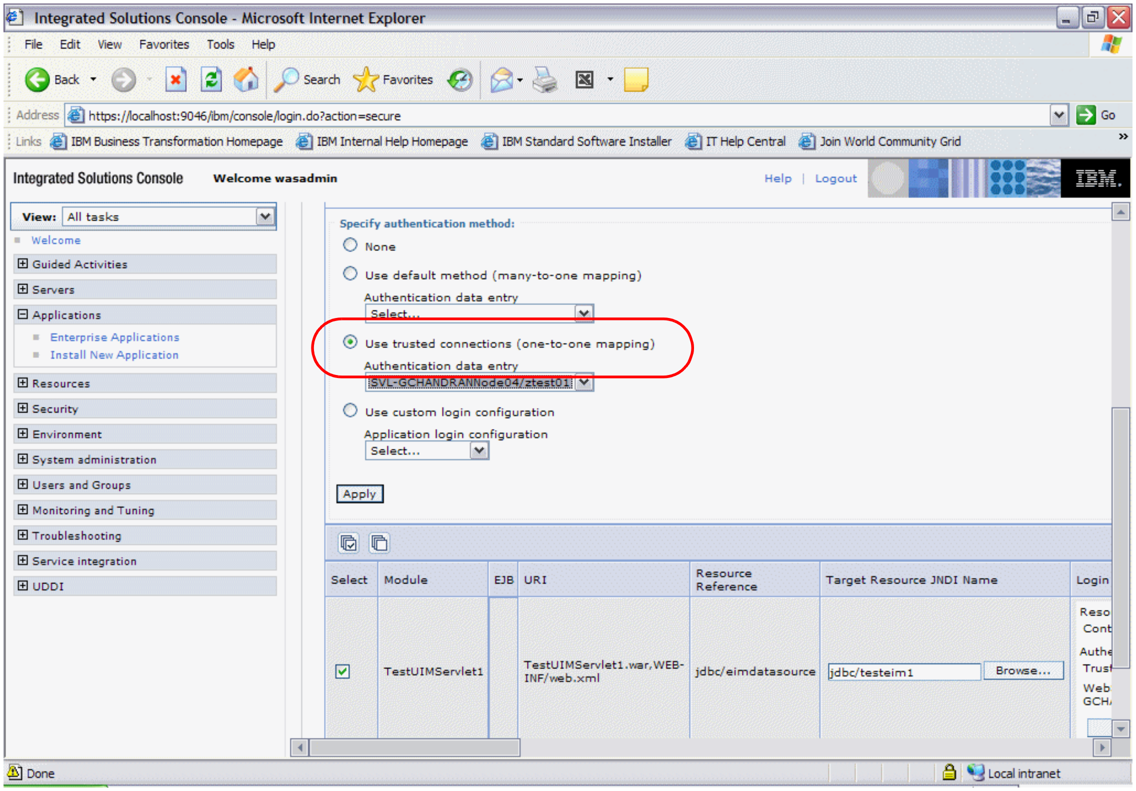
We use a simple Java servlet that connects to DB2 V9 for z/OS and selects some rows from a table.

9.3.1 Propagation of user IDs to DB2

As mentioned, WebSphere Application Server applications and most other distributed applications connect to DB2 for z/OS using the same authorization ID each time, namely the authorization ID of the application server process. We create an application that propagates the user ID to DB2.

The propagation of user IDs requires WebSphere Application Server V6.1. It is also available for CLI/ODBC.

First, the propagation of user IDs to DB2 can only happen in a trusted context. This is enabled from WebSphere Application Server. From the Applications panel, choose **Enterprise Applications** → **Test** (the name of our application) → **Resource references**. Select the authentication method **Use trusted connections**, as shown in Figure 9-9.


Integrated Solutions Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://localhost:9046/ibm/console/login.do?action=secure
Go
Links IBM Business Transformation Homepage IBM Internal Help Homepage IBM Standard Software Installer IT Help Central Join World Community Grid
Integrated Solutions Console Welcome wasadmin
Help | Logout
IBM.
View: All tasks
Welcome
Guided Activities
Servers
Applications
Enterprise Applications
Install New Application
Resources
Security
Environment
System administration
Users and Groups
Monitoring and Tuning
Troubleshooting
Service integration
UDDI
Done
Local intranet

Specify authentication method:

○ None

○ Use default method (many-to-one mapping)

Authentication data entry: Select...

● Use trusted connections (one-to-one mapping)

Authentication data entry: SVL-GCHANDRANNode04/ztest01

○ Use custom login configuration

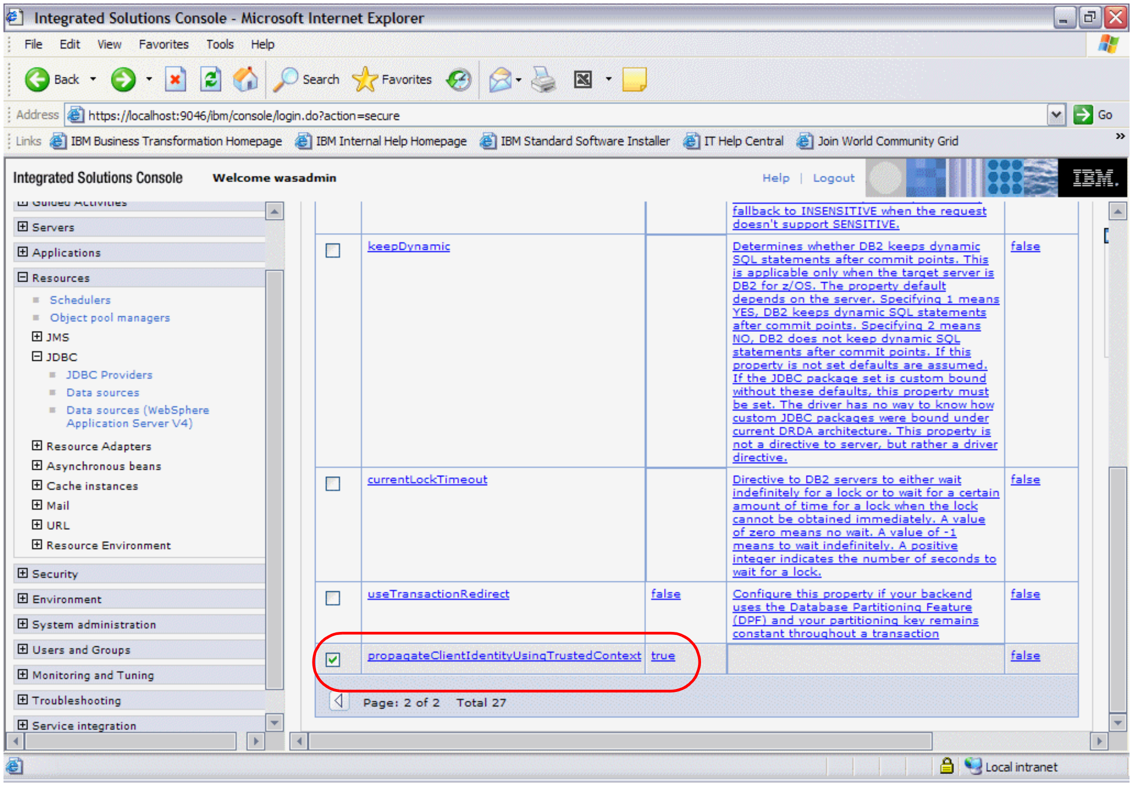
Application login configuration: Select...

Apply

Select	Module	EJB	URI	Resource Reference	Target Resource JNDI Name	Login
☑	TestUIMServlet1		TestUIMServlet1.war,WEB-INF/web.xml	jdbc/eimdatasource	jdbc/testeim1 Browse...	Reso Cont Authe Trust Web GCH/

Figure 9-9 Use trusted connections

Now the WebSphere Application Server is set up to use a trusted context. To enable the propagation of user IDs within the trusted context, from the Resources panel, choose **JDBC** → **Data sources** → **testeimdatasource** (the name of our data source) → **Custom properties**, as shown in Figure 9-10. We set the flag **propagateClientIdentityUsingTrustedContext**.


Integrated Solutions Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://localhost:9046/ibm/console/login.do?action=secure
Go
Links IBM Business Transformation Homepage IBM Internal Help Homepage IBM Standard Software Installer IT Help Central Join World Community Grid
Integrated Solutions Console Welcome wasadmin
Help | Logout
IBM.
Guided Activities
Servers
Applications
Resources
Schedulers
Object pool managers
JMS
JDBC
JDBC Providers
Data sources
Data sources (WebSphere Application Server V4)
Resource Adapters
Asynchronous beans
Cache instances
Mail
URL
Resource Environment
Security
Environment
System administration
Users and Groups
Monitoring and Tuning
Troubleshooting
Service integration
Local intranet

			fallback to INSENSITIVE when the request doesn't support SENSITIVE.	
□	keepDynamic		Determines whether DB2 keeps dynamic SQL statements after commit points. This is applicable only when the target server is DB2 for z/OS. The property default depends on the server. Specifying 1 means YES, DB2 keeps dynamic SQL statements after commit points. Specifying 2 means NO, DB2 does not keep dynamic SQL statements after commit points. If this property is not set defaults are assumed. If the JDBC package set is custom bound without these defaults, this property must be set. The driver has no way to know how custom JDBC packages were bound under current DRDA architecture. This property is not a directive to server, but rather a driver directive.	false
□	currentLockTimeout		Directive to DB2 servers to either wait indefinitely for a lock or to wait for a certain amount of time for a lock when the lock cannot be obtained immediately. A value of zero means no wait. A value of -1 means to wait indefinitely. A positive integer indicates the number of seconds to wait for a lock.	false
□	useTransactionRedirect	false	Configure this property if your backend uses the Database Partitioning Feature (DPF) and your partitioning key remains constant throughout a transaction	false
☑	propagateClientIdentityUsingTrustedContext	true		false

Page: 2 of 2 Total 27

Figure 9-10 Propagate client identity

We now need to define the trusted context in DB2. We do this with the DDL in Example 9-1.

Example 9-1 Create a trusted context

```
CREATE TRUSTED CONTEXT WASCTXT1
BASED UPON CONNECTION USING SYSTEM AUTHID ADMF001
ATTRIBUTES (ADDRESS '9.30.30.207')
WITH USE FOR USRT001, USRT002, USERT003
ENABLE;
```

The trusted context is named WASCTXT1 and it is reachable from 9.30.30.207, the IP address of the Windows server. The system authorization ID that is allowed to establish a connection through the trusted context is ADMF001, matching the user name entered previously in the Java authentication parameters of WebSphere Application Server.

When the trusted connection is established, the users USRT001, USRT002, and USRT003 are allowed to use the connection by switching the user ID. We kept the default with respect to AUTHENTICATION, which is without authentication, meaning that no password is necessary when switching to a new user.

Finally, we create RACF users with the same names as the Windows user IDs. Even though the user ID and password are not authenticated, RACF still needs to confirm the existence of the user ID and assign primary and secondary authorization IDs and CURRENT SQLID.

When created, we are ready to use the application with the new setup. We launch the Java servlet, logging on with user USRT001, and expect to get the same result from the application as before, only now the connection should be associated with USRT001.

More precisely, the following steps take place with respect to authentication:

1. The user logs on with user ID USRT001.
2. WebSphere Application Server authenticates the user ID and password.
3. Java security requests a connection to DB2 with user ID ADMF001 and the password.
4. DB2 calls in RACF to authenticate the user ID and password. RACF also assigns ADMF001 as the primary authorization ID and as the CURRENT SQLID. Secondary authorization IDs may also be assigned.
5. DB2 checks that the primary authorization ID ADMF001 is the system authorization ID of the trusted context, and the IP address 9.30.30.207 of the incoming request matches the “address” connection trust attribute of the trusted context. Then, the connection is established as trusted.
6. The application now issues a *switch user* request using USRT001. Note that this is not done explicitly by the Java servlet, but rather implicitly by WebSphere Application Server.
7. USRT001 is authenticated by RACF, and USRT001 is assigned as the primary authorization ID and as the CURRENT SQLID. Secondary, authorization IDs may also be assigned.
8. DB2 checks that the primary authorization ID USRT001 is allowed to use the connection, and the connection is initialized.

We use the DISPLAY THREAD command to get information about the connection, as shown in Example 9-2.

We see that an active thread exists with the authorization ID USRT001. The thread is established through trusted context WASCTXT1 and the system authorization ID associated with it is ADMF001.

We also see that the thread is established from 9.30.30.207, the IP address of the Windows server, and that the user ID associated with the thread is admf001.

Example 9-2 Display thread using trusted context

```
)DIS THD(*)
DSNV401I  ) DISPLAY THREAD REPORT FOLLOWS -
DSNV402I  ) ACTIVE THREADS - 822
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA *     6 db2jcc_appli USRT001 DISTSERV 0044    19
V485-TRUSTED CONTEXT=WASCTXT1,
     SYSTEM AUTHID=ADMF001,
     ROLE=*
V437-WORKSTATION=SVL-GCHANDRAN, USERID=admf001,
      APPLICATION NAME=db2jcc_application
V445-G91E1ECF.G778.BFCA149050B1=19 ACCESSING DATA FOR
  ::FFFF:9.30.30.207
```

9.3.2 Propagation of Windows user IDs to DB2 using EIM

We successfully propagated the user IDs to DB2, as shown in the previous section. The implementation, however, demanded that we create RACF user IDs with names matching those of the Windows user IDs—not a very elegant solution.

In this section, we expand on the propagation of user IDs by using EIM to map the Windows users to actual authorization IDs on z/OS. This requires z/OS V1.8.

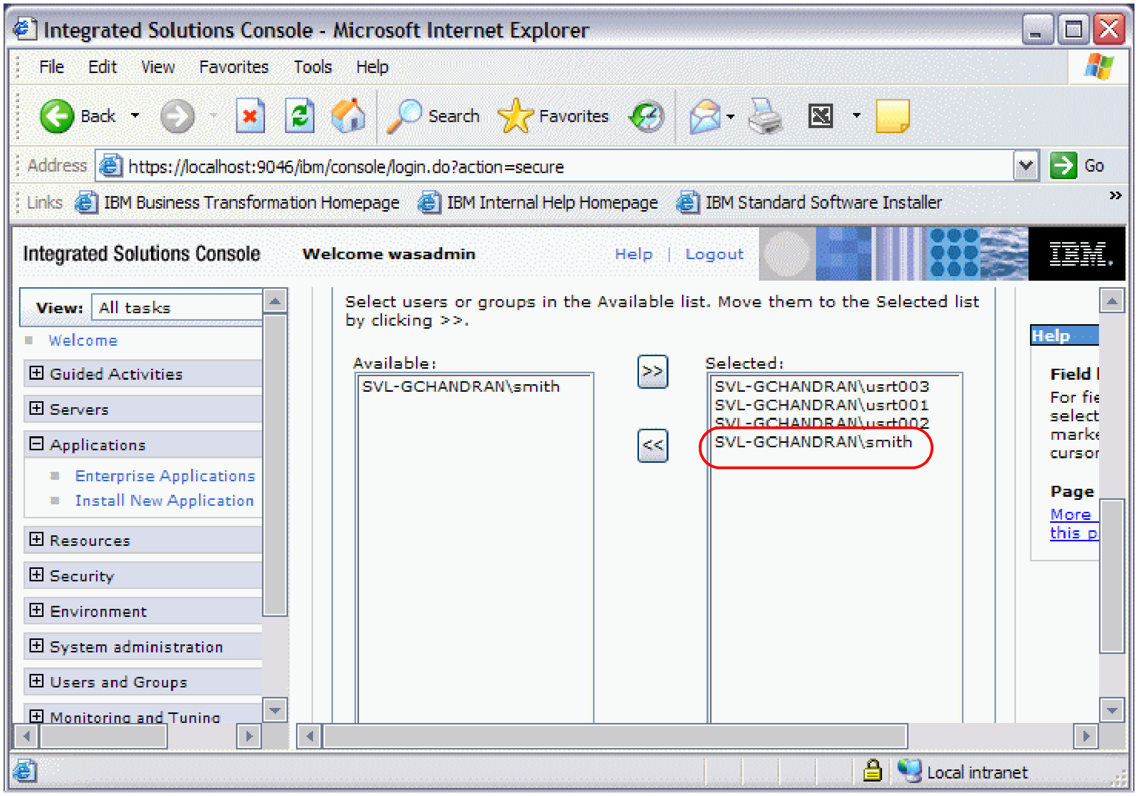
Identity mapping is necessary when the integration of servers (such as WebSphere Application Server and DB2) is needed, but the user registries are different and not shared between the systems.

We want to add a user SMITH who is not a RACF user ID and map him to a RACF user ADMF002. Then when we launch the Java servlet, we log on as SMITH, but the query is performed using the authorization ID of ADMF002.

First, we need to define the new user SMITH in the Windows registry.

Then, we add SMITH to the list of user IDs that are authorized to use the security role of eimsecurity (Figure 9-11).

To do this, from the Applications panel, select **Enterprise Applications** → **Test** (the name of our application) → **Security role to user/group mapping**.



Address: https://localhost:9046/ibm/console/login.do?action=secure

Select users or groups in the Available list. Move them to the Selected list by clicking >>.

Available:
SVL-GCHANDRAN\smith

Selected:
SVL-GCHANDRAN\usrt003
SVL-GCHANDRAN\usrt001
SVL-GCHANDRAN\usrt002
SVL-GCHANDRAN\smith

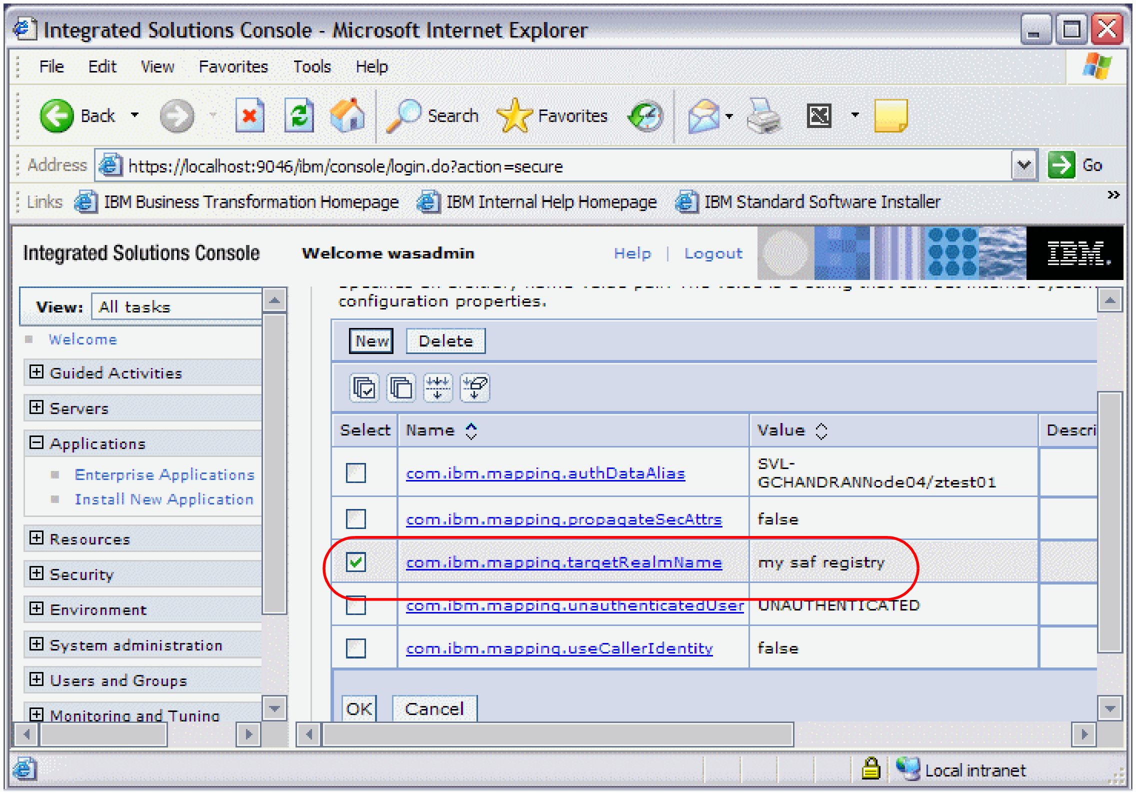
Figure 9-11 Adding another user

Next, we need WebSphere Application Server to request the mapping of user IDs to RACF users before validating these in RACF. The mapping takes place on z/OS by EIM and is initiated by RACF when requested to do so by WebSphere Application Server.

We supply a value for the parameter targetRealmName. If this parameter is blank or contains the host name of Windows system, a request for mapping is not made. If it does contain a value, mapping takes place.

To work with this parameter, from the Applications panel, select **Enterprise Applications** → **Test** (our own application) → **Resource reference** → **Mapping Properties**.

We fill in the value `my saf registry` (Figure 9-12).



Address: https://localhost:9046/ibm/console/login.do?action=secure

configuration properties.

Select	Name	Value	Descri
☐	com.ibm.mapping.authDataAlias	SVL-GCHANDRANNode04/ztest01	
☐	com.ibm.mapping.propagateSecAttrs	false	
☑	com.ibm.mapping.targetRealmName	my saf registry	
☐	com.ibm.mapping.unauthenticatedUser	UNAUTHENTICATED	
☐	com.ibm.mapping.useCallerIdentity	false	

Figure 9-12 Enabling mapping of users

After setting up the necessary things on WebSphere Application Server, we now turn to z/OS. For DB2 for z/OS to enable identity mapping, we need to set up the EIM, which is the component performing the mapping.

Setting up EIM on z/OS is a complex procedure and consists of the following tasks:

1. Set up the LDAP server with a TDBM back end.
2. Run the LDAP configuration utility.
3. Set up RACF for the LDAP server.
4. Set up the EIM domain.

For a detailed description of the tasks required, refer to Appendix C, “Enterprise Identity Mapping” on page 327.

After enabling EIM, we need to set up the user mapping. This is done in the EIM registry under UNIX System Services.

We add entries tying together SMITH and ADMF002, as shown in Example 9-3. There is a source entry and a target entry, and the two entries are tied by the WAS parameter of the -i option (identifier). The -u option (user) and the -r option (realm) identify the correct entry on the source side. The -u option on the target side then denotes the mapped user ID.

Example 9-3 EIM registry

```
eimadmin -aA -u "SMITH" -r "SVL-GCHANDRAN" -t SOURCE -i "WAS" \
-d 'ibm-eimdomainname=mydomain,o=IBM,c=US' \
-h ldap://9.30.114.83:3389 -b 'cn=LDAP Administrator' -w password

eimadmin -aA -u "ADMF002" -r "my saf registry" -t TARGET -i "WAS" \
-d 'ibm-eimdomainname=mydomain,o=IBM,c=US' \
-h ldap://9.30.114.83:3389 -b 'cn=LDAP Administrator' -w password
-o "db2/stlec2/9.30.114.83"
```

Finally, we change our trusted context definition so that it allows use of the trusted connection by ADMF002, the authorization ID that SMITH has been mapped into by EIM. This can be done with a simple ALTER command:

```
ALTER TRUSTED CONTEXT WASCTXT1
ADD USE FOR ADMF002;
```

Now, when we launch the Java servlet, logging on with Windows user ID SMITH, we expect the same result, only now ADMF002 is the authorization ID associated with the query.

The steps in the authentication conversation are as follows:

1. The user logs on with user ID SMITH.
2. WebSphere Application Server authenticates user ID and password.
3. Java security requests a connection to DB2 with the user ID ADMF001 and password.
4. DB2 calls in RACF to authenticate the user ID and password. RACF also assigns ADMF001 as the primary authorization ID and as the CURRENT SQLID. Secondary authorization IDs can also be assigned.
5. DB2 checks that the primary authorization ID ADMF001 is the system authorization ID of the trusted context and the IP address 9.30.30.207 of the incoming request matches the “address” connection trust attribute of the trusted context and then the connection is established as trusted.

6. The application now issues a **switch user** request using SMITH. Note that this is not done explicitly by the Java servlet, but rather implicitly by WebSphere Application Server.
7. Before authenticating the user ID, RACF recognizes that a mapping of user IDs must take place and calls EIM with the user ID SMITH.
8. EIM translates SMITH to ADMF002.
9. The user ID ADMF002 is authenticated by RACF, and ADMF002 is assigned as the primary authorization ID and as the CURRENT SQLID. Secondary authorization IDs can also be assigned.
10. DB2 checks that the primary authorization ID ADMF002 is allowed to use the connection, and the connection is initialized.

