



IBM Informix Security Guide



IBM Informix Security Guide

Note

Before using this information and the product it supports, read the information in "Notices" on page B-1.

This edition replaces SC23-7754-01.

This document contains proprietary information of IBM. It is provided under a license agreement and is protected by copyright law. The information contained in this publication does not include any product warranties, and any statements provided in this manual should not be interpreted as such.

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

© **Copyright International Business Machines Corporation 1996, 2008.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Introduction	vii
About This Publication	vii
Types of Users	vii
Prerequisites for a Secure Database Server	vii
What's New for Security in Dynamic Server, Version 11.50	ix
Documentation Conventions	x
Typographical Conventions	x
Feature, Product, and Platform Markup	x
Example Code Conventions	xi
Additional Documentation	xi
Compliance with Industry Standards	xi
Syntax Diagrams	xii
How to Read a Command-Line Syntax Diagram	xiii
Keywords and Punctuation	xiv
Identifiers and Names	xiv
How to Provide Documentation Feedback	xiv

Part 1. Securing Data

Chapter 1. Server Utility and Directory Security 1-1

Server Utility Security Checking on UNIX and Linux	1-1
Disabling Security Checking	1-2
Securing an Insecure Environment	1-2
Warning and Error Messages for Utility Security Checks	1-2
Users and Group Membership for Running Dynamic Server Utilities	1-3
INFORMIXDIR Directory Permissions	1-4
Security for Loading External Modules	1-5

Chapter 2. Network Data Encryption 2-1

Communication Support Modules for Data Transmission Encryption	2-1
Enabling Encryption with Communication Support Modules	2-1
CSM Configuration File	2-2
Encryption Ciphers and Modes	2-2
MAC Key Files	2-4
Generating a new MAC key file	2-4
MAC Levels	2-4
Switch Frequency	2-5
Network Data Encryption Syntax	2-5
Enterprise Replication and High Availability Network Data Encryption	2-11
Secure Sockets Layer Protocol	2-12
Manage Keystores and Digital Certificates with the IBM Global Security Kit	2-14
Configuring a Server Instance for Secure Sockets Layer Connections	2-14
Configuring a Client for SSL Connections	2-16
Configuring Server-to-Server Secure Sockets Layer Connections	2-17

Chapter 3. Column-Level Encryption 3-1

Encrypting Column Data	3-3
Example Showing How to Determine the Size of an Encrypted Column	3-3
Example Showing How to Encrypt a Column	3-4
Example Showing How to Query Encrypted Data	3-4

Chapter 4. Connection Security 4-1

Pluggable Authentication Modules for Systems Running on UNIX or Linux	4-2
The Name of the PAM Service	4-2

Authentication Modes with the PAM Module.	4-3
PAM Required Stack Size	4-3
Configuring a Database Server to Use PAM	4-3
LDAP Authentication Support on Windows	4-4
Installing and Customizing the LDAP Authentication Support Module	4-4
Configuring the LDAP Module	4-4
Configuring Dynamic Server for LDAP	4-5
Authentication Mode with the LDAP Module.	4-5
Authentication Module Deployment	4-5
Implicit Connections with Authentication Modules	4-5
Application Development for Authentication Modules	4-6
Distributed Transactions and Authentication Modules	4-7
Client APIs and Authentication Support Modules	4-8
Compatibility Issues with Authentication Modules	4-9
Simple Password Encryption	4-9
CSM Configuration File.	4-10
Configuring Password Encryption	4-10
SMI Tables and conscsm.cfg Setting	4-11
Example conscsm.cfg Entries for Password Encryption.	4-12
Single Sign-on Authentication.	4-12
Kerberos Authentication Protocol	4-13
Setting up an SSO Authentication Environment.	4-13
Clients Supporting SSO.	4-14
Preparing the Informix DBMS for Kerberos Authentication	4-14
Configuring an IDS Instance for SSO	4-15
Configuring ESQL/C and ODBC Drivers for SSO	4-18
Configuring JDBC Driver for SSO	4-19
Enterprise Replication and High Availability Connection Security.	4-20
Secure Local Connections to a Host.	4-21
Limiting Denial-of-Service Flood Attacks	4-21
LISTEN_TIMEOUT and MAX_INCOMPLETE_CONNECTIONS Configuration Parameters	4-22
Chapter 5. Discretionary Access Control	5-1
User Roles	5-1
Role Separation.	5-1
Default Roles	5-1
Granting privileges for a default role	5-2
Setting Permission to Create Databases	5-2
Security for External Routines (UDRs)	5-3
Enabling non-DBSAs to View SQL Statements a Session Is Executing	5-3
Chapter 6. Label-Based Access Control (Enterprise Edition)	6-1
Configuring Label-Based Access Control	6-1
How Security Labels Control Access	6-2
Database Security Administrator Role	6-3
Granting the Database Security Administrator Role	6-3
Revoking the Database Security Administrator Role	6-4
Security Label Components.	6-4
Security Label Component Type: ARRAY	6-5
Security Label Component Type: SET	6-6
Security Label Component Type: TREE	6-7
Creating Security Label Components.	6-8
Altering Security Label Components.	6-9
Security Policies	6-9
Creating Security Policies	6-9
Security Labels	6-10
Creating Security Labels	6-10
Granting Security Labels	6-11
Revoking Security Labels	6-11
Security Label Support Functions	6-12

Protecting Tables at the Row and Column Levels	6-12
Applying Row Level Protection	6-14
Applying Column Level Protection	6-14
Exemptions	6-15
Granting Exemptions	6-15
Revoking Exemptions	6-15
Maintaining a Label-Based Access Control Implementation	6-16
Dropping Security Objects	6-17
Renaming Security Objects.	6-18
IDS Security Considerations for Label-Based Access Control	6-19
Other IDS Functionality with Label-Based Access Control	6-22

Part 2. Auditing Data Security

Chapter 7. Overview of Auditing 7-1

Secure-Auditing Facility	7-1
Audit Events	7-1
Audit Masks.	7-1
Audit Process	7-3
Audit Trail	7-4
Roles for Database Server and Audit Administration	7-4
Audit Masks and Audit Instructions	7-4
User Masks	7-5
Template Masks	7-6
Audit Instructions	7-6
Audit Configuration	7-8
Auditing On or Off	7-9
Types of Auditing	7-9
Properties of Audit Files on UNIX	7-9
Windows Application Event Log.	7-10
Windows Message Server	7-11
Error Modes for Writing to an Audit File	7-11
Audit Configuration and the ADTCFG File	7-11
Access to the Audit Trail	7-12
Audit Analysis Overview	7-14
Importance of Audit Analysis.	7-14
Preparation for Audit Analysis	7-14
Strategies for Audit Analysis	7-15
Responses to Identified Security Problems	7-16
DBMS Security Threats	7-17
Primary Threats	7-17
Privileged Activity Threats.	7-17
Shared-Memory Connection Threats on UNIX	7-18
Threats from Malicious Software.	7-18
Remote-Access Threats	7-19
Obsolete-User Threats	7-19
Untrusted Software Used in a Privileged Environment	7-19
Distributed Database Configuration Threats	7-19

Chapter 8. Audit Administration. 8-1

Administrative Roles and Role Separation	8-1
Database Server Administrator.	8-1
Database System Security Officer	8-1
Audit Analysis Officer	8-2
Other Administrative Roles and Users	8-2
Using Role Separation	8-3
Auditing Setup	8-5
Setting Up the Default and Global Masks	8-6
Specifying a Directory for the Audit Trail (UNIX)	8-6
Setting the Error Mode	8-6

Setting the Audit Level	8-7
Activating Auditing	8-8
Audit Mask Maintenance	8-8
Creating Audit Masks.	8-8
Displaying Audit Masks	8-10
Modifying Audit Masks.	8-11
Deleting Audit Masks	8-11
Audit Configuration Maintenance	8-12
Displaying the Audit Configuration.	8-12
Starting a New Audit File	8-13
Changing Audit Levels	8-14
Changing the Audit Error Mode.	8-14
Turning Off Auditing	8-14

Chapter 9. Audit Analysis 9-1

Audit-Record Format	9-1
Audit Analysis Without SQL	9-2
Audit Analysis with SQL	9-3
Planning for SQL Audit Analysis	9-3
Revoking and Granting Privileges to Protect Audit Data	9-3
Preparing Audit Analysis Records for SQL Access	9-4
Interpreting Data Extracted from Audit Records	9-6

Chapter 10. Utility Syntax 10-1

The onaudit Utility	10-1
Showing Audit Masks	10-2
Modifying an Audit Mask	10-2
Creating or Adding an Audit Mask.	10-3
Deleting an Audit Mask	10-5
Starting a New Audit File	10-6
Showing the Audit Configuration	10-6
Changing the Audit Configuration	10-7
The onshowaudit Utility	10-9

Chapter 11. Audit Event Mnemonics and Fields 11-1

Audit-Event Mnemonics	11-1
Audit-Event Fields	11-5

Chapter 12. The ADTCFG File 12-1

ADTCFG File Conventions	12-1
ADTERR	12-2
ADTMODE.	12-2
ADTPATH	12-3
ADTSIZE	12-3

Part 3. Appendixes

Appendix. Accessibility A-1

Accessibility features for IBM Informix Dynamic Server	A-1
Accessibility Features	A-1
Keyboard Navigation	A-1
Related Accessibility Information.	A-1
IBM and Accessibility	A-1
Dotted Decimal Syntax Diagrams	A-1

Notices B-1

Trademarks	B-3
----------------------	-----

Index X-1

Introduction

About This Publication

This publication documents methods for keeping your data secure by preventing unauthorized viewing and altering of data or database objects.

This publication also documents the secure-auditing facility of the database server.

This manual is not a computer-security or trusted-facility-administration training manual.

Types of Users

This publication is written for the following users:

- Database administrators
- Database server administrators
- Operating system administrators
- Database system security officers

This manual is written with the assumption that you have the following background:

- A working knowledge of your computer, your operating system, and the utilities that your operating system provides
- Some experience working with relational databases or exposure to database concepts
- Some experience with computer programming
- Some experience with database server administration, operating-system administration, or network administration

If you have limited experience with relational databases, SQL, or your operating system, see the *IBM Informix Dynamic Server Getting Started Guide* for a list of supplementary titles.

Prerequisites for a Secure Database Server

You must ensure that the overall database environment is properly secured before implementing many of the features detailed in IBM® Informix® security documentation.

Database technology comprises interdependent components. Security must exist on each of these components at each layer in the environment to make a truly secure system. Security measures should include network, host, and physical security; identity management; and business controls.

For information about some steps you can take to help secure your database server and the data on your system, see the *IBM Data Server Security Blueprint*. Download the complete and most up-to-date blueprint at <http://www.ibm.com/software/data/db2imstools/solutions/security-blueprint.html>. A list of recommended precautions is reproduced in Figure 1 on page viii.

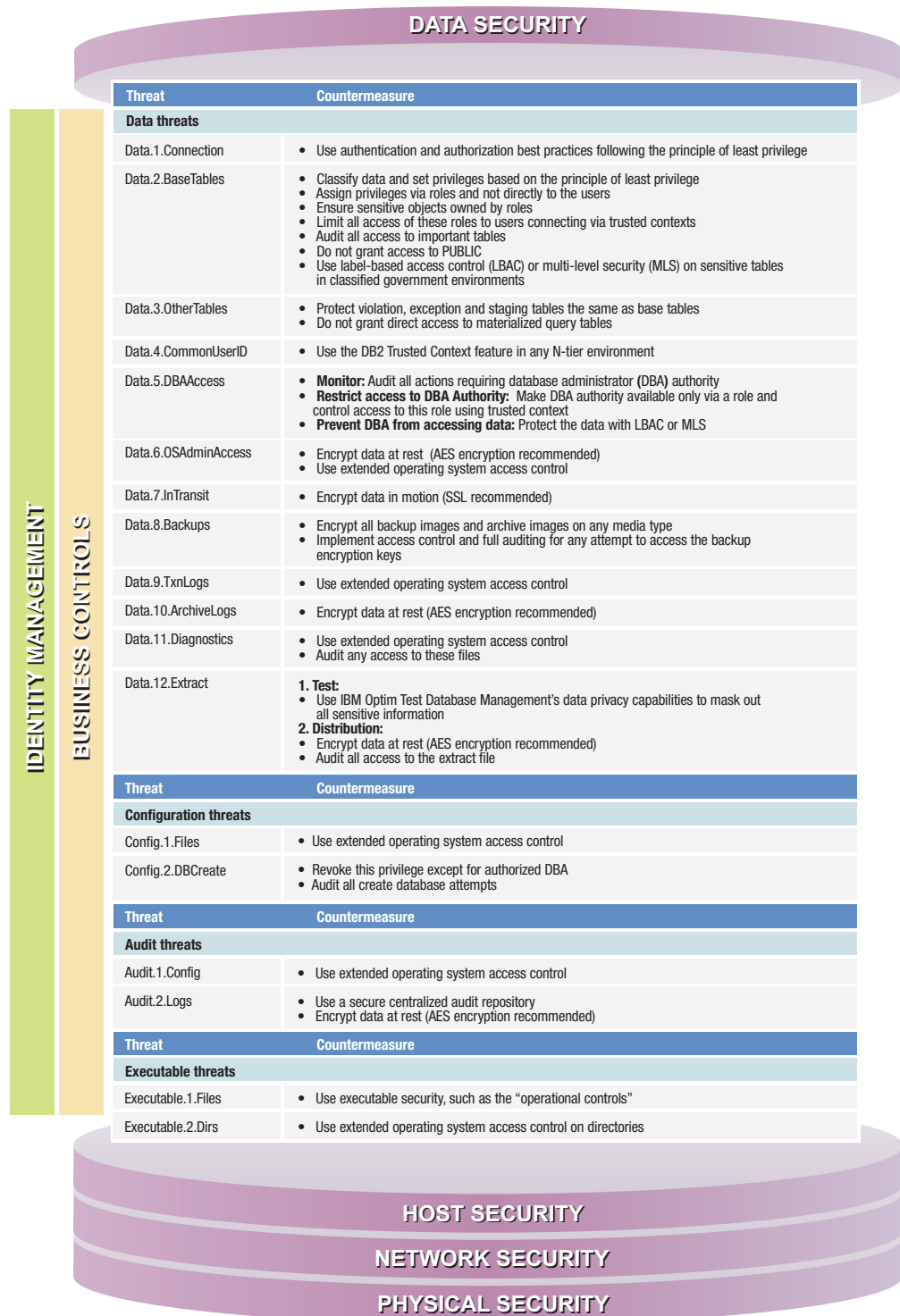


Figure 1. Security Precautions

What's New for Security in Dynamic Server, Version 11.50

For a comprehensive list of new features for this release, see the *IBM Informix Dynamic Server Getting Started Guide*. The following changes and enhancements are relevant to this publication.

Table 1. What's New in IBM Informix Security Guide for Version 11.50.xC2

Overview	Reference
<p>Server-Specific Audit Configuration File Functionality</p> <p>Each server instance uses its own adtcfg.servernumber file if the audit configuration parameters are changed with the onaudit utility. The new -n option for the onshowaudit utility reads the server-specific audit configuration (adtcfg.servernumber) file.</p>	See "The onshowaudit Utility" on page 10-9.

Table 2. What's New in IBM Informix Security Guide for Version 11.50.xC1

Overview	Reference
<p>Support for Encrypting Data by Using Secure Sockets Layer (SSL) Communications</p> <p>You can now configure IDS to use the SSL protocol, which encrypts data in TCP/IP connections between two points over a network. The SSL protocol is an alternative to the IDS-specific encryption Communication Support Module (CSM) and simple password CSM for CSDK clients. You must use SSL to encrypt data in communications between IDS and DRDA® clients.</p>	See "Secure Sockets Layer Protocol" on page 2-12.
<p>Single Sign-on Support Added</p> <p>With single sign-on, users authenticate themselves once and the authentication is carried securely over trusted and non-trusted networks. The users provide a valid user ID and password when they log in to a client computer, and they can access the database server and other SSO-enabled services without having to log in again. In the past, users had to log in multiple times.</p> <p>IDS delivers support for SSO in the Generic Security Services Communications Support Module (GSSCSM). GSSCSM works with Kerberos, which is a network authentication protocol. You must deploy a Kerberos authentication protocol that supports the Generic Security Services Application Programming Interface (GSSAPI) before you can use SSO with IDS. After you enable SSO, you benefit from centralized management of authentication.</p>	See "Single Sign-on Authentication" on page 4-12.

Documentation Conventions

This section describes the following conventions, which are used in the product documentation for IBM Informix Dynamic Server:

- Typographical conventions
- Feature, product, and platform conventions
- Syntax diagrams
- Command-line conventions
- Example code conventions

Typographical Conventions

This publication uses the following conventions to introduce new terms, illustrate screen displays, describe command syntax, and so forth.

Convention	Meaning
KEYWORD	Keywords of SQL, SPL, and some other programming languages appear in uppercase letters in a serif font.
<i>italics</i>	Within text, new terms and emphasized words appear in italics. Within syntax and code examples, variable values that you are to specify appear in italics.
boldface	Names of program entities (such as classes, events, and tables), environment variables, file names, path names, and interface elements (such as icons, menu items, and buttons) appear in boldface.
monospace	Information that the product displays and information that you enter appear in a monospace typeface.
KEYSTROKE	Keys that you are to press appear in uppercase letters in a sans serif font.
>	This symbol indicates a menu item. For example, “Choose Tools > Options ” means choose the Options item from the Tools menu.

Technical changes to the text are indicated by special characters depending on the format of the documentation:

HTML documentation

New or changed information is surrounded by blue ≧ and ≦ characters.

PDF documentation

A plus sign (+) is shown to the left of the current changes. A vertical bar (|) is shown to the left of changes made in earlier shipments.

Feature, Product, and Platform Markup

Feature, product, and platform markup identifies paragraphs that contain feature-specific, product-specific, or platform-specific information. Some examples of this markup follow:

Dynamic Server only: Identifies information that is specific to the Windows operating system

Windows only: Identifies information that is specific to the Windows operating system

This markup can apply to one or more paragraphs within a section. When an entire section applies to a particular product or platform, this is noted as part of the heading text, for example:

Table Sorting (Windows)

Example Code Conventions

Examples of SQL code occur throughout this publication. Except as noted, the code is not specific to any single IBM Informix application development tool.

If only SQL statements are listed in the example, they are not delimited by semicolons. For instance, you might see the code in the following example:

```
CONNECT TO stores_demo
...

DELETE FROM customer
  WHERE customer_num = 121
...

COMMIT WORK
DISCONNECT CURRENT
```

To use this SQL code for a specific product, you must apply the syntax rules for that product. For example, if you are using an SQL API, you must use EXEC SQL at the start of each statement and a semicolon (or other appropriate delimiter) at the end of the statement. If you are using DB–Access, you must delimit multiple statements with semicolons.

Tip: Ellipsis points in a code example indicate that more code would be added in a full application, but it is not necessary to show it to describe the concept being discussed.

For detailed directions on using SQL statements for a particular application development tool or SQL API, see the documentation for your product.

Additional Documentation

You can view, search, and print all of the product documentation from the IBM Informix Dynamic Server information center on the Web at <http://publib.boulder.ibm.com/infocenter/idshelp/v115/index.jsp>.

For additional documentation about IBM Informix Dynamic Server and related products, including release notes, machine notes, and documentation notes, go to the online product library page at <http://www.ibm.com/software/data/informix/pubs/library/>. Alternatively, you can access or install the product documentation from the Quick Start CD that is shipped with the product.







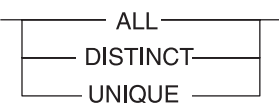
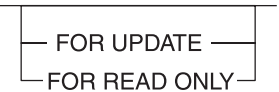
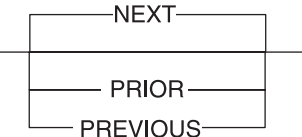
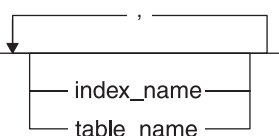

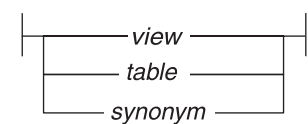
Compliance with Industry Standards

The American National Standards Institute (ANSI) and the International Organization of Standardization (ISO) have jointly established a set of industry standards for the Structured Query Language (SQL). IBM Informix SQL-based products are fully compliant with SQL-92 Entry Level (published as ANSI X3.135-1992), which is identical to ISO 9075:1992. In addition, many features of IBM Informix database servers comply with the SQL-92 Intermediate and Full Level and X/Open SQL Common Applications Environment (CAE) standards.

Syntax Diagrams

This guide uses syntax diagrams built with the following components to describe the syntax for statements and all commands other than system-level commands.

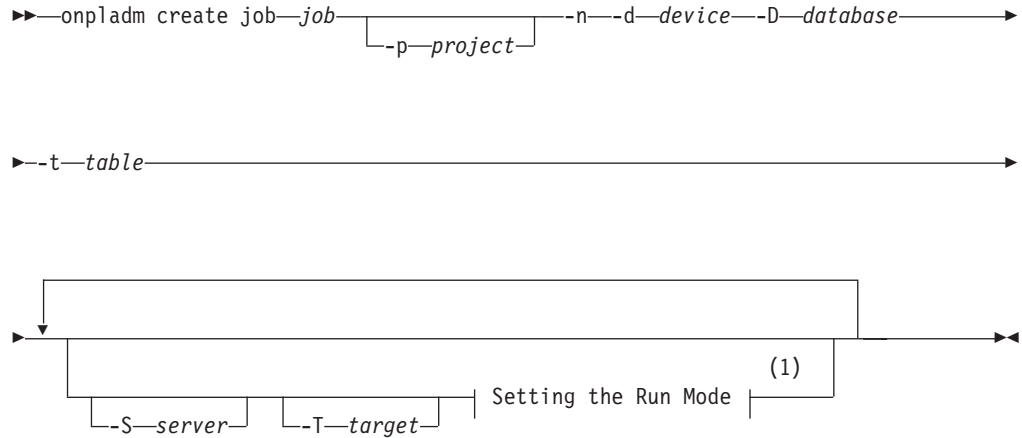
Table 3. Syntax Diagram Components

Component represented in PDF	Component represented in HTML	Meaning
	<code>>>-----</code>	Statement begins.
	<code>-----></code>	Statement continues on next line.
	<code>>-----</code>	Statement continues from previous line.
	<code>-----><</code>	Statement ends.
	<code>-----SELECT-----</code>	Required item.
	<code>--+-----+-- '-----LOCAL-----'</code>	Optional item.
	<code>---+-----ALL-----+--- +--DISTINCT-----+ '---UNIQUE-----'</code>	Required item with choice. One and only one item must be present.
	<code>---+-----+--- +--FOR UPDATE-----+ '--FOR READ ONLY--'</code>	Optional items with choice are shown below the main line, one of which you might specify.
	<code>.---NEXT-----. ---+-----+--- +---PRIOR-----+ '---PREVIOUS-----'</code>	The values below the main line are optional, one of which you might specify. If you do not specify an item, the value above the line will be used as the default.
	<code>.-----, v ---+-----+--- +---index_name---+ '---table_name---'</code>	Optional items. Several items are allowed; a comma must precede each repetition.
	<code>>>- Table Reference -><</code>	Reference to a syntax segment.
	<code> ---+-----view-----+--- +-----table-----+ '-----synonym-----'</code>	Syntax segment.

How to Read a Command-Line Syntax Diagram

The following command-line syntax diagram uses some of the elements listed in the table in Syntax Diagrams.

Creating a No-Conversion Job

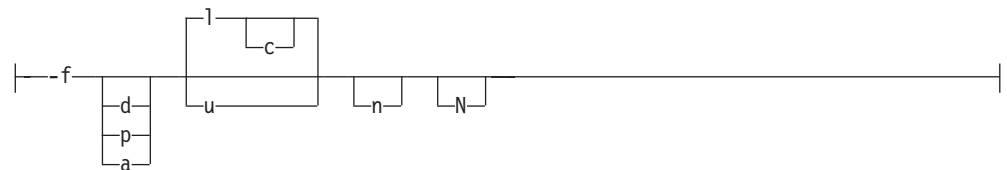


Notes:

- 1 See page Z-1

The second line in this diagram has a segment named “Setting the Run Mode,” which according to the diagram footnote, is on page Z-1. If this was an actual cross-reference, you would find this segment in on the first page of Appendix Z. Instead, this segment is shown in the following segment diagram. Notice that the diagram uses segment start and end components.

Setting the Run Mode:



To see how to construct a command correctly, start at the top left of the main diagram. Follow the diagram to the right, including the elements that you want. The elements in this diagram are case sensitive because they illustrate utility syntax. Other types of syntax, such as SQL, are not case sensitive.