We see the result in the DISPLAY THREAD in Example 9-4. Again, we see a thread in trusted context named WASCTXT1 with the system authorization ID ADMF001, but now the authorization ID of the thread is ADMF002.

Example 9-4 Display thread with mapped user in trusted context

```
)DIS THD(*)
DSNV401I  ) DISPLAY THREAD REPORT FOLLOWS -
DSNV402I  ) ACTIVE THREADS - 751
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA *     3 db2jcc_appli ADMF002  DISTSERV 0044    12
V485-TRUSTED CONTEXT=WASCTXT1,
     SYSTEM AUTHID=ADMF001,
     ROLE=*
 V437-WORKSTATION=SVL-GCHANDRAN, USERID=admf001,
      APPLICATION NAME=db2jcc_application
 V445-G91E1ECF.G74F.BFCA0E10A892=12 ACCESSING DATA FOR
  ::FFFF:9.30.30.207
```

9.3.3 Limiting privileges in DB2 to a single requester location

Up until now, we had to grant the privileges necessary to run the application to the individual authorization IDs running the application, either directly or through the use of secondary authorization IDs.

In both cases, these privileges are available regardless of how and from where the connection to DB2 is established. In other words, it is possible to log on to TSO with one of these authorization IDs and make use of the DB2 privileges from there, for example. Furthermore, the user is no longer restricted to the requests available from the application, but only by what privileges that user holds.

We want to prevent the WebSphere Application Server user IDs from exercising the privileges obtained for the use of the application, outside the application. Specifically, we want to make sure that the RACF user ADMF002 does not have any privileges in DB2, that is, that user is not able to issue SQL from anywhere except the Windows server and then only from within a trusted connection.

The Windows user SMITH obviously does not have any privileges in DB2; the user SMITH does not even exist in z/OS, and the system authorization ID ADMF001 need not have any privileges in DB2 either. Therefore, if we can clean up the privileges held by ADMF002, we have come a long way with regards to securing our DB2 data from unauthorized or unwanted access.

Therefore, we turn to the use of roles within the trusted context. We want to assign the user ID ADMF002 a role that can be used in the trusted context only, and which can hold the privileges necessary to run the application.

The creation of a trusted context with roles for the use of WebSphere Application Server involves no additional work on the middle layer.

In DB2, we need to:

1. Create the role and grant the necessary privileges to that role.
2. Alter the trusted context definition to supply a role for the switched user.
3. Revoke the privileges from the switched user.

We show the needed DDL in Example 9-5. We create a role RPAYROLL and grant it the necessary privileges. We then alter the trusted context to enable use of this role for users of the connection. Finally, we can revoke the privileges held by ADMF002.

Example 9-5 DDL for using a role

```
CREATE ROLE RPAYROLL;
GRANT SELECT ON TABLE PROD.EMP TO RPAYROLL;
ALTER TRUSTED CONTEXT WASCTXT1
ALTER DEFAULT ROLE RPAYROLL;
REVOKE SELECT ON TABLE PROD.EMP FROM ADMF002;
```

In the previous example, we made a default role for the whole trusted context. We could have chosen to define different roles for different users in order to tailor the privileges according to need.

One obvious such distinction is for applications where the management of database objects in DB2 is done from the application server, for example, creating new tables. The privileges needed for these operations must be separated from the privileges needed to use the application itself.

9.3.4 Improving encryption with SSL

The final security enhancement is of a different nature from the others. Up until now, we concentrated on authorized access to the system, how we limit it to what is necessary, and how we enable ourselves to monitor it.

We now turn to the encryption of data flowing between the application server and the database server to prevent unauthorized interception of data.

With DB2 V9 for z/OS, Secure Sockets Layer (SSL) encryption of data has been enabled, improving upon the existing techniques used by DRDA for encrypting data. SSL is a well-known protocol, widely in use to secure communications over TCP/IP sockets.

SSL works by allowing the client (in this case, WebSphere) and server (in this case, DB2) to send each other encrypted data that only they can decrypt. This is accomplished when the connection between the client and server is established. In a sequence of events generally known as the handshake, the server presents its public certificate for the client to accept or deny. If the certificate is accepted, the client and server agree on a hash for the duration of their conversation, which is used to encrypt the data.

Enabling SSL in DB2 for z/OS

The z/OS V1.7 Communications Server for TCP/IP introduces a service that implements the use of the SSL protocol on behalf of applications on z/OS, in this case, DB2. This service is called the Application Transparent Transport Layer Service (AT-TLS).

To enable SSL encryption on DB2 for z/OS, we perform the tasks described here.

We need to configure AT-TLS. This encompasses the following steps:

1. Enabling AT-TLS in the TCPIP profile.
2. Protecting the TCP/IP stack.
3. Configuring the policy agent. HandShakeRole must be set to `Server` or `ServerWithClientAuth`.

For further information about AT-TLS, refer to *z/OS Communications Server: IP Configuration Guide*, SC31-8775.

Then, define a secure port for TCP/IP connections to DB2. To do this, either specify the DRDA SECURE PORT field in the DDF panel 2 during DB2 installation, or update the SECPORT parameter of the DDF statement in the BSDS with DSNJU003.

Finally on the z/OS side, we have to generate a certificate to be used when authenticating the connection. To establish an SSL encrypted connection, a secure sockets layer handshake authenticates the server and the client and establishes an encryption method and a unique session key.

The certificate is generated using the following RACDCERT commands. First, a self-signed certificate is created:

```
RACDCERT CERTAUTH -
         GENCERT  -
         SUBJECTSDN(OU('SVL410 Server CA') -
                    O('IBM') -
                    L('SVL') -
                    SP('CA') -
                    C('USA')) -
         NOTAFTER(DATE(2030-12-31)) -
         WITHLABEL('SVL410 Server CA') -
         KEYUSAGE(CERTSIGN)
```

Then the certificate is associated with a server ID and signed:

```
RACDCERT ID(SYSDSP) -
         GENCERT    -
         SUBJECTSDN(CN('v14ec083.svl.ibm.com') -
                    OU('UTEC620') -
                    O('SVL410') -
                    C('USA')) -
         NOTAFTER(DATE(2030-12-31)) -
         WITHLABEL('SVL410 Server Certificate') -
         SIGNWITH(CERTAUTH LABEL('SVL410 Server CA'))
```

A key ring for holding certificates is created:

```
RACDCERT ID(SYSDSP) ADDRING(SVL410KEYRING)
```

The certificate is added to the key ring:

```
RACDCERT ID(SYSDSP) -
         CONNECT(CERTAUTH -
         LABEL('SVL410 Server CA') RING(SVL410KEYRING))
```

The certificate is made the default certificate:

```
RACDCERT ID(SYSDSP) -
         CONNECT(ID(SYSDSP) -
         LABEL('SVL410 Server Certificate') -
         RING(SVL410KEYRING) DEFAULT)
```

The certificate is exported to a sequential file called SYSADM.SVL410.CACERT for download to the application server:

```
RACDCERT CERTAUTH -
         EXPORT(LABEL('SVL410 Server CA')) -
         DSN('SYSADM.SVL410.CACERT')
```

The certificate looks like something like that shown in Example 9-6.

Example 9-6 Certificate

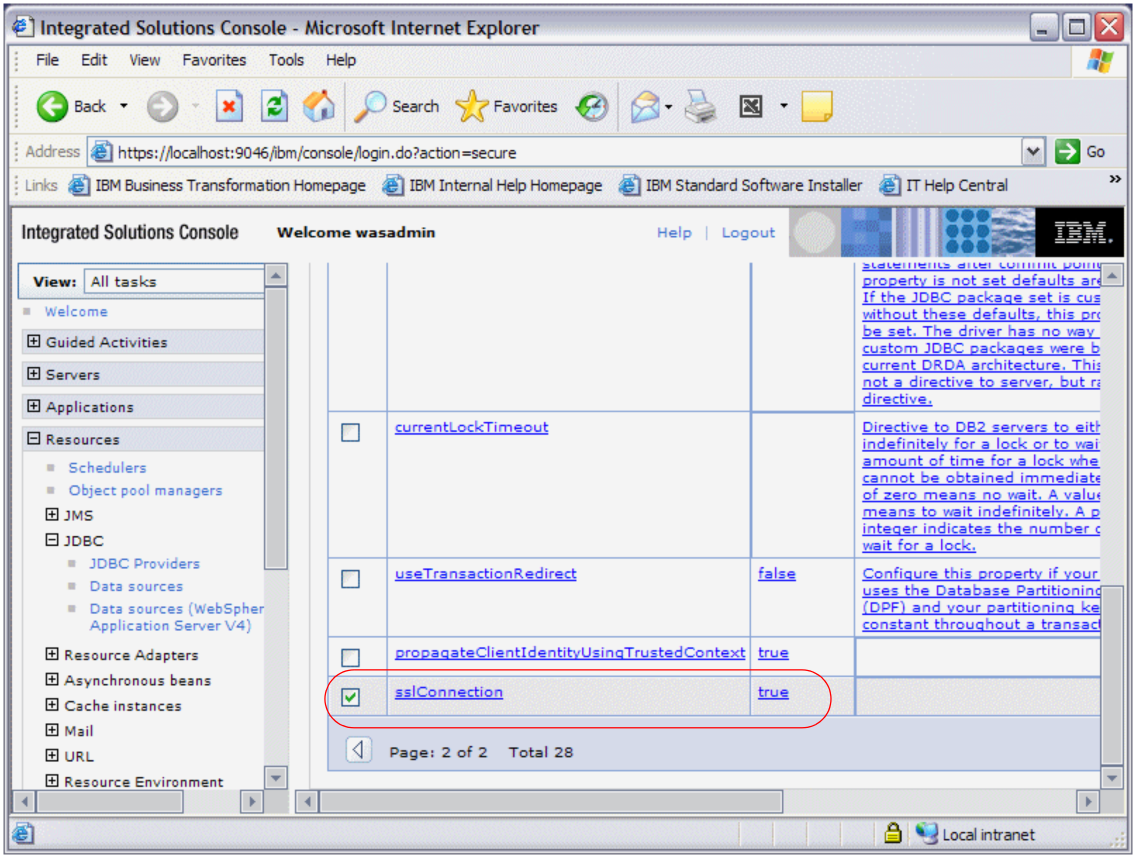
```
-----BEGIN CERTIFICATE-----
MIICnzCCAgigAwIBAgIBADANBgkqhkiG9w0BAQUFADBSMQwwCgYDVQQGEwNVU0Ex
CzAJBgNVBAgTAkNBMQwwCgYDVQQHEwNTVkwxDDAKBgNVBAoTA0lCTTEZMBcGA1UE
CxMQU1ZMNDEwIFNlcnZlciBDQTAeFw0wNjEyMjcwODAwMDBaFw0zMTAxMDEwNzU5
NTlaMFIxDDAKBgNVBAYTA1VTQTELMAkGA1UECBMCQ0ExDDAKBgNVBAcTA1NWTDEM
MAoGA1UEChMDSUJNMRkwFwYDVQQLExBTVkw0MTAgU2VydmVyIENBMIGfMA0GCSqG
SIb3DQEBAQUAA4GNADCBiQKBgQC8ArWfxo6nGoXFYuBwHBtLk8bobNFEw2y+B3Mb
TvFcHFwQ5wjeFJhBF9s32kPdOu2eEnlv3GopELYTC0FoB186niRld3R6D4JbslIb
FkgvirbsbMkoTrhypBk/584WRz2EvbPmYXZidd8bB5MfHOJI3vDwwhEFUeijT3XY
Iz4jCQIDAQABo4GEMIGBMD8GCWCGSAGG+EIBDQQyEzBHZW5lcmF0ZWQgYnkgdGhl
IFNlY3VyaXR5IFNlcnZlciBmb3Igei9PUyAoUkFDRikwDgYDVR0PAQH/BAQDAgEG
MA8GA1UdEwEB/wQFMAMBAf8wHQYDVR0OBBYEFPOOIODK8fc45XJz5sbuIdr/fCu9
MA0GCSqGSIb3DQEBBQUAA4GBACh46soRid0Yp+RtELr9qJ7UTeiTkeNMrJv2RC+/
i9+e4/zRPUXSFv6ShGqzfZW5h7UBRAuDbbwAmJ5KtgyYELLjPtCIFfaUjQ4AnOYy
Ohy/x4AXxBTmCeSgzIAmd+TviTq/OOxvqo3sRNNXERFzCJbM6MZ7GJgULxQJl4oo
R88v
-----END CERTIFICATE-----
```

This concludes the work needed on z/OS.

Enabling SSL in WebSphere

To implement SSL encryption on the application server, there are three key things that need to be done.

First, we need to set the Java connection property sslConnection. From the Resources panel, select **JDBC** → **Data Sources** → **testeimdatasource** (the name of our data source) → **Custom properties**, as shown in Figure 9-13.

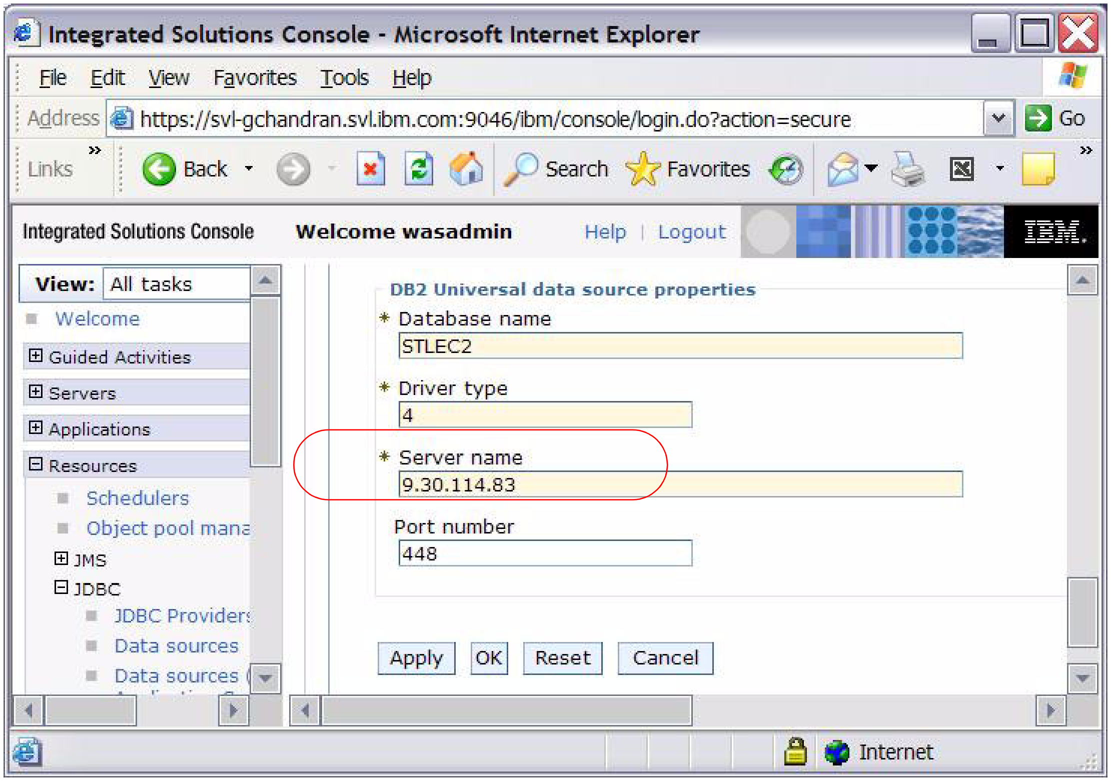


			statements after commit point property is not set defaults are If the JDBC package set is cus without these defaults, this pro be set. The driver has no way custom JDBC packages were b current DRDA architecture. This not a directive to server, but ra directive.
□	currentLockTimeout		Directive to DB2 servers to eith indefinitely for a lock or to wai amount of time for a lock whe cannot be obtained immediate of zero means no wait. A value means to wait indefinitely. A p integer indicates the number c wait for a lock.
□	useTransactionRedirect	false	Configure this property if your uses the Database Partitioning (DPF) and your partitioning ke constant throughout a transact
□	propagateClientIdentityUsingTrustedContext	true	
☑	sslConnection	true	

Page: 2 of 2 Total 28

Figure 9-13 Enabling SSL encryption

Next, we set up the data source properties to use the secure port for DB2 for z/OS. From the Resources panel, we select **JDBC** → **Data Sources** → **testeimdatasource** (the name of our data source) and change the port number from 446 to 448 (Figure 9-14).



DB2 Universal data source properties

* Database name: STLEC2

* Driver type: 4

* Server name: 9.30.114.83

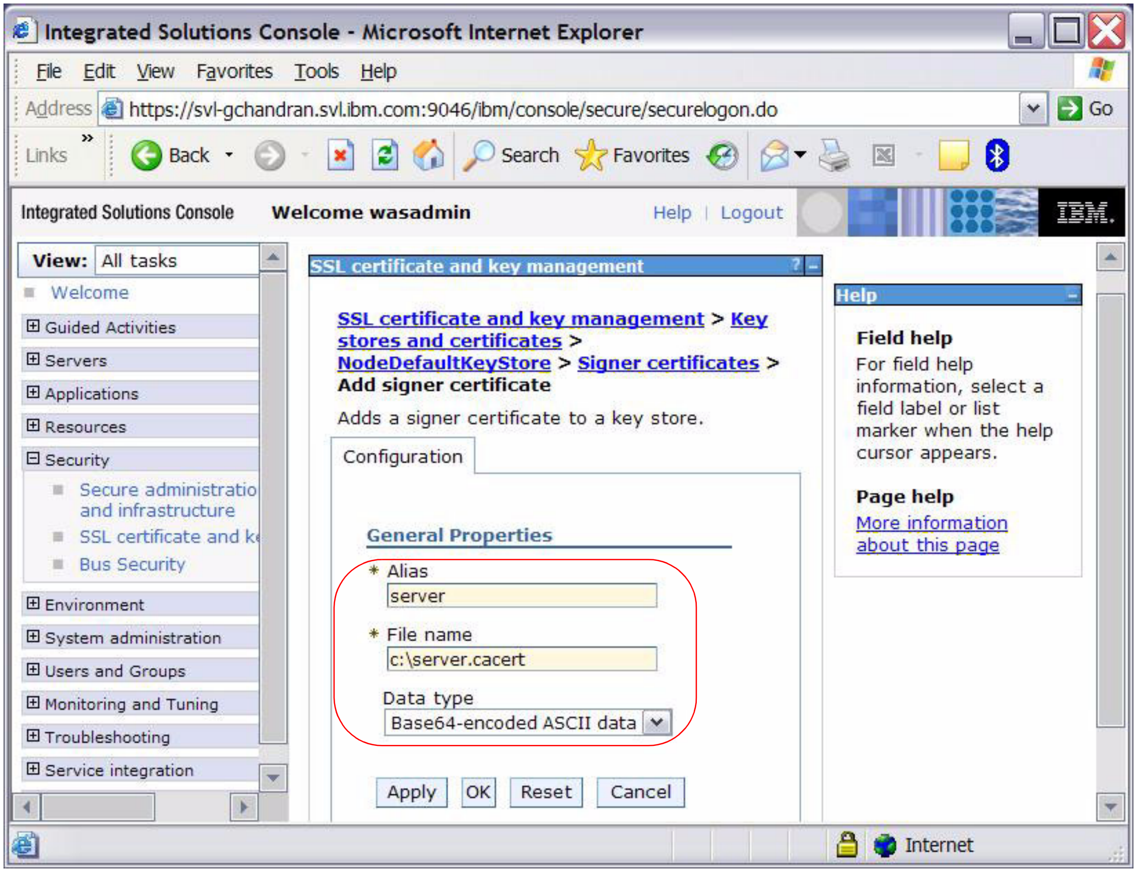
Port number: 448

Figure 9-14 Setting the secure port

Finally, we need to add the server certificate generated on the z/OS side to a key store, the equivalent of a key ring on z/OS, available to our application. The certificate that we exported to the sequential file SYSADM.SVL410.CACERT needs to be copied to our Windows server, using FTP, for example.

We add the certificate to a key store known by WebSphere Application Server. The default key store provided by WebSphere is sufficient, so all we have to do is to add the certificate.

Go to the Security panel. We select **SSL certificate and key management** → **Key stores and certificates**. We choose the default key store **NodeDefaultKeyStore** → **Signer certificates** → **Add signer certificate**. We then supply the file name and path for the certificate we copied down to the Windows server, as shown in Figure 9-15.



Integrated Solutions Console - Microsoft Internet Explorer

Address https://svl-gchandran.svl.ibm.com:9046/ibm/console/secure/securelogon.do

Integrated Solutions Console   Welcome wasadmin

SSL certificate and key management > Key stores and certificates > NodeDefaultKeyStore > Signer certificates > Add signer certificate

Adds a signer certificate to a key store.

Configuration

General Properties

* Alias: server

* File name: c:\server.cacert

Data type: Base64-encoded ASCII data

Figure 9-15 Adding certificate to the key store

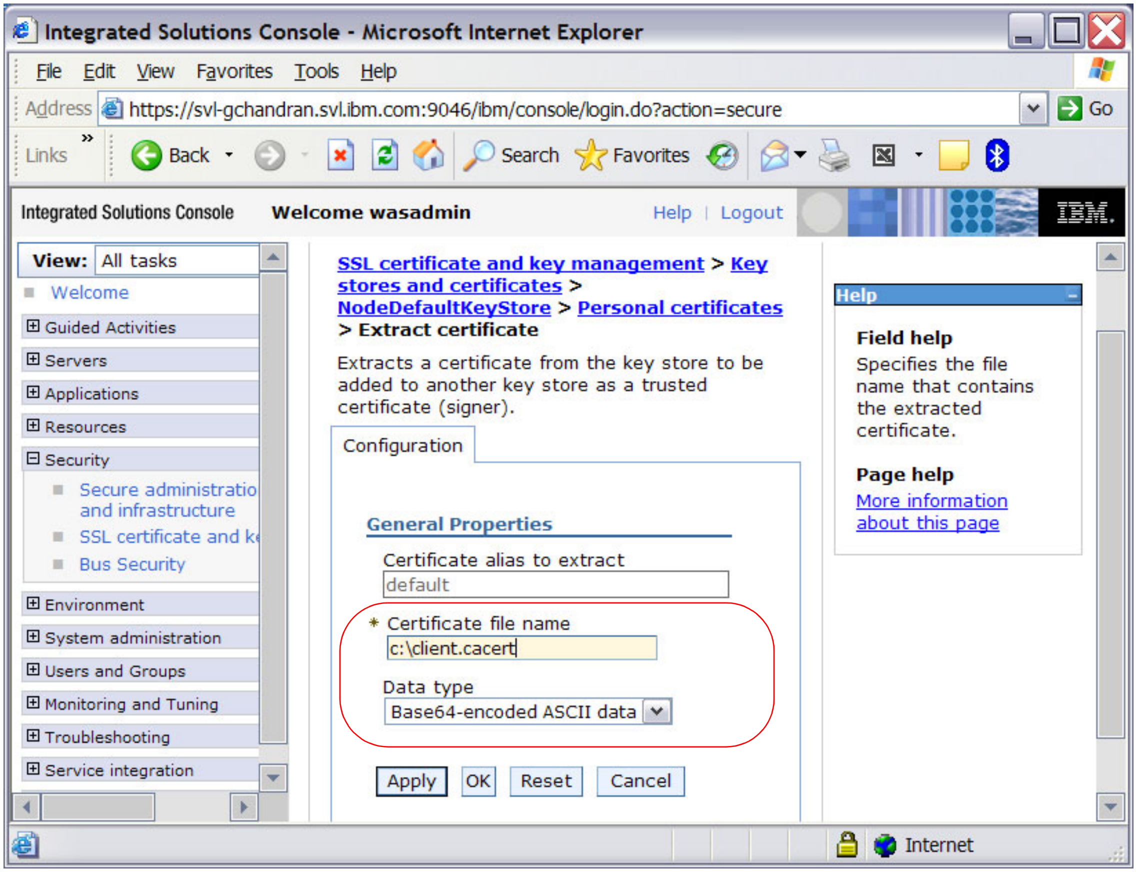
If the type of SSL encryption we chose for our connection is *server encryption*, we have now completed the setup. There is, however, the option of tightening security even more by using *server encryption with client authentication*, which requires the client to authenticate as a part of the handshake before setting an encryption key for the session.