The Creating a No-Conversion Job diagram illustrates the following steps:

1. Type **onpladm create job** and then the name of the job.
2. Optionally, type **-p** and then the name of the project.
3. Type the following required elements:
 - **-n**
 - **-d** and the name of the device
 - **-D** and the name of the database
 - **-t** and the name of the table

4. Optionally, you can choose one or more of the following elements and repeat them an arbitrary number of times:
 - **-S** and the server name
 - **-T** and the target server name
 - The run mode. To set the run mode, follow the Setting the Run Mode segment diagram to type **-f**, optionally type **d**, **p**, or **a**, and then optionally type **l** or **u**.
5. Follow the diagram to the terminator.

Keywords and Punctuation

Keywords are words reserved for statements and all commands except system-level commands. When a keyword appears in a syntax diagram, it is shown in uppercase letters. When you use a keyword in a command, you can write it in uppercase or lowercase letters, but you must spell the keyword exactly as it appears in the syntax diagram.

You must also use any punctuation in your statements and commands exactly as shown in the syntax diagrams.

Identifiers and Names

Variables serve as placeholders for identifiers and names in the syntax diagrams and examples. You can replace a variable with an arbitrary name, identifier, or literal, depending on the context. Variables are also used to represent complex syntax elements that are expanded in additional syntax diagrams. When a variable appears in a syntax diagram, an example, or text, it is shown in *lowercase italic*.

The following syntax diagram uses variables to illustrate the general form of a simple SELECT statement.

►►—SELECT—*column_name*—FROM—*table_name*—►►

When you write a SELECT statement of this form, you replace the variables *column_name* and *table_name* with the name of a specific column and table.

How to Provide Documentation Feedback

You are encouraged to send your comments about IBM Informix user documentation by using one of the following methods:

- Send e-mail to docinf@us.ibm.com.
- Go to the Information Center at <http://publib.boulder.ibm.com/infocenter/idshep/v115/index.jsp> and open the topic that you want to comment on. Click the feedback link at the bottom of the page, fill out the form, and submit your feedback.

Feedback from both methods is monitored by those who maintain the user documentation of Dynamic Server. The feedback methods are reserved for reporting errors and omissions in our documentation. For immediate help with a technical problem, contact IBM Technical Support. For instructions, see the IBM Informix Technical Support Web site at <http://www.ibm.com/planetwide/>.

We appreciate your suggestions.

Part 1. Securing Data

This section discusses methods for keeping your data secure by preventing unauthorized viewing and altering of data or other database objects.

Chapter 1. Server Utility and Directory Security

Informix Dynamic Server utilities and product directories are secure by default. You can also increase directory security for some environments with the `DB_LIBRARY_PATH` configuration parameter.

- The database server utilities make security checks before the database server starts. See “Server Utility Security Checking on UNIX and Linux.”
- Most IDS utilities run as secure users and belong to a secure group. See “Users and Group Membership for Running Dynamic Server Utilities” on page 1-3.
- The subdirectories under the `$INFORMIXDIR` directory have specific owners, directory permissions, and belong to a specific group depending on their security requirements. See “INFORMIXDIR Directory Permissions” on page 1-4.
- Use the `DB_LIBRARY_PATH` configuration parameter to control the location from which shared objects, such as external modules, can be loaded. See “Security for Loading External Modules” on page 1-5.

Server Utility Security Checking on UNIX and Linux

The database server utilities make security checks before the database server starts.

When you install a new version of your database server, you should follow the installation instructions to ensure that the permissions of all key files and directories are set appropriately.

To provide increased security, key server utilities check to determine if your environment is secure. Before the database server starts, the following settings must be unchanged from the settings established during installation:

- The permissions on `$INFORMIXDIR` and some directories under it.
For each directory, the database server checks that the directory exists, that it is owned by user **informix** and the correct group (as shown in “INFORMIXDIR Directory Permissions” on page 1-4), and that directory permissions do not include write permissions for the group or other users.
- The permissions on the `ONCONFIG` file.
The file must belong to the Database Server Administrator (DBSA) group. If the DBSA group is **informix** (the default group), the `ONCONFIG` file should be owned by user **informix**; otherwise, the ownership is not restricted. The file must not have write permissions for others.
- The permissions on the `sqlhosts` file.
Under the default configuration, the `sqlhosts` file is located in the `$INFORMIXDIR/etc` directory. The owner should be user **informix**, the group should be either the **informix** group or the DBSA group, and the file must not have public write permissions. If the file is specified through an `INFORMIXSQLHOSTS` environment variable, the owner and group are not checked; however, public write permissions are not permitted.
- Filename lengths.
The length of both the filenames `$INFORMIXDIR/etc/onconfig.std` and `$INFORMIXDIR/etc/$ONCONFIG` must be less than 256 characters.

If the tests for any of these conditions fail, the utilities exit with an error message.

Important: The database server does not read configuration files if they are publicly writable, unless the INFORMIXDIR environment variable is designated as secure in the `/etc/informix` directory. See information on “Disabling Security Checking” and “Securing an Insecure Environment.”

Disabling Security Checking

Although it is strongly recommended that you **never** disable security checking, you can partially disable the security checking for a database server residing in a specific \$INFORMIXDIR directory.

If you choose to disable the security checking even though it is recommended that you never do so, you should use the `ibmifmx_security.sh` script to limit the number of SUID and SGID programs on your system.

To disable security checking:

As the user **root**, run the `INFORMIXDIR/etc/informixdir-is-insecure` script. After this script runs successfully, the warning messages still appear when the utilities are run, but the programs continue. You can specify the value of INFORMIXDIR on the command line as an argument to the script. Thus, you do not need to set INFORMIXDIR in the **root** user environment.

The `informixdir-is-insecure` script creates a `/etc/informix` directory (if necessary) that is owned by **root** and has 555 permissions. In this directory, the script creates a file named `server-xx.xx.yyy` that has 444 permissions. The `xx.xx` portion of the filename is the major version number and `yyy` portion is the fix pack number: for example, `server-11.50.UC1`. This file lists the \$INFORMIXDIR values for which security checking is disabled.

Note: The format of the contents of the `server-xx.xx.yyy` files might change in future releases.

Securing an Insecure Environment

If the database server reports that your environment is no longer secure and the programs exit, you must reset the relevant directory permissions.

To secure an insecure environment:

As user **root**, run the `$INFORMIXDIR/etc/make-informixdir-secure` script. Although user **informix** has permission to run the script, the script cannot fix the problems unless the directory is owned by user **informix**. However, the error messages indicate what still needs to be fixed. The script also shows files and directories under \$INFORMIXDIR that belong to an unexpected owner or group or have public write permission.

Warning and Error Messages for Utility Security Checks

If the security check that a server utility performs at startup detects a problem, the security check returns an error message.

These messages are returned when the message file and internationalization support are not available; therefore, the error messages do not have error numbers and are not translated.

For the following problems, one of the following messages displays and the utility continues:

- INFORMIXDIR or ONCONFIG is too long. Maximum length for \$INFORMIXDIR/etc/\$ONCONFIG is 255 characters.
- INFORMIXSQLHOSTS is too long. Maximum length is 255 characters.
- ONCONFIG not set; TBCONFIG set. TBCONFIG will not be supported in future.

For the following problems, the utility exits and displays one of the following messages:

- User informix not found.
- Group informix not found.
- Could not access *logical-file filename*.
- *Logical-file filename* is not owned by user with id *UID*.
- *Logical-file filename* not owned by group with id *GID*.
- *Logical-file filename* has insecure mode *mode*.
- Could not access *logical-file filename*.

The following table defines the variables used in the messages above.

Variable	Explanation
<i>filename</i>	A name of the file or directory
<i>logical-file</i>	<p>Either ONCONFIG, INFORMIXSQLHOSTS, INFORMIXDIR, or INFORMIXDIR/<i>xxx</i> (where <i>xxx</i> is one of a number of subdirectories under \$INFORMIXDIR).</p> <p>For example, if INFORMIXDIR is set to /usr/informix, the message might read: INFORMIXSQLHOSTS /usr/informix/etc/sqlhosts is not owned by the user with id 1234.</p>
<i>mode</i>	An octal permissions value
<i>UID</i>	A number
<i>GID</i>	A number

Users and Group Membership for Running Dynamic Server Utilities

Most IDS utilities run as secure users and belong to a secure group.

The following Dynamic Server utilities are SUID **root** and SGID **informix**:

- **onaudit**
- **onbar_d**
- **ondblog**
- **onedcu**
- **oninit**
- **onmode**
- **ON-Monitor**
- **onshowaudit**
- **onsmsync**
- **onsnmp**
- **onsrvapd**

- **ontape**
- **snmpdm**

The following Dynamic Server utilities are SGID **informix**:

- **oncheck**
- **onedpdu**
- **onload**
- **onlog**
- **onparams**
- **onpload**
- **onspaces**
- **onstat**
- **onunload**
- **xtree**

Restriction: You cannot use the following utilities on HDR secondary servers, remote standalone (RS) secondary servers, or shared disk (SD) secondary servers:

- HPL
- dbload
- dbimport
- dbexport
- onload
- onunload
- archecker
- ondblog
- onshowaudit
- onaudit
- onparams
- onspaces
- onmonitor
- onperf
- onsnmp

INFORMIXDIR Directory Permissions

The subdirectories under the \$INFORMIXDIR directory have specific owners, directory permissions, and belong to a specific group depending on their security requirements.

The following table lists \$INFORMIXDIR directories and their respective owners, groups, and permissions.

Subdirectory	Owner	Group	Permissions
. (\$INFORMIXDIR)	informix	informix	755
bin	informix	informix	755
lib	informix	informix	755
gls	informix	informix	755
msg	informix	informix	755

Subdirectory	Owner	Group	Permissions
etc	informix	DBSA	775
aaodir	informix	AAO	775
tmp	informix	informix	770

See “Administrative Roles and Role Separation” on page 8-1 for more information about database server administrator (DBSA), audit analysis officer (AAO), and database system security officer (DBSSO) groups.

Security for Loading External Modules

Use the DB_LIBRARY_PATH configuration parameter to control the location from which shared objects, such as external modules, can be loaded.

Use the DB_LIBRARY_PATH configuration parameter to specify a comma-separated list of valid directory prefix locations from which the database server can load external modules, such as DataBlade® modules.

DB_LIBRARY_PATH takes effect when the database server is restarted after the parameter has been set.

The DB_LIBRARY_PATH configuration parameter allows you to control the location from which shared objects can be loaded, and it allows you to enforce policies and standards on the formats of the EXTERNAL NAME clause of the CREATE FUNCTION, CREATE PROCEDURE, and CREATE ROUTINE statements.

If the DB_LIBRARY_PATH configuration parameter is not set or is not present in the ONCONFIG file, security checks for loading external modules are not performed.

You should include in the DB_LIBRARY_PATH settings every file system in which your security policy authorizes DataBlade modules and UDRs to reside. DataBlade modules provided with Dynamic Server are stored under the **\$INFORMIXDIR/extend** directory. For extensibility to work properly when security is turned on, the string “\$INFORMIXDIR/extend” must be part of DB_LIBRARY_PATH.

For more information on the DB_LIBRARY_PATH configuration parameter, see the *IBM Informix Dynamic Server Administrator’s Reference*.

Chapter 2. Network Data Encryption

Use network encryption to encrypt data transmitted between server and client as well as between server and other server.

Encryption is the process of transforming data into an unintelligible form to prevent the unauthorized use of the data. To read an encrypted file, you must have access to a secret key or password that enables you to decrypt it. Unencrypted data is called *plain text*; encrypted data is called *cipher text*. A *cipher* is an encryption-decryption algorithm.

Communication Support Modules for Data Transmission Encryption

You can use the communication support modules (CSMs) to encrypt data transmissions, including distributed queries, over the network.

The encryption CSM (ENCCSM) provides network transmission encryption.

This option provides complete data encryption with a standard cryptography library, with many configurable options. A message authentication code (MAC) is transmitted as part of the encrypted data transmission to ensure data integrity. A MAC is an encrypted message digest.

CSMs have the following restrictions:

- You cannot use an encryption CSM and a simple password CSM simultaneously. For example, if you are using the simple password CSM, SPWDSCSM, and decide to encrypt your network data, you must remove the entries for the SPWDSCSM in your **concsm.cfg** and **sqlhosts** files.
- You cannot use either simple password CSM or encryption CSM over a multiplexed connection.
- Enterprise Replication and high-availability clusters (High-Availability Data Replication, remote standalone secondary servers, and shared disk secondary servers) support encryption, but cannot use a connection configured with a CSM. See “Enterprise Replication and High Availability Network Data Encryption” on page 2-11 for more information about this topic.
- Encrypted connections and unencrypted connections cannot be combined on the same port.

Secure Sockets Layer (SSL) communications, which encrypt data in end-to-end, secure TCP/IP and Distributed Relational Database Architecture™ (DRDA) connections between two points over a network, are an alternative to the IDS-specific encryption CSMs. For more information, see “Secure Sockets Layer Protocol” on page 2-12.

Enabling Encryption with Communication Support Modules

You must modify the **concsm.cfg** configuration file to use encryption with communication support modules.

Verify that the module can use a port that is not shared with an unencrypted connection before you enable network encryption.

To enable network encryption:

1. Add a line to the **concsm.cfg** configuration file. The **concsm.cfg** file must contain an entry for each Communications Support Module (of the same kind) that you are using.
2. Add an entry to the **options** column of the **sqlhosts** file or registry. For information on specifying the CSM in the **sqlhosts** file or registry, see the *IBM Informix Dynamic Server Administrator's Guide*.

CSM Configuration File

To use a communication support module (CSM), you must have a **concsm.cfg** file.

An entry in the **concsm.cfg** file is a single line and is limited to 1024 characters. After you describe the CSM in the **concsm.cfg** file, you can enable it in the **options** parameter of the **sqlhosts** file, as described in *IBM Informix Dynamic Server Administrator's Guide*.

The **concsm.cfg** file resides in the **etc** directory of **INFORMIXDIR** by default. If you want to store the file somewhere else, you can override the default location by setting the **INFORMIXCONCSMCFG** environment variable to the full pathname of the new location. For information on setting the environment variable **INFORMIXCONCSMCFG**, see the *IBM Informix Guide to SQL: Reference*.

Entries in the **concsm.cfg** file must conform to the following restrictions:

- The following characters are not allowed to be part of library pathnames:
 - = (equal sign)
 - " (double quote)
 - , (comma)
- White spaces cannot be used unless the white spaces are part of a pathname.

Encryption Ciphers and Modes

You must specify which ciphers and mode to use during encryption.

The cipher and mode that is used is randomly chosen among the ciphers that are common between the two servers. Make sure that all servers and client computers that participate in encrypted communication have ciphers and modes in common. Encryption is more secure if you include more ciphers and modes that the database server can switch between. For information on how to switch between ciphers, see "Switch Frequency" on page 2-5.

The Data Encryption Standard (DES) is a cryptographic algorithm designed to encrypt and decrypt data using 8-byte blocks and a 64-bit key.

The Triple DES (DES3) is a variation of DES in which three 64-bit keys are used for a 192-bit key. DES3 works by first encrypting the plain text using the first 64-bits of the key. Then the cipher text is decrypted using the next part of the key. In the final step, the resulting cipher text is re-encrypted using the last part of the key.

The Advanced Encryption Standard (AES) is a replacement algorithm that is used by the United States government.

Two encryption modes are:

- *Block Mode*, a method of encryption in which the message is broken into blocks and the encryption occurs on each block as a unit. Since each block is at least 8 bytes large, block mode provides the ability for 64-bit arithmetic in the encryption algorithm.
- *Stream Mode*, a method of encryption in which each individual byte is encrypted. It is generally considered to be a weak form of encryption.

A *blowfish* is a block cipher that operates on 64-bit (8-byte) blocks of data. It uses a variable size key, but typically, 128-bit (16-byte) keys are considered to be good for strong encryption. Blowfish can be used in the same modes as DES.

Important: It is strongly recommended that you do not specify specific ciphers. For security reasons, all ciphers should be allowed. If a cipher is discovered to have a weakness, you can exclude it.

Use the **allbut** option to list ciphers and modes to exclude. Enclose the **allbut** list in angled brackets (<>). The list can include unique, abbreviated entries. For example, **bf** can represent **bf1**, **bf2**, and **bf3**. However, if the abbreviation is the name of an actual cipher, then *only* that cipher is eliminated. Therefore, **des** eliminates only the DES cipher, but **de** eliminates **des**, **ede**, and **desx**.

The following **des**, **ede**, and **desx** ciphers are supported.

Cipher	Explanation	Blowfish Cipher	Explanation
des	DES (64-bit key)	bf1	Blowfish (64-bit key)
ede	Triple DES	bf2	Blowfish (128-bit key)
desx	Extended DES (128-bit key)	bf3	Blowfish (192-bit key)

Important: The cipher **desx** can only be used in **cbc** mode.

The following AES-encryption ciphers are supported.

Cipher	Explanation
aes	AES (128-bit key)
aes128	AES (128-bit key)
aes192	AES (192-bit key)
aes256	AES (256-bit key)

The following modes are supported.

Mode	Explanation
ecb	Electronic Code Book
cbc	Cipher Block Chaining
cfb	Cipher Feedback
ofb	Output Feedback

Because **ecb** mode is considered weak; it is only included if specifically requested. It is not included in the **all** or the **allbut** list.

MAC Key Files

The MAC key files contain encryption keys that are used to encrypt messages.

The database servers and client computers that participate in encryption normally require the same MAC key file. For information on how to switch between MAC keys, see “Switch Frequency” on page 2-5.

The default MAC key file is the built-in file provided by Dynamic Server. This file provides limited message verification (some validation of the received message and determination that it appears to have come from an Informix Dynamic Server client or server). A site-generated MAC key file performs the strongest verification. You can generate key files with the **GenMacKey** utility.

Each of the MAC key files is prioritized and negotiated at connect time. The prioritization for the MAC key files is based on their creation time by the **GenMacKey** utility. The built-in key file has the lowest priority.

Tip: If there are no MAC key files present, the built-in MAC key is used by default. However, by using a MAC key file, the default built-in MAC key is disabled.

Generating a new MAC key file

You can generate a new MAC key file to improve the reliability of message verification using encryption.

To generate a new MAC key file:

1. Execute the following command from the command line:

```
GenMacKey -o filename
```

The *filename* is the path and filename of the new MAC key file.

2. Update the central server's configuration to include the location of the new MAC key file in one of the following ways:
 - **Using encryption tags:** Edit the relevant line in the **concsn.cfg** file to add a path and filename to the **mac** tag. For instructions, see “The mac Tag” on page 2-7.
 - **Using encryption parameters:** Edit the encryption parameters file to alter the value of the **ENCCSM_MACFILES** parameter. For instructions, see “ENCCSM_MACFILES Parameter” on page 2-10.
3. If necessary, remove old MAC key file entries from the configuration.
4. Distribute the new MAC key file among all appropriate computers.

MAC Levels

MAC levels determine the type of MAC key generation.

The supported generation levels are:

- **high.** Uses SHA1 MAC generation on all messages.
- **medium.** Uses SHA1 MAC generation for all messages greater than 20 bytes long and XOR folding on smaller messages.
- **low.** Uses XOR folding on all messages.
- **off.** Does not use MAC generation.

The level is prioritized to the highest value. The **off** entry should only be used between servers when it is guaranteed that there is a secure network connection.

All servers and client computers that transmit encrypted communication must have at least one MAC level setting in common. For example, if one database server has a level of **high** and **medium** enabled and the other database server has only **low** enabled, then the connection attempt will fail. But if a database server has **high** and **medium** settings and the other database server has only the **medium** setting, the MAC generation levels support a connection.

Switch Frequency

The switch frequency defines when ciphers and or secret keys are renegotiated.

The default time that this renegotiation occurs is once an hour. By using switch options, you can set the time in minutes when the renegotiation occurs.

The longer that the secret key and encryption cipher remain in use, the more likely that the encryption rules might be broken by an attacker. To avoid this, cryptologists recommend periodically changing the secret key and cipher on long-term connections.

Network Data Encryption Syntax

You must specify network encryption libraries and options in the **concsm.cfg** file.

You can specify the following types of encryption options:

- DES and AES ciphers to use during encryption
- Modes to use during encryption
- Message authentication code (MAC) key files
- MAC levels
- Switch frequency for ciphers and keys

You can specify encryption options by using one of the following methods:

- “Using Encryption Tags in **concsm.cfg**”
- “Invoking an Encryption Parameters File in **concsm.cfg**” on page 2-9

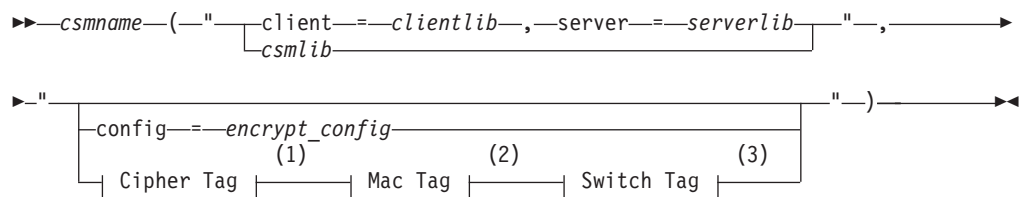
Using Encryption Tags in **concsm.cfg**

You can specify encryption options directly in the **concsm.cfg** file by specifying libraries and encryption tags.

To configure network encryption, use the following syntax to add one or more lines to the **concsm.cfg** file.

There are three encryption tags:

- The **cipher** tag
- The **mac** tag
- The **switch** tag



Notes:

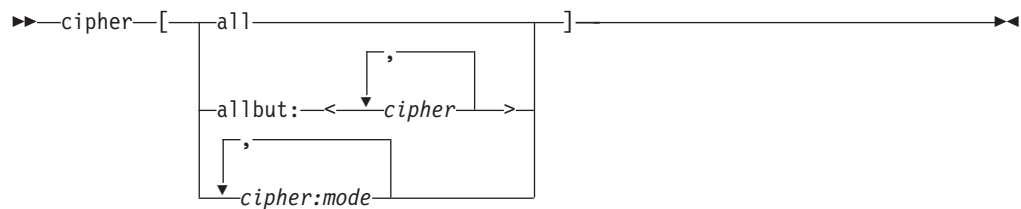
- 1 See "The cipher Tag."
- 2 See "The mac Tag" on page 2-7.
- 3 See "The switch Tag" on page 2-8.

Option	Description
client = <i>clientlib</i>	<p>The full path and name of the shared library that is the CSM on the client computer.</p> <p>Client computers use this CSM to communicate with the database server. The library provided by Dynamic Server is as follows:</p> <ul style="list-style-type: none"> • UNIX®: \$INFORMIXDIR/lib/client/csm/iencs11a.so • Windows® %INFORMIXDIR%\bin\client\iencs11a.dll
config = <i>encrypt_config</i>	<p>The full path and filename of the file in which the encryption parameters are defined. If the file does not exist, then the default values are used. No error is returned.</p> <p>For more information on using encryption parameters, see "Invoking an Encryption Parameters File in conccsm.cfg" on page 2-9.</p>
<i>csm</i> lib	<p>The full path and name of the shared library that is the CSM if the CSM is shared by both the database server and the client computers.</p> <p>The library provided by Dynamic Server is as follows:</p> <ul style="list-style-type: none"> • UNIX: \$INFORMIXDIR/lib/csm/iencs11a.so • Windows: %INFORMIXDIR%\bin\iencs11a.dll
<i>csm</i> name	<p>The name that you assign to the communications support module.</p> <p>For network encryption, use ENCCSM.</p>
server = <i>serverlib</i>	<p>The full path and name of the shared library that is the CSM on the database server. The library provided by Dynamic Server is usually installed in the following directories:</p> <ul style="list-style-type: none"> • UNIX \$INFORMIXDIR/lib/csm/iencs11a.so • Windows %INFORMIXDIR%\bin\iencs11a.dll

The cipher Tag:

The **cipher** tag specifies the ciphers and cipher modes to use for encryption.

The **cipher** tag can include the cipher options shown in the following syntax diagram.



Cipher Option	Description
all	Specifies to include all available ciphers and all available modes, except ECB mode. For example: cipher[all],...
allbut: <i><list of ciphers to exclude></i>	Specifies to include all ciphers except the ones in the list. For more information, see “Encryption Ciphers and Modes” on page 2-2. For example: cipher[allbut:<ecb,des>],... cipher[allbut:<cbc,bf>]
<i>cipher:mode</i>	Specifies one or more cipher and mode. For example: cipher[des:cbc,des:ofb]

The default value for the cipher field is:

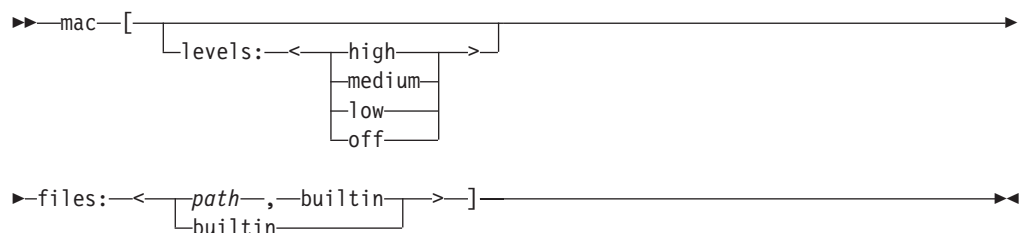
```
cipher[allbut:<ecb>]
```

For more information on ciphers and modes, see “Encryption Ciphers and Modes” on page 2-2.

The mac Tag:

The **mac** tag defines the MAC key files and the level of MAC generation to be used during the MAC generation.

The **mac** tag can include the MAC options shown in the following syntax diagram.



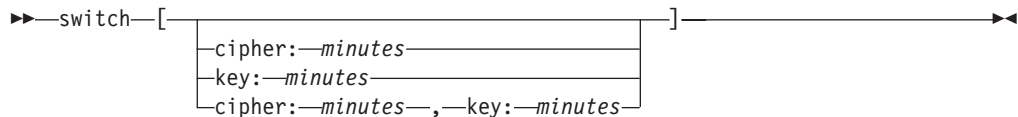
Mac Option	Description
levels	<p>Specifies a comma-separated list of MAC generation levels that the connection supports.</p> <p>For more information, see “MAC Levels” on page 2-4.</p>

Mac Option	Description
files	<p>Specifies a comma-separated list of the full path names of MAC key files.</p> <p>For more information, see “MAC Key Files” on page 2-4.</p> <p>For example:</p> <pre>mac[levels:<high,low>,files: </usr/local/bin/mac1.dat, /usr/local/bin/mac2.dat,builtin>]</pre>

The switch Tag:

The **switch** tag defines the frequency at which ciphers and or secret keys are renegotiated.

The **switch** tag can include the switch options shown in the following syntax diagram.



Switch Option	Description
cipher:minutes	Specifies the time in minutes between cipher renegotiation.
key:minutes	<p>Specifies the time in minutes between secret key renegotiation.</p> <p>For example:</p> <pre>switch[cipher:120,key:20]</pre>

For more information on switching ciphers and modes, see “Switch Frequency” on page 2-5.

Examples Using Encryption Tags:

Encryption tags specify values for network encryption.

These two examples illustrate possible tags in the **concsn.cfg** file to define the encryption CSM.

The following configuration string states to use all available ciphers except for any of the blowfish ciphers, and to not use any cipher in ECB mode:

```
ENCCSM("$INFORMIXDIR/lib/csm/iencs11a.so",
"cipher[allbut:<ecb,bf>]")
```

The following configuration string states to use the DES/CBC-mode, EDE/OFB-mode, and DESX/CBC-mode ciphers for this connection and also to switch the cipher being used every 120 minutes and renegotiate the secret key every 15 minutes:


```
ENCCSM("/$INFORMIXDIR/lib/csm/iencs11a.so",
"cipher[des:cbc,ede:ofb,desx:cbc],switch[cipher:120,key:15]")
```

Invoking an Encryption Parameters File in conccsm.cfg

You can configure encryption options by setting encryption parameters in a file and then invoking it in the **conccsm.cfg** file.

In the encryption parameters file that you specify in the **conccsm.cfg** file, each option has the following form:

parameter_name value

Use the following parameters to set encryption options:

- ENCCSM_CIPHERS: Ciphers to be used
- ENCCSM_MAC: MAC levels
- ENCCSM_MACFILES: MAC file locations
- ENCCSM_SWITCH: Cipher and key change frequency

The following restrictions apply to the parameter values:

- Each entry should be of the form *parameter_name value* separated by white spaces (for example, ENCCSM_MAC medium,high and ENCCSM_MACFILES /usr/local/bin/mac1.dat,/usr/local/bin/mac2.dat,builtin).

Note: White spaces are not allowed *within a value*.

- Each parameter should have one entry in the configuration file. If multiple entries exist, only the first entry is used.
- Default values are used if a parameter does not exist in the configuration file.
- Characters after a comment character (#) are ignored; however, the pathname value is not ignored.

ENCCSM_CIPHERS Parameter:

The ENCCSM_CIPHERS parameter specifies the ciphers and modes to use during encryption.

syntax ENCCSM_CIPHERS

all|allbut:<list of ciphers and
modes>|cipher:mode{,cipher:mode ...}

- all: Specifies to include all available ciphers and modes, except ECB mode. For example:
ENCCSM_CIPHERS all
- allbut:<list of ciphers and modes>: Specifies to include all ciphers and modes except the ones in the list. Separate ciphers or modes with a comma. For example:
ENCCSM_CIPHERS allbut:<cbc,bf>
- cipher:mode: Specifies the ciphers and modes. Separate cipher-mode pairs with a comma. For example:
ENCCSM_CIPHERS des3:cbc,des3:ofb
- default value: allbut:<ecb>

For more information on ciphers and modes, see “Encryption Ciphers and Modes” on page 2-2.

ENCCSM_MAC Parameter:

The ENCCSM_MAC parameter specifies the MAC level to use.

default value

medium

range of values

One or more of the following options, separated by commas:

- off does not use MAC generation.
- low uses XOR folding on all messages.
- medium uses SHA1 MAC generation for all messages greater than 20 bytes long and XOR folding on smaller messages.
- high uses SHA1 MAC generation on all messages.

For example: ENCCSM_MAC medium,high

For more information on MAC levels, see “MAC Levels” on page 2-4.

ENCCSM_MACFILES Parameter:

The ENCCSM_MACFILES parameter specifies the MAC key files to use.

default value

builtin

units pathnames, up to 1536 bytes in length

range of values

One or more full path and filenames separated by commas, and the optional **builtin** keyword. For example: ENCCSM_MACFILES /usr/local/bin/mac1.dat,/usr/local/bin/mac2.dat,builtin

For more information, see “MAC Key Files” on page 2-4.

ENCCSM_SWITCH Parameter:

The ENCCSM_SWITCH parameter defines the number of minutes between cipher and key renegotiation.

syntax ENCCSM_SWITCHcipher_switch_time,key_switch_time

- cipher_switch_time specifies the minutes between cipher renegotiation
- key_switch_time specifies the minutes between secret key renegotiation

default value

60,60

units minutes

range of values

positive integers

For more information, see “Switch Frequency” on page 2-5.

Example of Encryption Parameter File:

The encryption parameter file specifies values for encryption parameters.

The following example shows an encryption parameter file:

ENCCSM_CIPHERS	all
ENCCSM_SWITCH	120,60
ENCCSM_MAC	medium
ENCCSM_MACFILE	/usr/informix/etc/MacKey.dat

The following example illustrates a line in the **conccsm.cfg** file to specify encryption with a parameter file named **encrypt.txt**:

```
ENCCSM("usr/informix/lib/cms/iencs11a.so",
"config=/usr/lib/encrypt.txt")
```

Enterprise Replication and High Availability Network Data Encryption

You can configure network data encryption for Enterprise Replication and high availability clusters by using configuration parameters.

Important: You cannot start Enterprise Replication or high availability options on a network connection that is configured to use communication support module (CSM) encryption for client/server connections. CSM encryption must be configured to use a separate network port.

You can use Enterprise Replication and high availability encryption parameters to encrypt the data traffic between the servers participating in Enterprise Replication and high availability clusters (High-Availability Data Replication, remote standalone secondary servers, and shared disk secondary servers). High availability encryption works in conjunction with Enterprise Replication encryption and each operates whether the other is enabled or not.

The following configuration parameters configure encryption for Enterprise Replication and high availability clusters:

- ENCRYPT_CIPHERS: defines all ciphers and modes that can be used by the current database session
- ENCRYPT_MAC: controls the level of message authentication code (MAC) generation
- ENCRYPT_MACFILE: specifies a list of the full path names of MAC key files
- ENCRYPT_SWITCH: defines the frequency at which ciphers or secret keys are renegotiated
- ENCRYPT_CDR: sets the level of encryption for Enterprise Replication
- ENCRYPT_HDR: enables or disables HDR encryption
- ENCRYPT_SMX: sets the level of encryption for remote standalone and shared disk secondary servers

When working in conjunction with each other, high availability and Enterprise Replication share the same ENCRYPT_CIPHERS, ENCRYPT_MAC, ENCRYPT_MACFILE and ENCRYPT_SWITCH configuration parameters.

While an encrypted high availability or Enterprise Replication connection operates from server to server, CSM network encryption operates between client and server. Both types of encryption can run on the same network if configured as follows:

- One network port must be configured for high availability.
- The other network port must be configured for CSM connections.

For information on these configuration parameters, see *IBM Informix Dynamic Server Administrator's Reference*.

Secure Sockets Layer Protocol

The Secure Sockets Layer (SSL) protocol is a communication protocol that uses encryption to provide privacy and integrity for data communication through a reliable end-to-end secure connection between two points over a network.

You can use SSL for the following connections:

- IBM Data Server Driver for JDBC and SQLJ connections with Dynamic Server
- IBM Informix ESQL/C connections with Dynamic Server
- IBM Informix ODBC Driver connections with Dynamic Server
- DB-Access connections
- Enterprise Replication connections
- High-availability data replication (HDR) connections between an HDR primary server and one or more secondary servers of any type (HDR secondary, SD secondary, or RS secondary)
- Distributed transaction connections, which span multiple database servers
- The **dbexport**, **dbimport**, **dbschema**, and **dbload** utility connections
- Connection Manager connections between servers in a cluster

The SSL protocol provides these advantages over the Dynamic Server communication support modules (CSMs):

- SSL is a more widely used alternative to the IDS CSMs.
- You can use SSL for encrypted communication with both DRDA and SQLI clients. You can use the CSMs only for connections with SQLI clients; you cannot use them for connections with DRDA clients.

You can also configure the Encrypt and Simple Password Communications Support Modules (ENCCSM and SPWDSCSM) with SSL connections. However, because these CSMs provide encryption functionality, configuring the ENCCSM or SPWDSCSM with SSL involves additional effort with no extra benefit.

You can configure Pluggable Authentication Module (PAM) and the Generic Security Services Communications Support Module (GSSCSM), which uses the Kerberos 5 security protocol for single sign-on (SSO) with SSL connections.

IDS uses the libraries and utilities provided by the IBM Global Security Kit (GSKit) for SSL communication. Some operating systems, such as Mac OS X, do not support GSKit. Therefore, refer to the IDS Machine Notes for your operating system to verify whether you can use SSL on the Dynamic Server host computer.

Digital Certificates that Exchange Keys in SSL Connections

SSL uses *digital certificates*, which are electronic ID cards issued by a trusted party, to exchange keys for encryption and server authentication.

The trusted entity that issues a digital certificate is known as a *Certificate Authority* (CA).

The CA issues a digital certificate for only a limited time. When the expiration date passes, you must acquire another digital certificate.

With SSL, the data that moves between a client and server is encrypted using a *symmetric key* (secret or private key) algorithm. An *asymmetric key* (public key) algorithm is used for the exchange of the secret keys in the symmetric algorithm.

When a client attempts to connect to a secure server, an SSL *handshake* occurs. The handshake involves the following events:

1. The server sends its digital certificate to the client.
2. The client verifies the validity of the server digital certificate. For this to occur, the client must possess the digital certificate of the CA that issued the server digital certificate.

If the handshake succeeds, these events occur:

1. The client generates a random symmetric key and sends it to the server, in an encrypted form, using the asymmetric key in the server digital certificate.
2. The server retrieves the symmetric key by decrypting it.

Because the server and the client now know and can use the symmetric key, the server and client encrypt data for the duration of the session.

Keystores that Store SSL Keys and Digital Certificates

A *keystore* is a protected database that stores SSL keys and digital certificates. Both the client and server must have the keystore that stores the digital certificates used in SSL communication.

The Server Keystore and Its Configuration

The Dynamic Server keystore stores its digital certificate and the root CA certificate of all other servers that Dynamic Server is connecting to. The server keystore must be located in the **INFORMIXDIR/ssl** directory. The name of the keystore file must be *server_name.kdb*, where *server_name* is the value specified in the DBSERVERNAME configuration parameter.

Each Dynamic Server instance must have its own keystore.

Each certificate in the keystore has a unique label. When you set up Dynamic Server to use SSL, you must specify the name of the label of the Dynamic Server digital certificate in the SSL_KEYSTORE_LABEL configuration parameter in the ONCONFIG file. If you do not specify a label name in the SSL_KEYSTORE_LABEL configuration parameter, Dynamic Server uses the default certificate in the keystore for SSL communication. Only one certificate in a keystore is the default certificate.

The keystore is protected by a password that Dynamic Server must know so that Dynamic Server can retrieve its digital certificate for SSL communications. Dynamic Server stores its keystore password in an encrypted form in a stash (.sth) file in the **INFORMIXDIR/ssl** directory. The name of the keystore stash file must be *server_name.sth*.

The password for the keystore is mandatory, because this password protects the private key for the server.

The permissions on the **INFORMIXDIR/ssl/server_name.kdb** and **\$INFORMIXDIR/ssl/server_name.sth** files must be 600, with informix set as both the owner and the group, even though Dynamic Server does not enforce these permissions.

The Client Keystore and Its Configuration

The keystore on a Dynamic Server client stores the root CA certificates of all servers to which the client is connecting. A password for the keystore is optional on the client.

For Dynamic Server SQLI clients (ESQL/C, ODBC, DB-Access, and the **dbexport**, **dbimport**, **dbschema**, and **dbload** utilities), the location of the keystore and its stash file is not fixed. Instead, the **conssl.cfg** file in the **\$INFORMIXDIR/etc** directory specifies the keystore and the stash file for Dynamic Server clients.

The following table shows the client configuration parameters that are in the **conssl.cfg** file.

Table 2-1. Client Configuration Parameters in the **conssl.cfg** File

IDS Client Configuration Parameter	Description
SSL_KEYSTORE_FILE	This is the fully qualified file name of the keystore that stores the root CA certificates of all of the servers to which the client connects.
SSL_KEYSTORE_STH	This is the fully qualified file name of the stash file containing the encrypted keystore password.

If a **conssl.cfg** file does not exist or the **SSL_KEYSTORE_FILE** and **SSL_KEYSTORE_STH** configuration parameters are not set, the client uses **\$INFORMIXDIR/etc/client.kdb** and **\$INFORMIXDIR/etc/client.sth** as the default keystore and keystore stash file names for the client.

Manage Keystores and Digital Certificates with the IBM Global Security Kit

The IBM Global Security Kit (GSKit), which is bundled with and Dynamic Server and CSDK, provides the iKeyman utility. You can use the iKeyman utility to create keystores and manage digital certificates.

The iKeyman utility is a java utility that requires IBM JDK/JRE 1.3.1, 1.4.1 or higher with the Java™ Cryptography Extension (JCE) security packages installed.

For information on using the iKeyman utility, refer to the *IBM Global Security Kit Secure Sockets Layer Introduction and iKeyman User's Guide*.

The GSKit also provides a non-java utility that you can use to administer keystores and manage digital certificates. For more information on this utility, refer to the *GSKCapiCmd User's Guide*.

You can navigate to the GSKit guides from <http://www.ibm.com/software/webservers/httpservers/library/>.

Configuring a Server Instance for Secure Sockets Layer Connections

Configure an instance of Dynamic Server for Secure Sockets Layer (SSL) connections by adding connection information to the **sqlhosts** file, setting SSL configuration parameters, and configuring the keystore and the digital certificates it stores.

To configure an instance of IDS for SSL connections:

1. Update connection information in the **sqlhosts** file (UNIX) or the SQLHOSTS registry (Windows) to include information on SSL connections. Use the:
 - **onsocssl** protocol for ESQL/C, ODBC, DB-Access, **dbexport** utility, **dbimport** utility, **dbschema** utility, or **dbload** utility connections
 - **drsocssl** protocol for DRDA connections

The following table shows an example of an **sqlhosts** file configured for both SSL and non-SSL connections.

Table 2-2. Example of sqlhosts File Configured for SSL Connections

Server Name	Protocol	Host Name	Server Name
sf_on	onsoctcp	sanfrancisco	sf_serv
oak_on	onsocssl	oakland	oak_serv
sac_on	drsocssl	sacramento	sac_serv

For more information on the **sqlhosts** file and the SQLHOSTS registry, see the *IBM Informix Dynamic Server Administrator's Guide*.

2. Update configuration parameters in the ONCONFIG file, as follows:
 - a. Specify the name of the label of the server digital certificate in the SSL_KEYSTORE_LABEL configuration parameter.

The label can contain up to 512 characters. If you do not specify a label name, Dynamic Server will use the default certificate in the keystore.

For example, specify:

```
SSL_KEYSTORE_LABEL sf_ssl
```
 - b. Configure poll threads for SSL connections using the NETTYPE configuration parameter.

If you do not configure poll threads, Dynamic Server will start one poll thread.

For the protocol, specify socssl. The protocol format is **iiiPPP**, where **iii**=[ipc|soc|tli] and **PPP**=[shm|str|tcp|spx|imc|ssl].

For example, specify:

```
NETTYPE socssl,3,50,NET
```
 - c. Configure Encrypt Virtual Processors (VPs) for SSL encryption and decryption operations, using the VPCLASS parameter.

If Encrypt VPs are not configured, Dynamic Server will start one Encrypt VP the first time an SSL operation occurs.

You can also use the **onmode -p** command to add or drop Encrypt VPs when the database server is in online mode.

Tip: For large systems, configure multiple Encrypt VPs.
3. Set up a keystore and its password stash file and digital certificate, using the IBM Global Security Kit (GSKit) iKeyman utility or the related GSKCapiCmd tool, which does not require java to be installed on the system.

When you create the password, be sure to:

 - Choose the option to stash the password to a file.
 - Name the keystore as **servername.kdb**, where *servername* is value of the DBSERVERNAME configuration parameter.
 - Create the keystore and its stash file in the **INFORMIXDIR/ssl** directory.

- Set the permissions on the `INFORMIXDIR/ssl/<server_name>.kdb` and `$INFORMIXDIR/ssl/<server_name>.sth` files to 600, with `informix` set as both the owner and the group, even though Dynamic Server does not enforce these permissions.

For information on the keystore, the password stash file, and digital certifications, see “Secure Sockets Layer Protocol” on page 2-12.

For information on the `iKeyman` utility and the `GSKCapiCmd` tool, see “Manage Keystores and Digital Certificates with the IBM Global Security Kit” on page 2-14.

Configuring a Client for SSL Connections

Configure an ESQL/C, ODBC, DB-Access, **dbexport**, **dbimport**, **dbschema**, or **dbload** connection by adding connection information to the SQL HOSTS file, setting SSL configuration parameters, and configuring the keystore and the digital certificates it stores.

Prerequisite: For general information on Secure Sockets Layer (SSL) client connections, see “Secure Sockets Layer Protocol” on page 2-12.

To configure a client connection:

1. Update connection information in the `sqlhosts` file (UNIX) or the SQLHOSTS registry (Windows), using the **onsocssl** protocol for SSL SQLI client connections.

The following table shows an example of an `sqlhosts` file configured for these client connections.

Table 2-3. Example of `sqlhosts` File Configured for SSL SQLI Client Connections

Server Name	Protocol	Host Name	Server Name
sf_on	onsoctcp	sanfrancisco	sf_serv
oak_on	onsocssl	oakland	oak_serv

For more information on the `sqlhosts` file and the SQLHOSTS registry, see the *IBM Informix Dynamic Server Administrator's Guide*.

2. Create a **conssl.cfg** configuration file, using a text editor. The file must contain the following information:
 - `SSL_KEYSTORE_FILE` information that specifies the fully qualified file name of the keystore that stores the root CA certificates of all of the servers to which the client connects
 - `SSL_KEYSTORE_STH` information that specifies the fully qualified file name of the stash file containing the encrypted keystore password.

The format of the **conssl.cfg** file is:

Parameter Value # Comment

For example, the **conssl.cfg** file might contain this information:

```
SSL_KEYSTORE_FILE  /work/keystores/ssl_client.kdb  # Keystore file
SSL_KEYSTORE_STH   /work/keystores/ssl_client.sth   # Keystore stash file
```

3. Set up a keystore and its password stash file and digital certificate, using the IBM Global Security Kit (GSKit) `iKeyman` utility or the related `GSKCapiCmd` tool, which does not require `java` to be installed on the system.

When you create the password, be sure that:

- The name and location of the keystore and its stash file are as specified in the **conssl.cfg** file.

- Permissions on the keystore and its stash file are set to 666, even though the permissions are not enforced.

For information on the keystore, the password stash file, and digital certifications, see “Secure Sockets Layer Protocol” on page 2-12.

For information on the iKeyman utility and the GSKCapiCmd tool, see “Manage Keystores and Digital Certificates with the IBM Global Security Kit” on page 2-14.

4. Add the digital certificate of the Certificate Authority that issued the server digital certificate to the keystore.

Configuring Server-to-Server Secure Sockets Layer Connections

You can configure a high-availability data replication (HDR) primary server, an HDR secondary server, a shared disk (SD) secondary server, a remote standalone (RS) secondary server, an Enterprise Replication node, or a server involved in a distributed transaction connection for Secure Sockets Layer (SSL) connections.

To configure HDR servers, Enterprise Replication nodes, or servers involved in a distributed transaction:

1. Configure each server for SSL connections. Follow the steps in “Configuring a Server Instance for Secure Sockets Layer Connections” on page 2-14.
2. In each server keystore, add the root digital certificate that the Certificate Authority (CA) issued to the other servers to the server keystore.

For example, suppose you have three servers: serv1 (the primary server), serv2 (the secondary server), and serv3 (a shared disk secondary server). Each server has its own keystore and digital certificate (**serv1.kdb** and **serv1_label**, **serv2.kdb** and **serv2_label**, **serv3.kdb** and **serv3_label**).

Add the root certificates that the Certificate Authority (CA) issued to each server to the other servers, as follows.

1. Add the root certificates issued to serv2 and serv3 to the serv1 keystore.
2. Add the root certificates issued to serv1 and serv3 to the serv2 keystore.
3. Add the root certificates issued to serv1 and serv2 to the serv3 keystore.

For information on the keystore and digital certifications, see “Secure Sockets Layer Protocol” on page 2-12. For information on the iKeyman utility and the GSKCapiCmd tool, see “Manage Keystores and Digital Certificates with the IBM Global Security Kit” on page 2-14.

Chapter 3. Column-Level Encryption

You can use column-level encryption to store sensitive data in an encrypted format. After encrypting sensitive data, such as credit card numbers, only users who can provide a secret password can decrypt the data.

Use the built-in `ENCRYPT_AES()` and `ENCRYPT_TDES()` encryption functions to encrypt data in columns containing the following character data types or smart large object data types:

- CHAR
- NCHAR
- VARCHAR
- NVARCHAR
- LVARCHAR
- BLOB
- CLOB

You can also use the `SET ENCRYPTION PASSWORD` statement to set an encryption password for a session. If you do this, only users who can provide a secret password can view, copy, or modify encrypted data.

The built-in `ENCRYPT_AES()`, `ENCRYPT_TDES()`, `DECRYPT_CHAR()`, and `DECRYPT_BINARY()` encryption and decryption functions can use the session-level password if the password is not explicitly specified in the encryption or decryption function. If you use the `SET ENCRYPTION PASSWORD` statement, you do not need to provide the same password in every encryption or decryption function.

When you set an encryption password for a session, you can also specify a password hint. If you specify a hint, you can store the hint with the encrypted password or in another location. The password must be a minimum of 6 bytes and can be a maximum of 128 bytes. The password used for decryption must match the password used for encryption.

When you set encryption passwords for column data, you can specify these types of encryption:

- **Column Level Encryption.** All values in a specific column of a database table are encrypted with the same password (word or phrase), the same encryption algorithm, and the same cipher mode. For column-level encryption, you can store the hint outside the encrypted column, rather than repeating it in every row.

Tip: If encryption functions are not used, end users could enter unencrypted data into columns that are meant to contain encrypted data. To ensure that data entered into a field is always encrypted, use views and `INSTEAD OF` triggers.

- **Cell-Level Encryption** (also called *Row-Column* and *Set-Column Level Encryption*). Within a column of encrypted data, many different passwords, encryption algorithms, or modes are used. This type of encryption might be necessary to protect personal data.

Passwords and hints that you declare with `SET ENCRYPTION PASSWORD` are not stored as plain text in any table of the system catalog. To prevent other users from

accessing the plain text of encrypted data or of a password, you must avoid actions that might compromise the secrecy of a password:

- Unless your database is accessible only by a secure network, you must enable the Encryption Communication Support Module (ENCCSM) to protect data transmission between the database server and any client system.
- Do not index encrypted columns and do not create a functional index on a decrypted column. This would store plain-text data in the database, defeating the purpose of encryption.
- Do not store passwords in a trigger or in a user-defined routine (UDR) that exposes the password to the public. Use the session password before you activate the trigger, invoke the UDR, or pass any password as a parameter to a UDR.

When you set a password, Dynamic Server transfers the password and any hint to a 128-bit key that is used to encrypt the password and hint. Passwords and hints are not stored as clear text. The key is a time-based random value per instance. The database server starts the key when the server starts; the key is destroyed when the database server shuts down.

Although it is possible to store both encrypted and unencrypted data in a single column, your application must determine which rows contain encrypted data and which rows contain unencrypted data. In addition, the application must provide for using the correct code to handle the difference, because the built-in decryption functions fail if they are applied to unencrypted data. The simplest way to avoid this error is for all rows to use encryption in a column where any row is encrypted. For more information, see the *IBM Informix Guide to SQL: Syntax*.