To enable this encryption type, we set the HandShakeRole to `ServerWithClientAuth` in AT-TLS, as mentioned in the beginning of the chapter.

We then create a client certificate on the Windows server and import it to the DB2 for z/OS server key ring.

For the client certificate, we choose to use the default certificate provided by WebSphere Application Server in the default key store. To import it to z/OS, we first need to extract it from the key store.

From the Security panel, select **SSL certificate and key management** → **NodeDefaultKeyStore** → **Personal certificates** → **Extract certificates**. We extract the certificate in **Base64-encoded ASCII data**, supplying the path and file name of `c:\client.cacert`, as shown in Figure 9-16.


Integrated Solutions Console - Microsoft Internet Explorer

Address: https://svl-gchandran.svl.ibm.com:9046/ibm/console/login.do?action=secure

Integrated Solutions Console Welcome wasadmin

SSL certificate and key management > Key stores and certificates > NodeDefaultKeyStore > Personal certificates > Extract certificate

Extracts a certificate from the key store to be added to another key store as a trusted certificate (signer).

Configuration

General Properties

Certificate alias to extract: default

* Certificate file name: c:\client.cacert

Data type: Base64-encoded ASCII data

Help

Field help

Specifies the file name that contains the extracted certificate.

Page help

More information about this page

Figure 9-16 Extracting the client certificate

We then FTP the file containing the certificate to z/OS in ASCII format. We define the certificate to the key ring called SVL410KEYRING that is created at the server by issuing the following RACDCERT commands:

```
RACDCERT ID(SYSDSP) -
         ADD('USRT001.CLIENT.CACERT') TRUST -
         WITHLABEL('WebSphere Client Certificate')
RACDCERT ID(SYSDSP) -
         CONNECT(ID(SYSDSP) -
           LABEL('WebSphere Client Certificate') -
           RING(SVL410KEYRING))
```

We have now enabled SSL encryption with client authentication for connections between WebSphere Application Server and DB2 V9 for z/OS. As a final security measure, we want to make sure that any connection made using the trusted context uses SSL encryption. To do this, set the ENCRYPTION attribute to HIGH, as shown in the following alter statement:

```
ALTER TRUSTED CONTEXT WASCTXT1
ALTER ENCRYPTION HIGH;
```

This means that the minimum level of encryption for the connection is HIGH, corresponding to 128-bit SSL encryption.

Before launching the Java servlet, we ensure that DB2 has been set up to accept connections on a secure port by issuing a DISPLAY DDF command. Example 9-7 shows the output. We see that the secure port for TCP/IP connections is 448.

Example 9-7 Display DDF with secure port

```
)DIS DDF
DSNL080I  ) DSNLTDDF DISPLAY DDF REPORT FOLLOWS: 870
DSNL081I STATUS=STARTD
DSNL082I LOCATION           LUNAME            GENERICLU
DSNL083I STLEC2             -NONE             -NONE
DSNL084I TCPPORT=446   SECPORT=448   RESPORT=5001  IPNAME=STLEC2
DSNL085I IPADDR=::9.30.114.83
DSNL086I SQL    DOMAIN=v14ec083.svl.ibm.com
DSNL086I RESYNC DOMAIN=v14ec083.svl.ibm.com
DSNL099I DSNLTDDF DISPLAY DDF REPORT COMPLETE
```

After the servlet is launched, we can view the effect of the encryption setup on the z/OS console, as shown in Example 9-8. The messages show the handshake between the client and server.

Example 9-8   Handshake establishing SSL encryption

```
BPXF024I (OMVSKERN) Dec 11 22:15:39 TTLS 50331666 : 14:15:39 TCPIP 755
EZD1281I TTLS Map   CONNID: 0000013F LOCAL: ::FFFF:9.30.114.83..448
REMOTE: ::FFFF:9.30.30.165..1456 JOBNAME: V91ADIST USERID: SYSDSP
TYPE: InBound STATUS: Enabled RULE: V91ASecureServer ACTIONS:
V91ASecureGrpAct V91ASecureEnvAct **N/A**

BPXF024I (OMVSKERN) Dec 11 22:15:39 TTLS 50331666 : 14:15:39 TCPIP 756
EZD1283I TTLS Event GRPID: 00000002 ENVID: 00000000 CONNID: 0000013F
RC:    0 Connection Init

BPXF024I (OMVSKERN) Dec 11 22:15:39 TTLS 50331666 : 14:15:39 TCPIP 757
EZD1282I TTLS Start GRPID: 00000002 ENVID: 00000003 CONNID: 00000000
Environment Create ACTIONS: V91ASecureGrpAct V91ASecureEnvAct **N/A**

BPXF024I (OMVSKERN) Dec 11 22:15:39 TTLS 50331666 : 14:15:39 TCPIP 758
EZD1283I TTLS Event GRPID: 00000002 ENVID: 00000004 CONNID: 00000000
RC:    0 Environment Master Create 00000003

BPXF024I (OMVSKERN) Dec 11 22:15:39 TTLS 50331666 : 14:15:39 TCPIP 759
EZD1283I TTLS Event GRPID: 00000002 ENVID: 00000004 CONNID: 00000000
RC:    0 Environment Master Init 7EC6DB98

BPXF024I (OMVSKERN) Dec 11 22:15:39 TTLS 50331666 : 14:15:39 TCPIP 760
EZD1283I TTLS Event GRPID: 00000002 ENVID: 00000003 CONNID: 00000000
RC:    0 Environment Link 7EC6DB98 00000004

BPXF024I (OMVSKERN) Dec 11 22:15:39 TTLS 50331666 : 14:15:39 TCPIP 761
EZD1282I TTLS Start GRPID: 00000002 ENVID: 00000003 CONNID: 0000013F
Initial Handshake ACTIONS: V91ASecureGrpAct V91ASecureEnvAct **N/A**
HS-ServerWithClientAuth

BPXF024I (OMVSKERN) Dec 11 22:15:40 TTLS 50331666 : 14:15:39 TCPIP 762
EZD1283I TTLS Event GRPID: 00000002 ENVID: 00000003 CONNID: 0000013F
RC:    0 Initial Handshake 7EC0D918 7EC6DB98 TLSV1 05
```

9.3.5 A record trace

To conclude the implementation, let us briefly discuss the auditing possibilities now available for the application.

There are a number of trace records that have been modified to include information about the use of roles and trusted contexts. Also, a couple of new ones have been introduced. Among these are:

- IFCID 169, establishing a trusted connection from a requesting DB2 site to a server
- IFCID 269, establishing or reusing a trusted context
- IFCID 270, creating or altering a trusted context

We use our WebSphere Application Server application and the trusted context created earlier, which enables the use of the context for three users: USRT001, USRT002 and USRT003. We initiate the connection with system authorization ID ADMF001, switch to USERT001, perform a SELECT statement, switch to USERT002, perform an UPDATE and then a SELECT statement, and then terminate the connection.

Alongside the execution of this Java servlet, we run the following trace:

```
-STA TRACE (AUDIT) IFCID(83,143,144,145,269) DEST(GTF)
```

IFCID 83 is the record for IDENTIFY, IFCIDs 143 and 144 are FIRST READ and FIRST UPDATE, respectively, for audited tables, and IFCID 145 is DML STATEMENT.

We run the Java servlet, stop the trace, and format the output. This is done using IBM Tivoli® OMEGAMON® XE for DB2 Performance Expert/Monitor on z/OS V4.1.0, which added support for the new V9 IFCIDs and changes within existing IFCIDs. We format a simple record trace with the following SYSIN:

```
RECTRACE
LEVEL(LONG)
EXEC
```

In the trace, we first see the IDENTIFY with authid ADMF001 which is the SYSTEM AUTHID of the trusted context, and then the 269 record, which shows that a trusted context is being used. The connection type at this point is *established*.

We then see a new IDENTIFY with authid USRT001, which is the switched user, and another 269 record showing that a trusted context is used. This time the connection type is *reused*, that is, this is a switch of user ID. Note that ADMF001 is still associated with the connection.

Next, two records relate to the SQL performed. We see that DML has been performed with the authid of USRT001, and that it is a first read of an audited table.

There is another switch of users, as shown by records IDENTIFY and TRUSTED CONTEXT. Now, we connect to USRT002.

Finally, we see that USRT002 performs a first write and a first read of an audited table.

Example 9-9 USRT002 performing a first write and a first read of an audited table

```
1   LOCATION: STLEC2                           OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V4)                  PAGE: 1-1
       GROUP: N/P                                    RECORD TRACE - LONG                        REQUESTED FROM: NOT SPECIFIED
      MEMBER: N/P                                                                                           TO: NOT SPECIFIED
   SUBSYSTEM: V91A                                                                                 ACTUAL FROM: 12/12/06 18:13:09.3
 DB2 VERSION: V9                                                                                     PAGE DATE: 12/12/06
OPRIMAUTH CONNECT   INSTANCE      END_USER       WS_NAME                       TRANSACT
 ORIGAUTH CORRNAME  CONNTYPE      RECORD TIME    DESTNO ACE IFC  DESCRIPTION   DATA
 PLANNAME CORRNMBR                TCB CPU TIME              ID
 -------- -------- ----------- ----------------- ------ --- --- ------------- -------------------------------------------------------
 ADMF001  V91A     BFD833CC2075 'BLANK'          'BLANK'                       'BLANK'
 ADMF001  028.DBAA 'BLANK'     18:14:06.20115584    217   1  83 IDENTIFY   <-- NETWORKID:  G91E1ECF  LUNAME:  G42C      LUWSEQ:
 'BLANK'    02                 N/P                               END           REQUESTING LOCATION:  ::FFFF:9.30.30.2
                                                                               REQUESTING TIMESTAMP: N/P
                                                                               AR NAME: SVL-GCHANDRAN        PRDID: JCC V3 R1 MO
                                                                               RECOPT: 'BLANK' ACCESS:  SUCCESSFUL
                                                                               CURR SQLID : ADMF001
                                                                               ORIG AUTHID: ADMF001
                                                                               SECONDARY AUTHORIZATION IDS: N/P
                                                                               ACEE UTOKEN :&
                                                                                         ÌÌÌÌÌÌÌÌ ob{bÖÖÖÌÌÌÌÌÌÌÌmjalvvuÖººuÖÖÖÖ
                                                                               QW0083RT           0     QW0083RS           0
                                                                               QW0083CT  X'C4C9E2E340404040'

 ADMF001  SERVER   BFD833CC2075 admf001          SVL-GCHANDRAN                 db2jcc_application
 ADMF001  db2jcc_a DRDA        18:14:06.51319331    218   1 269 TRUSTED        NETWORKID:  G91E1ECF  LUNAME:  G42C      LUWSEQ:
 DISTSERV ppli                 N/P                              CONTEXT TRACE  REQUESTING LOCATION:  ::FFFF:9.30.30.2
                                                                               REQUESTING TIMESTAMP: N/P
                                                                               AR NAME: SVL-GCHANDRAN        PRDID: JCC V3 R1 MO
          !-----------------------------------------------------------------------------------------------------------------------
          !CONNECTION TYPE: ESTABLISHED     STATUS: SUCCESS   SQLCODE:          0   OBJECT OWNER: ROLE
          !SECURITY LABEL : N/P
          !-----------------------------------------------------------------------------------------------------------------------

 USRT001  SERVER   BFD833CC2075 admf001          SVL-GCHANDRAN                 db2jcc_application
 USRT001  db2jcc_a DRDA        18:14:06.64809717    222   1  83 IDENTIFY   <-- NETWORKID:  G91E1ECF  LUNAME:  G42C      LUWSEQ:
 DISTSERV ppli                 N/P                               END           REQUESTING LOCATION:  ::FFFF:9.30.30.2
                                                                               REQUESTING TIMESTAMP: N/P
                                                                               AR NAME: SVL-GCHANDRAN        PRDID: JCC V3 R1 MO
                                                                               RECOPT: 'BLANK' ACCESS:  SUCCESSFUL
                                                                               CURR SQLID : USRT001
                                                                               ORIG AUTHID: USRT001
                                                                               SECONDARY AUTHORIZATION IDS: N/P
                                                                               ACEE UTOKEN :&
                                                                                         ÌÌÌÌÌÌÌÌ ob{bÖÖÖÌÌÌÌÌÌÌÌ£º
                                                                               QW0083RT           0     QW0083RS           0
                                                                               QW0083CT  X'C4C9E2E340404040'
                               18:14:06.64977030    223   1 269 TRUSTED        db2jcc_application
                               N/P                              CONTEXT TRACE  NETWORKID:  G91E1ECF  LUNAME:  G42C      LUWSEQ:
                                                                               REQUESTING LOCATION:  ::FFFF:9.30.30.2
                                                                               REQUESTING TIMESTAMP: N/P
                                                                               AR NAME: SVL-GCHANDRAN        PRDID: JCC V3 R1 MO
          !-----------------------------------------------------------------------------------------------------------------------
          !CONNECTION TYPE: REUSED          STATUS: SUCCESS   SQLCODE:          0   OBJECT OWNER: ROLE
          !SECURITY LABEL : N/P
          !-----------------------------------------------------------------------------------------------------------------------

USRT001  SERVER   BFD833CC2075 admf001          SVL-GCHANDRAN                 db2jcc_application
 USRT001  db2jcc_a DRDA        18:14:08.32779111    243   1 145 AUDIT DML      NETWORKID:  G91E1ECF  LUNAME:  G42C      LUWSEQ:
```

```
 DISTSERV ppli                N/P                              STATEMENT     REQUESTING LOCATION:  ::FFFF:9.30.30.2
                                                                               REQUESTING TIMESTAMP: N/P
                                                                               AR NAME: SVL-GCHANDRAN        PRDID: JCC V3 R1 M0
                                                                               LOCATION NAME: STLEC2
                                                                               PKG COLLCT ID: NULLID
                                                                               PROGRAM NAME : SYSLN300
                                                                               TIME: X'5359534C564C3031'
                                                                               TYPE: SELECT - QUERY                 STMTÿ:
                                                                               HOST OPTIONS   X'0400000000000000'
                                                                               SQL TEXT: select empno, firstnme, lastname, workdept
                                                                                         from prod.emp
                                                                               DATABASE: 275                  TABLE OBID:
                                                                               ISOLATION: RS
                               18:14:08.33001118    244   1 144 AUDIT FIRST    db2jcc_application
                               N/P                              READ          NETWORKID:  G91E1ECF  LUNAME:  G42C      LUWSEQ:
                                                                               REQUESTING LOCATION:  ::FFFF:9.30.30.2
                                                                               REQUESTING TIMESTAMP: N/P
                                                                               AR NAME: SVL-GCHANDRAN        PRDID: JCC V3 R1 M0
                                                                               DATABASE: 275            LOGRBA: X'000000000000'
                                                                               PAGE SET: 2                      TABLE OBID:
1   LOCATION: STLEC2                           OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V4)                 PAGE: 1-13
       GROUP: N/P                                    RECORD TRACE - LONG                        REQUESTED FROM: NOT SPECIFIED
      MEMBER: N/P                                                                                           TO: NOT SPECIFIED
   SUBSYSTEM: V91A                                                                                 ACTUAL FROM: 12/12/06 18:13:09.3
 DB2 VERSION: V9                                                                                     PAGE DATE: 12/12/06
OPRIMAUTH CONNECT   INSTANCE      END_USER       WS_NAME                       TRANSACT
 ORIGAUTH CORRNAME  CONNTYPE      RECORD TIME    DESTNO ACE IFC  DESCRIPTION   DATA
 PLANNAME CORRNMBR                TCB CPU TIME              ID
 -------- -------- ----------- ----------------- ------ --- --- -------------- ------------------------------------------------------
 USRT002  SERVER   BFD833CC2075 admf001          SVL-GCHANDRAN                 db2jcc_application
 USRT002  db2jcc_a DRDA        18:14:35.11666047    258   1  83 IDENTIFY   <-- NETWORKID:  G91E1ECF  LUNAME:  G42C      LUWSEQ:
 DISTSERV ppli                N/P                              END           REQUESTING LOCATION:  ::FFFF:9.30.30.2
                                                                               REQUESTING TIMESTAMP: N/P
                                                                               AR NAME: SVL-GCHANDRAN        PRDID: JCC V3 R1 M0
                                                                               RECOPT: 'BLANK' ACCESS:  SUCCESSFUL
                                                                               CURR SQLID : USRT002
                                                                               ORIG AUTHID: USRT002
                                                                               SECONDARY AUTHORIZATION IDS: N/P
                                                                               ACEE UTOKEN :&
                                                                                            ÌÌÌÌÌÌÌÌ ob{bÖÖÖÌÌÌÌÌÌÌÌ£º
                                                                               QW0083RT           0     QW0083RS           0
                                                                               QW0083CT  X'C4C9E2E340404040'
                               18:14:35.11685111    259   1 269 TRUSTED        db2jcc_application
                               N/P                              CONTEXT TRACE  NETWORKID:  G91E1ECF  LUNAME:  G42C      LUWSEQ:
                                                                               REQUESTING LOCATION:  ::FFFF:9.30.30.2
                                                                               REQUESTING TIMESTAMP: N/P
                                                                               AR NAME: SVL-GCHANDRAN        PRDID: JCC V3 R1 M0
          !--------------------------------------------------------------------------------------------------------------------------
          !CONNECTION TYPE: REUSED          STATUS: SUCCESS   SQLCODE:          0   OBJECT OWNER: ROLE
          !SECURITY LABEL : N/P
          !--------------------------------------------------------------------------------------------------------------------------

 USRT002  SERVER   BFD833CC2075 admf001          SVL-GCHANDRAN                 db2jcc_application
 USRT002  db2jcc_a DRDA        18:14:35.54225611    267   1 145 AUDIT DML      NETWORKID:  G91E1ECF  LUNAME:  G42C      LUWSEQ:
 DISTSERV ppli                N/P                              STATEMENT     REQUESTING LOCATION:  ::FFFF:9.30.30.2
                                                                               REQUESTING TIMESTAMP: N/P
                                                                               AR NAME: SVL-GCHANDRAN        PRDID: JCC V3 R1 M0
                                                                               LOCATION NAME: STLEC2
                                                                               PKG COLLCT ID: NULLID
                                                                               PROGRAM NAME : SYSLN300
                                                                               TIME: X'5359534C564C3031'
                                                                               TYPE: UPDATE                         STMTÿ:
                                                                               HOST OPTIONS   X'0400000000000000'
                                                                               SQL TEXT: update prod.emp set workdept='F10' where
                                                                                         empno like '00001%'
                                                                               DATABASE: 275                  TABLE OBID:
                                                                               ISOLATION: RS
                               18:14:35.60239765    268   1 143 AUDIT FIRST    db2jcc_application
                               N/P                              WRITE         NETWORKID:  G91E1ECF  LUNAME:  G42C      LUWSEQ:
                                                                               REQUESTING LOCATION:  ::FFFF:9.30.30.2
                                                                               REQUESTING TIMESTAMP: N/P
                                                                               AR NAME: SVL-GCHANDRAN        PRDID: JCC V3 R1 M0
                                                                               DATABASE: 275            LOGRBA: X'000000000000'
                                                                               PAGE SET: 2                      TABLE OBID:

 USRT002  SERVER   BFD833CC2075 admf001          SVL-GCHANDRAN                 db2jcc_application
 USRT002  db2jcc_a DRDA        18:14:35.68349225    275   1 145 AUDIT DML      NETWORKID:  G91E1ECF  LUNAME:  G42C      LUWSEQ:
```

```
 DISTSERV ppli                 N/P                               STATEMENT     REQUESTING LOCATION:  ::FFFF:9.30.30.2
                                                                               REQUESTING TIMESTAMP: N/P
                                                                               AR NAME: SVL-GCHANDRAN        PRDID: JCC V3 R1 M0
                                                                               LOCATION NAME: STLEC2
                                                                               PKG COLLCT ID: NULLID
                                                                               PROGRAM NAME : SYSLN300
                                                                               TIME: X'5359534C564C3031'
                                                                               TYPE: SELECT - QUERY                 STMTý:
                                                                               HOST OPTIONS   X'0400000000000000'
                                                                               SQL TEXT: select empno, firstnme, lastname, workdept
                                                                                         from prod.emp
                                                                               DATABASE: 275                  TABLE OBID:
                                                                               ISOLATION: RS
                               18:14:35.68521034    276   1 144 AUDIT FIRST    db2jcc_application
                               N/P                               READ          NETWORKID:  G91E1ECF  LUNAME:  G42C      LUWSEQ:
                                                                               REQUESTING LOCATION:  ::FFFF:9.30.30.2
                                                                               REQUESTING TIMESTAMP: N/P
                                                                               AR NAME: SVL-GCHANDRAN        PRDID: JCC V3 R1 M0
                                                                               DATABASE: 275             LOGRBA: X'000000000000'
                                                                               PAGE SET: 2                       TABLE OBID:
ORECORD TRACE COMPLETE
```

10

RACF access control module

In this chapter, we describe several different scenarios where DB2 row-level support is used with the RACF access control module for DB2 in place.

10.1 z/OS environment

The scenarios that follow are a very simple create, insert, and select of a DB2 table with multilevel security implemented. We install the RACF access control module, use different profile definitions, and turn on and off selective multilevel security options. After reviewing these scenarios, you can see how the different multilevel security options and profile settings can affect the data that inserted in and selected out of a DB2 table.

Before proceeding with any scenarios, an understanding of the basic environment these scenarios will be running on is important.

We use a partial multilevel security environment as a base with only the SECLABEL class activated. Therefore, for the following scenarios, assume that any other multilevel security options are turned off, that is, the multilevel security options are set at SETR NOMLACTIVE and SETR NOMLS.

10.1.1 Security labels

An understanding of the SECLABELs used is, of course, important. As with the previous chapters, we run in a z/OS environment with the RACF database populated with the following, previously shown, SECLABEL:

```
L1N, L1A, L1B, L1C, L1AB
L2N, L2A, L2B, L2C, L2AB
L3N, L3A, L3B, L3C
L4N, L4A, L4B, L4C
```

As we have seen before, the Lx part of the SECLABEL is the security level of the SECLABEL, and the letters that follow are the different categories defined for the SECLABEL. If the defined category is N, no categories are defined to the SECLABEL. Therefore, SECLABEL L2AB is at security level 2, with categories A and B, while SECLABEL L4N is at security level 4, with no categories.

10.1.2 RACF access control module

As previously mentioned, we install the RACF access control module, but to have the exit work, when DB2 comes up, this module requires that at least one of the DB2 classes be active. To have the least amount of affect on our scenarios, we activate the DSNADM class, but do not define profiles in that class.

One thing to note is that for several of the following scenarios, different DB2 classes are turned on. For the RACF access control module to recognize the activation of these classes, you must stop and restart the DB2 subsystem;

otherwise, changes made in those classes will not be processed by the RACF access control module.

10.1.3 DB2

For the following scenarios, we use SPUFI to issue the SQL commands from general user DBUSER. So, all the necessary DB2 authority was provided to run SPUFI from the general user.

In addition, for these scenarios, we use the same create table and set of inserts SQL commands in each of the scenarios. We document these here and reference them from each of the scenarios.

We use a very basic create table job with the SECURITY LABEL column to define the DB2 table (Example 10-1). In addition, we define an employee ID and salary column as additional data to be saved in this table.

Example 10-1 Create table job

```
CREATE TABLE DBUSER.EMPLOYEE
   ( EMPID CHAR(10)  NOT NULL,
     SECLABEL CHAR(8) NOT NULL WITH DEFAULT AS SECURITY LABEL,
     SALARY DECIMAL(7,2)
    );
```

To populate the table with information, several inserts are done. Example 10-2 shows a very basic insert that adds 16 users. Each user has a unique SECLABEL, and the employee name will show which SECLABEL is to be associated with which user.