A query for encrypted data should specify an unencrypted column on which to select the rows. For information on queries, syntax, and reusing encrypted data, see the *IBM Informix Guide to SQL: Syntax*.

An encrypted value uses more storage space in a column than the corresponding plain text value. This occurs because all of the information needed to decrypt the value, except the encryption key, is stored with the value. Therefore, embedding zero bytes in the encrypted result is not recommended.

Dynamic Server includes an Encrypt Virtual Processor. If the encrypt option of the VPCLASS parameter is not defined in the ONCONFIG configuration file, the database server starts one Encrypt VP the first time that any encryption or decryption functions defined for column-level encryption are called. You can define multiple Encrypt VPs if necessary to decrease the time needed to start the database server. For more information, the configuration parameters chapter in the *IBM Informix Dynamic Server Administrator's Reference*.

When the database server is in online mode, you can use the **onmode -p** command to add or drop Encrypt VPs. For example, to add four more Encrypt VPs, use:

```
onmode -p 4 encrypt
```

To drop three Encrypt VPs, use:

```
onmode -p -3 encrypt
```

For more information, see the **onmode** utility chapter in the *IBM Informix Dynamic Server Administrator's Reference*.

Encrypting Column Data

You can store sensitive data in encrypted format.

Before you set the encryption password and encrypt data, you must be sure the encrypted data can fit in the column.

To encrypt a column:

1. Calculate the size of the encrypted column. If necessary, modify the column. For examples of two methods for calculating the size of an encrypted column, see “Example Showing How to Determine the Size of an Encrypted Column.”
2. Insert information on the encryption password into your code. Use the SET ENCRYPTION PASSWORD SQL statement to specify either a password or a password and a hint. Use the ENCRYPT_AES() or the ENCRYPT_TDES() function to define encrypted data. For an example of how to insert a password into your code and use the ENCRYPT function, see “Example Showing How to Encrypt a Column” on page 3-4.

Use the DECRYPT_BINARY(), and DECRYPT_CHAR() functions to query encrypted data. For an example of querying encrypted data, see “Example Showing How to Query Encrypted Data” on page 3-4.

See the *IBM Informix Guide to SQL: Syntax* for more informations about:

- The SET ENCRYPTION PASSWORD statement and the syntax to use to specify the password and the hint
- The ENCRYPT and DECRYPT functions

Example Showing How to Determine the Size of an Encrypted Column

The size of the column must be large enough to store the encrypted data.

The following example shows how the size of a **Credit Card** column is calculated:

```
DATA SIZE 16 bytes
ENCRYPTED DATA SIZE = (DATA SIZE + blocksize8) / blocksize8 *
blocksize8 = 24 bytes (integer operation)
OR ENCRYPTED DATA SIZE = (DATA SIZE - DATA SIZE% blocksize8 +
blocksize8 ) = 24 bytes
(For ENCRYPT_TDES, round up to (N + 1) * 8 bytes, for example
13 bytes round up to 16 bytes, 16 bytes to 24 bytes)
HEADER SIZE = 11 bytes (for Base64 encoding)
IV SIZE = 8 bytes (fixed size)
HINT SIZE = 32 bytes (maximum size)
ENCRYPTED HINT SIZE = 40 bytes (maximum size)

BASE64 SIZE = ((INPUT DATA SIZE + 2) / 3) * 4
(integer operation)
OR BASE64 SIZE = ((INPUT DATA SIZE + 2) -
(INPUT DATA SIZE + 2) % 3) / 3 * 4

TOTAL SIZE = HEADER SIZE
+ BASE64(IV SIZE + ENCRYPTED DATA SIZE + ENCRYPTED HINT)
= 11 + BASE64(8 + 24 + 40)
= 11 + (72 + 2) / 3 * 4
= 11 + 96 = 107
```

In the previous example, Initialization Vector (IV) is a pseudo-random series of bytes that is used to initiate encryption when using some cipher modes. IV size is the number of random series of bytes; for Dynamic Server, this is 8 bytes.

If the hint is not stored in the column, the total size in the previous example is 55 bytes.

Another way to determine the encrypted column size is to calculate as follows:

```
SELECT LENGTH(ENCRYPT_TDES
('1234567890123456",
      "password", "long...hint"))
FROM "informix".systables WHERE tabid = 1
```

Without the hint, you can calculate as follows:

```
SELECT LENGTH(ENCRYPT_TDES("1234567890123456",
"password", ""))
FROM "informix".systables
WHERE tabid = 1
```

Important: If the column size is smaller than the returned data size from ENCRYPT and DECRYPT functions, the encrypted data is truncated when it is inserted and it is not possible to decrypt the data (because the header will indicate that the length should be longer than the data received).

Example Showing How to Encrypt a Column

You can use the SET ENCRYPTION PASSWORD statement to restrict access to data in a column.

The following example shows how to use the encryption password in a column that contains a social security number:

```
create table emp
(   name char(40),
    salary money,
    ssn lvarchar(67)
);

set encryption password "one two three 123";
insert into emp values ("Alice", 50000, encrypt_aes
('123-456-7890'));
insert into emp values ("Bob", 65000, encrypt_aes
('213-656-0890'));
select name, salary, decrypt_char(ssn, "one two three 123")
from emp where name = 'Bob';
```

Example Showing How to Query Encrypted Data

You can query encrypted data with the DECRYPT function or the SET ENCRYPTION PASSWORD statement.

The following example shows how to use the decrypt function to query encrypted data:

```
select name, decrypt_char(ssn, "one two three 123") from emp;
or
set encryption password "one two three 123";
select name, salary, decrypt_char(ssn) from emp where name = 'Bob';
```

Chapter 4. Connection Security

You can prevent unauthorized connections to your database server, keep passwords secure, deploy single sign-on authentication, provide secure connections for Enterprise Replication and high availability clusters, keep local connections secure, and limit denial-of-service attacks. Most of these solutions can be used in conjunction with each other.

You can configure connection authentication using authentication modules. Depending on your platform, you can use one of the following authentication modules:

- Pluggable Authentication Module (PAM) for Dynamic Server systems running on UNIX or Linux®. These modules enable you to implement different authentication modules for different applications. See “Pluggable Authentication Modules for Systems Running on UNIX or Linux” on page 4-2.
- Lightweight Directory Access Protocol (LDAP) Authentication Support for Windows. Use the LDAP Authentication Support module when you want to use an LDAP server to authenticate users. See “LDAP Authentication Support on Windows” on page 4-4.

You can ensure that connection authentication passwords are secure by encrypting them by using a communication support module (CSM). The simple password CSM (SPWDCSM) provides password encryption. SPWDCSM is available on all platforms. See “Simple Password Encryption” on page 4-9.

If you want to support a single sign-on (SSO) environment, you can use the Generic Security Services CSM (GSSCSM) to implement a Kerberos authentication layer. In addition, the Kerberos protocol has several built-in features that can provide the same security benefits that simple password CSM and encryption CSM have. SSO authentication verifies a user's identity, and it facilitates centralized management of user IDs and passwords. If confidentiality and integrity services are enabled in GSSCSM, Kerberos authentication encrypts data transmissions and ensures transmissions are not altered between legitimate user and the database server.

Enterprise Replication and high availability connections cannot use authentication modules, but can function with these modules by restricting specific network ports to the replication and high availability connections. See “Enterprise Replication and High Availability Connection Security” on page 4-20.

You can configure IDS to check whether the ID of the user who is running the program matches the ID of the user who is trying to connect to the database. See “Secure Local Connections to a Host” on page 4-21.

You can limit the ability of denial-of-service attacks to prevent legitimate connections to the database server from being blocked. See “Limiting Denial-of-Service Flood Attacks” on page 4-21.

Pluggable Authentication Modules for Systems Running on UNIX or Linux

A Pluggable Authentication Module (PAM) is a well-defined framework for supporting different authentication modules originally developed by Sun Microsystems. PAM is supported in both 32- and 64-bit modes on Solaris, Linux, HP-UX and AIX®.

PAM enables system administrators to implement different authentication mechanisms for different applications. For example, the needs of a system like the UNIX login program might be different from an application that accesses sensitive information from a database. PAM allows for many such scenarios in a single machine, because the authentication services are attached at the application level.

In addition to enabling an application to select the authentication as needed, PAM permits module stacking. Many modules can be stacked one after another, thus enabling the application to be authenticated in multiple ways, before granting access. PAM provides a set of APIs to support authentication, account management, session management, and password management.

The system administrator can enable or disable the use of PAM. By default, the database server uses the traditional Informix authentication mechanism (which is based on the BSD rhosts mechanism) in order to avoid forcing major changes on users.

To use PAM with Dynamic Server:

- Your Informix database server must be on an operating system platform that supports PAM.
- Your client applications must be written using a sufficiently recent version of Client SDK.
- You must have the appropriate PAM service configured in the operating system.
- You must decide which PAM authentication method provides sufficient security: the client connection password, correct input to a challenge-response prompt (for example, a RADIUS authentication server), or a combination of both.
- *Linux only:* When you configure PAM to require both password and challenge-response authentication, the PAM service always ignores the password sent in the client connection request and prompts for the password a second time.
- If you require that an application authenticate in challenge-response mode before connecting to the database server, then design the application to handle the challenge prompt.
- You must ensure that Enterprise Replication and high availability clusters are not affected by PAM authentication.
- You must modify the server entry in the `sqlhosts` file for both the client application and the database server (if they are on separate machines or in separate locations on a single machine).

The Name of the PAM Service

The PAM service name identifies the PAM module.

This PAM module typically resides in the `/usr/lib/security` directory and its parameters are listed in the `/etc/pam.conf` file.

In Linux, the `/etc/pam.conf` file can be replaced with a directory called `/etc/pam.d`, where there is a file for each PAM service. If `/etc/pam.d` exists, `/etc/pam.conf` will be ignored by Linux. See the system documentation for the details of this configuration file.

Authentication Modes with the PAM Module

The PAM module determines whether a user can authenticate by providing a password, responding correctly to a challenge, or a combination of both.

The PAM implementation in Dynamic Server takes advantage of the fact that for explicit connection requests, the client sends a password to the server. You can set up PAM to make this password the only requirement for authentication to the server.

When you configure PAM to use the challenge-response protocol, authentication is complete after the user enters the correct reply to a question or other prompt. With this authentication mode, an application must be designed to respond to the challenge prompt correctly before connecting to the database server. You can set up PAM authentication to use the challenge-response mode only, so that PAM ignores the client connection password.

Linux only: If PAM is configured to authenticate users with the challenge-response protocol, the password from the client is ignored always. The PAM service on Linux prompts for the user password a second time if both password and challenge-response authentication are enabled.

PAM Required Stack Size

You can customize the stack size available for PAM modules.

The PAM feature loads operating system or third-party PAM modules (shared libraries) into the **informix** user thread. The stack size requirements of these PAM modules cannot be predicted. For instance, on Linux some modules need more than 128K of stack space. Use the `PAM_STACKSIZE` configuration parameter to customize the stack size for PAM modules.

For example, set `PAM_STACKSIZE` in the `ONCONFIG` file as follows:

```
PAM_STACKSIZE 64 # Stack size needed for the PAM modules
(K Bytes)
```

On UNIX, the default value of `PAM_STACKSIZE` is 32 KB.

On Linux, the default value is 128 KB plus the value of the `STACKSIZE` configuration parameter.

Configuring a Database Server to Use PAM

To configure a server to use PAM, the system administrator must know:

- The name of the PAM module.
- Whether the PAM module will raise a challenge in addition to accepting a simple username and password combination.

The following example shows an `sqlhosts` entry with illustrative names:

```
Authentication mode: challenge
ifxserver2 otlitcp servermc portnum2 options
  where options are "s=4, pam_serv=(pam_pass), pamauth=(challenge)"
PAM service: pam_password (Needs only a password)
Authentication mode: password
ifxserver2 otlitcp servermc portnum2 options
  where options are "s=4, pam_serv=(pam_pass), pamauth=(password)"
```

LDAP Authentication Support on Windows

LDAP Authentication on Windows is set up and configured like the Pluggable Authentication Module (PAM) that is used on UNIX and Linux. Use the LDAP Authentication Support module when you want to use an LDAP server to authenticate your system users. The module contains source code that you can modify for your specific LDAP Authentication Support module.

The authentication module is a DLL that usually resides in the %INFORMIXDIR%\dbssodir\lib\security directory. The parameters of the module are listed in the %INFORMIXDIR%\dbssodir\pam.conf file. The source code for a fully functional LDAP Authentication Module and samples of the required configuration files are included in the %INFORMIXDIR%\demo\authentication directory.

The LDAP Authentication Module provides single-module authentication only. The module does not support features such as module stacking. The system administrator can enable or disable the authentication.

Installing and Customizing the LDAP Authentication Support Module

Before you can use the Dynamic Server LDAP Authentication Module to create your authentication module, you must have an LDAP server and the LDAP client-side system. Examples of LDAP systems are IBM Directory Server and openLDAP.

Your LDAP client-side system typically includes LDAP libraries and header files. These libraries and header files are required to compile the LDAP module.

To customize the LDAP Authentication Support module:

1. Customize the **pam_ldap.c** file that is included with Dynamic Server.
2. Compile the **pam_ldap.c** file into a DLL and place it in a secure directory.

Tip: Place the **pam_ldap.c** file in the %INFORMIXDIR%\dbssodir\lib directory.

Your installation also includes a template of a configuration file, **pam_ldap_tmpl**, for the LDAP module. This configuration file contains site specific information. You should store site-specific information in this configuration file, because the file enables a single LDAP module to work in different settings.

Configuring the LDAP Module

Use the template of a PAM configuration file to configure your LDAP module.

To configure your LDAP module:

1. Copy the template file to %INFORMIXDIR%\dbssodir\etc and name it **pam.conf**.
2. Customize the file to accommodate your local security settings. See the template file, **pam.conf_tmpl**, for details about how to customize the file.

Configuring Dynamic Server for LDAP

To configure a server to use an LDAP Authentication Support module, edit the **sqlhosts** file. The system administrator must know:

- The name of the module.
- Whether the module will raise a challenge in addition to accepting a simple username and password combination.

The following example shows an **sqlhosts** entry with descriptive names:

```
PAM service: pam_chal

Authentication mode: challenge
ifxserver1 otlitcp servermc portnum1
s=4, pam_serv=(pam_chal), pamauth=(challenge)

PAM service: pam_password (Needs only a password)

Authentication mode: password
ifxserver2 otlitcp servermc portnum2
s=4, pam_serv=(pam_pass), pamauth=(password)
```

Authentication Mode with the LDAP Module

The LDAP Authentication Support module determines whether a simple password is sufficient or other challenges are required. Implementation of the module in Dynamic Server takes advantage of the fact that for explicit connections, a password is sent to the server by the client. This password can be used to satisfy the LDAP Authentication Support module in cases where a simple password is used. If the authentication mode involves responding to single or multiple challenges, the applications must be able to respond to the challenges.

Authentication Module Deployment

When you use authentication modules, you should consider the following issues:

- “Implicit Connections with Authentication Modules”
- “Application Development for Authentication Modules” on page 4-6
- “Distributed Transactions and Authentication Modules” on page 4-7
- “Client APIs and Authentication Support Modules” on page 4-8
- “Compatibility Issues with Authentication Modules” on page 4-9

Implicit Connections with Authentication Modules

Authentication responses to authentication modules, such as PAM and LDAP, expect a password. However, in implicit connections to the database server, there is no password.

PAM and LDAP are challenge oriented systems, in that the authentication response (the password) is supplied in response to a message from the authentication module. Implicit connections can work under PAM and LDAP only in challenge mode. Implicit connections in password mode will result in failure.

Application Development for Authentication Modules

The authentication method depends on the PAM or LDAP Authentication Support module installed.

The authentication method can involve challenge and response. When the PAM or LDAP Authentication Support module raises a challenge, these processes occur:

1. The database server forwards the challenge to the client.
2. The application must respond to the challenge using a callback function that is provided by an API in the IBM Informix Client Software Development Kit (Client SDK) (Client SDK), such as the Java Database Connectivity (JDBC) Driver.
3. If the server to which the client is connecting is set up for challenge, the application must register a callback function with a Client SDK component.
4. When the Client SDK API receives a challenge from the server, the challenge is forwarded to the application by the callback function.
5. The application must respond to the challenge.
6. The Client SDK component forwards the response to the database server.

The application must be prepared to respond to multiple challenges and cannot assume the number of challenges or the challenges themselves.

Syntax of the Callback Function:

```
mint ifx_pam_callback(mint (*callbackfunc_ptr)(char *challenge,  
char *response, mint msg_style))
```

char *challenge

the character buffer in which the challenge is given by the server. The size of this is fixed at 512 bytes, defined by PAM_MAX_MSG_SIZE in the **pam_appl.h** file.

char *response

the character buffer in which the response is provided by the user. The size of this is fixed at 512 bytes, defined by PAM_MAX_RESP_SIZE in the **pam_appl.h** file.

int msg_style

contains a number that indicates the type of the message given by the server. Based on the type of the response, the application can take appropriate action in the callback function.

The client application must register the callback function before making the first connection. If the callback function is not registered when the first connection is made to the database server, and the server responds, then ESQL/C returns error -1809.

The example below shows a very simple program that first registers a callback function and then unregisters it.

```
#include <stdio.h>  
#include <security/pam_appl.h>  
  
static int user_callback(char *challenge, char *response,  
int msg_style);  
  
int main(void)  
{  
    EXEC SQL char passwd[]="password";  
    int retval = 0;
```

```

/* first register the callback */
retval = ifx_pam_callback(user_callback);

if (retval == -1)
{
    printf("Error in registering callback\n");
    return (-1);
}
else
{
    EXEC SQL database test; /* successful connection */
    /* Note that this is an implicit connection. So, the
     * application should be ready to respond to challenges.*/
    printf ("sqlcode on pam connect = %d\n", SQLCODE);
}

retval = ifx_pam_callback(NULL); /* unregister the callback
                                * function */

if (retval == -1)
{
    printf("Error in registering callback\n");
    return (-1);
}
else
{
    /* This connection throw error -1809, since the callback
     * function was unregistered statement */
    EXEC SQL database test;
    printf ("sqlcode on connect = %d\n", SQLCODE);
}
return 0;
}

static int user_callback(char *challenge, char *response,
int msg_style)
{
    switch (msg_style)
    {
        /* If the msg_style is PAM_PROMPT_ECHO_OFF, the
         * application should not echo the user's response. */
        case PAM_PROMPT_ECHO_OFF:
        case PAM_PROMPT_ECHO_ON :
            printf("%s: %d:", challenge, msg_style);
            scanf("%.*s", PAM_MAX_RESP_SIZE, response);
            break;

        case PAM_ERROR_MSG:
        case PAM_TEXT_INFO:
        default:
            printf("%s: %d\n", challenge, msg_style);
            break;
    }
    return 0;
}

```

Distributed Transactions and Authentication Modules

When IDS initiates a distributed connection after the session is established, it cannot respond to authentication challenges because the timing is unpredictable. Also, the password needed to connect to the local server might not be the same as the password needed to connect to the remote server. Consequently, authentication for distributed (I-Star) connections must be completed by the remote server on the

basis of trust. The remote server must trust the local server and the remote administrators must explicitly permit the user to connect from the local server to the remote server.

The **sysauth** table in the **sysuser** database on a server records the trusted remote servers and the host on which those servers run and controls incoming connections from other servers. If PAM or an LDAP Authentication Support Module is enabled in the remote servers, the system administrator can enter authorized users in the **sysauth** table in the **sysuser** database for each remote server.

Database: **sysuser**

Table: **sysauth**

Table 4-1. Schema of the sysauth Table

Column	Type
username	CHAR(32)
groupname	CHAR(32)
servers	VARCHAR(128)
hosts	VARCHAR(128)

The table can contain multiple rows for a single user to permit connections from different servers and hosts. A unique index exists on the combination of username, servers, and hosts, none of which allow nulls. The **groupname** column should be empty; any value in the column is ignored.

For example, to permit the server to accept distributed transactions from a user known as *user1* from database server *server1* running on host *host1.example.com*:

```
insert into sysauth values ("user1", NULL, "server1", "host1.example.com");
```

For forward compatibility, ensure that each row in the table identifies one user name, one IDS server name, and one host name. Do not use comma-separated or space-separated lists of server or host names in one entry.

Client APIs and Authentication Support Modules

Only specific IBM Informix client APIs support PAM and LDAP Authentication Support modules. To use the other APIs when an authentication module is enabled on Dynamic Server, you can connect to a DBSERVERALIAS.

The following IBM Informix client APIs support PAM and LDAP Authentication Support modules:

- ESQL/C
- ODBC
- JDBC

The other APIs do not support PAM and LDAP Authentication Support modules. To use them against a version of Dynamic Server that has an enabled authentication module, connect to a DBSERVERALIAS that does not have the PAM parameters in the **sqlhosts** file.

The following client APIs, tools and applications do not support PAM or LDAP Authentication Support modules:

- LibC++
- Client DataBlade API
- OLE DB
- Visual Basic Applications using ODBC
- Ilogin and ODBC Test connection
- Informix Server Administrator

For more information on ESQL/C, ODBC, and JDBC, see the *IBM Informix ESQL/C Programmer's Manual*, the *IBM Informix ODBC Driver Programmer's Manual*, and the *IBM Informix JDBC Driver Programmer's Guide*.

Compatibility Issues with Authentication Modules

Only specific IBM Informix products support authentication modules. To use the other products when an authentication module is enabled on Dynamic Server, you can connect to a DBSERVERALIAS.

Not all IBM Informix products and tools support PAM or LDAP authentication:

- IBM Informix-4GL does not have a mechanism for identifying callback functions and therefore cannot directly support PAM or LDAP authentication. However, if IBM Informix-4GL uses the correct version of CSDK, you can write C code that can be called from IBM Informix-4GL to handle the challenge and response protocol. To implement PAM, migrate to the new CSDK version, modify your applications to register a callback that can handle challenges and responses, and recompile your application.
- Products such as Informix SQL will not handle the challenge and response protocol.
- The DB-Access, **dbexport**, **dbimport**, **dbload**, and **dbschema** utilities support PAM. If they receive a challenge, they pass the challenge to the user and wait for a response. This is repeated for each challenge that the PAM module raises.
- The **onmode**, **onstat**, and **oncheck** server administration utilities do not use PAM. However, because these utilities operate on all Dynamic Server ports, the utilities can function with a PAM-enabled port.
- Other server utilities do not support PAM.

If you are using any tools that do not support PAM or LDAP authentication modules, then make connections to a DBSERVERALIAS that does not have the PAM parameters in the SQLHOSTS file.

Simple Password Encryption

The simple password communication support module (SPWDCSM) provides password encryption.

This encryption protects a password when it must be sent between the client and the database server for authentication. SPWDCSM is available on all platforms.

You cannot use password encryption with encryption CSM (ENCCSM). For example, if you are using the SPWDCSM and decide to encrypt your network data, you must remove the entries for the SPWDCSM in your **concsn.cfg** and **sqlhosts** files.

You cannot use simple password CSM over a multiplexed connection.

CSM Configuration File

To use a communication support module (CSM), you must have a **concsm.cfg** file.

An entry in the **concsm.cfg** file is a single line and is limited to 1024 characters. After you describe the CSM in the **concsm.cfg** file, you can enable it in the **options** parameter of the **sqlhosts** file, as described in *IBM Informix Dynamic Server Administrator's Guide*.

The **concsm.cfg** file resides in the **etc** directory of **INFORMIXDIR** by default. If you want to store the file somewhere else, you can override the default location by setting the **INFORMIXCONCSMCFG** environment variable to the full pathname of the new location. For information on setting the environment variable **INFORMIXCONCSMCFG**, see the *IBM Informix Guide to SQL: Reference*.

Entries in the **concsm.cfg** file must conform to the following restrictions:

- The following characters are not allowed to be part of library pathnames:
 - = (equal sign)
 - " (double quote)
 - , (comma)
- White spaces cannot be used unless the white spaces are part of a pathname.

Configuring Password Encryption

For password encryption, you must specify password encryption libraries and connection options.

Syntax

To configure password encryption, use the following syntax to add a line to the **concsm.cfg** file.

```
►—csmname—(—"——client———clientlib——,——server———serverlib——"—,——►
               |_____|
               csmlib
►—"——"——,—"——"——")——►
   |_____|
   global_options
   |_____|
   conn_options
```

Option	Description
client=clientlib	<p>Specifies the full path and name of the shared library that is the CSM on the client computer.</p> <p>Client computers use this CSM to communicate with the database server. The library provided by Dynamic Server is \$INFORMIXDIR/lib/client/csm/libixspw.so</p>

Option	Description						
<i>conn_options</i>	<p>The <i>conn_options</i> option can be set as follows:</p> <table> <tr> <th>Setting</th><th>Result</th></tr> <tr> <td>p=1</td><td>The password is mandatory for authentication.</td></tr> <tr> <td>p=0</td><td>The password is not mandatory. If the client provides it, the password is encrypted and used for authentication.</td></tr> </table> <p>An unknown option placed in <i>conn_options</i> results in a context initialization error.</p> <p>You can put a null value in the <i>conn_options</i> field, for example: "". For CSDK before version 2.3, if the <i>conn_options</i> field is null, the default behavior is p=1. For CSDK version 2.3 and later, if the <i>conn_options</i> field is null, the default behavior is p=0.</p>	Setting	Result	p=1	The password is mandatory for authentication.	p=0	The password is not mandatory. If the client provides it, the password is encrypted and used for authentication.
Setting	Result						
p=1	The password is mandatory for authentication.						
p=0	The password is not mandatory. If the client provides it, the password is encrypted and used for authentication.						
<i>csmllib</i>	<p>The full path and name of the shared library that is the CSM if the CSM is shared by both the database server and the client computers.</p> <p>The library provided by Dynamic Server is \$INFORMIXDIR/lib/csm/libixspw.so.</p>						
<i>csmname</i>	<p>The name that you assign to the communications support module</p> <p>For example, a name could be SPWDCSM.</p>						
<i>global_options</i>	This option is not currently used.						
server=serverlib	<p>Specifies the full path and name of the shared library that is the CSM on the database server.</p> <p>The library provided by Dynamic Server is usually installed in the following directories:</p> <ul style="list-style-type: none"> • UNIX: \$INFORMIXDIR/libcsm/libixspw.so • Windows: %INFORMIXDIR%\bin\libixspw.dll 						

SMI Tables and conccsm.cfg Setting

If you want to build the SMI tables when you bring up the database server (**oninit -i**), do not specify the **p=1** option in the database server CSM entry in the **conccsm.cfg** file. The **oninit** process does not have a password for the **informix** or **root** user ID. If you specify the **p=1** option in the **conccsm.cfg** file for the database server, you receive the following error message:

```
-5013 CSM: cannot obtain credential:
authentication error.
```

To specify that the password is mandatory for the database server CSM when the SMI tables are not yet built:

1. Do not specify the **p=1** option in the **conccsm.cfg** entry.
2. Bring up the database server with the **oninit -i** command to build the SMI tables.
3. Bring down the database server.
4. Specify the **p=1** option in the database server CSM entry in the **conccsm.cfg** file.

5. Start the database server again with the **oninit** command.

Example **concsbm.cfg** Entries for Password Encryption

You must specify parameters and fields in the **concsbm.cfg** file for password encryption.

The following two examples illustrate the two alternatives for parameters that you must enter in the **concsbm.cfg** file to define the Simple Password Communication Support Module.

Alternative 1:

```
SPWDCSM
("client=/usr/informix/lib/client/csm/libixspw.so,server=/usr/informix/lib/csm/
libixspw.so", "", "")
```

Alternative 2:

```
SPWDCSM("/usr/informix/lib/csm/
libixspw.so", "", "")
```

The following example shows the *conn_options* field set to 0, so no password is necessary:

```
SPWDCSM("/work/informix/csm/libixspw.so","", "p=0")
```

Single Sign-on Authentication

Single sign-on is an authentication feature that bypasses the need to provide user name and password after a user logs in to the client computer's operating system.

IDS delivers support for single sign-on (SSO) in the Generic Security Services Communications Support Module (GSSCSM) and uses the Kerberos 5 security protocol.

With SSO, authentication for the DBMS and other SSO-enabled services happens when a user first logs in to the client computer (or domain, in the case of Windows). The Kerberos implementation validates the user credentials. Kerberos authentication generates a system of secret keys that store login credentials. When a user action tries to access a Dynamic Server database, an exchange of ticket-granting tickets (TKTs) allows database access without a login prompt.

Single sign-on authentication uses both of the following open computing standards:

- Generic Security Services Application Programming Interface (GSSAPI): an API defined by Internet Engineering Task Force (IETF) standard RFC 2743 for client-server authentication
- Kerberos security protocol: RFC 1510 that defines a typical key exchange mechanism. Applications can use the Kerberos service to authenticate their users and exchange cryptographic keys containing credentials.

SSO also includes support for confidentiality and integrity services, so an SSO environment does not need to have other Dynamic Server CSMs. With confidentiality enabled in GSSCSM, the data transmitted to and from the SSO-authenticated user is encrypted and can be viewed only by the user logged in with the authorized credentials. Integrity service ensures that data sent between user and the DBMS is not altered during transmission.

GSSCSM does not function with the simple password and encryption modules (SPWDCSM and ENCCSM). SSO implemented with GSSCSM supports PAM and LDAP, but does not support mutual authentication.

Kerberos Authentication Protocol

For single sign-on, the user login process and authentication must employ a Kerberos 5 network infrastructure, including a dedicated Key Distribution Center computer.

A complete description of the Kerberos security protocol, as well as how to configure it specifically for your system, are beyond the scope of this documentation. This topic orients users new to Kerberos implementations to the starting points for gathering required information.

Overview of Kerberos

Kerberos is a third-party network authentication protocol that employs a system of shared secret keys to securely authenticate a user in an unsecured network environment. The application server and client exchange encrypted keys (*tickets*), instead of a clear-text user ID and password pair, to establish a user's credentials on the network. A separate server referred to as the *Key Distribution Center* (KDC) issues a ticket after verifying the validity of a user login.

Each user, or *principal* in Kerberos terms, possesses a private encryption key that is shared with the KDC. Collectively, the set of principals and computers registered with a KDC are known as a *realm*.

An encrypted service ticket stores a user's credentials. The database server unencrypts the ticket to verify that the credentials are associated with a user login authorized for access. While a valid service ticket exists on the network, the IDS instance authorizes logged-in user access to the DBMS. The Kerberos protocol has the following security features:

- Service tickets exist on the network for a limited duration.
- Only the client and the server can unencrypt these tickets, which reduces risk if they are intercepted from the network.
- Input of user name and password is limited to the initial login session, reducing the risk of possible interception of clear-text credentials.

Administration of user IDs is simplified because the KDC hosts a central repository for principals. However, the disadvantage of this centralization is that it allows for a single point-of-attack by hackers. You need to weigh Kerberos' advantages against this potential threat for your own environment.

Setting up an SSO Authentication Environment

Establishing SSO authentication for Dynamic Server involves configuration of a secured Key Distribution Center computer and connectivity files, along with generation of client and server service principals.

The overall process in deploying Kerberos SSO for Informix is as follows:

1. Configure the computers on the network to function with the Kerberos 5 authentication protocol. This involves setup of a secured computer to host the Key Distribution Center (KDC). It is possible that your network already has been set up with a Kerberos mechanism.

2. Create client user principals and the Dynamic Server service principal in the KDC (see “Preparing the Informix DBMS for Kerberos Authentication”).
3. Configure the SQLHOSTS information and Generic Security Services communications support module (GSSCSM) on the computer hosting the database server (see “Configuring an IDS Instance for SSO” on page 4-15).
4. Configure the Dynamic Server service principal key and ensuring it is on the computer hosting the database server.
5. Configure a database client program that functions with GSSCSM (see “Clients Supporting SSO”).

Clients Supporting SSO

Client programs that are available in the IBM Informix Client Software Development Kit (Client SDK) can connect to Dynamic Server with SSO.

Refer to the *IBM Informix Client Products Installation Guide* for an overview of the Client SDK.

You can use the following clients with SSO:

IBM Informix ESQL/C

IBM Informix ODBC Driver

IBM Informix JDBC Driver with Sun Java Developer Kit (Sun JDK) version 1.4 onwards

IBM Informix DB-Access

Refer to “Configuring ESQL/C and ODBC Drivers for SSO” on page 4-18 and “Configuring JDBC Driver for SSO” on page 4-19 for how to set up the client programs.

Preparing the Informix DBMS for Kerberos Authentication

Configure your login process and user authentication to function with a Kerberos 5 mechanism before you set up Dynamic Server for single sign-on.

Dynamic Server SSO requires installation and setup of a Kerberos 5 authentication mechanism on the client and server computers of your network. For details on setting up your network according to the Kerberos standard, refer to the documentation provided with the installed Kerberos product.

Important: Use a secure computer for the Key Distribution Center to ensure the safety of the passwords and encryption keys. Limit access to specific users and, if possible, do not use the computer for other tasks.

JDBC Driver client sites: Read “Configuring JDBC Driver for SSO” on page 4-19 before you do the following steps.

You must have **kadmin** privileges (UNIX and Linux) or domain administrator rights (Windows) to complete steps 3 on page 4-15, 4 on page 4-15, and 5 on page 4-15.

1. *For sites that are enabling a new Kerberos 5 setup for SSO:* Run the sample client and server programs if they are available with your Kerberos product. This helps eliminate setup errors in the network infrastructure.

2. Verify that the clocks of all computers to be involved with SSO authentication are synchronized. Kerberos typically does not function when there is a clock discrepancy of five minutes or more between computers.
3. Create the Dynamic Server service and client principals on the Key Distribution Center (KDC) with the **kadmin** utility (UNIX and Linux) or with Active Directory (Windows). Remember the following as you create principals:
 - a. All principals to be used with Dynamic Server must be in the same realm or trusted realms.
 - b. All principals must map to database server user IDs. For example, if you have user5@payroll.jkenterprises as a principal, user5 must exist as an operating system user and payroll.jkenterprises.com as a fully qualified host name.
4. *UNIX and Linux only:* Add the server service principal key to the keytab file and transfer the file to the Dynamic Server host computer.
5. *UNIX and Linux only:* Put the keytab file into the default keytab file location.

Configuring an IDS Instance for SSO

Complete the following tasks for the server side of your system to enable SSO functionality with Dynamic Server:

1. “Set sqlhosts Information for SSO”
2. “Set up the conccsm.cfg File for SSO”
3. “Ensure Keytab File Has the Required Key (UNIX and Linux)” on page 4-17
4. “Verify Dynamic Server Uses Kerberos Authentication for SSO” on page 4-17

Set sqlhosts Information for SSO

This task configures the sqlhosts connectivity options so that your Dynamic Server instance can support single sign-on.

Configure the SSO connection to function with a dbserveralias. The dbserveralias that you set must match an existing value of the DBSERVERALIASES configuration parameter.

The main action of this task is to set the options field of the sqlhosts information to s=7,csm=(GSSCSM). To modify the SQLHOSTS information:

1. Open the **sqlhosts** file (UNIX and Linux) or the SQLHOSTS registry key (Windows) on the computer hosting the database server. See the *IBM Informix Dynamic Server Administrator's Guide* for details on how to set SQLHOSTS information.
2. Create a SQLHOSTS entry to the dbserveralias by specifying onsoctcp in the NETTYPE field and s=7,csm=(GSSCSM) in the OPTIONS field. For example, the following entry creates a Kerberos service for the fictional company JK Enterprises if the port number is already defined in **\$INFORMIXDIR/etc/services**:

```
ol_home2data onsoctcp jkent-005 s=7,csm=(GSSCSM)
```

You will need to configure the SQLHOSTS information on the client computer similarly. If you are using SSO in an environment where both database server and your client program are on the same computer, then you have no other SQLHOSTS tasks to complete.

Set up the conccsm.cfg File for SSO

You must specify the credentials encryption libraries of Dynamic Server in the communications support module configuration file to enable SSO. In addition, you

control whether or not SSO functions with Kerberos-defined confidentiality and integrity services in this configuration file.

Syntax

To configure the communications support module (CSM) for SSO, use the following syntax to add a line to **\$INFORMIXDIR/etc/concsm.cfg** (UNIX and Linux) or **%INFORMIXDIR%\etc\concsm.cfg** (Windows). For more information about the **concsm.cfg** file and CSM syntax rules, see “CSM Configuration File” on page 4-10.

```

>> csmname—(—"client"—clientlib—,server—serverlib—"—,
               |csmlib|
"global_options"—,—"conn_options"—)

```

Option	Description
<i>csmname</i>	<p>The name that you assign to the communications support module</p> <p>For example, a name could be GSSCSM.</p>
<i>csmlib</i>	<p>The full path and name of the shared library that is the CSM if the CSM is shared by both the database server and the client computers.</p> <p>The library provided by Dynamic Server is \$INFORMIXDIR/lib/csm/libixgss.so (UNIX and Linux) and %INFORMIXDIR%\bin\libixgss.dll (Windows).</p>
client= <i>clientlib</i>	<p>Specifies the full path and name of the shared library that is the CSM on the client computer.</p> <p>Client computers use this CSM to communicate with the database server. The library provided by Dynamic Server is \$INFORMIXDIR/lib/client/csm/libixgss.so (UNIX and Linux) and %INFORMIXDIR%\bin\libixgss.dll (Windows).</p>
server= <i>serverlib</i>	<p>Specifies the full path and name of the shared library that is the CSM on the database server.</p> <p>The library provided by Dynamic Server is usually installed in the following directories:</p> <ul style="list-style-type: none"> • UNIX: \$INFORMIXDIR/lib/csm/libixgss.so • Windows: %INFORMIXDIR%\bin\libixgss.dll
<i>global_options</i>	This option is not currently used.

Option	Description										
<i>conn_options</i>	<p>You can configure Kerberos-defined confidentiality and integrity services here or do nothing to accept the defaults. If you enter any values, you can do this for one service or for both services. The settings must be entered as comma-separated values. The <i>conn_options</i> in the concsm.cfg file are as follows:</p> <table> <tr> <th>Setting</th><th>Result</th></tr> <tr> <td>c=0</td><td>Confidentiality service of the Generic Security Services (GSS) API is disabled.</td></tr> <tr> <td>c=1</td><td>Confidentiality service is enabled. This is the default setting.</td></tr> <tr> <td>i=0</td><td>Integrity service of the GSS-API is disabled.</td></tr> <tr> <td>i=1</td><td>Integrity service is enabled. This is the default setting.</td></tr> </table>	Setting	Result	c=0	Confidentiality service of the Generic Security Services (GSS) API is disabled.	c=1	Confidentiality service is enabled. This is the default setting.	i=0	Integrity service of the GSS-API is disabled.	i=1	Integrity service is enabled. This is the default setting.
Setting	Result										
c=0	Confidentiality service of the Generic Security Services (GSS) API is disabled.										
c=1	Confidentiality service is enabled. This is the default setting.										
i=0	Integrity service of the GSS-API is disabled.										
i=1	Integrity service is enabled. This is the default setting.										

Ensure Keytab File Has the Required Key (UNIX and Linux)

Add the service principal key generated in the Key Distribution Center to the credentials information stored in the keytab file on the Dynamic Server host computer, and then validate that all necessary credentials are stored in this file.

Before you can complete this task, verify that you comply with the following prerequisites:

- A valid Dynamic Server service principal has been created on the Key Distribution Center (KDC) computer. Typically, a Kerberos principal is created by using the **kadmin** utility. Refer to your Kerberos documentation for further information.
- Client principals also exist on the KDC computer.

Important: Protect your system from intruders by maintaining appropriate security measures, such as controlling access to the keytab file.

1. Add the service principal key to the keytab file on the KDC computer.
2. Transfer the file to the keytab file for the DBMS, typically a separate computer hosting Dynamic Server.

The keytab File (UNIX and Linux):

All Kerberos server computers on UNIX and Linux must have a keytab file to authenticate with the Key Distribution Center.

A keytab file is an encrypted copy of the Dynamic Server service key. This file must be on the computer hosting Dynamic Server so that the DBMS can authenticate with the Key Distribution Center (KDC) and can accept the client's security context.

For instructions on adding a key to the keytab file, refer to the documentation provided with the Kerberos product.

Verify Dynamic Server Uses Kerberos Authentication for SSO

Before you set up the SQLHOSTS information and **concsm.cfg** file for the client computer in a single sign-on implementation, verify that your login service is correctly configured to use Kerberos authentication.

The client user principal and service principals must exist in the Key Distribution Center (KDC) to authenticate using the Kerberos tickets. Also, the KDC daemon must be running.

1. Log in using Kerberos authentication, which typically generates the required user credentials (ticket-granting ticket) for SSO on all platforms. However, if you are working on UNIX or Linux, you can also employ the **kinit** utility to obtain a ticket-granting ticket (TGT). For example, the following command could generate a TGT for the user named admin in the realm payroll.jkenterprises.com:

```
% /usr/local/bin/kinit admin@payroll.jkenterprises.com
```

2. Use the **klist** utility to view the credentials cache from the KDC and verify the existence of a valid ticket for the user ID. A valid ticket looks similar to the following example:

```
Ticket cache: FILE:/tmp/krb5cc_200
Default principal: admin@payroll.jkenterprises.com
```

```
Valid starting    Expires
01/30/08 09:45:28 01/31/08 09:45:26
Service principal
krbtgt/payroll.jkenterprises.com@jkenterprises.com
```

3. After Dynamic Server accepts a connection request, verify that a valid ticket-granting service (TGS) is present. The TGS is required for the server service principal. The following is an example of output using the **klist** utility, with ol_home2data/jkent-005.payroll.jkenterprises.com as the Dynamic Server service principal.

```
Ticket cache: FILE:/tmp/krb5cc_200
Default principal: admin@payroll.jkenterprises.com
```

```
Valid starting    Expires
01/30/08 09:45:28 01/31/08 09:45:26
Service principal
krbtgt/payroll.jkenterprises.com@jkenterprises.com
```

```
01/30/08 09:48:31 01/31/08 09:45:26
ol_home2data/jkent-005.payroll.jkenterprises.com@jkenterprises.com
```

Configuring ESQ/C and ODBC Drivers for SSO

The steps for preparing the SQLHOSTS information and the Generic Security Services (GSS) CSM configuration file for ESQ/C and ODBC and a client computer are similar to the corresponding server side setup procedures.

Complete the tasks outlined in “Configuring an IDS Instance for SSO” on page 4-15 before working on your client.

See “Clients Supporting SSO” on page 4-14 for a list of clients that support SSO with Dynamic Server.

1. Complete any setup steps specific to the client software you are using. This includes the following:
 - a. *for ESQ/C:* Include a SQLHOSTS entry specifying onsoctcp in the NETTYPE field and s=7,csm=(GSSCSM) in the OPTIONS field matching the same information for the Kerberos service in the server’s SQLHOSTS information. For example, the following entry could be valid for the company JK Enterprises if the port number is already set in **\$INFORMIXDIR/etc/services**:

```
ol_sso_krb onsoctcp jkent-005 ol_sso_svce s=7,csm=(GSSCSM)
```


- b. *for ODBC on UNIX and Linux:* Add Options=s=7,csm=(GSSCSM) to the connection settings for the SSO-enabled database server entry in the **odbc.ini** file, as illustrated in the following example:

```
Driver=/usr/informix/lib/cli/iclit09b.so
Description=IBM INFORMIX ODBC DRIVER
Database=stores_demo
ServerName=ol_sso_krb
Options="s=7,csm=(GSSCSM)"
```

- c. *for ODBC on Windows:*

- 1) Open SETNET32 (Start > All Programs > IBM Informix Client-SDK > Setnet32) and select the Server Information tab.
 - 2) Provide the connectivity information for the database server that is configured for Kerberos authentication. The entries in the fields of the Server Information tab correspond with the information in the SSO entry for the SQLHOSTS registry key, including s=7,csm=(GSSCSM) in the Options field.
 - 3) Open the ODBC Driver manager program.
 - 4) On the General tab provide your Data Source Name (DSN) details, and on the Connection tab select the SSO-enabled instance in the Server Name field.
2. Create the communications support module (CSM) configuration file for the client computer. This file must be named **\$INFORMIXDIR/etc/concsm.cfg** on UNIX and Linux platforms, and **%INFORMIXDIR%\etc\concsm.cfg** on Windows. Read the “CSM Configuration File” on page 4-10 information for details about file requirements.
3. Add a line to **concsm.cfg** for the client computer shared libraries and, if you choose, for the global and connection options. Refer to “Set up the concsm.cfg File for SSO” on page 4-15 for how to enter this configuration information.

Configuring JDBC Driver for SSO

When JDBC Driver is the client for SSO, use the `DriverManager.getConnection()` method, with an SSO connection property set to the Dynamic Server service principal.

Using IBM Informix JDBC Driver as the SSO client has been developed and tested with Sun Java Developer Kit (Sun JDK) 1.4 only.

1. Set the `DriverManager.getConnection()` method with the SSO options. The following example illustrates valid syntax for one database URL:

```
=jdbc:informix-sqli://payroll.jkenterprises.com:9555/test:
informixserver=ol_jk_ent1;CSM=(SSO=ol_jk_ent1@jkenterprises.com,ENC=true);
```

ENC in the database URL determines whether Generic Security Services (GSS) encryption is enabled or not. By default, the setting is ENC = true (encryption enabled). See the *IBM Informix JDBC Driver Programmer's Manual* for details about the `DriverManager.getConnection()` method and database URLs.

2. Create a login configuration file before running the application with the following entry:

```
com.sun.security.jgss.initiate {
    com.sun.security.auth.module.Krb5LoginModule required
    useTicketCache=true doNotPrompt=true;
};
```

Refer to your Kerberos documentation about login modules for additional options.

3. Provide the login configuration file with the -D option to run the application. The following example illustrates the format for the command, where *IfmxLog.conf* is the full path and name to the login configuration file and *TestSso* is the Java class name:

```
java -Djava.security.auth.login.config=IfmxLog.conf TestSso
```

Enterprise Replication and High Availability Connection Security

You can increase security for Enterprise Replication and high availability cluster (High-Availability Data Replication, remote standalone secondary servers, and shared disk secondary servers) connections configuring a dedicated port in the INFORMIXSQLHOSTS file.

Add **s=6** to the options (fifth) field in the INFORMIXSQLHOSTS files to indicate that the corresponding port accepts only Enterprise Replication or high availability cluster connection requests. Any other type of connection request will be rejected with error number -25539, invalid connection type.

Here is an outline of a INFORMIXSQLHOSTS file entry:

```
dbservername nettype hostname servicename s=6
```

For example:

```
ifxer1 oltlitcp mc001 er_port s=6,Other_ER_Parameters
```

When you set **s=6**, the Enterprise Replication or high availability cluster connection requests are authenticated using a new mechanism. The system administrator should create a file **hosts.equiv** file in the **\$INFORMIXDIR/etc** directory and add the names of the participating Enterprise Replication and HDR nodes (host names, as would be found in the third column of the INFORMIXSQLHOSTS file) in that file, one per line. The format of the file is similar to the UNIX **/etc/hosts.equiv** file. The file should be owned by user **informix**, belong to group **informix**, and the permissions should be restricted so that at most user **informix** can modify the file (using octal permissions, one of the values 644, 640, 444, or 440 is appropriate).

If the configuration is such that the replicating servers are on the same machine, then the **\$INFORMIXDIR/etc/hosts.equiv** file is not needed.

The following restrictions apply to this security option:

- For Enterprise Replication or high availability cluster-only ports, **s=6** should be the only security option present. No other security option (**s=0,1,2,3,4,5**) should be used when **s=6** is used.
- This option is specific to the database server environment for Enterprise Replication and high availability clusters, so it should not be used in the client environment. Clients will return an error if this option is set in SQLHOSTS file and the client attempts to use the associated server name.

Recommendation: Dedicate the database server name or a database server alias for administering the secure connectivity. For example, if you are using a high availability cluster, execute the **onmode -d primary secondary_servername** command with INFORMIXSERVER set to the secure database server or alias name. Then execute the **ontape** or **onbar** restore commands (for example, **ontape -p**) that are part of high availability cluster initialization using a different, non-secure INFORMIXSERVER setting. Likewise, use a different, non-secure INFORMIXSERVER for other client applications, such as DB-Access.

Single sign-on implemented with the Generic Security Services communication support module (GSSCSM) does not function in Enterprise Replication, High-Availability Data Replication, and other high availability cluster environments.

For information on high availability clusters, see *IBM Informix Dynamic Server Administrator's Guide*.

For information on encrypting Enterprise Replication and high availability cluster communications, see “Enterprise Replication and High Availability Network Data Encryption” on page 2-11.

For information on Enterprise Replication, see *IBM Informix Dynamic Server Enterprise Replication Guide*.

Secure Local Connections to a Host

The SECURITY_LOCALCONNECTION configuration parameter enables a database server administrator (DBSA) to set up security checking for local connections with the same host.

The table below shows the settings of the SECURITY_LOCALCONNECTION configuration parameter that you can use.

Table 4-2. SECURITY_LOCALCONNECTION Configuration Parameter Setting

Setting	Explanation
0	No security checking occurs.
1	Dynamic Server compares the user ID of the owner trying to connect with the connection user ID. If these do not match, Dynamic Server rejects the connection.
2	Dynamic Server performs the same checking that is performed when SECURITY_LOCALCONNECTION is set to 1. In addition, Dynamic Server gets the peer port number from the network API and verifies that the connection is coming from the client program. If you set SECURITY_LOCALCONNECTION to 2, you must have SOCTCP or IPCSTR network protocols.

If SECURITY_LOCALCONNECTION is set to 1 or 2, Dynamic Server establishes a connection only if the connection meets the requirements of the security check.