Example 10-2 Insert table job

```
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP SYSLOW', 'SYSLOW', 32000 );
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP SYSHGH', 'SYSHIGH', 99500 );

INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L1A ', 'L1A', 11100 );
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L1B ', 'L1B', 11200 );
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L1C ', 'L1C', 11300 );

INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L2A ', 'L2A', 22100 );
```

```
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L2B ', 'L2B', 22200 );
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L2C ', 'L2C', 22300 );

INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L3A ', 'L3A', 33100 );
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L3B ', 'L3B', 33200 );
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L3C ', 'L3C', 33300 );

INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L4A ', 'L4A', 44100 );
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L4B ', 'L4B', 44200 );
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L4C ', 'L4C', 44300 );

INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L1AB', 'L1AB', 99100 );
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L2AB', 'L2AB', 99200 );
```

10.2 Scenarios

We go through seven different scenarios. In each scenario, we turn on or off a different option, or define the RACF profiles with different SECLABEL information. Then, we do a basic create and insert, and the results, stored in the DB2 table, are displayed. In each scenario, we use the same basic set of commands so that you can see the different results based on the environment that has been set up.

The scenarios are:

1. SETR MLS not active.
2. SETR MLS active.
3. SETR MLS not active. RACF profile protection used without SECLABELs in profiles.
4. SETR MLS not active. RACF profile protection used with SECLABELs in profiles.
5. SETR MLS active. RACF profile protection used without SECLABELs in profiles.

6. SETR MLS and MLACTIVE active. RACF profile protection used.
7. SETR MLS active. RACF profile protection used with special SECLABELs in profiles.

10.2.1 Scenario 1: SETR MLS not active

In this scenario:

- No RACF profiles are defined for the DB2 classes.
- RACF access control module is active.
- The SECLABEL class is active, but not SETR MLS or MLACTIVE.
- The user DBUSER is logged on with a SECLABEL of L3A.

We do a basic create table (Example 10-1 on page 281) and insert to populate the table (Example 10-2 on page 281).

Now listing the created table, you only see the rows that the user's SECLABEL will dominate. In this case, L3A's SECLABEL dominates rows with SECLABEL L3A, L2A, L1A, and SYSLOW. Therefore, only 4 out of the 16 rows are displayed.

Example 10-3 User with SECLABEL L3A selecting data from table

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
      ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+---------+
EMPID          SALARY  SECLABEL
---------+---------+---------+---------+---------+---------+---------+
EMP L1A      11100.00  L1A
EMP L2A      22100.00  L2A
EMP L3A      33100.00  L3A
EMP SYSLOW   32000.00  SYSLOW
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+
```

Logging on again with a SECLABEL of SYSHIGH, and doing the same list, the same user is able to display all 16 rows (Example 10-4).

Example 10-4 User with SECLABEL SYSHIGH selecting data from table

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
      ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+
EMPID          SALARY  SECLABEL
---------+---------+---------+---------+---------+---------+
EMP L1A      11100.00  L1A
EMP L1AB     99100.00  L1AB
EMP L1B      11200.00  L1B
EMP L1C      11300.00  L1C
EMP L2A      22100.00  L2A
EMP L2AB     99200.00  L2AB
EMP L2B      22200.00  L2B
EMP L2C      22300.00  L2C
EMP L3A      33100.00  L3A
EMP L3B      33200.00  L3B
EMP L3C      33300.00  L3C
EMP L4A      44100.00  L4A
EMP L4B      44200.00  L4B
EMP L4C      44300.00  L4C
EMP SYSHGH   99500.00  SYSHIGH
EMP SYSLOW   32000.00  SYSLOW
DSNE610I NUMBER OF ROWS DISPLAYED IS 16
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
```

Note that the insert job in Example 10-2 on page 281 allowed all the rows to be added with whatever SECLABEL the user provided, though the user was only allowed to see the rows that he dominated.

Logging on again with a SECLABEL of L1AB, and doing the same list, the same user is able to display four rows (Example 10-5). Once again, the user only displays the rows that he dominates, though this set of displayed rows is different from the set of rows displayed when the user’s SECLABEL was L3A.

Example 10-5 User with SECLABEL L1AB selecting data from table

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
      ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+
EMPID          SALARY  SECLABEL
---------+---------+---------+---------+---------+---------+
EMP L1A      11100.00  L1A
EMP L1AB     99100.00  L1AB
EMP L1B      11200.00  L1B
EMP SYSLOW   32000.00  SYSLOW
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
```

10.2.2 Scenario 2: SETR MLS active

In this scenario, the SETR MLS option (that is, *no-write down*) is on to see how things work differently. In this scenario:

- No RACF profiles are defined for the DB2 classes.
- RACF access control module is active.
- The SECLABEL class is active, and SETR MLS is on, but not SETR MLACTIVE.
- The user DBUSER is logged on with a SECLABEL of L3A.

Once again, we do a basic create table (Example 10-1 on page 281, and Example 10-2 on page 281) and insert to populate the table.

Nothing looks different when creating or inserting the rows into the table, but when listing the data, things get interesting. The SECLABELs for the inserted rows all change to L3A, the SECLABEL of the user. Because all the inserted rows have the SECLABEL of L3A, the user (who issued the insert) can see all 16 rows he or she inserted.

Example 10-6 User listing data inserted by user with SECLABEL L3A and SETR MLS

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
       ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+
EMPID          SALARY  SECLABEL
---------+---------+---------+---------+---------+---------+
EMP L1A      11100.00  L3A
EMP L1AB     99100.00  L3A
EMP L1B      11200.00  L3A
EMP L1C      11300.00  L3A
EMP L2A      22100.00  L3A
EMP L2AB     99200.00  L3A
EMP L2B      22200.00  L3A
EMP L2C      22300.00  L3A
EMP L3A      33100.00  L3A
EMP L3B      33200.00  L3A
EMP L3C      33300.00  L3A
EMP L4A      44100.00  L3A
EMP L4B      44200.00  L3A
EMP L4C      44300.00  L3A
EMP SYSHGH   99500.00  L3A
EMP SYSLOW   32000.00  L3A
DSNE610I NUMBER OF ROWS DISPLAYED IS 16
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100

---------+---------+---------+---------+---------+---------+
```

Because SETR MLS is in effect, the issuing user cannot change the classification of the data that he or she is inserting; therefore, all the data this user inserts into the table will inherit the SECLABEL of the issuer, in this case, L3A.

10.2.3 Scenario 3: SETR MLS not active, RACF profile protection used

With the previous scenarios, we set a basic baseline. Even though the RACF access control module is in place and active, because no RACF profiles were defined in the DB2 classes, the exit did nothing in actual access protection, and deferred all its decisions to DB2. In the next set of scenarios, we define profiles in the DB2 classes, and the RACF access control module determines some of the security authorizations.

In this scenario:

- RACF profiles are defined for some of the DB2 classes, as documented here.
- RACF access control module is active.
- The SECLABEL class is active, but not SETR MLS or SETR MLACTIVE.
- The user DBUSER is logged on with a SECLABEL of L3A.

Keeping things simple, we create a few profiles in the MDSNTB and MDSNDB classes. From a RACF special user, the following commands are issued:

```
RDEFINE MDSNTB ** UACC(NONE)
RDEFINE MDSNDB ** UACC(NONE)
SETR CLASSACT(MDSNTB MDSNDB)
```

Because DB2 classes are being activated, a restart of the DB2 address space is needed so that the RACF access control module knows that these classes are active. As an additional precaution, a SETR RACLIST(MDSNTB MDSNDB) will be done to refresh any residual data in storage.

When trying to create a table, the command fails (Example 10-7). This failure is because the RACF profiles have been defined to prevent anyone from creating any tables.

Example 10-7 Unauthorized user attempting to create table

```
CREATE TABLE DBUSER.EMPLOYEE
   ( EMPID CHAR(10)  NOT NULL,
     SECLABEL CHAR(8) NOT NULL WITH DEFAULT AS SECURITY LABEL,
     SALARY DECIMAL(7,2)
    );
---------+---------+---------+---------+---------+---------+---------+---------
```

```
DSNT408I SQLCODE = -551, ERROR:  DBUSER DOES NOT HAVE THE PRIVILEGE TO PERFORM
         OPERATION CREATE TABLE ON OBJECT DSNDB04
DSNT418I SQLSTATE   = 42501 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXODD2 SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = 50  0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'00000032'  X'00000000'  X'00000000'  X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
---------+---------+---------+---------+---------+---------+---------+---------
```

The following ICH408I message is generated (Example 10-8).

Example 10-8   ICH408I message from unauthorized creation of table

```
ICH408I USER(DBUSER  ) GROUP(SYS1    ) NAME(####################)
  DSND.DSNDB04.CREATETAB CL(MDSNDB  )
  INSUFFICIENT ACCESS AUTHORITY
  FROM ** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
```

To permit the user DBUSER to take any database and table actions, the RACF profiles need to be updated. From the RACF special user, the following commands are issued to permit DBUSER to make any database and table actions:

```
PERMIT ** ID(DBUSER) CLASS(MDSNDB) ACCESS(UPDATE)
PERMIT ** ID(DBUSER) CLASS(MDSNTB) ACCESS(UPDATE)
SETR RACLIST(MDSNDB MDSNTB) REFRESH
```

Because the RACF access control module knows that the MDSNDB and MDSNTB classes are active, a restart of the DB2 address space is not required, but the SETR RACLIST REFRESH is needed to pick up these changes. Now that the DB2 user has been authorized to create tables in a database, the previous create command works fine.

DBUSER, as the owner of table, can insert rows into the table without any additional authority needed, even if RACF profiles denied him access to insert them. To have a user who is not the owner insert rows into the table, we use another DB2 user (RACFU01) to insert rows into the table.

The user RACFU01 logs on with a SECLABEL of L2C (a different SECLABEL from DBUSER). Once again, RACF fails the command to insert these rows (Example 10-9 on page 288). This fails because the RACF profiles have been defined to prevent anyone (but DBUSER) from inserting rows into any table, as can be seen in the failure messages (Example 10-10 on page 288).

Example 10-9 Unauthorized user attempting to insert rows into a table

```
INSERT INTO DBUSER.EMPLOYEE
      VALUES ( 'EMP SYSLOW', 'SYSLOW', 32000 ) ;
---------+---------+---------+---------+---------+---------+---------+---------
DSNT408I SQLCODE = -551, ERROR:  RACFU01 DOES NOT HAVE THE PRIVILEGE TO PERFORM
         OPERATION INSERT ON OBJECT DBUSER.EMPLOYEE
DSNT418I SQLSTATE   = 42501 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXOIN SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = -600  0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'FFFFFDA8'  X'00000000'  X'00000000'  X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
---------+---------+---------+---------+---------+---------+---------+---------
```

Example 10-10 ICH408I message from unauthorized insert into table

```
ICH408I USER(RACFU01 ) GROUP(SYS1    ) NAME(####################)
  DSND.DBUSER.EMPLOYEE.INSERT CL(MDSNTB  )
  INSUFFICIENT ACCESS AUTHORITY
  FROM ** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
```

To permit the user RACFU01 to take any table actions, the RACF profiles need to be updated. From the RACF special user, the following commands are issued to permit RACFU01 to make any table actions:

```
PERMIT ** ID(RACFU01) CLASS(MDSNTB) ACCESS(UPDATE)
SETR RACLIST(MDSNTB) REFRESH
```

Now that RACFU01 has been authorized to insert rows into a table, the insert commands work fine. Also, by giving this user this authority, he will also be allowed to select to view the rows of the table. Due to the fact that RACFU01 is logged on with SECLABEL L2C, he can only see 3 out of the 16 rows.

Example 10-11 User with SECLABEL L2C selecting data from table

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
       ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+
EMPID          SALARY  SECLABEL
---------+---------+---------+---------+---------+---------+
EMP L1C      11300.00  L1C
EMP L2C      22300.00  L2C
EMP SYSLOW   32000.00  SYSLOW
DSNE610I NUMBER OF ROWS DISPLAYED IS 3
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
```

Note that the RACF profiles do *not* have SECLABELs assigned to them. Even in this case, RACF allowed the table to be created and data to be inserted into the table.

10.2.4 Scenario 4: SETR MLS not active, RACF profile protection with SECLABELs in profiles

As was mentioned earlier, SECLABELs were not assigned to the RACF profiles that were used. In this scenario:

- RACF profiles are defined with SECLABELs for some of the DB2 classes, as documented here.
- RACF access control module is active.
- The SECLABEL class is active, but not SETR MLS or SETR MLACTIVE.
- The user DBUSER is logged on with a SECLABEL of L3A.
- The user RACFU01 is logged on with a SECLABEL of L2C.

In this scenario, we maintain the same security environment as we had in 10.2.3, “Scenario 3: SETR MLS not active, RACF profile protection used” on page 286. That is, the following profiles have been created:

```
MDSNDB ** UACC(NONE)
but DBUSER has UPDATE access
MDSNTB ** UACC(NONE)
but RACFU01 has UPDATE access
and DBUSER has UPDATE access
```

To use a SECLABEL in a RACF profile, the profile MDSNDB ** is updated with the SECLABEL *L3B*. To do this, issue the following commands from a RACF special user:

```
RALTER MDSNDB ** SECLABEL(L3B)
SETR RACLIST(MDSNDB) REFRESH
```

When trying to create a table, the command fails (Example 10-12). This fails because the MAC check does not let someone with a SECLABEL that is not dominated by the caller create the table. The protecting RACF profile has a SECLABEL of *L3B*, which is not dominated by DBUSER's SECLABEL of *L3A*, as can be seen in the following messages (Example 10-13).

Example 10-12 Unauthorized user attempting to create table

```
CREATE TABLE DBUSER.EMPLOYEE
   ( EMPID CHAR(10)  NOT NULL,
     SECLABEL CHAR(8) NOT NULL WITH DEFAULT AS SECURITY LABEL,
     SALARY DECIMAL(7,2)
    );
---------+---------+---------+---------+---------+---------+---------+---------
DSNT408I SQLCODE = -551, ERROR:  DBUSER DOES NOT HAVE THE PRIVILEGE TO PERFORM
         OPERATION CREATE TABLE ON OBJECT DSNDB04
DSNT418I SQLSTATE   = 42501 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXODD2 SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = 50  0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'00000032'  X'00000000'  X'00000000'  X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
---------+---------+---------+---------+---------+---------+---------+---------
```

Example 10-13 ICH408I message from unauthorized creation of table

```
ICH408I USER(DBUSER  ) GROUP(SYS1    ) NAME(####################)
   DSND.DSNDB04.CREATETAB CL(MDSNDB  )
   INSUFFICIENT SECURITY LABEL AUTHORITY
   FROM ** (G)
   ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
```

Changing the MDSNDB ** profile to have a SECLABEL L2AB also fails, because DBUSER's SECLABEL of L3A will not dominate it. The same is true if the profile SECLABEL is L4A. Changing the MDSNDB ** profile to have a SECLABEL of L2A, which can be dominated by DBUSER's SECLABEL of L3A, allows DBUSER to create the table.

In addition to giving the MDSNDB profile a SECLABEL, the MDSNTB profile is also given a SECLABEL. The profile MDSNTB ** is updated with the SECLABEL L3B. To do this, issue the following commands from a RACF special user:

```
RALTER MDSNTB ** SECLABEL(L3B)
SETR RACLIST(MDSNTB) REFRESH
```

Using RACFU01, when trying to insert a row into a table, the command fails (Example 10-14). Again, this failure is because the MAC check does not let someone with a SECLABEL that is not dominated by the caller create the table. The protecting RACF profile has a SECLABEL of L3B, which is not dominated by RACFU01's SECLABEL of L2C, as can be seen in the following messages (Example 10-15).

Example 10-14 Unauthorized user attempting to insert rows into a table

```
INSERT INTO DBUSER.EMPLOYEE
      VALUES ( 'EMP SYSLOW', 'SYSLOW', 32000 ) ;
---------+---------+---------+---------+---------+---------+---------+---------
DSNT408I SQLCODE = -551, ERROR:  RACFU01 DOES NOT HAVE THE PRIVILEGE TO PERFORM
         OPERATION INSERT ON OBJECT DBUSER.EMPLOYEE
DSNT418I SQLSTATE   = 42501 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXOIN SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = -600  0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'FFFFFDA8'  X'00000000'  X'00000000'  X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
---------+---------+---------+---------+---------+---------+---------+---------
```

Example 10-15 ICH408I message from unauthorized insert into a table

```
ICH408I USER(RACFU01 ) GROUP(SYS1    ) NAME(####################)
  DSND.DBUSER.EMPLOYEE.INSERT CL(MDSNTB  )
  INSUFFICIENT SECURITY LABEL AUTHORITY
  FROM ** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
```

Changing the MDSNTB ** profile to have a SECLABEL L2AB also fails, because RACFU01's SECLABEL of L2C does not dominate it. The same is true if the profile SECLABEL is L3C. Changing the MDSNTB ** profile to have a SECLABEL of L2C, which can be dominated by RACFU01's SECLABEL of L2C, allows RACFU01 to insert rows into the table.

With RACFU01 authorized to insert rows into a table, the previous insert command executes successfully (Example 10-16). In addition, the updated profile also grants RACFU01 the authority to select to view the rows of the table. Because RACFU01 is logged on with SECLABEL *L2C*, we can only see 3 out of the 16 rows for the same reason.

Example 10-16 User with SECLABEL L2C selecting data from table

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
       ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+
EMPID          SALARY  SECLABEL
```

```
---------+---------+---------+---------+---------+---------+
EMP L1C      11300.00  L1C
EMP L2C      22300.00  L2C
EMP SYSLOW   32000.00  SYSLOW
DSNE610I NUMBER OF ROWS DISPLAYED IS 3
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
```

With the SECLABEL for MDSNTB ** being L2C, the user DBUSER does not dominate this profile with his SECLABEL of L3A. As a result, this user does not have the RACF authority to insert anything into the table. But because this user is the owner of the table, the user bypasses *all* the RACF authority checking (including MAC) and is allowed to insert rows into the table, as can be seen by the following inserts (Example 10-17).

Example 10-17 Additional INSERTs issued by DBUSER whose SECLABEL does not dominate the SECLABEL of the protecting profile in the MDSNTB class

```
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L1AXXX', 'L1A', 19999 ) ;
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L1BXXX', 'L1B', 29999 ) ;
INSERT INTO DBUSER.EMPLOYEE
  VALUES ( 'EMP L1CXXX', 'L1C', 39999 ) ;
```

As the owner, DBUSER also has the authority to select to view rows of this table. Because DBUSER is logged on with SECLABEL L3A, he can only see 5 out of the 19 rows (the original 16 plus 3 new ones we just added) to which his SECLABEL allows him.

Example 10-18 User with SECLABEL L3A selecting data from table

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
      ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+
EMPID          SALARY  SECLABEL
---------+---------+---------+---------+---------+---------+
EMP L1A      11100.00  L1A
EMP L1AXXX   19999.00  L1A
EMP L2A      22100.00  L2A
EMP L3A      33100.00  L3A
EMP SYSLOW   32000.00  SYSLOW
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
```

10.2.5 Scenario 5: SETR MLS active, RACF profile protecting without SECLABELs in profile

In this scenario, the SETR MLS option (that is, *no-write down*) is on to see how things work differently. In this scenario:

- RACF profiles are defined as documented here.
- RACF access control module is active.
- The SECLABEL class is active, and SETR MLS is on, but not SETR MLACTIVE.
- The user DBUSER is logged on with a SECLABEL of L3A.
- The user RACFU01 is logged on with a SECLABEL of L2C.

In this scenario, we maintain the same security environment as we had in 10.2.3, "Scenario 3: SETR MLS not active, RACF profile protection used" on page 286. That is, the following profiles have been created:

```
MDSNDB ** UACC(NONE)
but DBUSER has UPDATE access
MDSNTB ** UACC(NONE)
but RACFU01 has UPDATE access
and DBUSER has UPDATE access
```

A basic create table executes successfully. In this case, the profile protecting the create (MDSNDB **) is defined without a SECLABEL, and the user creating the table (DBUSER) has the SECLABEL L3A.

Next, an insert into the table executes successfully as well. The profile protecting the insert (MDSNTB **) is defined without a SECLABEL, and the user inserting into the table (RACFU01) has the SEDCLABEL L2C.

Basically, nothing looks different when creating or inserting the row into the table, compared to scenario 3, but when listing the data, things get interesting. The SECLABEL for the inserted rows all change to *L2C*, the SECLABEL of the inserting user. Because all the inserted rows have the SECLABEL of L2C, the user (who issued the insert) can see all 16 rows she inserted (Example 10-19 on page 294).

Example 10-19 User with SECLABEL L2C selecting data from table

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
       ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+
EMPID          SALARY  SECLABEL
---------+---------+---------+---------+---------+---------+
EMP L1A      11100.00  L2C
EMP L1AB     99100.00  L2C
EMP L1B      11200.00  L2C
EMP L1C      11300.00  L2C
EMP L2A      22100.00  L2C
EMP L2AB     99200.00  L2C
EMP L2B      22200.00  L2C
EMP L2C      22300.00  L2C
EMP L3A      33100.00  L2C
EMP L3B      33200.00  L2C
EMP L3C      33300.00  L2C
EMP L4A      44100.00  L2C
EMP L4B      44200.00  L2C
EMP L4C      44300.00  L2C
EMP SYSHGH   99500.00  L2C
EMP SYSLOW   32000.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 16
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
```

Because there is no SECLABEL for MDSNTB **, the MAC check does not occur, and the user DBUSER can also add rows to the table. Using the previous example to add new rows, the user successfully adds the new rows. But as in the case above, she adds them with her SECLABEL of L3A (Example 10-20).

Example 10-20 User with SECLABEL L3A selecting data from table

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
      ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+
EMPID          SALARY  SECLABEL
---------+---------+---------+---------+---------+---------+
EMP L1AXXX   19999.00  L3A
EMP L1BXXX   29999.00  L3A
EMP L1CXXX   39999.00  L3A
DSNE610I NUMBER OF ROWS DISPLAYED IS 3
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
```

10.2.6 Scenario 6: SETR MLS and SETR MLACTIVE active, RACF profile protection

In this scenario, the SETR MLS option is on, in addition to the SETR MLACTIVE option, to see how things work differently. In this scenario:

- RACF profiles are defined as documented here.
- RACF access control module is active.
- The SECLABEL class is active, and SETR MLS and SETR MLACTIVE are both active.
- The user DBUSER is logged on with a SECLABEL of L3A.
- The user RACFU01 is logged on with a SECLABEL of L2C.

In this scenario, we maintain the same security environment as we had in 10.2.3, “Scenario 3: SETR MLS not active, RACF profile protection used” on page 286. That is, the following profiles have been created:

```
MDSNDB ** UACC(NONE)
but DBUSER has UPDATE access
MDSNTB ** UACC(NONE)
but RACFU01 has UPDATE access
and DBUSER has UPDATE access
```

When trying to create a table, the command fails (Example 10-21). This failure is because there is no SECLABEL in the MDSNDB ** profile and one is required when SETR MLACTIVE is on. The failure can be seen in the following messages (Example 10-22 on page 296).

Example 10-21 Unauthorized user attempting to create table

```
CREATE TABLE DBUSER.EMPLOYEE
   ( EMPID CHAR(10)  NOT NULL,
     SECLABEL CHAR(8) NOT NULL WITH DEFAULT AS SECURITY LABEL,
     SALARY DECIMAL(7,2)
    );
---------+---------+---------+---------+---------+---------+---------+---------
DSNT408I SQLCODE = -551, ERROR:  DBUSER DOES NOT HAVE THE PRIVILEGE TO PERFORM
         OPERATION CREATE TABLE ON OBJECT DSNDB04
DSNT418I SQLSTATE   = 42501 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXODD2 SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = 50  0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'00000032'  X'00000000'  X'00000000'  X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
---------+---------+---------+---------+---------+---------+---------+---------
```

Example 10-22   ICH408I message from unauthorized creation of table

```
ICH408I USER(DBUSER  ) GROUP(SYS1    ) NAME(####################)
   DSND.DSNDB04.CREATETAB CL(MDSNDB  )
   INSUFFICIENT SECURITY LABEL AUTHORITY
   FROM ** (G)
   ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
```

Changing the MDSNDB ** profile to have a SECLABEL of L3A allows DBUSER to create the table. To do this, issue the following commands from a RACF special user:

```
RALTER MDSNDB ** SECLABEL(L3A)
SETR RACLIST(MDSNDB) REFRESH
```

Using RACFU01, when trying to insert a row into a table, the command fails (Example 10-23). Again, this failure is because there is no SECLABEL in the MDSNTB ** profile and one is required when SETR MLACTIVE is on. Example 10-24 shows the failure messages.

Example 10-23   Unauthorized user attempting to insert rows into a table

```
INSERT INTO DBUSER.EMPLOYEE
      VALUES ( 'EMP SYSLOW', 'SYSLOW', 32000 ) ;
---------+---------+---------+---------+---------+---------+---------+---------
DSNT408I SQLCODE = -551, ERROR:  RACFU01 DOES NOT HAVE THE PRIVILEGE TO PERFORM
         OPERATION INSERT ON OBJECT DBUSER.EMPLOYEE
DSNT418I SQLSTATE   = 42501 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXOIN SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = -600  0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'FFFFFDA8'  X'00000000'  X'00000000'  X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
---------+---------+---------+---------+---------+---------+---------+---------
```

Example 10-24   ICH408I message from unauthorized insert into a table

```
ICH408I USER(RACFU01 ) GROUP(SYS1    ) NAME(####################)
  DSND.DBUSER.EMPLOYEE.INSERT CL(MDSNTB  )
  INSUFFICIENT SECURITY LABEL AUTHORITY
  FROM ** (G)
  ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
```

Changing the MDSNTB ** profile to have a SECLABEL of L2C allows RACFU01 to insert rows into the table. To do this, issue the following commands from a RACF special user:

```
RALTER MDSNTB ** SECLABEL(L2C)
SETR RACLIST(MDSNTB) REFRESH
```

When inserting rows into the table, nothing looks different, but when listing the data, things get interesting. The SECLABEL for the inserted rows all change to L2C, the SECLABEL of the user RACFU01 (Example 10-25).

Example 10-25 User with SECLABEL L2C selecting data from table

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
       ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+
EMPID         SALARY  SECLABEL
---------+---------+---------+---------+---------+---------+
EMP L1A      11100.00  L2C
EMP L1AB     99100.00  L2C
EMP L1B      11200.00  L2C
EMP L1C      11300.00  L2C
EMP L2A      22100.00  L2C
EMP L2AB     99200.00  L2C
EMP L2B      22200.00  L2C
EMP L2C      22300.00  L2C
EMP L3A      33100.00  L2C
EMP L3B      33200.00  L2C
EMP L3C      33300.00  L2C
EMP L4A      44100.00  L2C
EMP L4B      44200.00  L2C
EMP L4C      44300.00  L2C
EMP SYSHGH   99500.00  L2C
EMP SYSLOW   32000.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 16
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
```

Now with the SECLABEL for MDSNTB ** being L2C, the user DBUSER does not dominate this profile with his SECLABEL of L3A. Therefore, this user does not have the RACF authority to insert anything into the table. As shown in 10.2.4, “Scenario 4: SETR MLS not active, RACF profile protection with SECLABELs in profiles” on page 289, with SETR MLS off, the user, even though he does not have RACF authority to insert into the table, was able to insert new rows into the table.

With SETR MLS on, this is not the case. Even the owner of the table must have the RACF authority to add a row into a table. In 10.2.5, “Scenario 5: SETR MLS active, RACF profile protecting without SECLABELs in profile” on page 293, where SETR MLS was on, DBUSER, the owner, was able to insert rows into the table, but in that case, the MDSNTB ** profile did not have a SECLABEL assigned to it. Now that the profile is required to have a SECLABEL (because

SETR MLACTIVE is on), and DBUSER does not dominate that SECLABEL, that user will not be able to insert new rows into the table.

Using the same Example 10-17 on page 292 to add new rows, DBUSER fails to add any new rows and generates a ICH408I error message (Insufficient Security Label Authority), as shown in Example 10-27.

Example 10-26 Unauthorized user attempting to insert rows into a table

```
INSERT INTO DBUSER.EMPLOYEE
     VALUES ( 'EMP L1AZZZ', 'L1A', 19999 ) ;
---------+---------+---------+---------+---------+---------+---------+---------
DSNT408I SQLCODE = -551, ERROR:  DBUSER DOES NOT HAVE THE PRIVILEGE TO PERFORM
         OPERATION INSERT ON OBJECT DBUSER.EMPLOYEE
DSNT418I SQLSTATE   = 42501 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXOIN SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = -600  0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'FFFFFDA8'  X'00000000'  X'00000000'  X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION
---------+---------+---------+---------+---------+---------+---------+---------
```

Example 10-27 ICH408I message from unauthorized insert into a table

```
ICH408I USER(DBUSER  ) GROUP(SYS1    ) NAME(####################)
  DSND.DBUSER.EMPLOYEE.INSERT CL(MDSNTB  )
  INSUFFICIENT SECURITY LABEL AUTHORITY
  FROM ** (G)
  ACCESS INTENT(UPDATE )  ACCESS ALLOWED(NONE   )
```

Note: Turning on SETR MLS (write-down authority) does not grant any OWNER privileges, as we have seen. It does not grant any MATCH privileges either.

10.2.7 Scenario 7: SETR MLS active, RACF profile protection used with special SECLABELs in profiles

In this scenario, SETR MLS is active, and RACF profile protection is in use with special SECLABELs in profiles.

RACF has a few special SECLABELs, two of which we look at here, SYSMULTI and SYSNONE. As defined earlier, SYSMULTI is equivalent to any SECLABEL that it is tested against. SYSNONE is used when no SECLABEL is required for the resource, but because a MAC check needs to be done, one is provided here (in short, it acts like SYSMULTI in this case).

Again, in this scenario, we maintain the same environment that existed in 10.2.3, “Scenario 3: SETR MLS not active, RACF profile protection used” on page 286.

In this scenario, the SETR MLS option (that is, *no-write down*) is on, and we use the special SECLABELs to see how things work with them. In this scenario:

- RACF profiles are defined as documented here.
- RACF access control module is active.
- The SECLABEL class is active, and SETR MLS is on, but not SETR MLACTIVE.
- The user DBUSER is logged on with a SECLABEL of L3A.
- The user RACFU01 is logged on with a SECLABEL of L2C.

In this scenario, we maintain the same security environment as we had in 10.2.3, “Scenario 3: SETR MLS not active, RACF profile protection used” on page 286. That is, the following profiles have been created:

```
MDSNDB ** UACC(NONE)
but DBUSER has UPDATE access
MDSNTB ** UACC(NONE)
but RACFU01 has UPDATE access
and DBUSER has UPDATE access
```

To introduce some of the special SECLABELs with the SETR MLS option, the MDSNDB ** profile is updated with the SECLABEL SYSMULTI. To do this, issue the following commands from a RACF special user:

```
RALTER MDSNDB ** SECLABEL(SYSMULTI))
SETR RACLIST(MDSNDB) REFRESH
```

A basic create table executes successfully. In this case, the profile protecting the create (MDSNDB **) is defined with SECLABEL SYSMULTI, which is equivalent to the SECLABEL of the user creating the table (DBUSER with SECLABEL L3A).

For the inserts, the MDSNTB ** profile is also updated with a SECLABEL, but in this case, SYSNONE. To do this, issue the following commands from a RACF special user:

```
RALTER MDSNTB ** SECLABEL(SYSNONE)
SETR RACLIST(MDSNTB) REFRESH
```

Now inserting into the table executes successfully as well. The profile protecting the insert (MDSNTB **) is defined with the SECLABEL SYSNONE, which is equivalent to the SECLABEL of the user inserting into the table (RACFU01 with SECLABEL L2C). Now when listing the data, as shown before with SETR MLS on, the data that is inserted has been changed. The SECLABEL for the inserted rows all change to L2C, the SECLABEL of the user RACFU01 (Example 10-28).

Example 10-28 User with SECLABEL L2C selecting data from table

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
       ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+
EMPID          SALARY  SECLABEL
---------+---------+---------+---------+---------+---------+
EMP L1A      11100.00  L2C
EMP L1AB     99100.00  L2C
EMP L1B      11200.00  L2C
EMP L1C      11300.00  L2C
EMP L2A      22100.00  L2C
EMP L2AB     99200.00  L2C
EMP L2B      22200.00  L2C
EMP L2C      22300.00  L2C
EMP L3A      33100.00  L2C
EMP L3B      33200.00  L2C
EMP L3C      33300.00  L2C
EMP L4A      44100.00  L2C
EMP L4B      44200.00  L2C
EMP L4C      44300.00  L2C
EMP SYSHGH   99500.00  L2C
EMP SYSLOW   32000.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 16
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
```

The user DBUSER, the owner of the table, can also add rows to the table. Using the previous example to add new rows, the user successfully adds the new rows. But as in the previous example, adds w them with the SECLABEL of L3A (Example 10-29).

Example 10-29 User with SECLABEL L3A selecting data from table that user just inserted

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
      ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+
EMPID         SALARY  SECLABEL
---------+---------+---------+---------+---------+---------+
EMP L1AXXX   19999.00  L3A
EMP L1BXXX   29999.00  L3A
EMP L1CXXX   39999.00  L3A
DSNE610I NUMBER OF ROWS DISPLAYED IS 3
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
```

Having DBUSER log on to a user with a higher SECLABEL (say SYSHIGH), he can now see all 19 rows that were added to this table (Example 10-30).

Example 10-30 User with SECLABEL SYSHIGH selecting data from table

```
SELECT EMPID, SALARY, SECLABEL FROM DBUSER.EMPLOYEE
      ORDER BY EMPID;
---------+---------+---------+---------+---------+---------+
EMPID         SALARY  SECLABEL
---------+---------+---------+---------+---------+---------+
EMP L1A      11100.00  L2C
EMP L1AB     99100.00  L2C
EMP L1AXXX   19999.00  L3A
EMP L1B      11200.00  L2C
EMP L1BXXX   29999.00  L3A
EMP L1C      11300.00  L2C
EMP L1CXXX   39999.00  L3A
EMP L2A      22100.00  L2C
EMP L2AB     99200.00  L2C
EMP L2B      22200.00  L2C
EMP L2C      22300.00  L2C
EMP L3A      33100.00  L2C
EMP L3B      33200.00  L2C
EMP L3C      33300.00  L2C
EMP L4A      44100.00  L2C
EMP L4B      44200.00  L2C
EMP L4C      44300.00  L2C
EMP SYSHGH   99500.00  L2C
EMP SYSLOW   32000.00  L2C
DSNE610I NUMBER OF ROWS DISPLAYED IS 19
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+
```

10.3 Conclusion

When mixing row-level support and the RACF/DB2 exit, be very careful. As stated earlier, planning is key.

With SETR MLS on, security becomes highly restrictive. The DB2 row-level support only allows users to enter data at their SECLABEL. And with the RACF access control module in place, the only way to truly take advantage of row-level support is by using the SECLABELs SYSNONE or SYSMULTI to protect DB2 operations. However, to protect an entire table with the same SECLABEL, you can avoid using row-level support, but protect the entire table with a profile, and use the appropriate SECLABEL to protect actions against that table.

With SETR MLS off (that is, SETR NOMLS), security is less restrictive. Users can insert data at any SECLABEL, as long as they have the authority to update the table, though they are only allowed to see information they have the authority to look at. However, users can easily insert data with the wrong classification, and as a result, declassify or mis-classify data, allowing the wrong people to see it.

Hopefully, through these examples, you have a basic concept of how to use row-level support with the RACF access control module and how they interact with each other.

A

Trusted context syntax

We provide the full description of the DDL for creating a trusted context.[1]

The CREATE TRUSTED CONTEXT statement defines a trusted context at the current server.

Invocation

This statement can be embedded in an application program or issued interactively. It is an executable statement that can be dynamically prepared only if DYNAMICRULES run behavior is implicitly or explicitly specified.

Authorization

The privilege set we define here must include SYSADM authority.

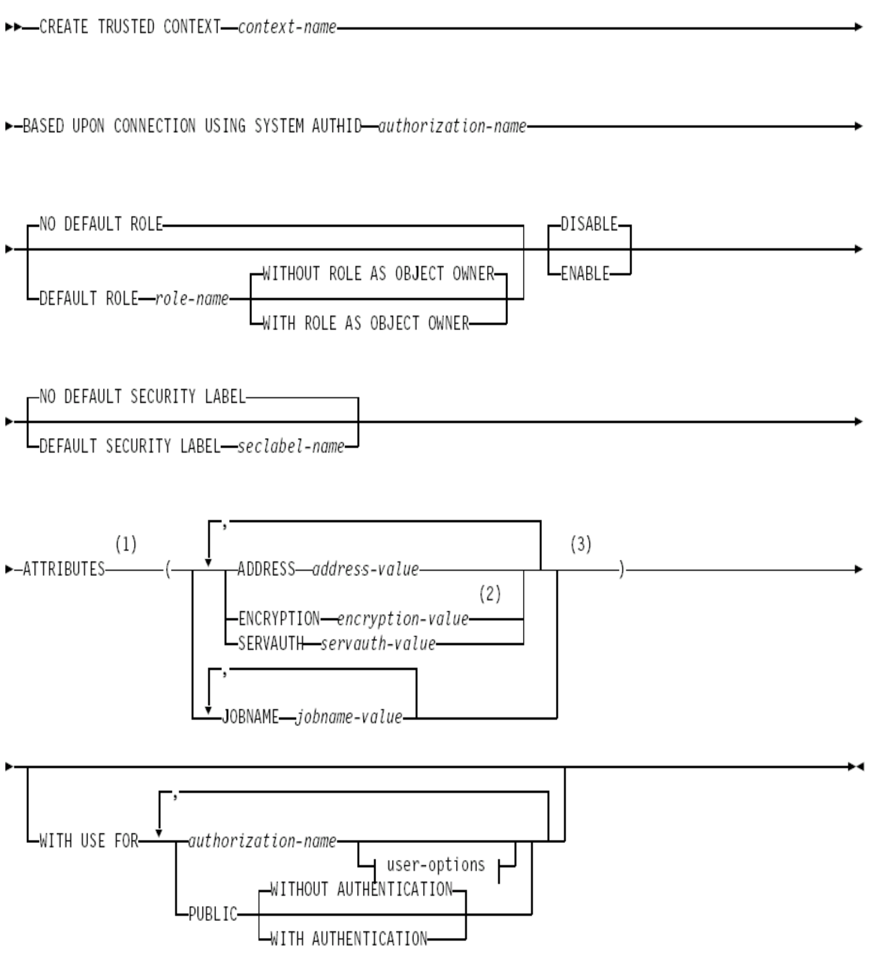
Privilege set

If the statement is embedded in an application program, the privilege set is the privileges that are held by the of the owner of the plan or package. If the application is bound in a trusted context with the ROLE AS OBJECT OWNER clause specified, a role is the owner. Otherwise, an authorization ID is the owner.

If the statement is dynamically prepared, the privilege set is the privileges that are held by the SQL authorization ID of the process unless the process is within a trusted context and the ROLE AS OBJECT OWNER clause is specified. In that

[1] This information is from *DB2 for z/OS SQL Reference*, SC18-9854.

case, the privileges set is the privileges that are held by the role that is associated with the primary authorization ID of the process. See Figure A-1 and Figure A-2 on page 305.



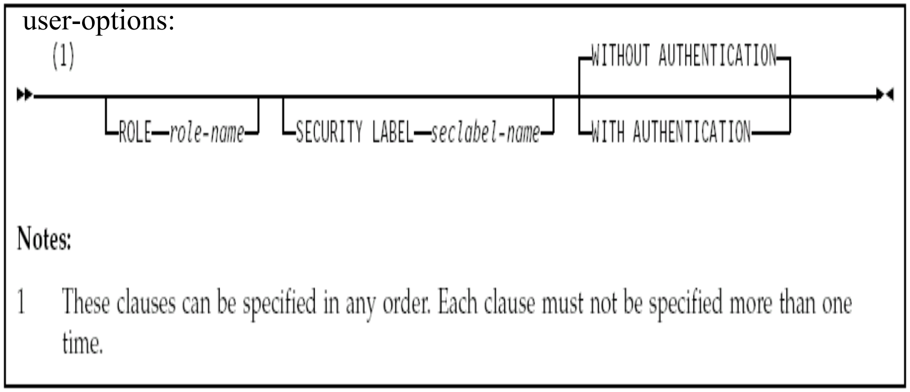
Notes:

1 This clause and the clauses that follow can be specified in any order. Each clause must not be specified more than one time.

2 ENCRYPTION must not be specified more than one time.

3 Each pair of attribute name and corresponding value must be unique.

Figure A-1 Create trusted context



Notes:

1 These clauses can be specified in any order. Each clause must not be specified more than one time.

Figure A-2 Create trusted context continued

Description

Use the following values for the trusted context:

- *context-name*

  Names the trusted context. The name must not identify a trusted context that exists at the current server.

- BASED UPON CONNECTION USING SYSTEM AUTHID

  Specifies that the context is a connection that is established by the authorization ID that is specified by *authorization-name*. The system authorization ID is the primary authorization ID. For a remote connection, it is derived from the system user ID that is provided by an external entity, such as a middleware server. For a local connection, the system authorization ID is derived depending on the sources, as specified in Table A-1.

Table A-1 System authorization ID for a local connection

Source of local connection	System authorization ID
Started task (RRSAF)	USER parameter on JOB statement or RACF USER
TSO	TSO logon ID
BATCH	USER parameter on JOB statement

authorization-name must not be associated with an existing trusted context.

- NO DEFAULT ROLE or DEFAULT ROLE *role-name*

  Specifies whether a default role is associated with a trusted connection that is based on the specified trusted context.

  - NO DEFAULT ROLE

    Specifies that the trusted context does not have a default role. The authorization ID of the process is the owner of any object that is created using a trusted connection that is based on this trusted context. That authorization ID must possess all of the privileges that are necessary to create that object.

    NO DEFAULT ROLE is the default.

  - DEFAULT ROLE *role-name*

    Specifies that role-name is the role for the trusted context. *role-name* must identify a role that exists at the current server. This role is used with the user in a trusted connection that is based on the specified trusted context when the user does not have a user-specified role that is defined as part of the definition of this trusted context.

- WITHOUT ROLE AS OBJECT OWNER or WITH ROLE AS OBJECT OWNER

  Specifies whether a role is used as the owner of objects that are created using a trusted connection that is based on the specified trusted context. The authorization ID of the process is the owner of any object that is the created using a trusted connection that is based on this trusted context. That authorization ID must possess all of the privileges that are necessary to create the object.

  - WITHOUT ROLE AS OBJECT OWNER

    Specifies that a role is not used as the owner of the objects that are created using a trusted connection that is based on the specified trusted context.

    WITHOUT ROLE AS OBJECT OWNER is the default.

  - WITH ROLE AS OBJECT OWNER

    Specifies that the context-assigned role is the owner of the objects that are created using a trusted connection that is based on this trusted context and that role must possess all of the privileges that are necessary to create the object. The context-assigned role is the role that is defined for the user within this trusted context, if one is defined. Otherwise, the role is the default role that is associated with the trusted context. The role is also used as the grantor for any GRANT statements that are issued and the revoker for any REVOKE statement that are issued using a trusted connection that is based on this trusted context.

- DISABLE or ENABLE

  Specifies whether the trusted context is created in the enabled or disabled state:

  - DISABLE

    Specifies that the trusted context is disabled when it is created. A trusted context that is disabled is not considered when a trusted connection is established. DISABLE is the default.

  - ENABLE

    Specifies that the trusted context is enabled when it is created.

- NO DEFAULT SECURITY LABEL or DEFAULT SECURITY LABEL *seclabel-name*

  Specifies whether the trusted connection has a default security label:

  - NO DEFAULT SECURITY LABEL

    Specifies that the trusted context does not have a default security label.

  - DEFAULT SECURITY LABEL *seclabel-name*

    Specifies that *seclabel-name* is the default security label for the trusted context and is the security label that is used for multilevel security verification. *seclabel-name* must identify one of the RACF SECLABEL values that is defined for the SYSTEM AUTHID. This security label is used for a trusted connection that is based on the specified trusted context when the user does not have a specific security label defined as part of the definition of this trusted context. In this case, *seclabel-name* must also identify one of the RACF SECLABEL values that is defined for the user.

- ATTRIBUTES

  Specifies a list of one or more connection trust attributes that are used to define the trusted context:

  - ADDRESS *address-value*

    Specifies the actual communication address that is used by the connection to communicate with the database manager. The protocol supported is only for TCP/IP. The ADDRESS attribute can be specified multiple times, but each *address-value* must be unique.

    When establishing a trusted connection, if multiple values are defined for the ADDRESS attribute for a trusted context, a candidate connection is considered to match this attribute if the address that is used by a connection matches any of the defined values for the ADDRESS attribute of the trusted context.

    address-value specifies a string constant that contains the value that is associated with the ADDRESS trust attribute. *address-value* must be an

IPv4 address, an IPv6 address, or a secure domain name with a length no greater than 254 bytes. No validation of *address-value* is done at the time the CREATE TRUSTED CONTEXT statement is processed. *address-value* must be left justified within the string constant.

- An IPv4 address is represented as a dotted decimal address. An example of an IPv4 address is 9.112.46.111.
- An IPv6 address is represented as a colon hexadecimal address. An example of an IPv6 address is 2001:0DB8:0000:0000:0008:0800:200C:417A. This address can also be expressed in a compressed form as 2001:DB8::8:800:200C:417A.
- A domain name is converted to an IP address by the domain name server where a resulting IPv4 or IPv6 address is determined. An example of a domain name is www.ibm.com. The gethostbyname socket call is used to resolve the domain name.

– ENCRYPTION *encryption-value*

encryption-value specifies a string constant that contains the value that is associated with the ENCRYPTION trust attribute. *encryption-value* must be left justified within the string constant. ENCRYPTION must not be specified more than one time in the statement. *encryption-value* must be one of the following values:

- NONE, which specifies that no specific level of encryption is required.
- LOW, which specifies that a minimum of light encryption is required. LOW corresponds to 64-bit DRDA encryption.
- HIGH, which specifies that strong encryption is required. HIGH corresponds to SSL encryption.

Table A-2 summarizes when a trusted context can be used depending on the encryption that is used by the existing connection. If the trusted context cannot be used for the connection, an error code is returned.

Table A-2 Summary of when trusted context can be used by an existing connection

Encryption that is used by the existing connection	**Value of the ENCRYPTION clause for the trusted context**	**Can the trusted context be used for the connection?**
No encryption	NONE	Yes
No encryption	LOW	No
No encryption	HIGH	No
Low encryption (64 bit)	NONE	Yes
Low encryption (64 bit)	LOW	Yes

Encryption that is used by the existing connection	Value of the ENCRYPTION clause for the trusted context	Can the trusted context be used for the connection?
Low encryption (64 bit)	HIGH	No
High encryption (128 bit)	NONE	Yes
High encryption (128 bit)	LOW	Yes
High encryption (128 bit)	HIGH	Yes

– JOBNAME *jobname-value*

Specifies the z/OS job name or started task name (depending on the source of the address space) for local applications. The JOBNAME attribute can be specified multiple times, but each *jobname-value* must be unique.

jobname-value specifies a string constant that contains the value that is associated with the JOBNAME trust attribute. *jobname-value* is an EBCDIC 8-byte job name or started task name. *jobname-value* must be left justified in the string constant. Table A-3 lists possible values for the job name (depending on the source of the address space).

Table A-3 Job name for local connection

Source of the address space	Job name
RRSAF	Job name or started task name
TSO	TSO logon ID
BATCH	Job name on JOB statement

– SERVAUTH *servauth-value*

Specifies the name of a resource in the RACF SERVAUTH class. This resource is the network access security zone name that contains the IP address of the connection that is used to communicate with DB2. The SERVAUTH attribute can be specified multiple times, but each *servauth-value* must be unique.

servauth-value specifies a string constant that contains the value that is associated with the SERVAUTH trust attribute. *servauth-value* is an EBCDIC 64-byte RACF SERVAUTH CLASS resource name. *servauth-value* must be left justified in the string constant. No validation of *servauth-value* is done at the time the CREATE TRUSTED CONTEXT statement is processed.

- WITH USE FOR

  Specifies who can use a trusted connection that is based on the specified trusted context. If the definition of a trusted context allows access by PUBLIC and a list of users, the specifications for a user take higher precedence than the specification for PUBLIC. For example, assume that a trusted context is defined that allows access by both PUBLIC WITH AUTHENTICATION and JOE WITHOUT AUTHENTICATION. If the trusted context is used by JOE, authentication is not required. However, if the trusted context is used by GEORGE, authentication is required.

  – *authorization-name*

    Specifies that the trusted connection can be used by the specified *authorization-name*. This is the DB2 primary authorization ID. The *authorization-name* must not be specified more than one time in the WITH USE FOR clause.