Limiting Denial-of-Service Flood Attacks

Dynamic Server has multiple listener threads (**listen_authenticate**) to limit denial-of-service (DOS) attacks.

These threads authenticate client requests, while the main listener thread only accepts the incoming requests and forks new threads for authentication.

You can use the MAX_INCOMPLETE_CONNECTIONS configuration parameter to configure the number of the threads authenticating at any point in time.

You can use the LISTEN_TIMEOUT configuration parameter to configure the timeout value for incomplete connections.

DOS attacks can occur when you use external mechanisms such as Telnet to connect to the port reserved for a database server. For example, if you use Telnet to connect to the port reserved for a database server service, but do not send data, and a separate session attempts to connect to the server through an application such as DB-Access, the listener thread is blocked while waiting for information from the Telnet session and the listener thread cannot accept the connection to the application used in the second session. If during the waiting period, an attacker launches a distributed DOS (DDOS) attack in a loop, you can receive a flood attack on the connection leading to poor connection performance.

LISTEN_TIMEOUT and MAX_INCOMPLETE_CONNECTIONS Configuration Parameters

You can use configuration parameters to reduce the risk of a hostile, denial-of-service (DOS) flood attack.

You can customize the following configuration parameters:

- **LISTEN_TIMEOUT**. Sets the incomplete connection timeout period. The default incomplete connection timeout period is 60 seconds.
- **MAX_INCOMPLETE_CONNECTIONS**. Restricts the number of incomplete requests for connections. The default maximum number of incomplete connections is 1024.

If you do not set the **LISTEN_TIMEOUT** and **MAX_INCOMPLETE_CONNECTIONS** configuration parameters and a flood of unauthorized attacks occurs, the Listener VP might become insecure and it might not be able to listen to a valid request in a timely manner.

If you set the **LISTEN_TIMEOUT** and **MAX_INCOMPLETE_CONNECTIONS** configuration parameters and someone tries to break into the system and reaches the maximum limit specified, the following information in the online message log tells you that the system is under attack:

```
%d incomplete connection at this time.  
System is under attack through invalid clients  
on the listener port.
```

Depending on the machine capability of holding the threads (in number), you can configure **MAX_INCOMPLETE_CONNECTIONS** to a higher value and depending on the network traffic, you can set **LISTEN_TIMEOUT** to a lower value to reduce the chance that the attack can reach the maximum limit.

You can use the **onmode -wm** or **onmode -wf** commands to change the values of these configuration parameters while the server is online. For more information, see the *IBM Informix Dynamic Server Administrator's Reference*.

Chapter 5. Discretionary Access Control

Discretionary access control verifies whether the user who is attempting to perform an operation has been granted the required privileges to perform that operation. You can perform the following types of discretionary access control:

- Create user roles to control which users can perform operations on which database objects. See “User Roles.”
- Control who is allowed to create databases. See “Setting Permission to Create Databases” on page 5-2.
- Prevent unauthorized users from registering user-defined routines. See “Security for External Routines (UDRs)” on page 5-3.
- Control whether other users besides the DBSA are allowed to view executing SQL statements. See “Enabling non-DBSAs to View SQL Statements a Session Is Executing” on page 5-3.

User Roles

A role is a work-task classification, such as payroll or payroll manager. Each defined role has privileges on the database object granted to the role. You use the CREATE ROLE statement to define a role.

After you create a role, you use the GRANT statement to grant privileges to one or more users associated with the role name.

When a role is granted to a user, the role grantor or the role grantee (user) must use the SET ROLE statement to activate the role. Only then does the user have the privileges of the role.

For more information on creating and using roles, see the *IBM Informix Guide to SQL: Syntax*.

Role Separation

When you install a database server instance, you implement role separation by setting the INF_ROLE_SEP environment variable to a non-zero integer value. Role separation enforces separating administrative tasks by people who run and audit the database server. If INF_ROLE_SEP is not set, then user **informix** can perform all administrative tasks.

You cannot switch on role separation by resetting the environment after the server instance has been installed without role separation, and you cannot selectively implement role separation on only some of the databases of the same database server.

For more information on the INF_ROLE_SEP environment variable, see the *IBM Informix Guide to SQL: Reference*. For more information about role separation, see “Using Role Separation” on page 8-3.

Default Roles

An administrator can define a default role to assign to individual users or to the PUBLIC group for a particular database.

The default role is automatically applied when a user establishes a connection with the database. This enables a user to connect to a database without issuing a SET ROLE statement.

Each user has whatever privileges are granted to the user individually, as well as the privileges of the default role. A user can switch from the current individual role to the default role using the SET ROLE DEFAULT statement.

If different default roles are assigned to a user and to PUBLIC, the default role of the user takes precedence. If a default role is not assigned to a user, the user only has individually granted and public privileges.

Granting privileges for a default role

To define and grant privileges for a default role:

1. Choose an existing role in the current database to use as a default role or create the role that you want to use as a default role. Use the CREATE ROLE *rolename* statement to create a new role in the current database.
2. Use the GRANT statement to grant privileges to the role.
3. Grant the role to a user and set the role as the default user or PUBLIC role, using the syntax GRANT DEFAULT ROLE *rolename* TO *username* or GRANT DEFAULT ROLE *rolename* TO PUBLIC.

Use the REVOKE DEFAULT ROLE statement to disassociate a default role from a user.

A user must use the SET ROLE DEFAULT statement to change any other current role to the default role.

See the *IBM Informix Guide to SQL: Syntax* for more information on using these statements.

Setting Permission to Create Databases

Use the DBCREATE_PERMISSION configuration parameter to give specified users permission to create databases and thus prevent other users from creating databases.

If you do not set the DBCREATE_PERMISSION configuration parameter, any user can create a database.

The **informix** user always has permission to create databases.

To set permission to create databases:

1. To restrict the ability to create databases to the **informix** user, add the following line to the ONCONFIG file:
DBCREATE_PERMISSION informix
2. You can include multiple instances of DBCREATE_PERMISSION in the ONCONFIG file to give additional users permission to create databases. For example, to grant permission to a user named **watsonjay**, add this line to the ONCONFIG file:
DBCREATE_PERMISSION watsonjay

Security for External Routines (UDRs)

External routines with shared libraries that are outside the database server can be security risks. External routines include user-defined routines (UDRs) and the routines in DataBlade modules.

A database server administrator (DBSA), the user **informix** by default, can implement security measures that establish which users can register external routines. This prevents unauthorized users from registering the external routines.

Use the `IFX_EXTEND_ROLE` configuration parameter to restrict the ability of users to register external routines.

The default value of the `IFX_EXTEND_ROLE` configuration parameter is 1 (or 0n).

When the `IFX_EXTEND_ROLE` configuration parameter is set to 0n:

- You can grant a user privileges to create or drop a UDR that has the `EXTERNAL` clause.
- The `EXTEND` role is operational and you can grant a user privileges to create or drop an external routine that has the `EXTERNAL` clause.

When you grant the `EXTEND` role to a specific user, the **sysroleauth** system catalog table is updated to reflect the new built-in role.

After you set the `IFX_EXTEND_ROLE` configuration parameter to 0n, a DBSA can use the following syntax to grant and revoke privileges to and from specific users.

- `GRANT extend To username`
- `REVOKE extend From username`

If you do not want to restrict UDR access, set the `IFX_EXTEND_ROLE` configuration parameter to 0 (or 0ff). When the `IFX_EXTEND_ROLE` parameter is set to 0ff, the `EXTEND` role is not operational and any user can register external routines.

The **dbimport** utility, in particular, is affected when the `IFX_EXTEND_ROLE` configuration parameter is set to 0n because a user who uses **dbimport** to create a new database has not been given an extend role on that database.

For more information, see the *IBM Informix Guide to SQL: Syntax*.

Enabling non-DBSAs to View SQL Statements a Session Is Executing

The **onstat** commands that show the SQL statement text that a session is executing are normally restricted to DBSA users. You can set the `UNSECURE_ONSTAT` configuration parameter to 1 to remove this restriction. Removing this restriction enables other users to execute **onstat -ses**, **onstat -stm**, **onstat -ssc**, and **onstat -sql** commands.

The `UNSECURE_ONSTAT` configuration parameter takes effect when the database server is shut down and restarted.

Chapter 6. Label-Based Access Control (Enterprise Edition)

Label-based access control (LBAC) is an implementation of multi-level security (MLS) that enables you to control who has read access and who has write access to individual rows and columns of data.

MLS systems process information with different security levels, permit simultaneous access by users with different security clearances, and allow users access only to information for which they have authorization. MLS is a well-known implementation of mandatory access control (MAC). If you hold the *database security administrator (DBSECADM)* role in IDS, you can configure the LBAC objects to meet your security requirements:

1. *Security policies.* You attach a security policy to a table that you want to protect from unauthorized access. To create a security policy, you define security labels that determine who can access the table's data. You can have one or more security policies on your system, depending on your organization's needs.
2. *Security labels.* You associate security labels with one or more objects in a table (*data labels*) and with users (*user labels*). When a user attempts to access an LBAC-protected table object, the system compares the user label to the data label to determine if the user can have access. If the user was not granted any label, access in most circumstances is automatically blocked.
3. *Security label components.* Security label components are the building blocks of LBAC security policies. You use these components to form security policies, which, in combination with security labels, represent different user access privileges. The variety of security label components that you can create, and the flexibility that you have in constructing security policies and security labels, offers you flexibility in the way you design your organization's LBAC solution.

LBAC complements discretionary access control (DAC). When a user attempts to access a protected table, IDS enforces two levels of access control. The first level is DAC. With DAC, IDS verifies whether the user attempting to access the table has been granted the required privileges to perform the requested operation on that table. The second level is LBAC, which controls access at the row level, column level, or both levels. The combination of DAC privileges and LBAC-protected data access granted to a user is referred to as the user's *credentials*.

This feature is available only for Dynamic Server Enterprise edition. For details on the differences between editions, see the following Web site: <http://www.ibm.com/software/data/informix/ids/ids-ed-choice/>.

Configuring Label-Based Access Control

The general procedure involves a few SQL-based tasks that define precise but flexible database security objects.

Before you implement label-based access control (LBAC), you need to identify the data that needs to be protected, who can access that data, and what tables cannot be protected.

The following list outlines the major tasks in setting up a basic implementation with IDS:

1. The database server administrator (DBSA) grants the DBSECADM role.

2. The DBSECADM defines the security objects:
 - a. Creates security label components to define the attributes of sensitive data, as well as the corresponding attributes of users who are allowed to have read access or write access to this data.
 - b. Creates security policies to reflect the organization's restrictions about who can access protected data.
 - c. Creates security labels for the security policies.
 - d. Grants security labels to users who must have access to the protected data.
 - e. *To protect new tables:* Uses the CREATE TABLE statement with the SECURITY POLICY clause and specifies how security objects protect data at the row level, column level, or at both levels.
 - f. *To protect existing tables:* Uses the ALTER TABLE statement with the ADD SECURITY POLICY clause and specifies how security objects protect data at the row level, column level, or at both levels.

Tables to Exclude from LBAC Protection

LBAC does not protect the following categories of tables:

- virtual-table interface (VTI) tables
- tables with virtual-index interface (VII)
- temporary (TEMP) tables
- typed tables
- hierarchical tables

How Security Labels Control Access

Security labels rely on security label components to store information about the classification of data and about which users have access authority.

Label-based access control (LBAC) works by comparing the labels that you have associated with users against labels that you have associated with data using a predefined rule set (IDSLBACRULES). You construct these labels with security label components, which represent different levels of data classification and access authority. Before you design an LBAC implementation, you must know how the labels store information in the components and how user operation and component type affect label comparison.

LBAC compares *values* for each user and data label when someone attempts access to a protected table. A user without a security label has a NULL value. When you create a security label, you select its values by choosing *elements* from each security label component that is part of the policy. Variations in the way you choose to group the elements provide the differing values among labels that contain the same components.

LBAC compares, one-by-one, each component value of a user label to the corresponding component value in the data label. The comparison between labels is done in the sequence that the components are listed in the labels. The comparison determines if the user label component meets the appropriate IDSLBACRULE criterion for access. When *all* the values in the user label meet the criteria for access, the user label *dominates* the data label and can work with the protected data. If any user label values do not dominate, then the user's credentials do not fit the criteria of the protecting security label. LBAC denies protected-data

access to a user with a NULL value, unless the DBSECADM has granted the user an exemption to the security policy protecting the table.

When a user attempts to retrieve data from an LBAC-protected table with a SELECT operation, the comparison follows Read Access Rules.

Read Access Rules:

- IDSLBACREADARRAY: The array component of the user security label must be greater than or equal to the array component of the data security label. The user can read data only at or below the level of the value in the array component of the user label, where level refers to the value's relative ranking in the order of array elements.
- IDSLBACREADSET: The user security label set component must include every element in the value for the set component of the data security label.
- IDSLBACREADTREE: The tree component of the user security label must include at least one of the elements in the value for the tree component of the data security label or an ancestor of one such element.

When a user attempts an INSERT, UPDATE, or DELETE operation, the comparison follows Write Access Rules.

Write Access Rules:

- IDSLBACWRITEARRAY: The array component of the user security label must be equal to the array component of the data security label. The user can write data only at the level of the value in the array component of the user label, where level refers to the value's relative ranking in the order of array elements.
- IDSLBACWRITESET: The user security label set component must include every element in the value for the set component of the data security label.
- IDSLBACWRITETREE: The tree component of the user security label must include at least one of the elements in the value for the tree component of the data security label or an ancestor of one such element.

Database Security Administrator Role

The database security administrator role (DBSECADM) is required to create and maintain label-based access control security objects.

DBSECADM is a powerful server-level role that has the following responsibilities for all databases running on the IDS installation:

- Create, drop, alter, and rename security label components
- Create, drop, and rename security policies
- Create, drop and rename security labels
- Attach security policies to tables, as well as detach security policies
- Grant security labels to users, as well as revoke security labels
- Grant and revoke exemptions from security policies
- Grant and revoke the SETSESSIONAUTH privilege

Granting the Database Security Administrator Role

A DBSA uses the GRANT DBSECADM statement to give database security administrator authority to a user.

You must be a DBSA to grant DBSECADM.

Grant the DBSECADM role by issuing the GRANT DBSECADM statement, as described in the *IBM Informix Guide to SQL: Syntax*.

The following statement gives DBSECADM authority to user sam:

```
GRANT DBSECADM TO sam;
```

Revoking the Database Security Administrator Role

A DBSA uses the REVOKE DBSECADM statement to take away database security administrator authority from a user who previously was granted this role.

You must be a DBSA to revoke the DBSECADM role. You must know the login name from whom you want to revoke the DBSECADM role.

Revoke the DBSECADM role by issuing the REVOKE DBSECADM statement, as described in *IBM Informix Guide to SQL: Syntax*.

The following statement revokes DBSECADM authority from user sam:

```
REVOKE DBSECADM FROM sam;
```

Security Label Components

Security label components are security objects for defining security policies. The elements of these components are used to define security labels, which control access to protected tables.

Security label components represent any criteria that your organization might use to decide if a user should have access to a table row or column. Typical examples of such criteria include:

- How much authority the user has in the organization
- Which confidential data, if any, the user is entitled to read or write
- To which department the user belongs
- Whether the user is involved in a particular project

Before you create security label components, you must know how your organization's privacy plan corresponds with a data classification scheme. You also must identify the security policy and security labels that you will build from the components. Data classifications that you implement through label-based access control (LBAC) map to the elements that you list when you create security label components. When a user attempts to access protected data, the label values of a user is compared to the label values of the row or column. Security label components, and their elements that are used in the security labels, specify these values.

Types of Security Label Components

There are three types of security label components:

- ARRAY: Each element represents a point on an ordered scale of relative values (see "Security Label Component Type: ARRAY" on page 6-5)
- SET: Each element represents one member of an unordered set (see "Security Label Component Type: SET" on page 6-6)
- TREE: Each element represents a node in a tree-like hierarchy (see "Security Label Component Type: TREE" on page 6-7)

As you design an LBAC solution, you identify the security label component type that best reflects the relationship among varying authority levels and groups of users. A basic LBAC implementation can draw on the organization's existing categorizations to name and group the elements, so that the elements are entities the organization already uses. As an overview, the following examples briefly describe the way security label components can function in two different situations.

Component Reflecting a Strictly Ranked Data Classification Scheme

Example: If you are creating a security label component to represent a simple, linear ranking of data-access classifications, you use a component of type ARRAY. An ARRAY-type security label component that represents four data-access classifications could have the following elements: Top Secret, Secret, Confidential, and Unclassified.

Component Reflecting an Organizational Chart

Example: The executive management of a fictional information-services corporation in the United States named "JK Enterprises" wants to limit access to specific rows of data on a database to which all employees have access. JK Enterprises has branched its national organization into regions and subregions. Much of JK Enterprises' privacy policy to be implemented with LBAC allows or denies access based on the user's affiliation with a regional level. The higher-level regions encompass larger areas of the organization. For example, an employee designated as part of the West regional level is entrusted with more authority than employees designated with the subordinate Southwest, California, and Pacific Northwest regional levels. The security label component type that best suits this set of criteria is TREE. Therefore, the user with DBSECADM authority at JK Enterprises creates a security label component named `region` and identifies the following as some of the elements for the component:

- West
- Southwest
- California
- Pacific Northwest

Because the regions of JK Enterprises encompass the entire United States, the four regions listed above compose a partial list of elements. The diagram in "Security Label Component Type: TREE" on page 6-7 illustrates all the elements of this company's `region` security label component.

Security Label Component Type: ARRAY

Security label component type ARRAY represents a ranked group of elements.

The elements in an ARRAY component represent an ordered scale of relative values; the first element listed has the highest value and the last has the lowest.

The maximum number of elements in an ARRAY type of security label component is 64. An ARRAY component of a security label has the value of only one of its elements when it is compared following the IDSLBACRULES.

Example: If the fictional company JK Enterprises defines security label component `level` as a ranking of the company's four different privacy levels, as in the following statement:

```
CREATE SECURITY LABEL COMPONENT level
  ARRAY [ 'Top Secret', 'Secret', 'Confidential', 'Unclassified' ];
```

Then Figure 6-1 illustrates the order of the elements:

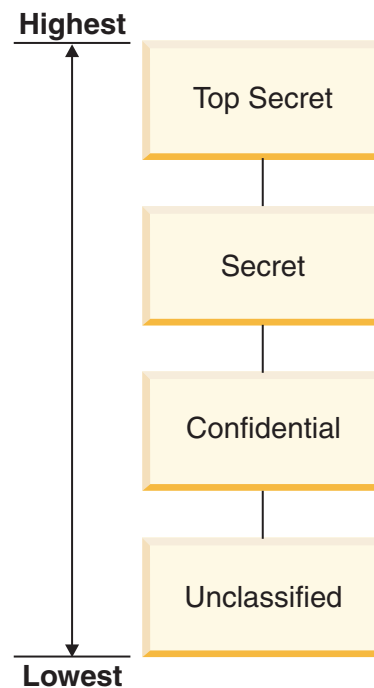


Figure 6-1. Relationship of Elements in an ARRAY Example

When an ARRAY component in the user label is compared to an ARRAY component of a data label:

- *For Read Access:* The IDSLBACREADARRAY rule lets the user component dominate when its value is *greater than or equal to* the value of the component in the data security label. The user can read data only at or below the level of the value in the array component of the user label.
- *For Write Access:* The IDSLBACWRITEARRAY rule lets the user component dominate when its value is *equal to* the value of the component in the data label. The user can write data only at the level of the value in the array component of the user label.

Security Label Component Type: SET

Security label component type SET is used to represent a group of unordered elements.

A SET-type security label component consists of an unordered list of elements. There is no ranking or other relationship among elements in this type of component.

The maximum number of elements that can exist in a SET is 64. The value of a SET-type component in a security label can consist of one or more elements.

The following SQL statement creates a SET-type security label component named function with three elements:

```
CREATE SECURITY LABEL COMPONENT function
  SET {'Developer', 'Administrative', 'Legal'};
```

When a SET component in the user label is compared to a SET component of a data label:

- *For Read Access:* The IDSLBACREADSET rule lets the user label dominate when the SET component of the user security label includes all the elements of the value for the SET component of the data security label.
- *For Write Access:* The IDSLBACWRITESET rule lets the user label dominate when the SET component of the user security label includes all the elements of the value for the SET component of the data security label.

Security Label Component Type: TREE

Security label component type TREE contains a group of elements that represent a family of parent-child relationships.

The elements in this type of security label component can be thought of as being in a tree. The first element you specify for a TREE-type component is ROOT, which represents the highest level of authority. Then you specify the other elements sequentially to follow the different levels of children and grandchildren that you want in the component.

The maximum number of elements in a TREE security label component is 64. The value of a TREE component in a label can be one or more of its nodes.

Example: JK Enterprises decides that its levels of authority to access protected data needs to follow its organizational chart. The company could use this scheme to outline its TREE security label component. The following is an example of a statement creating the region security label component:

```
CREATE SECURITY LABEL COMPONENT region
TREE ( 'USA Headquarters' ROOT,
      'West' UNDER 'USA Headquarters',
      'Central' UNDER 'USA Headquarters',
      'East' UNDER 'USA Headquarters',
      'Pacific Northwest' UNDER 'West',
      'California' UNDER 'West',
      'Pacific Southwest' UNDER 'West',
      'North Central' UNDER 'Central',
      'South Central' UNDER 'Central',
      'Northeast' UNDER 'East',
      'Mid Atlantic' UNDER 'East',
      'Southeast' UNDER 'East');
```

Figure 6-2 on page 6-8 illustrates the relationships among the TREE component elements in this example.

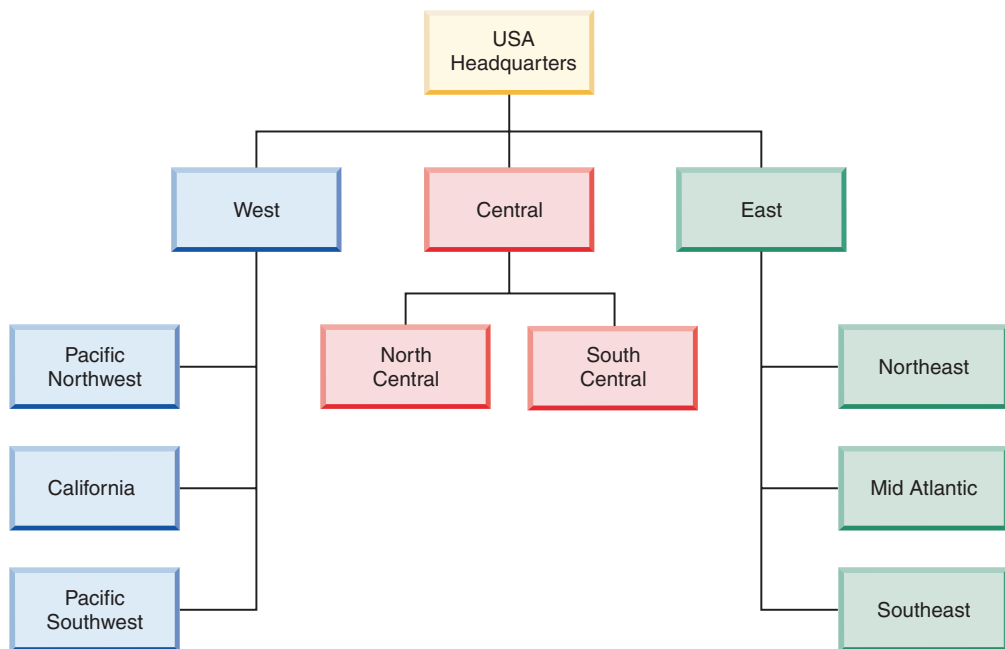


Figure 6-2. Relationship of Elements in a TREE Example

When a user label with one or more TREE components is compared to a data label with TREE components:

- *For Read Access:* The IDSLBACREADTREE rule lets the user label dominate and have read access when the label's TREE component includes at least one of the elements in the value for the tree component of the data label or the ancestor of one such element.
- *For Write Access:* The IDSLBACWRITETREE rule lets the user label dominate and have write access when each of the label's TREE components includes at least one of the elements in the value for the tree component of the data label or the ancestor of one such element.

Creating Security Label Components

The CREATE SECURITY LABEL COMPONENT statement defines this database security object.

You must hold the DBSECADM role to create security label components.

When you create a security label component you must provide the following:

- A name for the component
- The type of component it is (ARRAY, SET, or TREE)
- A complete list of elements

Create a security label component by issuing the CREATE SECURITY LABEL COMPONENT statement, as described in *IBM Informix Guide to SQL: Syntax*.

The following is an example of a CREATE SECURITY LABEL COMPONENT statement that creates a SET-type component with name department and elements Marketing, HR, and Finance:

```
CREATE SECURITY LABEL COMPONENT department
SET {'Marketing', 'HR', 'Finance'};
```


Altering Security Label Components

The ALTER SECURITY LABEL COMPONENT statement adds one or more new elements to an existing component.

You must hold the DBSECADM role to add one or more elements to a security label component, and you must know what type of component it is.

When you alter a security label component, remember the following:

- A security label component can consist of no more than 64 elements.
- If the component you want to alter is of type ARRAY or TREE, you need to know the relationships that all the elements of the resulting component will have with one another.
- The ALTER SECURITY LABEL COMPONENT statement cannot modify or drop any existing elements.

Add one or more elements to a security label component by issuing the ALTER SECURITY LABEL COMPONENT statement, as described in *IBM Informix Guide to SQL: Syntax*.

The following is an example of a ALTER SECURITY LABEL COMPONENT statement that adds to a SET-type component the elements Training, QA, and Security:

```
ALTER SECURITY LABEL COMPONENT department
  ADD SET {'Training', 'QA', 'Security'};
```

Security Policies

Security policies are database objects that you create and use to protect tables from unauthorized access.

A security policy is a named database object defined by a group of security label components.

A security policy is attached to one or more tables to allow only users with valid label-based access control credentials to read or write protected data. A user has valid credentials when the user has a security label that dominates when compared to a labeled row or column following the IDSLBACRULES. A security policy has no effect on data that has no security label.

No more than one security policy can be attached to a table, and a security policy can include no more than 16 security label components. You attach a security policy to a table via a clause in a CREATE TABLE or ALTER TABLE statement. See “Protecting Tables at the Row and Column Levels” on page 6-12 for how to attach the policy to a table.

Creating Security Policies

You create security policies after you have created security label components.

You must hold the DBSECADM role to create a security policy. The maximum number of security label components with which you can build a security policy is 16.

The order in which you list security label components when you create a security policy does not indicate any sort of precedence or other relationship among the components, but it is important to know the order when creating security labels with built-in SECLABEL functions.

Create a security policy by issuing the CREATE SECURITY POLICY statement, as described in *IBM Informix Guide to SQL: Syntax*.

The following is an example of this SQL statement, where company is the security policy name and region and department are security label components used in the policy:

```
CREATE SECURITY POLICY company  
  COMPONENTS region, department;
```

Security Labels

Security labels are objects applied to rows and columns in order to protect these data, and granted to users to give them access to protected data.

When users try to access protected data, label-based access control compares the user label to the data label. The process of this comparison is detailed in “How Security Labels Control Access” on page 6-2.

When you create a security label:

- You identify to what security policy the label belongs.
- You assign a value for each security label component in the security policy.

You apply just one label to a row or column. For a given security policy, you typically grant to a user one label to define both read and write access. But you can grant a user one label for read access and a different label for write access to data protected by the same security policy. When the read-access label differs from the write-access label granted to a user, the user can only write to data objects that can be accessed by the user’s read-access label.

Creating Security Labels

The CREATE SECURITY LABEL statement defines a new security label for a specified security policy.

You must hold the DBSECADM role to create a security label.

When you create a security label, you do the following:

- Specify a security policy to which the label belongs.
- Identify the components of that policy.
- Identify one or more elements of each component.
- Name the label.

Create a security label by issuing the CREATE SECURITY LABEL statement, as described in *IBM Informix Guide to SQL: Syntax*.

The following is an example of a CREATE SECURITY LABEL statement:

```
CREATE SECURITY LABEL company.label2  
  COMPONENT level 'Secret',  
  COMPONENT function 'Administrative',  
  COMPONENT region 'Southwest';
```

This statement defines label2 in security policy company.

Granting Security Labels

The GRANT SECURITY LABEL statement grants a security label to a user or to a list of users.

You must hold the DBSECADM role to grant a label to users. Users specified in a GRANT SECURITY LABEL statement cannot be the DBSECADM who issues it.

When you issue the GRANT SECURITY LABEL statement, you can optionally specify that the users receive the label for read access, write access, or all access. If you do not specify access, then the statement grants users an all-access label.

If a user is granted a different security label for read access than for write access, then the values given for the security label components must follow these rules:

- For security label components of type ARRAY, the value must be the same in both security labels.
- For security label components of type SET, the values given in the security label used for WRITE access must be a subset of the values given in the security label used for READ access. If all of the values are the same, this is considered a subset, and is allowed.
- For security label components of type TREE, every element in the TREE component of the security label for write access must be either an element or a descendent of an element in the TREE component of the security label for read access.

To grant a security label, refer to the documentation about the GRANT SECURITY LABEL statement in *IBM Informix Guide to SQL: Syntax*.

In the following example of this SQL statement, label2 of the company security policy is granted to user maria.

```
GRANT SECURITY LABEL company.label2
    TO maria;
```

Revoking Security Labels

The REVOKE SECURITY LABEL statement revokes a security label from a user or from a list of users.

You must hold the DBSECADM role to issue the REVOKE SECURITY LABEL statement.

When you issue the statement, you optionally can do the following:

- Revoke every security label of a security policy from users.
- Specify read-access or write-access label, or both labels, if the users have two different labels for a security policy.

Revoke a security label by issuing the REVOKE SECURITY LABEL statement, as described in *IBM Informix Guide to SQL: Syntax*.

In the following example of this SQL statement, label2 of the company security policy is revoked from user maria.

```
REVOKE SECURITY LABEL company.label2
    FROM maria;
```

Security Label Support Functions

Security label support functions are expressions for manipulating security labels.

You typically use the security label support functions (SECLABEL functions) to specify a label in data-manipulation (DML) operations on protected table rows. In these operations, however, the security label support functions do not provide any more access to protected data than is already provided by your security credentials. There are three built-in functions for label-based access control in IDS:

- The SECLABEL_BY_NAME function enables you to provide a security label directly by specifying its name.
- The SECLABEL_BY_COMP function enables you to provide a security label directly by specifying its component values.
- The SECLABEL_TO_CHAR function returns a security label in the security label string format.

You can reference a security label with these functions by providing one of the following:

- A name, as declared in the CREATE SECURITY LABEL or RENAME SECURITY LABEL statement.
- A list of values for each component of the security policy of the security label.
- An internal encoded value that the IDSSECURITYLABEL data type stores.

These functions can convert between the various forms of a security label.

See the *IBM Informix Guide to SQL: Syntax* for more information about and examples of security label support functions.

Protecting Tables at the Row and Column Levels

Protect rows and columns by associating them with security objects via clauses in the CREATE TABLE and ALTER TABLE statements.

After you have created the security objects required for your label-based access control (LBAC) implementation, you need to apply them to the tables that you want to protect. The main actions to protect the data at this stage are the following:

- Attach a security policy to each table containing data to be protected by LBAC.
- Associate the necessary rows and columns with security labels.

Data in a table can only be protected by security labels that are part of the security policy protecting the table. Data protection, including attaching a security policy to a table, can be done when creating the table or later by altering the table.

Protected Table with Row Level Granularity

A table can be marked as protected with row level granularity during CREATE TABLE or ALTER TABLE by attaching a security policy and by specifying the security label column. The security label column must be of the IDSSECURITYLABEL data type.

If users attempt to access a row to which they do not have the required LBAC credentials, the system responds to the users as if the row did not exist.

Protected Table with Column Level Granularity

A database table can be marked as protected with column level granularity during CREATE TABLE or ALTER TABLE by attaching a security policy to such table and by attaching a security label to one or more columns of that table. When a column is associated with a security label, that column is referred to as a *protected column*. The security policy attached to the table affects what security label can be applied to the column.

If users attempt to access a column to which they do not have the required LBAC credentials, the system generates an error message.

Security Label Column (IDSSECURITYLABEL Data Type)

The column holding the label for row level granularity must be of the IDSSECURITYLABEL data type. Only a user who holds the DBSECADM role can create, alter, or drop a column of this data type. IDSSECURITYLABEL is a built-in DISTINCT OF VARCHAR(128) data type. A table that has a security policy can have only one IDSSECURITYLABEL column.

The following cannot be applied to a security label column:

- Referential constraints
- Check constraints
- Primary key or unique constraints if the security label column is the only column in constraint
- Column protection
- Encryption

For more information about the IDSSECURITYLABEL data type, see the *IBM Informix Guide to SQL: Reference* and *IBM Informix Guide to SQL: Syntax*.

Simultaneous Row and Column Level Protection on a Table

A protected table can be defined with both row and column level granularities. If both row and column granularity are applied to a table, then LBAC enforces column before row level access control.

You can apply row and column level protection on a table in a single statement rather than issuing separate statements for the two granularities when you do the either of the following:

- When you create a new LBAC-protected table
- When you alter a table to add row level protection in addition to the existing column level protection

The following is an example of a CREATE TABLE statement and a ALTER TABLE statement that set up two tables with both row and column level protection.

```
CREATE TABLE T5
  (C1 IDSSECURITYLABEL,
   C2 int,
   C3 char (10) COLUMN SECURED WITH label6)
SECURITY POLICY company;

ALTER TABLE T6
ADD ( C1 IDSSECURITYLABEL),
MODIFY (C2 INT COLUMN SECURED WITH label7),
ADD SECURITY POLICY company;
```

For more information about how these statements work, see “Applying Row Level Protection,” “Applying Column Level Protection,” *IBM Informix Guide to SQL: Reference*, and *IBM Informix Guide to SQL: Syntax*.

Applying Row Level Protection

Protect row level data by associating the table with a security policy and inserting an IDSSECURITYLABEL-type column.

There are two methods for applying row level protection:

1. For a *new* table: Use the CREATE TABLE statement with the appropriate IDSSECURITYLABEL and SECURITY POLICY clauses, as described in *IBM Informix Guide to SQL: Syntax*.
2. For an *existing* table: Use the ALTER TABLE statement with the appropriate IDSSECURITYLABEL and ADD SECURITY POLICY clauses, as described in *IBM Informix Guide to SQL: Syntax*.

The following is an example of a statement that applies row level protection when you create a new table (T1), using the security policy named company and the security label named label2.

```
CREATE TABLE T1
  (C1 IDSSECURITYLABEL,
   C2 int,
   C3 char (10))
  SECURITY POLICY company;
```

The following statement provides an example of applying row-level protection on a table (T2) that already exists on the database, using the security policy named company. The default value for C1 is label3.

```
ALTER TABLE T2
  ADD (C1 IDSSECURITYLABEL DEFAULT label3),
  ADD SECURITY POLICY company;
```

Applying Column Level Protection

Protect column level data by associating the table with a security policy and attaching a security label to one or more columns.

There are two methods for applying column level protection:

1. For a *new* table: Use the CREATE TABLE statement with the COLUMN SECURED WITH and SECURITY POLICY clauses, as described in *IBM Informix Guide to SQL: Syntax*.
2. For an *existing* table: Use the ALTER TABLE statement with the MODIFY (*your_column* COLUMN SECURED WITH) and ADD SECURITY POLICY clauses, as described in *IBM Informix Guide to SQL: Syntax*.

The following is an example of a statement that applies column level protection when a new table (T3) is created, using the security policy named company and a security label named label4.

```
CREATE TABLE T3
  (C1 CHAR (8),
   C2 int COLUMN SECURED WITH label4,
   C3 char (10))
  SECURITY POLICY company;
```

The following statement provides an example of applying column-level protection on a table (T4) that already exists on the database, using the security policy named `company` and a security label named `label5`.

```
ALTER TABLE T4
  MODIFY (C1 CHAR (8) COLUMN SECURED WITH label5),
  ADD SECURITY POLICY company;
```

Exemptions

Exemptions modify security credentials of users by disabling one or more of the IDSLBACRULES for a component type in a security policy.

Since exemptions are based on a security label component type for a particular security policy, this exemption does not apply outside that security policy. Within the security policy, the exemption applies to *all* instances of the component type.

Exemptions can be useful for letting trusted users do administrative work for which otherwise it would be cumbersome to grant all necessary label-based access control credentials. For example, if your job is to classify incoming data, a typical practice would be for the DBSECADM to grant you exemptions so that you can write to any data row in the security policy.

If users hold an exemption to every rule of a security policy, then they will have complete access to all data protected by that policy.

Exemptions provide very powerful access. Do not grant them without careful consideration.

Granting Exemptions

The GRANT EXEMPTION statement gives a user an exemption from one or more access rules of a security policy.

You must hold the DBSECADM role to grant exemptions.

Grant an exemption by issuing the GRANT EXEMPTION statement, as described in the *IBM Informix Guide to SQL: Syntax*.

The following statement grants user `maria` an exemption from the IDSLBACWRITETREE rule in security policy `company`:

```
GRANT EXEMPTION
ON RULE IDSLBACWRITETREE
FOR company
TO maria
```

To grant a user exemptions from all IDSLBACRULES of a security policy, specify `ALL` in place of the policy name in the statement. Typically, this type of exemption is practical for a user who is responsible for loading and unloading data in protected tables.

Revoking Exemptions

The REVOKE EXEMPTION statement revokes from a user an exemption on one or more access rules of a security policy.

You must hold the DBSECADM role to revoke exemptions.

Revoke an exemption by issuing the REVOKE EXEMPTION statement, as described in the *IBM Informix Guide to SQL: Syntax*.

The following statement revokes from user maria an exemption from the IDSLBACWRITETREE rule in security policy company:

```
REVOKE EXEMPTION ON RULE IDSLBACWRITETREE FOR company FROM maria
```

To revoke all IDSLBACRULES exemptions that a user has for a security policy, specify ALL in place of the policy name in the statement.

Maintaining a Label-Based Access Control Implementation

Optimizing database performance can require adjusting the values of configuration parameters for security policies and user credentials.

LBAC Cache Tuning

Poor performance of a database with tables protected by label-based access control (LBAC) can indicate that the system is needlessly relying on disk operation more than on LBAC-related caching to retrieve information from memory.

Fine-tuning one or more of the following parameters in the ONCONFIG file can improve performance for queries frequently executed on protected tables. For example, if the value for the PLCY_HASHSIZE parameter is set too low, there are not enough hash buckets allocated for security policy information caching and consequently some database performance involving LBAC-protected tables declines.

PLCY_HASHSIZE

The PLCY_HASHSIZE parameter specifies the number of hash buckets in the security policy information cache.

onconfig.std value

31

units

kilobytes

range of values

Any positive integer

takes effect

When the database server is shut down and restarted

refer to

IBM Informix Dynamic Server Performance Guide for information about configuration effects on memory

PLCY_POOLSIZE

The PLCY_POOLSIZE parameter specifies the maximum number of entries in each hash bucket of the security policy information cache.

default onconfig.std value

127

units

kilobytes

range of values

16 or higher positive integer value

takes effect

When the database server is shut down and restarted

refer to

IBM Informix Dynamic Server Performance Guide for information about configuration effects on memory

USRC_HASHSIZE

The USRC_HASHSIZE parameter specifies the number of hash buckets in the LBAC credential memory cache. This memory cache holds information about users' LBAC credentials.

onconfig.std value

31

units

kilobytes

range of values

Any positive integer

takes effect

When the database server is shut down and restarted

refer to

IBM Informix Dynamic Server Performance Guide for information about configuration effects on memory

USRC_POOLSIZE

The USRC_POOLSIZE parameter specifies the maximum number of entries in each hash bucket of the LBAC credential memory cache. This memory cache holds information about users' LBAC credentials.

default onconfig.std value

127

units

kilobytes

range of values

16 or higher positive integer value

takes effect

When the database server is shut down and restarted

refer to

IBM Informix Dynamic Server Performance Guide for information about configuration effects on memory

Dropping Security Objects

Use the DROP SECURITY statement to remove a security label component, a security policy, or a security label from the database.

You must hold the DBSECADM role to remove a security object.

Three valid keyword definitions of the DROP SECURITY statement are as follows:

1. DROP SECURITY POLICY *policy* removes a security policy; this can be used in RESTRICT and CASCADE modes
 - Example: DROP SECURITY POLICY company removes the policy named company from the database
2. DROP SECURITY LABEL *policy.label* removes a security label; this can be used in RESTRICT mode
 - Example: DROP SECURITY LABEL company.label2 removes the label named label2.
3. DROP SECURITY LABEL COMPONENT *component* removes a security label component; this can be use in RESTRICT mode
 - Example: DROP SECURITY LABEL COMPONENT department removes the component department.

For more information about the DROP SECURITY statement, including details about the RESTRICT and CASCADE modes, see *IBM Informix Guide to SQL: Syntax*.

When the DROP SECURITY statement executes successfully, the database server deletes any rows that reference the name or the numeric identifier of the specified object from the tables of the system catalog, including the following tables:

- **syssecpolicies** for security policies
- **sysseclabels** for security labels
- **sysseclabelcomponents** for security label components

Renaming Security Objects

Use the RENAME SECURITY statement to rename a security policy, a security label, or a security label component.

You must hold the DBSECADM role to rename a security object.

The three valid clauses for the RENAME SECURITY statement are as follows:

1. POLICY *old_name* TO *new_name* renames a security policy
 - Example: RENAME SECURITY POLICY company TO subsidiary; renames the policy named company to subsidiary
2. LABEL *security_policy.old_name* TO *new_name* renames a security label; in this statement you also indicate the security policy to which the label belongs
 - Example: RENAME SECURITY LABEL subsidiary.label8 TO label9; renames label8 to label9, which belongs to security policy subsidiary
3. LABEL COMPONENT *old_name* TO *new_name* specifies a security label component
 - Example: RENAME SECURITY LABEL COMPONENT department TO division; renames the component department to division.

For more information about the RENAME SECURITY statement, see *IBM Informix Guide to SQL: Syntax*.

The RENAME SECURITY statement replaces the *old_name* with the specified *new_name* in the table of the system catalog in which the renamed security object is registered:

- **syssecpolicies.secpolicyname** for security policies
- **sysseclabels.seclabelname** for security labels
- **sysseclabelcomponents.compname** for security label components.

This statement does not, however, change the numeric value of the `syssecpolicies.secpolicyid`, `sysseclabels.seclabelid`, or `sysseclabelcomponents.compoid` of the renamed security object.

IDS Security Considerations for Label-Based Access Control

The wide range of IDS capabilities requires certain precautions and planning to ensure protected tables can be accessed appropriately.

The following actions require holding the DBSECADM role after you have implemented label-based access control (LBAC) on your database server:

- Using the SETSESSIONAUTH privilege (see “SET SESSION AUTHORIZATION”)
- Exporting schema and data (see “The dbschema, dbexport, and dbimport Utilities” on page 6-20)
- Importing data (see “The dbschema, dbexport, and dbimport Utilities” on page 6-20)

These actions require the user to have read and write access credentials:

- Backing up and restoring with **onbar** and **ontape** utilities (see “Backup and Restore” on page 6-20)
- “Data Loading and Unloading” on page 6-20

To prevent unauthorized access to protected tables, take extra precautions with the following database operations and objects:

- “The onlog Utility” on page 6-21
- “The oncheck Utility” on page 6-21
- “Enterprise Replication” on page 6-21
- “Data Definition Language (DDL) Operations” on page 6-21
- “INSERT INTO . . . SELECT FROM Statement” on page 6-21
- High Performance Loader .RET and .FLT files (see “Data Loading and Unloading” on page 6-20)
- “Temporary Tables Created by the INTO TEMP Clause” on page 6-21
- “User-Defined Routines” on page 6-21 created with DBA keywords

SET SESSION AUTHORIZATION

The SET SESSION AUTHORIZATION statement allows you to assume the identity of another user, including the user’s LBAC credentials for protected tables.

IDS 11.10 and later versions that have label-based access control (LBAC) capability handles the SETSESSIONAUTH privilege differently from legacy versions of the database server that did not have LBAC functionality. The newer versions of IDS require the DBSECADM to grant the SETSESSIONAUTH privilege. Because the SETSESSIONAUTH privilege can be used to assume the LBAC credentials of another user, the DBSECADM should be careful in granting the SETSESSIONAUTH privilege.

If the database server has been converted from a legacy version that did not support LBAC, users who held the DBA privilege are automatically granted the SETSESSIONAUTH access privilege for PUBLIC in the migration process. You

must initialize the converted server as a version that supports LBAC security policies to remove the SETSESSIONAUTH privilege from all DBAs and enable the DBSECADM role to grant this privilege.

For more information about how SET SESSION AUTHORIZATION operates with LBAC, see *IBM Informix Guide to SQL: Syntax*.

Backup and Restore

Users who are responsible for backing up or restoring protected data with an **onbar** and **ontape** utilities must have LBAC read-and-write access credentials for the corresponding server tables. LBAC security remains intact during backup and after being restored on the server, but to protect the saved backup data you need to take other precautions.

IDS allows restoration of a specific table or set of tables that have previously been backed up with **onbar** or **ontape**. These tables can be restored to a specific point in time. During table-level restore of LBAC-protected tables, ensure that the schema command files specify the security policy with the target table. As a protected target table will be created during the restore, the user running the table level restore must hold the DBSECADM role. Also, LBAC rules will be enforced when the INSERT statement from the schema command file is executed to load the target table. If the entire table is to be restored, the user must possess the necessary LBAC credentials.

You cannot use the **archecker** utility to perform a table-level restore.

The dbschema, dbexport, and dbimport Utilities

LBAC rules will be enforced on protected tables when the **dbschema** and **dbexport** utilities are run. Only those rows will be unloaded where the user's security label dominates the column label, row label, or both. Since both **dbschema** and **dbexport** utilities need to read LBAC catalogs, the user running these utilities must have the appropriate LBAC credentials or exemptions to access the data.

The **dbimport** utility creates and populates a database from text files. The user importing LBAC-protected data with this utility must have the DBSECADM role. After the import process is complete, the DBSECADM role does not have any exemptions that were defined prior to the import process.

Data Loading and Unloading

IDS provides a number of ways to load and unload data. Some of these methods are:

- **dbload** utility
- **onpload** and **ipload** utilities for High-Performance Loader (HPL)

LBAC rules are applied when these statements and utilities are executed on protected tables. The user's security label must dominate the column label, row label, or both. If an entire table is to be loaded/unloaded, then the user must have the necessary LBAC credentials to read and write all the labeled rows and columns. Alternatively, the DBSECADM can grant an exemption to the user so that the security policy protecting the tables can be bypassed.

Rows that are rejected when the **onpload** utility is executed are dumped to .REJ and .FLT files. Take the necessary precautions to prevent unauthorized access to

these files. For express-mode loads using HPL, the rows are inserted directly to table extents skipping the SQL layer. The user running the express-mode load must be granted the necessary exemptions to bypass the security policy.

You cannot use the **onload** and **onunload** utilities with LBAC.

The onlog Utility

The **onlog** utility displays all or selected portions of the logical log. This command can take input from selected log files, the entire logical log, or a backup tape of previous log files. The log records can expose data that is protected by LBAC on a live database. Take precautions to ensure data is not exposed by misuse of this utility.

The oncheck Utility

The **oncheck** utility can display pages from tables or chunks, which can expose data that is protected by LBAC on a live database. Take precautions to ensure data is not exposed by misuse of this utility.

Enterprise Replication

You cannot apply LBAC to a table participating in Enterprise Replication. Also, you cannot define an Enterprise Replication replicate on a table that is protected by LBAC.

Data Definition Language (DDL) Operations

LBAC does not restrict users on your system from performing data definition language (sometimes called *data definition statements*) operations. For example, a user whom has not been granted security policy credentials or an exemption can run TRUNCATE TABLE or DROP TABLE on an LBAC-protected table.

INSERT INTO . . . SELECT FROM Statement

When the INSERT INTO . . . SELECT FROM statement is used on an LBAC-protected table to create another table, ensure that the new table is protected by the same security policy used to protect the source table. Otherwise, the new table can potentially expose data in violation of your organization's privacy policy. Note that this potential data exposure can happen if the statement is used to create a permanent table, or to create a temporary table and then inserted into a permanent one.

Temporary Tables Created by the INTO TEMP Clause

The INTO TEMP clause of the SELECT statement creates a temporary table to hold the query results. If the table being selected from is a protected table, the query-result data in the intermediate temporary table is not protected by LBAC. Take the necessary precautions to ensure that the data in the temporary table is not exposed to unauthorized users.

User-Defined Routines

IDS allows registration of user-defined routines (UDRs) with the DBA keyword. If a user is granted the execute privilege on a UDR, the database server automatically grants the user temporary DBA privileges that are enabled only when the user is

executing the UDR. The user executing the DBA UDR assumes the identity of a DBA for the duration of the UDR and will therefore have the DBA's user label during that time. Avoid using protected tables in DBA UDRs.

Other IDS Functionality with Label-Based Access Control

IDS has non-security functionality that operates seamlessly with label-based access control.

IDS label-based access control (LBAC) is designed to work smoothly with all parts of the database server and without excessive user intervention to contain unauthorized data exposure. The following areas of IDS are highlighted to address potential areas of concern.

High-Availability Clusters

High-availability clusters (High-Availability Data Replication, shared disk secondary servers, and remote standalone secondary servers) provide a way to provide one or more copies of the database server. LBAC objects created on a database of the primary server are replicated to the secondary servers. All tables protected on the primary server will also be protected on the secondary servers.

Distributed Queries

IDS allows you to query more than one database on the same database server or across multiple database servers. This type of query is called a distributed query. LBAC rules will be applied to distributed queries involving protected tables and local synonyms of remote protected tables. Queries issued from a non-LBAC server but involving LBAC-protected tables on a different server will also require that the user have the necessary LBAC credentials to access the protected data on the other server.