    If this *authorization-name* is the same as the authorization name that is specified in the SYSTEM AUTHID clause, the role or the security label that is specified in the WITH USE FOR clause takes precedence over the default values that are specified in the SYSTEM AUTHID clause.

    If the *authorization-name* is the same as the authorization name that is specified in the SYSTEM AUTHID and if authentication is required either by specification of the AUTHENTICATION clause or by setting the value of the TCP/IP Already Verified subsystem parameter to NO, the authentication requirement takes precedence when establishing a trusted connection. For example, if *authorization-name* is the same as the authorization name that is specified in the SYSTEM AUTHID and the WITHOUT AUTHENTICATION clause is specified, but the TCP/IP Already Verified subsystem parameter is set to NO, an authentication token is required for SYSTEM AUTHID when the trusted connection is established. If the *authorization-name* is the SYSTEM AUTHID and the WITH AUTHENTICATION clause is specified, but the TCP/IP Already Verified subsystem parameter is set to YES, an authentication token is still required for SYSTEM AUTHID.

  – ROLE *role-name*

    Specifies that *role-name* is the role that is used when a trusted connection is using the specified trusted context. *role-name* must identify a role that exists at the current server. The role that is explicitly specified for the user overrides any default role that is associated with the trusted context.

  - SECURITY LABEL *seclabel-name*

    Specifies that *seclabel-name* is the security label to use for multilevel security verification when the trusted connection is used by the specified *authorization-name*. The *seclabel-name* must be one of the RACF SECLABEL values that is defined for the user. The security label that is explicitly specified for the user overrides any default security label that is associated with the trusted context.

- PUBLIC

  Specifies that a trusted connection that is based on the specified trusted context can be used by any user. All users that are using a trusted connection that is defined with PUBLIC use the privileges that are associated with the default role for the associated trusted context. If the default role is not defined for the trusted context, there is no role associated with the users that use a trusted connection that is based on the specified trusted context.

  If the default security label for the trusted context is defined, all users that are using the trusted context must have the security label defined as one of the RACF SECLABEL values for the user. The default security label is used for multilevel security verification with all users that are using the trusted context.

- WITHOUT AUTHENTICATION or WITH AUTHENTICATION

  Specifies whether use of the trusted connection requires authentication of the user.

  - WITHOUT AUTHENTICATION

    Specifies that use of a trusted connection by the user does not require authentication. WITHOUT AUTHENTICATION is the default.

  - WITH AUTHENTICATION

    Specifies that use of a trusted connection requires the authentication token with the authorization ID to authenticate the user. If a trusted connection is established locally, the authentication token is the password that is provided by the CONNECT statement with the USER and USING clauses.

    If the trusted connection is established from a remote client, the authentication token can be one of the following tokens:

    - Password
    - RACF PassTicket
    - Kerberos token

Notes

See the following considerations:

- Owner privileges: There are no specific privileges on a trusted context.
- Specifying a role in the definition of a trusted context: The definition of a trusted context can designate a role for a specific authorization ID, and a default role for use for an authorization ID for which a specific role has not been specified in the definition of the trusted context. This role can be used with a trusted connection that is based on the trusted context, but it does not make the role available outside of a trusted connection that is based on the trusted context. When an SQL statement that is not a CREATE, GRANT, or REVOKE statement is issued using a trusted connection, the privileges that are held by a role that is in effect for the authorization ID within the definition of the associated trusted context are considered in addition to other privileges that are directly held by the authorization ID of the statement. The CREATE, GRANT, and REVOKE statements only consider the privileges of the role that is in effect for the trusted connection, or the authorization ID of the statement if a role is not in effect for the trusted connection. If ROLE AS OBJECT OWNER is in effect for a trusted connection, the role that is in effect for the authorization ID for the trusted connection becomes the owner of any object that is created while using the trusted connection.
- When a newly created trusted context takes effect: The newly created trusted context takes effect after the CREATE TRUSTED CONTEXT statement is committed. If the CREATE TRUSTED CONTEXT statement results in an error or is rolled back, no trusted context is created.

Examples

We provide the following examples.

Example 1

The following statement creates a trusted context called CTX1, which is based on a connection and can only be used by users JOE and SAM. Authentication information is required for JOE to use the trusted connection. The trusted context specifies a default role called CTXROLE. However, when JOE uses the trusted connection, the default role is overridden by the user role, ROLE1. When SAM uses the trusted connection, SAM uses the default role. CTX1 is enabled when it is created.

```
CREATE TRUSTED CONTEXT CTX1
    BASED UPON CONNECTION USING SYSTEM AUTHID ADMF001
    ATTRIBUTES (ADDRESS '9.30.131.203', ENCRYPTION 'LOW')
    DEFAULT ROLE CTXROLE
    ENABLE
    WITH USE FOR SAM, JOE ROLE ROLE1 WITH AUTHENTICATION;
```

Example 2

The following statement creates a trusted context, CTX2, for a started task, WASPROD. CTX2 is based on a connection, can be used by user SALLY, specifies a default role CTXROLE, and is enabled when it is created. SALLY uses the default role that is associated with the trusted context.

```
CREATE TRUSTED CONTEXT CTX2
BASED UPON CONNECTION USING SYSTEM AUTHID ADMF002
ATTRIBUTES (JOBNAME 'WASPROD') DEFAULT ROLE CTXROLE WITH ROLE AS OBJECT
OWNER
ENABLE
WITH USE FOR SALLY;
```

B

RACF options that control the use of security labels

This appendix provides detailed descriptions of the RACF settings that control a multilevel security system. Much of this appendix is based on the information in *z/OS Planning for Multilevel Security and the Common Criteria,* GA22-7509-05.

You can use the RACF SETROPTS command to control how security labels are used on your system. Different SETROPTS options control different aspects of security label function. We describe these options in the following sections:

- COMPATMODE and NOCOMPATMODE
- MLACTIVE and NOMLACTIVE
- MLFSOBJ
- MLIPCOBJ
- MLNAMES and NOMLNAMES
- MLQUIET and NOMLQUIET
- MLS and NOMLS
- MLSTABLE and NOMLSTABLE
- SECLABELAUDIT and NOSECLABELAUDIT
- SECLABELCONTROL and NOSECLABELCONTROL
- SECLBYSYSTEM and NOSECLBYSYSTEM

Before you can activate most of these options, the SECLABEL class must be active. We discuss activating the SECLABEL class in 2.7, “Using security labels” on page 19.

COMPATMODE and NOCOMPATMODE

If a subject used a pre-RACF 1.9 protocol that did not, or was not able to, specify a security label, the COMPATMODE option allows subjects to access a resource, provided the user is authorized to use a security label that would allow the access, regardless of whether the user is using the security label at the time of the authorization check. The NOCOMPATMODE option requires that the user be using a security label that allows the access in order to be granted access. NOCOMPATMODE is in effect when a RACF database is first initialized using IRRMIN00.

The COMPATMODE option only applies if the creator of the ACEE for the user had an old RACINIT parameter list. The only reason to use COMPATMODE is to prevent an old application from failing while you test. Correct the application to use a current RACINIT protocol, or replace it with an application that does.

Guideline: Do not set the COMPATMODE option.

MLACTIVE and NOMLACTIVE

Use the MLACTIVE and NOMLACTIVE options to control whether security labels are required for certain resources. The MLACTIVE option requires security labels for most resources other than resources related to z/OS UNIX and for all users entering the system.

Requirement: The SECLABEL class *must* be active before you can use the MLACTIVE options.

The MLACTIVE option has two suboptions, FAILURES and WARNING:

- MLACTIVE(FAILURES) specifies that RACF is to reject any request to access a resource in the classes listed in Figure B-1 on page 318 that does not have a security label. For a detailed description of the access checking methodology, see *z/OS Security Server RACF Security Administrator's Guide,* SA22-7683. RACF also rejects any attempt by a user to enter the system without a security label, unless the user is authorized to use the SYSLOW security label, in which case the user runs with the SYSLOW security label. In addition, a user task running in a server address space must have a security label that is equivalent to the security label of the address space.
- MLACTIVE(WARNING) specifies that RACF is to issue a warning for any request to access a resource in a class listed in Figure B-1 on page 318 that does not have a security label, but allow the access if the request passes the discretionary access check. RACF also issues a warning for any attempt by a user to enter the system without a security label, but allows the user to enter the system. In addition, a user task running in a server address space must have a security label that is equivalent to the security label of the address space.

The NOMLACTIVE option specifies that security labels are not required for resources in the classes listed in Figure B-1 on page 318.

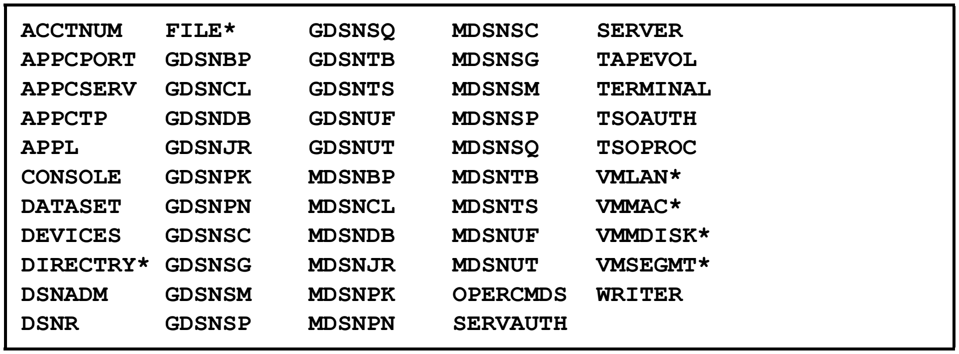
You can set MLACTIVE(WARNING) temporarily when you are setting up multilevel security to verify that you have assigned security labels to all of the users and resources that require them. Then, set MLACTIVE(FAILURES) when you are ready to enforce the use of security labels.

Before you activate MLACTIVE(FAILURES), ensure that you have done the following tasks:

1. You defined security labels by defining profiles in the RACF SECLABEL class.
2. You authorized all the users to use the security labels they will need.
3. You assigned security labels to all applications and started tasks that act as servers, authenticating users in their address spaces. If the users can operate at multiple security labels, assign SYSMULTI to the started tasks. This includes started tasks that are marked trusted, such as JES2. The security label for a started task must be assigned prior to the task starting. If a task is only started at IPL time, you might need to re-IPL before you activate MLACTIVE(FAILURES) to ensure that the task has a security label.
4. You assigned security labels to all data sets.
5. You assigned security labels to all profiles in the classes shown in Figure B-1 on page 318.
6. You activated and RACLISTed the SECLABEL class.

7. You temporarily ran with MLACTIVE(WARNING) set to verify that you have completed your setup correctly.

Figure B-1 shows resource classes that require a security label when MLACTIVE(FAILURES) is active. Classes marked with an asterisk (*) are VM classes and are not relevant on a z/OS system.



```
ACCTNUM   FILE*     GDSNSQ    MDSNSC    SERVER
APPCPORT  GDSNBP    GDSNTB    MDSNSG    TAPEVOL
APPCSERV  GDSNCL    GDSNTS    MDSNSM    TERMINAL
APPCTP    GDSNDB    GDSNUF    MDSNSP    TSOAUTH
APPL      GDSNJR    GDSNUT    MDSNSQ    TSOPROC
CONSOLE   GDSNPK    MDSNBP    MDSNTB    VMLAN*
DATASET   GDSNPN    MDSNCL    MDSNTS    VMMAC*
DEVICES   GDSNSC    MDSNDB    MDSNUF    VMMDISK*
DIRECTRY* GDSNSG    MDSNJR    MDSNUT    VMSEGMT*
DSNADM    GDSNSM    MDSNPK    OPERCMDS  WRITER
DSNR      GDSNSP    MDSNPN    SERVAUTH
```

Figure B-1 Resource classes

When the MLACTIVE option is active, if a user creates a profile in any of the classes listed in Figure B-1, the system assigns a security label to the profile. If the command that creates the profile does not specify a security label, the issuing user’s current security label is used.

MLFSOBJ

Use the MLFSOBJ option to control whether security labels are required for z/OS UNIX files and directories. The MLFSOBJ option has two suboptions, ACTIVE and INACTIVE:

- MLFSOBJ(ACTIVE) specifies that when the SECLABEL class is active, only trusted or privileged started tasks can access files and directories that do not have security labels. For information about how security labels are assigned to z/OS UNIX files and directories, see Chapter 3, “MLS” on page 27.
- MLFSOBJ(INACTIVE) specifies that files and directories do not require security labels.

If you issue the command SETROPTS MLFSOBJ without specifying ACTIVE or INACTIVE, the default value is ACTIVE.

Before you activate MLFSOBJ(ACTIVE), ensure that you have done the following tasks:

1. You defined security labels by defining profiles in the RACF SECLABEL class.
2. You authorized all users to use the security labels they will need.
3. You assigned security labels to all z/OS UNIX files and directories.
4. You activated and RACLISTed the SECLABEL class.
5. It is a good idea to re-IPL after you have assigned security labels to all users and activated the SECLABEL class to ensure that all file systems have security labels.

Requirement: The SECLABEL class must be active before you can activate the MLFSOBJ option.

Guideline: Run with MLFSOBJ(ACTIVE) set.

MLIPCOBJ

Use the MLIPCOBJ option to control whether security labels are required for interprocess communication. The MLIPCOBJ option has two suboptions, ACTIVE and INACTIVE:

- MLIPCOBJ(ACTIVE) specifies that when the SECLABEL class is active, all IPC objects must have a security label. Those that do not can only be accessed by trusted or privileged started tasks.
- MLIPCOBJ(INACTIVE) specifies that IPC objects do not require a security label.

If the SECLABEL class is active, security labels are assigned to IPC objects during object creation, and security labels are checked before access is allowed to an IPC object that has a security label. However, as long as the MLIPCOBJ option is not active, any IPC object that is running without a security label can be accessed. When you activate the MLIPCOBJ option, IPC objects running without a security label can no longer be accessed. Before you activate the MLIPCOBJ option, let your system run with the SECLABEL class active to allow the system to assign security labels to IPC objects as they are created. Run until you are sure that all active IPC objects have been created by users who have a security label. Or re-IPL to be certain that all IPC objects have security labels.

Before you activate MLIPCOBJ(ACTIVE), ensure that you have done the following tasks:

1. You defined security labels by defining profiles in the RACF SECLABEL class.
2. You authorized all users to use the security labels they will need.
3. You activated and RACLISTed the SECLABEL class.
4. You ensured that all IPC objects have security labels, by either re-IPLing after you assigned security labels to all users and activated the SECLABEL class, or running with the SECLABEL class active until you are sure that all IPC objects have security labels.

Requirement: The SECLABEL class must be active before you can activate the MLIPCOBJ option.

Guideline: Run with MLIPCOBJ(ACTIVE) set.

MLNAMES and NOMLNAMES

Use the MLNAMES and NOMLNAMES options to control whether the name-hiding function is in effect:

- The MLNAMES option specifies that the name-hiding function is active, with the following results:
  - Users cannot view the names of z/OS UNIX files and directories that their current security label does not give them authority to read.
  - Users cannot view the names of data sets that a mandatory access check followed by a discretionary access check does not allow them to read.
  - Users listing catalogs or directories cannot see the names of resources that they cannot currently read.
  - Users cannot read a VTOC directly unless they have been given authorization to the profile in the FACILITY class that protects the VTOC.
- The NOMLNAMES option specifies that the name-hiding function is not in effect. Users can view the names of files, directories, and data sets regardless of their authority to read them.

Guidelines:

- If you do not have a need to protect the names of data sets, files, and directories, run with the NOMLNAMES option set. Because the MLNAMES option can adversely affect performance, do not run with it active unless you need the protection it provides.
- If you need to protect the names of data sets, files, and directories, run with the MLNAMES option set.

MLQUIET and NOMLQUIET

Use the MLQUIET and NOMLQUIET options to control whether the system is in a tranquil state. When the MLSTABLE option is active, authorized users cannot make changes to security labels or change the security labels associated with resources until the security administrator sets the MLQUIET option.

- The MLQUIET option prevents users other than SPECIAL users, console operators, and started procedures from logging on, starting new jobs, or accessing resources. This option prevents these users from using the RACROUTE AUTH, DEFINE, and VERIFY requests.
- The NOMLQUIET option resumes normal processing.

Requirement: The SECLABEL class must be active before you can activate the MQUIET option.

Guideline: Run with NOMLQUIET active. Set the MLQUIET option temporarily when you need to change profiles in the SECLABEL class or change the SECLABEL field in profiles.

MLS and NOMLS

Use the MLS and NOMLS options to control whether users who do not have the write-down privilege can write down. The MLS option, together with the write-down privilege, helps prevent declassification of data. The MLS option has two suboptions, FAILURES and WARNING:

- MLS(FAILURES) specifies that RACF is to reject any request to write down, unless the user issuing the request has write-down mode active.

- MLS(WARNING) specifies that RACF is to issue a warning for any request to write down, and allow the request. (Exception: z/OS UNIX files and directories do not support the WARNING mode. MLS(WARNING) has the same effect as MLS(FAILURES) for z/OS UNIX files and directories.)
- NOMLS specifies that requests to write down are allowed, and users can copy data to a lower security label.

You can set MLS(WARNING) temporarily when you are setting up multilevel security to verify that you have given all users who need it the write-down privilege. Then, set MLS(FAILURES) when you are ready to prevent write-down by unauthorized users.

Before you activate MLS(FAILURES), ensure that you have done the following tasks:

1. You defined security labels by defining profiles in the RACF SECLABEL class.
2. You authorized all users to use the security labels they will need.
3. You assigned security labels to all data sets.
4. If you need to allow some users to write down, you activated the write-down by user privilege by creating the profile IRR.WRITEDOWN.BYUSER in the FACILITY class, you gave users who require the write-down privilege access to the profile, and you activated and RACLISTed the FACILITY class.
5. You activated and RACLISTed the SECLABEL class.
6. You temporarily ran with MLS(WARNING) set to verify that you completed your setup correctly.

Requirement: The SECLABEL class must be active before you can activate the MLS option.

Guideline: Run with the MLS(FAILURES) option active.

MLSTABLE and NOMLSTABLE

These options control whether authorized users can make changes to security labels or change the security labels associated with resources while the system is not quiesced:

- The MLSTABLE option prevents authorized users from doing the following actions while the system is not quiesced:
  - Changing profiles in the SECLABEL class with the RALTER command
  - Changing the SECLABEL field in profiles

    Security labels can only be changed when any possible users of the security labels are logged off and the security administrator has issued the RACF command SETROPTS MLQUIET.
- NOMLSTABLE specifies that there are no restrictions on when authorized users can change security labels.

Guideline: Run with the MLSTABLE option active.

SECLABELAUDIT and NOSECLABELAUDIT

You can specify that the SECLABEL profile's auditing options are to be used in addition to the auditing options specified for the user or the resource.

This additional auditing occurs whenever an attempt is made to access or define a resource protected by a profile, file security packet (FSP), or IPC security packet (ISP) that has a security label specified, or whenever a user running with a security label attempts to access or define a resource. If the user and resource have different security labels, auditing occurs if either security label's options specify auditing. If both security labels' options specify auditing, the auditing done is based on the options specified for the resource's security label.

For example, to specify auditing of all failed accesses to resources that have a security level of EAGLE, and all failed accesses by users that have a security label EAGLE, issue the following command:

```
RALTER SECLABEL(EAGLE) AUDIT(FAILURES(READ))
```

The SECLABELAUDIT and NOSECLABELAUDIT options determine whether RACF does the additional auditing specified on the SECLABEL profiles. You must have the RACF AUDITOR attribute to issue a SETROPTS command specifying these options. The options are:

- SECLABELAUDIT specifies that RACF is to do the additional auditing specified in the profiles in the SECLABEL class.
- NOSECLABELAUDIT specifies that RACF is not to do the additional auditing specified in the profiles in the SECLABEL class.

For more information about the SECLABELAUDIT option, see *z/OS Security Server RACF Auditor's Guide,* SA22-7684.

SECLABELCONTROL and NOSECLABELCONTROL

These options limit the users who can specify the SECLABEL operand on RACF commands that modify security labels or change the security labels associated with resources.

The SECLABELCONTROL option allows only certain users to specify the SECLABEL operand: Users with the SPECIAL attribute can specify the SECLABEL operand on any RACF command. Some examples of these commands are:

- Change a profile in the SECLABEL class with the RALTER command.
- Change the SECLABEL field of a profile.
- Issue an ADDSD, ALTDSD, or DELDSD command that causes the security label of a data set to change.

The NOSECLABELCONTROL option allows any user to change the SECLABEL field in a profile, as long as the user has at least READ access authority to the associated SECLABEL profile, unless the MLSTABLE option is in effect.

Guideline: Run with the SECLABELCONTROL option active.

SECLBYSYSTEM and NOSECLBYSYSTEM

These options control the definition of security labels in a sysplex on a system image basis. For more information about using system-specific security labels, see 2.10, “Using system-specific security labels in a sysplex” on page 25.

The SECLBYSYSTEM option specifies that security labels in a sysplex can be defined on a system image basis. When SECLBYSYSTEM is active, the SMF ID values specified in the member list of the profiles in the SECLABEL class determine whether a security label is valid for each system. Security labels that are not valid for a system are considered inactive and cannot be used or listed by users without the SPECIAL or AUDITOR attributes on that system. After activating the SECLBYSYSTEM option, you must issue a SETROPTS RACLIST(SECLABEL) REFRESH command to complete the activation of security labels by system.

The NOSECLBYSYSTEM option specifies that security labels in a sysplex are not defined on a system image basis. No distinction by system is made when defining security labels, and all security labels are valid on all systems that share the RACF database. They are unique across the whole sysplex.

Requirement: This option cannot be activated if the SECLABEL class is not active.

C

Enterprise Identity Mapping

In this appendix, we discuss Enterprise Identity Mapping (EIM).

The problem

Today's network environments are made up of a complex group of systems and applications, resulting in the need to manage multiple user registries. These user registries are intended to be used by applications to achieve user identification and authentication. The authenticated user ID is eventually used for access control as performed by the application itself, or the middleware it runs on, or by the local operating system. In today's typical heterogeneous configurations, this ends up in the situation shown in Figure C-1. In this figure, the many different platforms in the installation are represented with the local identity they have been set up with for the user John N Smith. For instance, John N Smith is known as JohnSM in the AIX 5L user registry, implying that John N Smith must be identified as JohnSM by applications running on his behalf on the AIX 5L platforms. Also in this example, applications running on behalf of John N Smith on the z/OS platform must use the SAF user ID SMITH1.

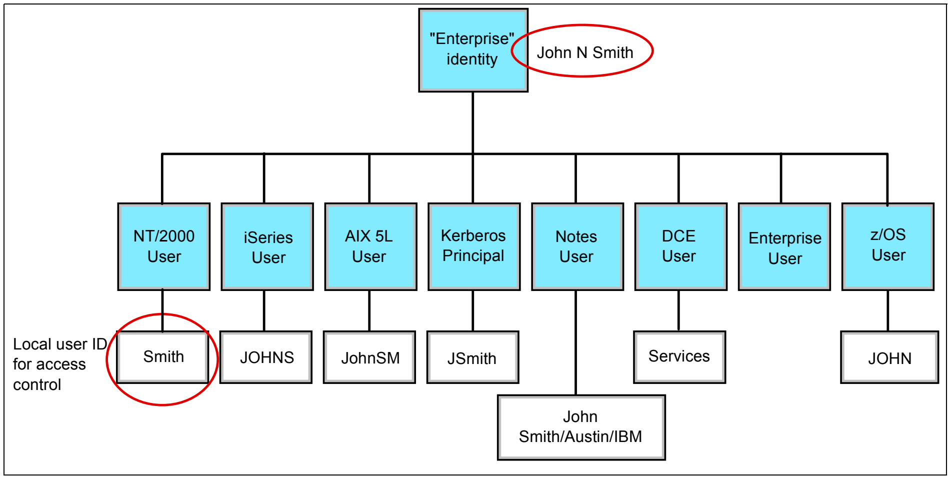


Figure C-1 A heterogeneous configuration

Typically, these multiple user registries have a design and implementation specific to each type of platform, or even application because some of them might have been implemented at the application level. They have grown over time along with the user population and the introduction of new systems and applications, yielding large administrative problems that affect users, administrators, and application developers when it comes to keeping track of a single entity and its many representations in the installation.

Systems-specific local user IDs and registries, with their underlying mechanisms, are not going to unify in a common format across all vendors, at least for many years. What seems the most natural way to proceed is for a user to authenticate to an installation using a *network identity*. It would be John N Smith in this example, and it would then be left to the involved applications to obtain from a trusted source the locally meaningful representation (that is, user ID) of this user.

Enterprise Identity Mapping was introduced as part of the IBM Autonomic Computing Initiative and is an IBM eServer infrastructure technology that allows administrators and application developers to address this problem more easily and inexpensively than previously possible. EIM enables products and applications to run in a more secure way in today's extremely open environments, as expected for the on demand operating infrastructure.

EIM accomplishes this by providing a central place to store mappings between user IDs that are defined in different registries in an installation. These mappings indicate:

- A relationship between user IDs, that is, user IDs that belong to the same person or entity within the enterprise.
- Or the mapping can represent a transformation of one or more user IDs to an application-specific user ID.

In the EIM terminology, the unique name given at an enterprise level for a user or an entity is the EIM identifier. EIM defines associations between an EIM identifier and user IDs in registries that are part of OS platforms, applications, and middleware.

Applications, typically servers, can then use an EIM API to find a mapping that transforms the installation level user ID initially used for authentication to a local user ID, which can in turn be used to access local resources.

Important: As the name implies, EIM provides identity mapping only. EIM is not to take care of the user authentication itself, which must obviously be performed ahead of identity mapping. It is left to the application deployment strategy to decide on how user authentication should be performed and how inter-application trust can be implemented. It appears today that Kerberos authentication nicely fits the authentication needs in an EIM environment.

The benefits of the EIM approach

The benefits are:

- The application server does not need to invent yet another user registry; it can use existing protocols for authenticating users and existing resource access managers to control the use of resources.
- The application server does not need to store these mappings in side files.

- The mappings can be used by more than one application.
- The mappings can span different platforms in the enterprise.
- Multi-tier applications can be written that will not ask the user for a platform-specific identity. They can get this identity without going into an identification and authentication process that can expose passwords.

Note: EIM is often mistaken for a user ID management solution similar to products such as Tivoli Identity Manager. An identity management solution is able to work with all aspects of a user ID, passwords, and user authorities. EIM only contains the names of user IDs and none of the other attributes. It has insufficient information for authentication or authorization.

Recent enhancements to EIM

Changes to the EIM domain can be logged to SMF. Lookup operations can be logged, which was introduced in z/OS V1R7. The RACF SMF unload utility can process EIM events into a tabular format or an XML document.

The EIM implementation concepts

A simplified view of the EIM conceptual implementation is shown in Figure C-2 on page 332. In this figure, the installation is composed of a z/OS, two OS/400®, and one AIX 5L system. The user ID mapping is kept in an LDAP directory called an EIM domain controller accessible by remote EIM clients (all these terms are explained in more detail later). In this example, John Smith (of whose EIM identifier is John N Smith) is running a client application on his workstation that is eventually to send a request to a server running in SYSA. Both the client application and server use Kerberos authentication.

The process is:

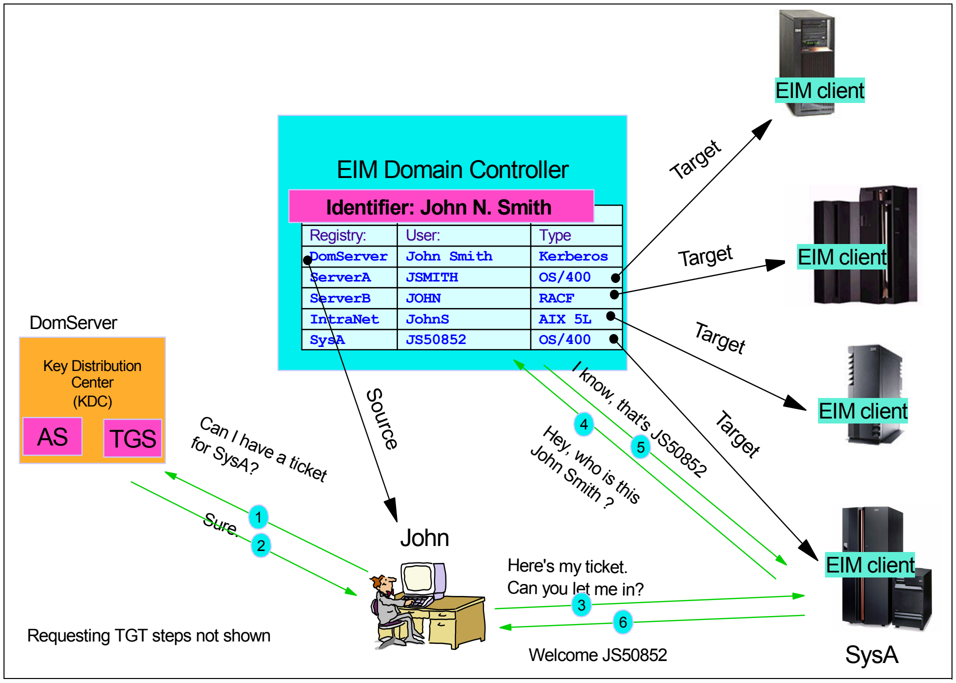
1. John Smith authenticated to the Kerberos authentication server (not shown in the picture) and now requests a service ticket to the Kerberos ticket granting server to get access to the server on SYSA.
2. The Kerberos service ticket is delivered to the client application and contains John Smith's Kerberos principal name, which for this example is John Smith@DomServer.

3. John Smith’s client application requests a service on SYSA, using the Kerberos service ticket to authenticate. The Kerberos-enabled server on SYSA decrypts and parses the service ticket and extracts John Smith’s principal name, John Smith, which is expected to be meaningless to the local OS/400 user registry because it has not been implemented with Kerberos semantics. Note that the successful ticket parsing process is also a proof of proper client authentication by the Kerberos authentication server.
4. The server application on SYSA then acts as an EIM client, using the local EIM API, and requests the EIM domain controller to map the principal name John Smith from the source DomServer to a user ID in the target SYSA registry.
5. The EIM domain controller looks up its directory and finds the mapping information for the specified source and target identities. In this example, the principal name John Smith returns JS50852 for John N Smith’s SYSA local identity.

   Note: The EIM domain controller proceeds by first mapping the source identity, John Smith in this example, to the EIM identifier, John N Smith, and then maps the EIM identifier to the corresponding user ID in the target registry, that is, SYSA’s registry.

6. The server application on SYSA then acknowledges the John Smith authentication and identity mapping to the client application. It is expected that the SYSA server application is now to run on behalf of user JS50852.

In this model, if the request was passed on to another tier for additional processing, SYSA would pass it with the initially authenticated identity, that is, John Smith from DomServer to the other system, which in turn would invoke the EIM domain controller for a local mapping.


EIM Domain Controller
Identifier: John N. Smith

Registry:	User:	Type
DomServer	John Smith	Kerberos
ServerA	JSMITH	OS/400
ServerB	JOHN	RACF
IntraNet	JohnS	AIX 5L
SysA	JS50852	OS/400

EIM client
Target
EIM client
Target
Target
EIM client
Target
EIM client
SysA
DomServer
Key Distribution Center (KDC)
AS
TGS
Source
Can I have a ticket for SysA?
1
Sure.
2
John
Hey, who is this John Smith ?
4
I know, that's JS50852
5
Here's my ticket. Can you let me in?
3
6
Welcome JS50852
Requesting TGT steps not shown

Figure C-2 The EIM conceptual implementation

EIM components

The basic components of EIM are:

- The EIM domain controller: The domain controller is an IBM Tivoli Directory Server or IBM z/OS Integrated Security Services LDAP Server that contains one or more EIM domains. These can be:
  - The OS/400 V5R2 LDAP server on iSeries™
  - The z/OS V1R4 LDAP server on zSeries
  - The IBM Directory Services on all platforms, including AIX 5L on pSeries®, Windows 2000 on xSeries®, and Linux

  The information in the domain controller is created and maintained by an administrative utility:
  - OS/400 V5R2 iSeries Navigator
  - z/OS V1R4 with SPE OW57137 and later
  - AIX 5L V5R3

- EIM domain: The domain contains the mappings for an enterprise. It is located by the URL for the LDAP server. The name of the domain must be descriptive of the enterprise. An example of an EIM domain's URL is:

  ```
  ldap://some.host/ibm-eimDomainName=My Business,c=fr
  ```

- The EIM client: This is the code that implements the EIM lookup APIs and the administrative APIs used for creating, modifying, displaying, and removing information from an EIM domain.
  The EIM client uses the EIM API that is provided with the eServer's operating systems or that are downloadable from the Internet. As of the writing of this book, the EIM client APIs are available beginning in C/C++ or Java with these minimum systems levels:

  - OS/400 V5R2
  - z/OS V1R4 with SPE OW57137
  - AIX 5L R5V2
  - Microsoft Windows 2000, with the IBM Directory V4.1 client
  - Linux SLES8 on PPC64, Red Hat 7.3 on i386™, or SLES7 on zSeries; with the IBM Directory V4.1 client or OpenLDAP v2.0.23 client
    EIM applications no longer require APF authorization as of z/OS V1R7 or later.

  Note that the Windows and Linux clients must be downloaded from:

  http://www.ibm.com/servers/eserver/security/eim/availability.html

- The EIM identifier: This is the name that represents the unique name of an individual or entity within the enterprise. It can be a name or number or some combination of the two that represents a person or entity in your company. The EIM identifier is the anchor point for all mappings.

- The EIM registry: This is the logical representation of a user registry that exists on one of the systems in the network. Examples of user registries are RACF (or equivalent), LDAP (which contains bind IDs and passwords), and Lotus Notes®. An EIM registry only contains user IDs and EIM information needed for mappings with EIM identifiers. It does not contain any of the other information, such as passwords or user attributes, that normally exist in a user registry.

- The EIM associations: These are the relationships between a user ID and an EIM identifier. There are three kinds of associations:

  - Source association: The registry user ID can be used as a source, that is, a starting point, for a lookup operation. Note that it is assumed here that the source identity has been properly authenticated by the application.

  - Target association: The registry user ID can be returned as the target value for a lookup operation. It is expected that this user ID is intended to be used for access control by the application.

    The EIM domain controller also provides fields for application-specific information that can be used to refine the lookup results.

  - Administrative association: The user ID is kept in the EIM domain controller for administration purposes only. However, it is not recognized as a source or target user ID in a mapping lookup. In addition, it is not returned as a result of a mapping lookup.

  User IDs can be defined with both source and target and, if desired, with administrative associations.

- EIM policies: These are a special kind of association that can be used by an EIM lookup operation. If the EIM lookup API cannot locate a specific EIM association, the API looks for a policy. The policy acts as a default or many-to-one mapping between a source registry and a target registry. There are three kinds of policies:

  - A certificate filter policy associated with an x.509 registry
  - A registry policy
  - A domain policy

  Policies have been implemented with z/OS V1.6.

- EIM lookup operations: They retrieve a mapping from the specified EIM domain. There are three kinds of lookups:

  - eimGetTargetFromSource returns a user ID in the target registry when a source registry and source user ID is provided.
  - eimGetTargetFromidentifier returns a user ID in the target registry when only the EIM identifier is provided.
  - eimGetAssociatedIdentifier returns the EIM identifier that has an association with the given registry and user ID.

Miscellaneous additional information

For more details about EIM concepts, see the IBM eServer Information Center:

http://www.ibm.com/servers/eserver/security/eim/

An EIM domain controller can run on any platform where IBM Tivoli Directory Server can be installed. To get a complete list of those platforms, go to http://www.ibm.com and search for IBM Tivoli Directory Server. The z/OS Integrated Security Services LDAP Server can also serve as an EIM domain controller.

An application that uses EIM to locate mappings needs access to the EIM client. Depending on the platform, it is either part of the operating system or available from IBM as a download. There are two kinds of EIM clients: one is written in

C/C++ and the other is written in Java. The IBM eServer Information Center has the latest information about supported platforms and releases.

The EIM domain controller

An EIM controller uses an LDAP directory server that supports the LDAP (Version 3) protocol. It must also understand the following attributes:

- ibm-entryUUID attribute
- ibmattributetypes: aclEntry, aclPropagate, aclSource, entryOwner, ownerPropagate, ownerSource
- New attribute types and object classes for EIM (provided through schema updates)

The z/OS V1R4 Security Server LDAP, with APAR OW55078, is the initial z/OS level that can host an EIM domain controller using the TDBM back end, that is, the domain controller data resides in DB2 tables managed by LDAP. The SDBM back end, giving controlled access to the RACF database, is not required. New EIM functions may require new levels of LDAP EIM schemas to be installed.

The ibm-entryuuid attribute

The UUID concept is inherited from Distributed Computing Environment (DCE) and stands for Universal Unique Identifier. This is a unique identifier generated by the LDAP server for any entry that is created or modified and does not already have a unique identifier assigned. The unique identifier is stored in the ibm-entryuuid attribute. The ibm-entryuuid attribute is replicated to servers that support the ibm-entryuuid attribute. A utility, **ldapassuuids**, is provided to create the ibm-entryuuids for existing entries when migrating from previous releases.

For uniqueness, the z/OS LDAP server can be set up to use a local network adapter MAC address as a seed to generate the UUIDs. For more information, see the z/OS LDAP reference book *z/OS Integrated Security Services LDAP Server Administration and Use*, SC24-5923.

Overview of EIM interactions

Figure C-3 on page 336 shows these interactions.

The EIM administrator builds and manages the directory trees representing EIM domains, using the **eimadmin** utility, or having a program do it by calling the EIM client administrative API. Note that we strongly recommend letting the **eimadmin** utility, or the administrative API, perform this management task and not to try using direct LDAP operations such as **ldapadd**, **ldapdelete**, and so on.

The EIM domain controller LDAP directory has been set up with the EIM schemas.

An EIM-enabled application uses the EIM client API to interrogate the EIM domain controller. The EIM API converts the EIM function into LDAP requests that are transparently issued by the LDAP client. Notice the local registry name, as defined by the EIM and client platform administrators and as recorded in the domain controller, is cached in z/OS and made available to the client API.

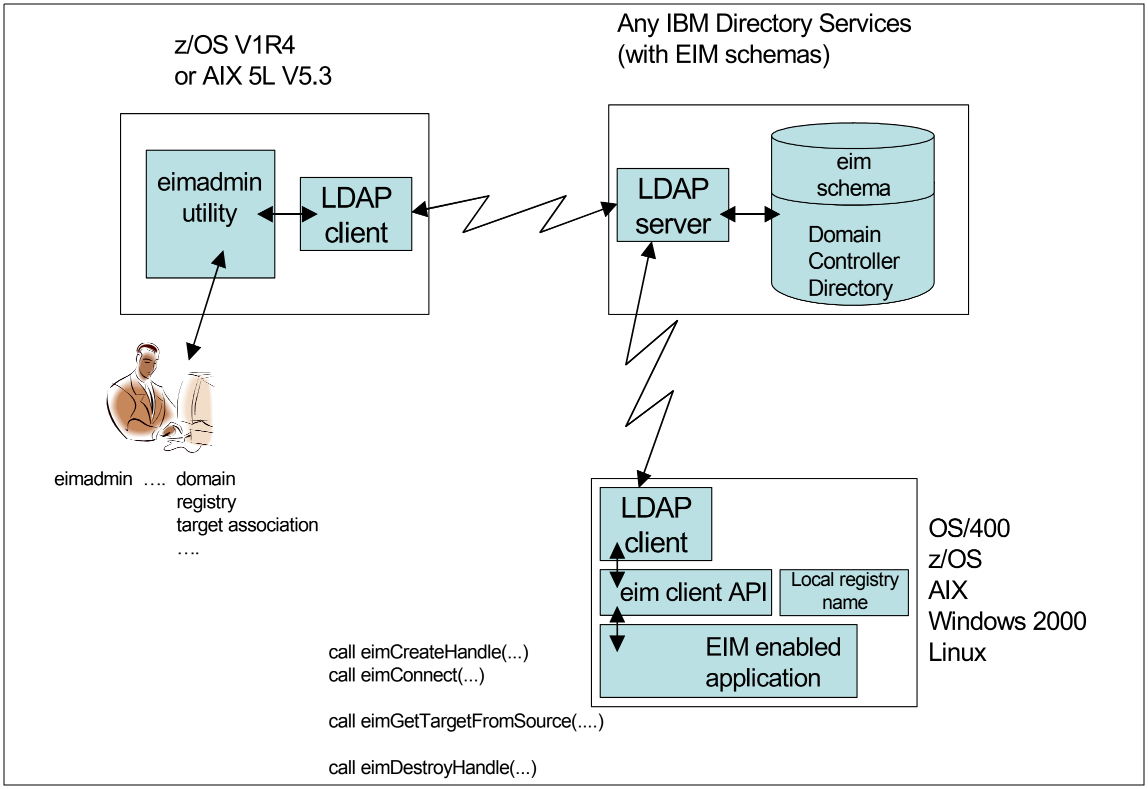


Figure C-3 EIM infrastructure and interactions overview

The EIM APIs

There are two groups of APIs: The administration API that is used by **eimadmin** or a client program and the EIM lookup API, which implements the mapping functions.

Administrative APIs

The administrative APIs are:

- Create, delete, change, and list a domain
- Create and destroy connections with a domain
- Add, delete, change, and list identifiers
- Add, delete, change, and list system and application registries
- Add, remove, and list associations
- Change and list registry users
- Add, remove, query, and list access to domains, identifiers, registries, and users

Runtime lookup APIs

The runtime lookup APIs are:

- Create and destroy connections with a domain
- eimGetTargetFromSource
- eimGetTargetFromIdentifier
- eimGetAssociatedIdentifiers

Content of the EIM domain controller

The following objects are represented by entries in the domain controller:

- Domain
- Identifiers
- Registries
- Associations
- Policies

The directory tree

Figure C-4 on page 338 shows an example of a directory tree for the domain MopBooks. The subtrees to the EIM domain entry contain:

- EIM identifiers: Remember that a source user ID is pointing to an EIM identifier, and therefore all existing source associations are represented at the EIM identifier level.
- Registries: These are the representations of the EIM identifier local to a system or application. The lookup target associations are represented at the registry level.
- Groups: These are EIM access control groups, which are further described in "Access control to EIM objects" on page 339.

Note also the heavy use of the UUIDs to establish cross references between entries.

Note: Again, these directory entries are built and maintained through the **`eimadmin`** utility or the administration API. Do not try to manage directly using your own LDAP operations.

Note that the **`eimadmin`** facility accepts inputs from file. Users can consider building this file from a RACF DBUNLOAD for mass creation of user IDs mappings in an EIM domain controller.

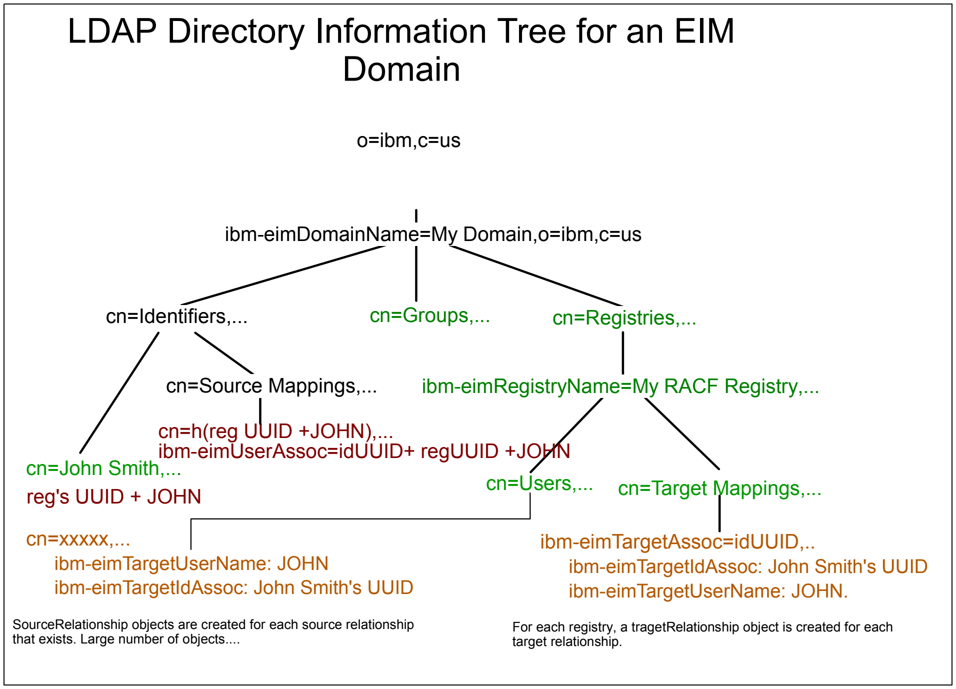

LDAP Directory Information Tree for an EIM Domain
o=ibm,c=us
ibm-eimDomainName=My Domain,o=ibm,c=us
cn=Identifiers,...
cn=Groups,...
cn=Registries,...
cn=Source Mappings,...
ibm-eimRegistryName=My RACF Registry,...
cn=h(reg UUID +JOHN),...
ibm-eimUserAssoc=idUUID+ regUUID +JOHN
cn=John Smith,...
reg's UUID + JOHN
cn=Users,...
cn=Target Mappings,...
cn=xxxxx,...
ibm-eimTargetUserName: JOHN
ibm-eimTargetIdAssoc: John Smith's UUID
ibm-eimTargetAssoc=idUUID,..
ibm-eimTargetIdAssoc: John Smith's UUID
ibm-eimTargetUserName: JOHN.
SourceRelationship objects are created for each source relationship that exists. Large number of objects....
For each registry, a tragetRelationship object is created for each target relationship.

Figure C-4 An example of an EIM directory tree

Access controls to the EIM domain controller and its contents

The access to the EIM domain controller and its contents is based on the accessing user's distinguished name and its proper authentication.

User authentication on z/OS LDAP

A user accessing an EIM domain controller running on z/OS can authenticate with:

- The LDAP simple bind. The bind distinguished name and the bind password flow over the TCP/IP session in clear text or protected by SSL encryption.

- Alternatively, the simple bind can use a password protected by the CRAM-MD5 hashing.
- A Kerberos service ticket, with the Kerberos principal name of the client. The Kerberos protocol itself ensures proper authentication of the client.
- A client digital certificate, over an SSL connection, that contains the client distinguished name. The SSL protocol itself ensures proper authentication of the client.

Note: Much information, such as the LDAP bind and authentication data and the domain to be used, can be entered for security and information management reasons in RACF profiles used when connecting to the EIM domain controller. We explain this in “Using RACF profiles to keep EIM default parameters” on page 346.

Access control to EIM objects

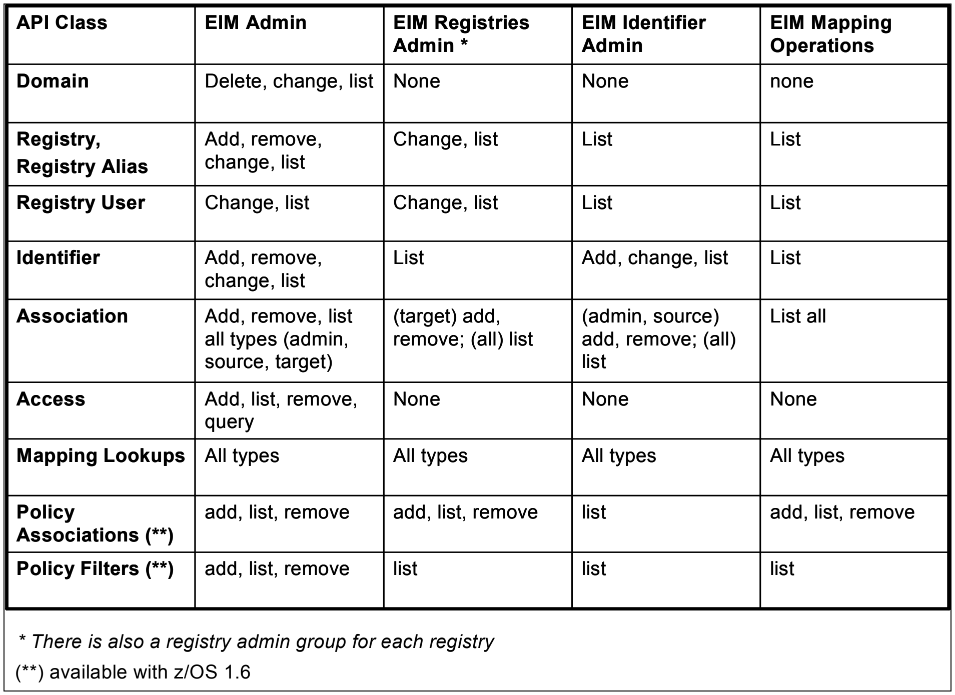
The LDAP objects in the EIM domain controller are subject to access controls by LDAP ACLs that grant access to the authenticated user according to the EIM access control group of which the user is a member. The predefined access control groups are established at the EIM domain level and are, as of the writing of this book:

- EIM administrator: This access control group allows the user to manage all of the EIM data within this EIM domain.
- EIM registries administrator or registry x administrator: This access control group allows the user to manage all EIM registries, or only one specific registry, definitions, and target associations.
- EIM identifier administrator: This access control group allows the user to add and change EIM identifiers and manage source and administrative associations.
- EIM mapping operations: This access control group allows users to conduct EIM lookup operations. A user with this access control can perform the following functions:
  - Perform EIM lookup operations.
  - Retrieve associations, EIM identifiers, and EIM registry definitions.

The LDAP administrator has full access to any object in the directory, and only an LDAP administrator can create a new domain.

Users are assigned to access control groups by the eimAddAccess API call, or by the **`eimadmin`** utility with the **`-C`** parameter.

A user invoking the EIM API is authorized to specific functions, as shown in Figure C-5.



API Class	EIM Admin	EIM Registries Admin *	EIM Identifier Admin	EIM Mapping Operations
Domain	Delete, change, list	None	None	none
Registry, **Registry Alias**	Add, remove, change, list	Change, list	List	List
Registry User	Change, list	Change, list	List	List
Identifier	Add, remove, change, list	List	Add, change, list	List
Association	Add, remove, list all types (admin, source, target)	(target) add, remove; (all) list	(admin, source) add, remove; (all) list	List all
Access	Add, list, remove, query	None	None	None
Mapping Lookups	All types	All types	All types	All types
Policy Associations (**)	add, list, remove	add, list, remove	list	add, list, remove
Policy Filters (**)	add, list, remove	list	list	list

** There is also a registry admin group for each registry*

(**) available with z/OS 1.6

Figure C-5 EIM access controls

Setting up the LDAP directory to act as an EIM domain controller

In this section, we discuss setting up the LDAP directory to act as an EIM domain controller.

LDAP configurations

The EIM domain controller can take advantage of the usual LDAP configurations, aiming at enhancing availability or simplifying the name space management. These are:

- LDAP server with replicas: The replica is another usable instance of the master directory. The master-replica configuration can be implemented at z/OS; beginning with z/OS V1.6, you can also use a peer-to-peer replication configuration.
- LDAP directory with referrals: A referral is a pointer in a directory to another directory so that the naming space can split across several LDAP servers. LDAP V3 referrals can be used with the z/OS LDAP server.

- To achieve very high availability, z/OS also enables you to configure a single directory to be served by multiple LDAP servers located in members of a sysplex.

Note: The choice of LDAP servers to host an EIM domain controller is dictated by the availability of the following required attributes:

- ibm-entryUUID
- aclEntry
- aclPropagate
- aclSource
- entryOwner
- entryPropagate
- entrySource

EIM schemas

In z/OS, the EIM LDAP object classes and attributes definitions are delivered in the schema.IBM.ldif file in /usr/lpp/ldap/etc/. They are, as of the writing of this book:

- Objectclasses
  - ibm-eimDomain
  - ibm-eimIdentifier
  - ibm-eimRegistry
  - ibm-eimSystemRegistry
  - ibm-eimApplicationRegistry
  - ibm-eimRegistryUser
  - ibm-eimSourceRelationship
  - ibm-eimTargetRelationship
  - ibm-eimDefaultPolicy
  - ibm-eimDomainName
  - ibm-eimFilterPolicy
  - ibm-eimPolicyListAux
- Attributes
  - ibm-eimDomainName
  - ibm-eimAdditionalInformation
  - ibm-eimAdminUserAssoc
  - ibm-eimDomainVersion
  - ibm-eimRegistryAliases
  - ibm-eimRegistryEntryName
  - ibm-eimRegistryName
  - ibm-eimRegistryType
  - ibm-eimSourceUserAssoc
  - ibm-eimTargetIdAssoc

- ibm-eimTargetUserName
- ibm-eimUserAssoc
- ibm-eimFilterType
- ibm-eimFilterValue
- ibm-eimPolicyStatus

EIM administration tools

The EIM domain controller administration can be performed by programs using the administration API, or can be done from the z/OS UNIX console with the **eimadmin** line command. Figure C-6 provides an example of the **eimadmin** utility, and Figure C-7 on page 343 shows a synthetic view of keywords and parameters.

```
PATKAP: /u/patkap>eimadmin -lD -d "ibm-eimdomainname=MopBooks,o=moppssc,c=fr" -h
            ldap://10.11.12.13:389 -b cn=ldapadmin -w secret
                   domain name: MopBooks
                     domain DN: ibm-eimdomainname=MopBooks,o=moppssc,c=fr
                      policies: DISABLED
```

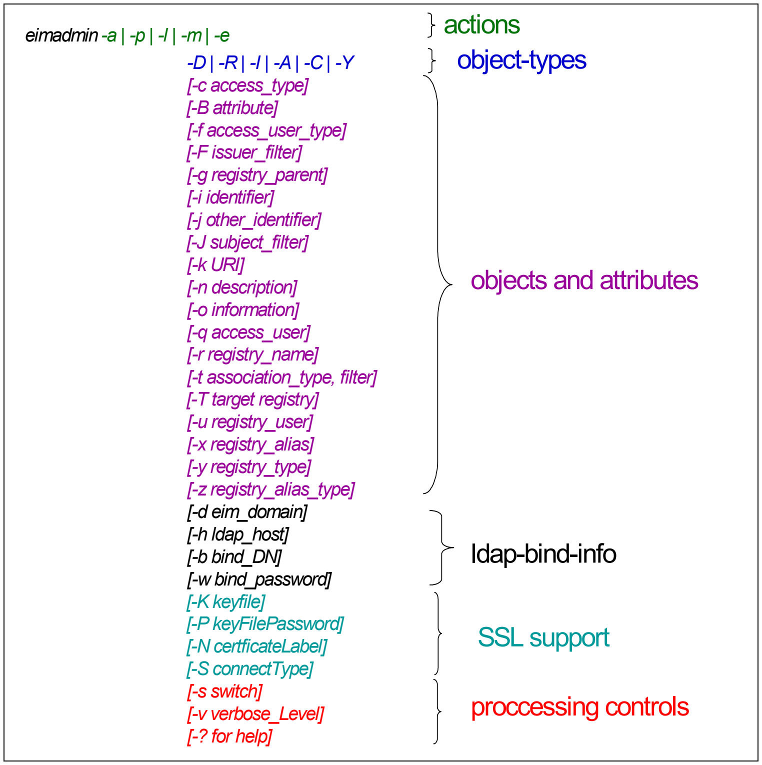
Figure C-6 Example of eimadmin use

The following actions can be invoked using **eimadmin**:

- Add an object.
- Delete an object.
- List objects (for example, list directories, list registries, and so forth).
- Modify attributes associated with objects.
- Erase attributes.

The following objects can be administered:

- Domains
- Registries
- Identifiers
- Associations
- Access authorities
- Policies



```
eimadmin -a | -p | -l | -m | -e
                    -D | -R | -I | -A | -C | -Y
                    [-c access_type]
                    [-B attribute]
                    [-f access_user_type]
                    [-F issuer_filter]
                    [-g registry_parent]
                    [-i identifier]
                    [-j other_identifier]
                    [-J subject_filter]
                    [-k URI]
                    [-n description]
                    [-o information]
                    [-q access_user]
                    [-r registry_name]
                    [-t association_type, filter]
                    [-T target registry]
                    [-u registry_user]
                    [-x registry_alias]
                    [-y registry_type]
                    [-z registry_alias_type]
                    [-d eim_domain]
                    [-h ldap_host]
                    [-b bind_DN]
                    [-w bind_password]
                    [-K keyfile]
                    [-P keyFilePassword]
                    [-N certficateLabel]
                    [-S connectType]
                    [-s switch]
                    [-v verbose_Level]
                    [-? for help]
```

Figure C-7 eimadmin utility and parameters

Note that the **eimadmin** utility can use an input file containing several actions that will all be performed on the single invocation of the utility.

An interesting attribute is **-y**, which is the registry type. As of the writing of this book, the **-y** attribute can have the following predefined values:

- RACF
- OS400
- KERBEROS (for case ignore)
- KERBEROSX (for case exact)
- AIX
- NDS
- LDAP
- PD (Policy Director)
- WIN2K
- X509
- LINUX
- DOMINOS
- DOMINOL

Refer to the “EIM registry definition” topic in *z/OS Integrated Security Services Enterprise Identity Mapping (EIM) Guide and Reference*, SA22-7875, to understand what it takes to create your own EIM registry type.

The EIM client

EIM applications on z/OS must be APF authorized. Requiring APF authorization prevents inadvertent or malicious use of EIM APIs to change information in an EIM domain or to extract unauthorized information, which means that you must set the APF authorization extended attribute for each EIM application program residing in HFS files (EIM applications no longer require APF-authorization as of z/OS V1R7 or later). This attribute is set by using the **`extattr`** command.

> **Important:** As mentioned before, an EIM client performs lookup operations within a specific EIM domain to retrieve a user ID to be presumably used for access control. We also explained that EIM does not do user or entity authentication. The assumption is, therefore, that all lookup parameters are provided by trusted applications, which properly proceeded with user or entity authentication whenever it was necessary.

An application can perform one of two types of EIM lookup operations based on the type of information the application supplies as the source of the EIM lookup operation: a user identity or an EIM identifier.

The client application can use these calls, provided its identity is in the lookup access control group:

- eimGetTargetFromSource
- eimGetTargetFromIdentifier
- eimGetAssociatedIdentifiers

If successful, the lookup operation returns either an EIM identifier or a user identifier in the specified registry.

The application supplies a user identity as a source

The application must also supply:

- The EIM registry definition name for the source user identity
- The EIM registry definition name that is the target of the EIM lookup operation

To be used as the source in an EIM lookup operation, a user identity must have a source association defined for it.

The application supplies an EIM identifier as a source

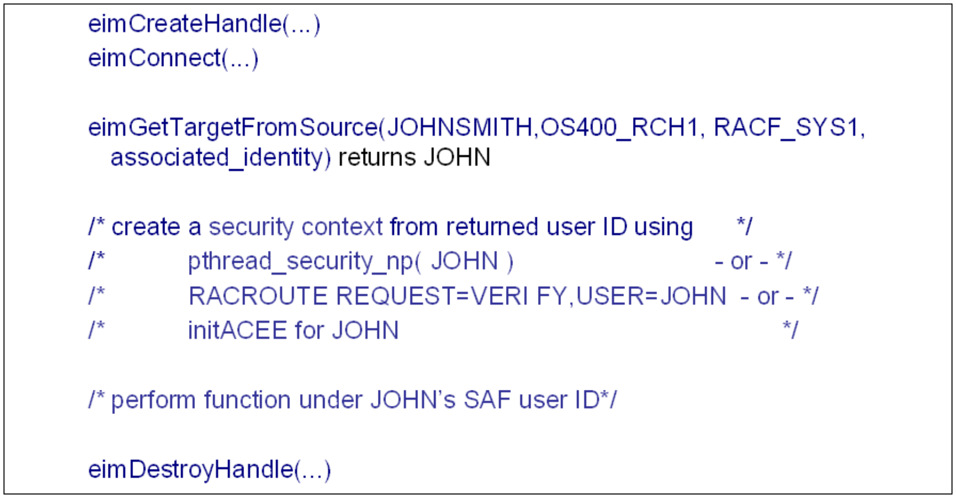
The application must also supply the EIM registry definition name that is the target of the EIM lookup operation.

For a user identity to be returned as the target of either type of EIM lookup operation, the user identity must have a target association defined for it.

The supplied information is passed to the EIM lookup operation that searches the EIM domain controller for the source association that matches the supplied information. Based on the EIM identifier (supplied to the API or determined from the source association information), the EIM lookup operation then searches for a target association for that identifier that matches the target EIM registry definition name, and provides application-specific information if any were defined.

Typically, the client application proceeds with a sequence of actions, as shown in Figure C-8 on page 346. In this example, the application is executed on z/OS and has received a request from an OS400 system.

1. The eimCreateHandle() call allocates an EimHandle structure, which identifies the EIM connection during its lifetime and maintains per-connection information. The EimHandle structure is passed on subsequent calls to other EIM operations. When the ldapURL parameter is NULL, EIM attempts to read the domain name, LDAP URL, and bind DN from RACF profiles, as described in “Using RACF profiles to keep EIM default parameters” on page 346.
2. The eimConnect() call performs the bind to the LDAP domain controller and establishes the search tree base at the specified EIM domain. Connection information provided in the call specifies what kind of authentication is to be performed for the LDAP bind and whether the communication is protected with SSL. If the simple bind credentials in the eimConnectInfo structure are null, and connection information was obtained from RACF on the eimCreateHandle, a call to the R_DCEKEY callable service is made to decrypt the LDAP bind password stored in RACF.
3. The eimtargetgetFromSource() function in this example requests a mapping from identity JOHNSMITH in registry OS400_RCH1 to the corresponding identity in registry RACF_SYS1.
4. After the z/OS local identity has been obtained, it can be used to be attached to the task, process, or thread serving the request (assuming that the client application has the proper privilege to do so).
5. When done, the eimDestroyHandle() function frees resources associated with the EimHandle and closes connections to the EIM domain controllers.



```
eimCreateHandle(...)
eimConnect(...)

eimGetTargetFromSource(JOHNSMITH,OS400_RCH1, RACF_SYS1,
  associated_identity) returns JOHN

/* create a security context from returned user ID using      */
/*         pthread_security_np( JOHN )                   - or - */
/*         RACROUTE REQUEST=VERI FY,USER=JOHN  - or - */
/*         initACEE for JOHN                                    */

/* perform function under JOHN's SAF user ID*/

eimDestroyHandle(...)
```

Figure C-8 Typical client application

Using RACF profiles to keep EIM default parameters

In this section, we review using RACF profiles to keep EIM default parameters.

LDAP and domain bind information

Several profiles can be used to store in RACF an LDAP URL, the bind distinguished name, and the bind password, which can then be retrieved by the EIM services. The choice of the profile is dictated by the intent to either establish a system default that is used for EIM only or that can be used by other functions (such as z/OS PDAS or PKI Services), or to establish a default setup for a specific application user ID. These profiles are searched in the following order, as shown in Figure C-9 on page 347, by EIM services that have been invoked without specifying the domain controller or domain in their parameter list:

1. Application user ID-specific default

   The user-specific default is in a profile in the LDAPBIND class, with an arbitrary name that must also appear in the EIM segment of the USER profile. An example of definition of this profile is:

   ```
   RDEFINE LDAPBIND BUCKSDOMAIN +
   EIM(DOMAINDN('ibm-eimDomain=Bucks Domain.o=ibm.c=us')) +
   OPTIONS(ENABLE)) +
   PROXY(LDAPHOST(ldap://another.big.host) +
   BINDDN('cn=EIM Application Lookups') BINDPW('secret'))
   ```

   To establish this profile as a default for a specific user SERVERID, the EIM segment in the USER profile can be updated as follows:

   ```
   ALTUSER SERVERID EIM(LDAPPROF(BUCKSDOMAIN))
   ```

2. System-wide default

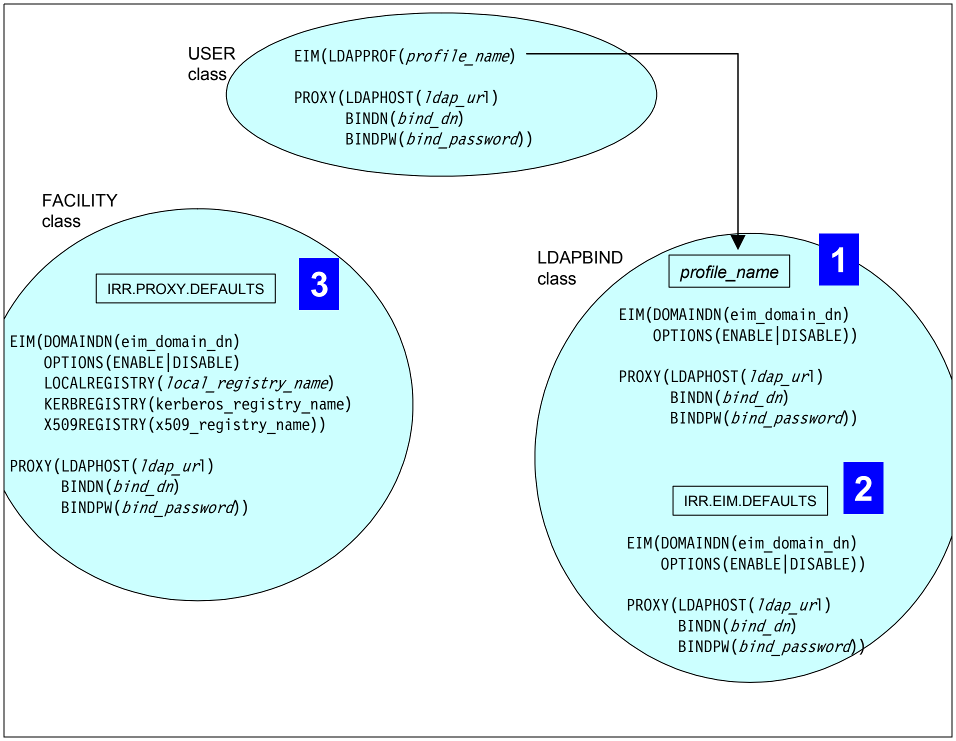
   The IRR.EIM.DEFAULTS profile in the LDAPBIND class establishes a default. An example of a definition of this profile is:

   ```
   RDEFINE LDAPBIND IRR.EIM.DEFAULTS +
   EIM(DOMAINDN('ibm-eimDomain=Joes Domain.o=ibm.c=us')) +
   OPTIONS(ENABLE)) +
   PROXY(LDAPHOST(ldap://some.big.host) +
   BINDDN('cn=EIM Lookup') BINDPW('secret'))T
   ```

3. System-wide default

   IRR.PROXY.DEFAULTS profile in the FACILITY class. An example of a definition of this profile is:

   ```
   RDEFINE FACILITY IRR.PROXY.DEFAULTS +
   EIM(DOMAINDN('ibm-eimDomain=Joes Domain.o=ibm.c=us')) +
   OPTIONS(ENABLE)) +
   PROXY(LDAPHOST(ldap://some.big.host) +
   BINDDN('cn=EIM Lookup') BINDPW('secret'))
   ```



```
EIM(LDAPPROF(profile_name)

PROXY(LDAPHOST(ldap_url)
      BINDN(bind_dn)
      BINDPW(bind_password))
```

```
EIM(DOMAINDN(eim_domain_dn)
    OPTIONS(ENABLE|DISABLE)
    LOCALREGISTRY(local_registry_name)
    KERBREGISTRY(kerberos_registry_name)
    X509REGISTRY(x509_registry_name))

PROXY(LDAPHOST(ldap_url)
      BINDN(bind_dn)
      BINDPW(bind_password))
```

```
EIM(DOMAINDN(eim_domain_dn)
    OPTIONS(ENABLE|DISABLE))

PROXY(LDAPHOST(ldap_url)
      BINDN(bind_dn)
      BINDPW(bind_password))
```

```
EIM(DOMAINDN(eim_domain_dn)
    OPTIONS(ENABLE|DISABLE))

PROXY(LDAPHOST(ldap_url)
      BINDN(bind_dn)
      BINDPW(bind_password))
```

Figure C-9 Keeping EIM LDAP and domain information in RACF profiles

The EIM and PROXY segment keywords and subkeywords combine to define the EIM domain, the LDAP host it resides on, and the bind information required by the EIM services to establish a connection with an EIM domain. The EIM services attempt to retrieve this information when it is not explicitly supplied through the invocation parameters.

Managing the local registry name

We assume here that the registry name is also specified in the EIM domain controller, presumably by using the **`eimadmin`** utility.

The local registry name can be set in the FACILITY IRR.PROXY.DEFAULTS profile by the RACF administrator for retrieval by the following functions, when the local registry is not indicated in the parameter list:

- eimGetTargetFromSource
- eimGetIdentifierFromSource
- eimGetAssociatedIdentifiers

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see “How to get IBM Redbooks” on page 350. Note that some of the documents referenced here may be available in softcopy only.

- *DB2 UDB for z/OS Version 8 Performance Topics*, SG24-6465
- *Rational Application Developer V6 Programming Guide*, SG24-6449
- *WebSphere Application Server V6.1: System Management and Configuration*, SG24-7304
- *z/OS Distributed File Service zSeries File System Implementation z/OS V1R7*, SG24-6580

Other publications

These publications are also relevant as further information sources:

- *DB2 UDB for z/OS RACF Access Control Module Guide*, SC18-7433
- *DB2 Universal Database for z/OS Administration Guide*, SC18-7413
- *DFSMS/MVS DFSMSdfp Storage Administration Reference*, SC26-4920
- *Print Services Facility for z/OS Security Guide*, S544-3291
- *z/OS Communications Server: IP Configuration Guide*, SC31-8775
- *z/OS Communications Server: IP Configuration Reference*, SC31-8776
- *z/OS DFSMS Object Access Method Planning, Installation, and Storage Administration Guide for Object Support*, SC35-0426
- *z/OS DFSMS Object Access Method Planning, Installation, and Storage Administration Guide for Tape Libraries*, SC35-0427
- *z/OS DFSMSdss Storage Administration Guide*, SC35-0423
- *z/OS DFSMShsm Implementation and Customization Guide*, SC35-0418

- *z/OS Integrated Security Services Enterprise Identity Mapping (EIM) Guide and Reference*, SA22-7875
- *z/OS Integrated Security Services LDAP Server Administration and Use*, SC24-5923
- *z/OS JES2 Initialization and Tuning Guide*, SA22-7532
- *z/OS Planning for Multilevel Security and the Common Criteria*, GA22-7509
- *z/OS Security Server RACF Auditor's Guide,* SA22-7684
- *z/OS TSO/E Customization*, SA22-7783

Online resources

These Web sites are also relevant as further information sources:

- Enterprise Entity Mapping

  `http://www.ibm.com/servers/eserver/security/eim/availability.html`
- Common Criteria Security Certification

  `http://www.ibm.com/systems/z/security/ccs_certification.html`
- Security Evaluations for IBM Products

  `http://www.ibm.com/security/standards/st_evaluations.shtml`
- National Security Agency and the Central Security Service

  `http://www.nsa.gov/ia/industry/niap.cfm`
- Consolidated Service Test and the RSU

  `http://www.ibm.com/servers/eserver/zseries/zos/servicetst/mission.html`

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

Securing DB2 and Implementing MLS on z/OS

IBM®

Securing DB2 and Implementing MLS on z/OS

Redbooks

DB2 V9 trusted context and roles

Multilevel security: What, why, and how

Vanguard Administrator: Simplifying MLS implementations

Today's computing environment is subject to increasing regulatory pressures and potentially malicious attacks. Regulatory compliance, security, and audit are in the daily headlines and growing more prominent.The security of the information to which you have been entrusted has never been more critical. The reality of compliance is too complex. Compliance demands that you work carefully to set up a strong, comprehensive set of policies and controls. That means controls that consider operational data, financial data, unstructured data, spreadsheets, e-mail, and business intelligence data.

We have a responsibility to secure all business data and especially sensitive customer data. Security can be difficult to manage. IBM DB2 for z/OS already resides on one of the most secure platforms in the industry. IBM System z servers are routinely used by enterprises around the world to support their mission-critical applications. The mainframe's strengths in security stem in part from its history of supporting sensitive data for large enterprises, resulting in security features being built into its design for many decades. It also benefits from a system-wide approach with security capabilities built into the hardware, operating systems, databases, key middleware and more. Its highly evolved layers and security management components give it a fundamental advantage over other systems.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6480-01 ISBN 0738486116