Fragmentation

Fragmentation is a database server feature that allows you to control where data is stored at the table level using a fragmentation strategy. IDS ensures that the source and targets tables have the required identical LBAC security objects for attaching and detaching fragments.

Synonyms and Views

Views and synonyms can be created on existing tables and views that reside in the current database, or in another database of the local database server or of a remote database server. LBAC rules will be applied when a user attempts to access data through views and synonyms on protected tables.

Violations Tables

IDS provides a facility to track rows that violate constraints. The `START VIOLATIONS TABLE` statement creates a special violations table that holds nonconforming rows that fail to satisfy constraints and unique indexes during `INSERT`, `UPDATE`, and `DELETE` operations on target tables. In order to prevent unauthorized exposure of protected data through a violations table, IDS secures the violation table with same security policy as the target table when the `START VIOLATIONS TABLE` statement is executed.

Referential Integrity Scans

LBAC rules are applied when the ON DELETE CASCADE option is specified and when an INSERT statement to a child table generates a referential integrity scan on the parent table.

Part 2. Auditing Data Security

This section discusses how to audit the security of your database.

Chapter 7. Overview of Auditing

This chapter provides an overview of auditing and of auditing terminology. It describes audit events, explains in detail how audit masks are configured and used, and indicates how to perform audit analysis. It also introduces the various audit administration roles.

Secure-Auditing Facility

Auditing creates a record of selected activities that users perform. An audit administrator who analyzes the audit trail can use these records for the following purposes:

- To detect unusual or suspicious user actions and identify the specific users who performed those actions
- To detect unauthorized access attempts
- To assess potential security damage
- To provide evidence in investigations, if necessary
- To provide a passive deterrent against unwanted activities, as long as users know that their actions might be audited

Important: Make sure that users know that every action they perform against the database can be audited and that they can be held responsible for those actions.

You cannot use auditing to track transactions to reconstruct a database. The database server has archive and backup facilities for that purpose. The *IBM Informix Backup and Restore Guide* explains these facilities.

Audit Events

Any database server activity that could potentially alter or reveal data or the auditing configuration is considered an *event*. The database server secure-auditing facility lets you audit and keep a record of events either when they succeed or fail, or simply when the activity is attempted. You can identify each audit event by a four- or five-letter event code called an *audit-event mnemonic*. Chapter 11, “Audit Event Mnemonics and Fields,” on page 11-1 lists the audit-event mnemonics and describes the events that you can audit with the secure-auditing facility.

You can specify events that you want to audit in an *audit mask*. Auditing is based on the notion of audit events and audit masks.

Audit Masks

Audit masks specify those events that the database server should audit. You can include any event in a mask. The masks are associated with user IDs, so that specified actions that a user ID takes are recorded. Global masks `_default`, `_require`, and `_exclude` are specified for all users in the system.

Before you use auditing, you need to specify which audit events to audit. To specify audited events, add the events to the masks. You also need to perform other tasks, which Chapter 8, “Audit Administration,” on page 8-1 describes.

The database server does not provide auditing for objects or processes. For example, you cannot ask the database server to audit all access attempts on a certain object. You can, however, filter audit records from the audit trail based on objects with the audit-analysis tools, which Chapter 9, “Audit Analysis,” on page 9-1, describes.

Figure 7-1 represents a set of audit masks. The actual masks and their features are explained in “Audit Masks and Audit Instructions” on page 7-4.

After installation:

- Create audit masks
- Turn on auditing

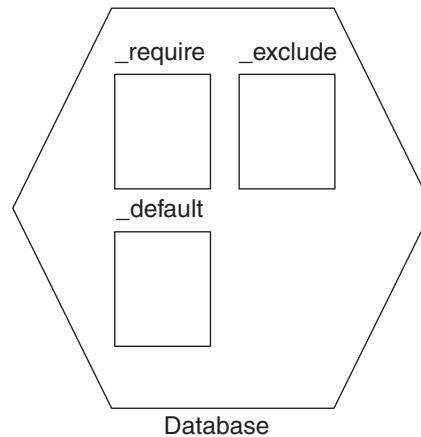


Figure 7-1. Audit Masks After Installation

After installation is complete, you can create the audit masks and turn on auditing.

Important: If auditing is off, the database server does not audit any events, even if events are specified in the masks.

In addition to the three masks that Figure 7-1 shows, you can specify *user masks* for individual users. User masks enable you to audit some users more than others and target different types of activities for different users. Except for the audit administrator who maintains the masks, a user cannot tell which events are being audited. For a description of user masks, see page “User Masks” on page 7-5.

You can also create *template masks* to create new user masks. For a description of template masks, see page “Template Masks” on page 7-6.

Masks and their events are called *auditing instructions*, as Figure 7-2 on page 7-3 shows. You have significant flexibility regarding the auditable facets of Dynamic Server. You can select anything from minimal audit instructions, in which no events are audited, to maximal audit instructions, in which all security-relevant database server events are audited for all users.

Defining masks:

- You must specify the events to audit within one or more audit masks.
- You can create masks for individual users.
- You can change the audit instructions during regular system operation.
- You can change a single mask during regular system operation.

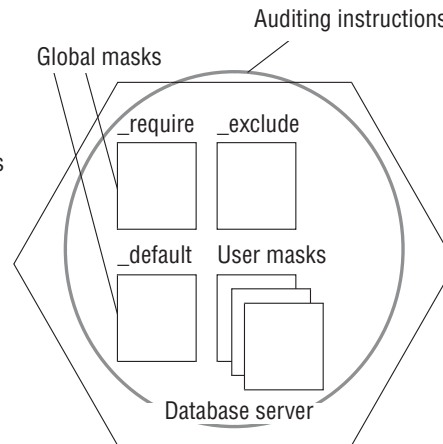


Figure 7-2. The Auditing Instructions

After you define the auditing instructions and turn on auditing, you can modify one or more audit masks as needs change and you identify potential security threats. For information on how to change audit masks, see Chapter 8, “Audit Administration,” on page 8-1.

Audit Process

When you turn on auditing, the database server generates *audit records* for every event that the auditing instructions specify, as Figure 7-3 shows.

The database server stores the audit records in a file called an *audit file*, as Figure 7-3 shows. The collection of audit records makes up the *audit trail*. (The audit trail might consist of more than one audit file.)

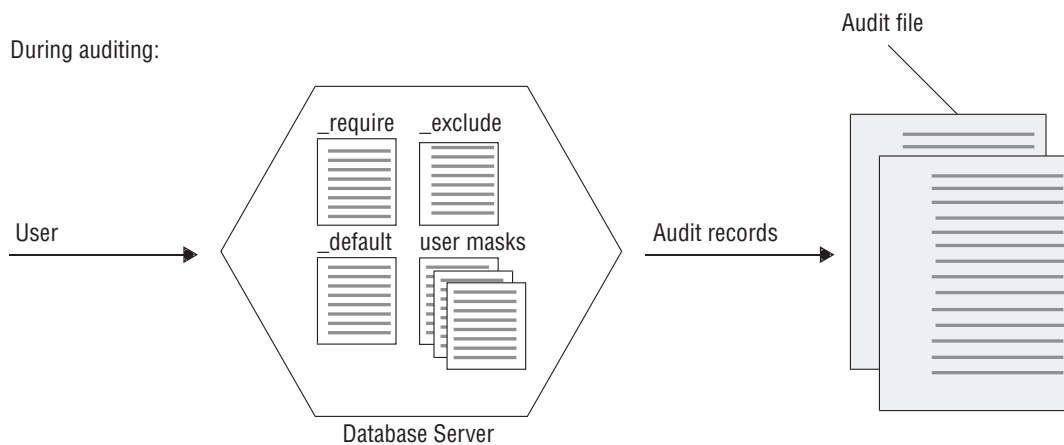


Figure 7-3. The Audit Process

An audit administrator needs to specify and maintain the *audit configuration*, which includes the following information:

- The audit mode
- How the database server behaves if it encounters an error when writing audit records to the audit trail
- For UNIX, the directory in which the audit trail is located
- For UNIX, the maximum size of an audit file before the database server automatically starts another audit file

These topics are explained in “Audit Configuration” on page 7-8.

The database server generates audit records and writes them to the audit file or to an event log regardless of whether the client user that performs the audited action is local or remote. The database server includes both the user login and database server name in every audit record to help pinpoint a specific initiator and action.

In high availability clusters, only the primary database server performs secure auditing and produces an audit trail. The **onaudit** utility runs on the secondary servers but does not audit any of the audit events.

Audit Trail

Review the audit trail regularly. The database server offers a data-extraction utility, **onshowaudit**, that you can use to select audit data for specific users or database servers.

After you extract data, you can specify that it be formatted to load into a database for subsequent manipulation with SQL. “Audit Analysis Overview” on page 7-14 explains this process.

Roles for Database Server and Audit Administration

The operating-system administrator (OSA) can set up the following roles for database server administration and audit administration, in addition to any administrative roles that your operating system might have:

- The database server administrator (DBSA) maintains and tunes the database server
- An audit administrator can have either or both of the following roles:
 - Database system security officer (DBSSO), who specifies and maintains the audit masks
 - Audit analysis officer (AAO), who turns auditing on and off, sets up and maintains the audit configuration, and reads and analyzes audit-trail data

Although role separation provides more secure auditing, these roles are optional. Before the database server software is installed, the OSA, or whoever installs the database server, decides whether to have separate or combined DBSSO and AAO roles for audit administration and who should perform each role.

For detailed information about roles and role separation, see “Using Role Separation” on page 8-3. For information about setting up role separation and creating a user group for each role, see your *IBM Informix Installation Guide*.

Audit Masks and Audit Instructions

As described in “Audit Masks” on page 7-1, an *audit mask* specifies a set of events to be audited when a user performs them. Audit events are derived from a combination of user and global masks. Chapter 11, “Audit Event Mnemonics and Fields,” on page 11-1 lists the set of auditable events. The set of events is fixed, but you use masks to specify only the ones that you need to audit.

The following table lists four types of audit masks.

Mask Type	Mask Name
-----------	-----------

Individual user masks

username

Default mask

_default

Global masks

_require and **_exclude**

Template masks

_maskname

The following section describes the first three kinds of masks. For a description of template masks, see page “Template Masks” on page 7-6.

User Masks

The global masks are always applied to user actions that are performed during a session in which auditing is turned on. Audit masks are applied in the following order:

1. An individual user mask or if none, the **_default** mask
2. The **_require** mask
3. The **_exclude** mask

When a user initiates access to a database, the database server checks whether an individual user mask exists with the same *username* as the account that the user uses. If an individual user mask exists, the database server reads the audit instructions in it first and ignores the **_default** mask. If no individual user mask exists, the database server reads and applies the audit instructions in the **_default** mask to that user.

In addition to default and individual masks, the database server reads and applies the audit instructions in the **_require** and **_exclude** masks. These masks are *global* because they apply to all users. Audit events in the **_require** mask are audited, even if they are not found in the **_default** or individual user masks. Audit events in the **_exclude** mask are not audited, even if the previously read masks specifically require them.

Important: If the audit instructions of these masks conflict, the instructions in the last mask to be read are used. Masks are read in the following order: *username*, **_default**, **_require**, and **_exclude**.

Users cannot tell if individual user masks exist for their accounts. Also, users do not need to do anything to enable auditing of their actions. Once an audit administrator turns on auditing, it operates automatically and users cannot disable it.

When the database server is installed, no audit masks exist. An audit administrator must specify all masks, including the default mask and the global masks.

Important: Actions that the *DBSA*, an audit administrator, or user **informix** generally performs are potentially dangerous to the security of the database server. To reduce the risk of an unscrupulous user abusing the **informix** account, it is recommended that the actions of **informix** always be audited. This procedure is intended to prevent an unscrupulous user from using **informix** to tamper with auditing or from granting discretionary access to another unscrupulous user.

Template Masks

As you become accustomed to the types of auditing that seem useful at your site, you might notice that certain auditing practices occur repeatedly. You can create *template* audit masks to help set up auditing for situations that recur or for various types of users.

For example, you might define a template mask called `_guest` and copy it to individual user masks for people who use your database server for a short time. You can copy a template mask to a user mask and modify it at the same time, perhaps turning off events that were audited in the template mask.

Important: All template mask names must be unique, contain fewer than eight characters, and begin with an underscore (`_`). These naming rules distinguish template masks from individual user masks.

You cannot create template masks with the following names because the database server already uses them:

- `_default`
- `_require`
- `_exclude`

When the database server is installed, no template masks exist. The number of template masks you can create is unlimited.

Audit Instructions

An audit administrator sets the audit instructions that the database server performs. The administrator must set an amount of auditing that is comprehensive enough to prove useful but not so exhaustive that it adversely affects system resources. When role separation exists, the DBSSO creates audit masks and the AAO configures mandatory auditing for the DBSA and the DBSSO. You can find advice on how to set the audit instructions in *A Guide to Understanding Audit in Trusted Systems* (published by the National Computer Security Center, NCSC-TG-001, June 1988).

This section suggests how to choose events to audit, how to set the audit instructions, and how the choices affect performance. For details of how to create and modify audit masks, see Chapter 8, “Audit Administration,” on page 8-1.

All the audit masks that the database server uses are stored in the system-monitoring interface (SMI) **sysaudit** table in the **sysmaster** database. The masks are updated automatically when the database server is upgraded to a newer version. Although information stored in the **sysmaster** database is available through SQL, you should use the **onaudit** utility for all audit-mask creation and maintenance. (See Chapter 10, “Utility Syntax,” on page 10-1.) Also, see the description of the **sysmaster** database in the *IBM Informix Administrator's Reference*.

Resource and Performance Implications

The amount of database server auditing enabled at any given time has a direct effect on operating-system resources and database server performance. Audit records that the database server generates are stored on disk. The greater the number of audit records generated, the more disk space required (for storage), and the greater the amount of CPU time required to process audit records (for storage, viewing, deletion, archiving, and restoration).

How system resources and performance are affected depends on these factors:

- Number of users/events audited
- Processor configuration
- System and user load
- Disk space
- Workload

For example, a system with parallel-processing capabilities, several terabytes of available disk space, 64 users, and full auditing might experience little degradation in performance and a relatively small disk-space ratio for audit data. However, a single-processor configuration with low disk space, multiple users, and full auditing might experience significant system-resource degradation and relatively rapid disk-space consumption by the audit trail.

From a system performance standpoint, the greatest overhead is incurred when you audit all database server security-related events that all users perform. Full auditing could severely degrade system performance and response time as well as require a significant amount of disk space for audit-record storage (depending on the amount of database server user activity). However, full auditing provides the most audit information and thus reduces the security risk.

You can turn off auditing to eliminate the effect on system performance, but then auditing cannot contribute to system security. At a minimum, you should audit the initiation of new user sessions.

The database server event that, if audited, has the most significant effect on system performance and disk space is Read Row (RDRW). In an established database that is primarily accessed by users who search for information, every row presented to every user generates an audit record. On a high-volume system, this quickly produces large numbers of audit records.

Special Auditing Considerations

Certain certification and accreditation organizations require that the installation process itself be audited. After configuring the operating system to accept audit data, the OSA should make sure that the AAO audits the actions taken during installation.

Level of Auditing Granularity

The Dynamic Server secure-auditing facility can audit the following events at the fragment level of granularity and shows additional information for fragmented objects:

- **Alter Table (ALTB).** The partition list that follows the alter-table operation is in the event record.
- **Create Index (CRIX).** The index can be fragmented; the event record includes fragmentation information.
- **Create Table (CRTB).** The table can be fragmented; the event record includes fragmentation information.
- **Delete Row (DLRW).** The partition and the record ID within the partition appear in the event record.
- **Insert Row (INRW).** The partition and the record ID within the partition appear in the event record.
- **Read Row (RDRW).** The partition and the record ID within the partition appear in the event record.

- **Update Row (UPRW).** The partition and the record ID within the partition appear in the event record.

Attention: Use row-level auditing only when absolutely necessary. Row-level auditing slows the database server dramatically and fills audit directories quickly.

For more information on the fields in an audit-event record, see Chapter 11, “Audit Event Mnemonics and Fields,” on page 11-1.

In addition, the database server audits the following events to the RESTRICT/CASCADE level:

- Drop Table (DRTB)
- Drop View (DRVW)
- Revoke Table Access (RVTB)

For more information on the corresponding SQL statements, see the *IBM Informix Guide to SQL: Syntax*.

Use of Various Masks

The **_require** mask can be a valuable tool because it audits every database server user for the events that are specified in this mask. You can use this mask to perform the bulk of the auditing. The **_require** mask enables you to make rapid changes to the auditing configurations for all users by adding or removing items from this one mask.

The **_exclude** mask is also useful. It is read last, so its contents take precedence over the instructions in the other masks. As the name implies, the audit events that you specify in the **_exclude** mask are excluded from auditing. This exclusion is true of every event, including those specified in the **_require** mask. The Read Row audit event, for example, is a good candidate for the **_exclude** mask. Read Row is a common event that can generate huge amounts of potentially useless data in the audit trail.

How you use the **_default** and individual user masks depends on the number of users and their activities. For example, if you have only a few users, you might want to give each one an individual mask. You might then use the **_default** mask to audit events that are initiated by users who do not normally use your database, and configure the **_default** mask with a high level of security. To offset any detrimental effects on system performance, set up less-comprehensive individual user masks for frequent users. Or, if you have many users and do not want to create many individual user masks, leave the **_default** mask empty and rely on the **_require** mask for most of your auditing.

Audit Configuration

The AAO can monitor the audit configuration, as Chapter 8, “Audit Administration,” on page 8-1 describes. Setting the audit configuration consists of performing the following tasks:

- Turning auditing on or off
- Specifying audit modes
- Using the **ADTCFG** file
- On UNIX, determining properties of the audit files

Sections that follow describe these topics.

Auditing On or Off

An audit administrator determines whether auditing is on or off. Auditing is turned off by default when the database server is installed. As Chapter 8, “Audit Administration,” on page 8-1, describes, the AAO can turn auditing on and off at any time, by using the **onaudit** utility, which Chapter 10, “Utility Syntax,” on page 10-1, describes. The database server can be in either online or quiescent mode for the changes to take effect.

When the AAO turns on auditing, all sessions, new and current, start auditing auditable events. Both existing sessions and new sessions produce records. All user sessions that are started thereafter also produce audit records.

Similarly, when the AAO turns off auditing, auditing stops for all existing sessions, and new sessions are not audited. If the AAO turns off auditing and then turns it on again while the database server is in online mode, existing sessions resume producing audit records.

Types of Auditing

When the AAO turns on auditing, the AAO can set the ADTMODE parameter in the ADTCFG file to specify the type and level of auditing.

For details, see “Changing the Audit Configuration” on page 10-7 and see Chapter 12, “The ADTCFG File,” on page 12-1. For more information on auditing administration, see “Administrative Roles and Role Separation” on page 8-1.

Auditing Modes

When you turn on auditing, you can set the ADTMODE parameter to 0, 1, 3, 5, or 7 in the ADTCFG file to specify the type and level of auditing.

For example, if you set the ADTMODE configuration parameter to 1 in your ADTCFG file, auditing is turned on automatically during database server initialization. After you turn on auditing, the database server records only the audit events defined in the audit masks.

The AAO configures auditing and specifies an error mode, in case an error occurs when an audit record is stored.

Properties of Audit Files on UNIX

As “Audit Process” on page 7-3 describes, with database server-managed auditing on UNIX, the database server writes audit records to audit files in an audit trail. This section describes the audit files in more detail.

Location of Audit Files

The audit files are located in a directory that you specify with the **onaudit** utility or the ADTPATH configuration parameter in the \$INFORMIXDIR/aaodir/adtcfg UNIX file.

If you change the audit path, the change takes effect immediately for all existing sessions. This feature enables you to change the directory when the database server is in online mode, which is useful if the file system that contains the existing audit files becomes full.

Keep the file system that holds the audit trail cleaned out so that ample storage space is always available.

New Audit Files

When the database server writes an audit record, the database server appends the record to the current audit file. If you bring the database server out of online mode and then put it back, the database server continues to use the same audit file. The database server starts a new audit file only under the following conditions:

- When the file reaches a specified size
- When you manually direct the database server to start a new audit file, as Chapter 8, “Audit Administration,” on page 8-1 describes
- When you start database server-managed auditing

The database server starts a new audit file at the default size of 10,240 bytes, which is the minimum size for audit files. (The **adtcfg.std** file might list a value of 50,000 bytes as a guideline.) You can change this file size at any time when auditing is on, even when the database server writes to an audit file, as Chapter 10, “Utility Syntax,” on page 10-1 describes.

The optimal size for audit files depends on your configuration. Larger files contain more data, which results in fewer files to review. However, the trade-off is that large files are more difficult to manipulate.

Audit File Names

No matter how you start a new audit file, it follows the same naming convention.

The naming convention is *dbservername.integer*, where *dbservername* is the database server name as defined in the **ONCONFIG** file, and *integer* is the next integer. The series starts with 0.

For example, if a new audit file is started for a database server **maple**, and the last audit file was saved in the file **maple.123**, then the next audit file is called **maple.124**. If **maple.124** already exists, the next available number is used. The names are unique to a specific audit directory, so you can have **auditdir1/maple.123** and **auditdir2/maple.123**, and so on.

Windows Application Event Log

Windows systems provide an event-logging facility as a common repository for logging events and other useful information. The event-logging facility also provides a user interface to filter, view, and back up the information that is stored there.

In versions of Windows earlier than Windows XP, applications with appropriate permissions could write to the security log and to the system log. In Windows XP and later versions, however, applications cannot write to the Windows Security Event log. Auditing messages from the database server are now sent to a log file, whose directory path can be specified using the **onaudit** utility. The default pathname is **%INFORMIXDIR%\aaodir**.

Any messages that the database server writes to its log file are also written to the Windows Application Event log.

Windows Message Server

Dynamic Server for Windows runs as a service under the **informix** user account.

The Dynamic Server **Message Server** service communicates with the database server through the named pipes interprocess communications mechanism to receive information and to write it to the Windows Application Event log, as well as to the log file `%INFORMIXDIR%\%INFORMIXSERVER%.log`.

The database server starts **Message Server** when an instance of the database server first needs to write a message to the event log. **Message Server** does not terminate automatically when an instance of the database server terminates.

Error Modes for Writing to an Audit File

If the database server encounters an error when it writes to the audit file, it can behave in various ways called *error modes*. You can change the error mode, as “Setting the Error Mode” on page 8-6 describes, at any time during database server operation, even after an error occurs. See also the discussion of **onaudit** error modes in Chapter 10, “Utility Syntax,” on page 10-1.

Halt Error Modes

When the database server is in a *halt* error mode (1 or 3), it does not allow the session that received the error to continue processing after it writes to the audit trail. The database server might even terminate the session or shut down, depending on the error mode. Descriptions of halt error modes follow:

- Mode 1: A thread is suspended but the session continues when the audit record is successfully written.
- Mode 3: The database server shuts down and the user session cannot continue.

Processing for the session does not continue until the error condition is resolved.

Continue Error Mode

When the database server is in *continue* error mode (0), it allows the session that received the error to continue processing after it writes to the audit trail. However, the audit record that was being written when the error occurred will be lost. The database server writes an error to the message log stating that an error made while writing an audit record has occurred.

If the error continues to occur, all subsequent attempts to write to the audit trail also generate messages in the message log, which can quickly grow very large.

Audit Configuration and the ADTCFG File

Configuration parameters in the **ADTCFG** file specify the properties of the audit configuration. These configuration parameters are **ADTERR**, **ADTMODE**, **ADTPATH**, and **ADTSIZE**.

The pathname for the default **ADTCFG** file follows.

Environment

ADTCFG Pathname

UNIX `$INFORMIXDIR/aaodir/adtcfg`

Windows

%INFORMIXDIR%\aaodir\adtcfg

If you edit the **ADTCFG** file to change the audit parameters, the audit configuration is not changed until you reinitialize shared memory. If you use the **onaudit** utility to change the audit configuration, the changes occur immediately.

Changes made with **onaudit** are written to an **adtcfg.servernum** companion file. (SERVERNUM is a parameter in the **ONCONFIG** file, which the *IBM Informix Administrator's Reference* describes.) The configuration changes take effect in the server immediately. The current and subsequent server instance refers to the **adtcfg.servernum** file for the audit configuration parameters instead of the file **adtcfg**.

For detailed information about the **ADTCFG** configuration parameters, see Chapter 12, "The ADTCFG File," on page 12-1.

Access to the Audit Trail

Standard users should not be able to view or alter audit files. The audit trail (that is, the audit files) should be accessed only with the **onshowaudit** utility, which has its own protection, as follows:

- With role separation on, only an AAO can run **onshowaudit**.
- With role separation off on UNIX, only user **informix**, a member of the **informix** group, or user **root** can run **onshowaudit**.
- With role separation off on Windows, only user **informix** can run **onshowaudit**.

Access to Audit Files on UNIX

The following characteristics control access to audit files in a UNIX environment and protect them from being accidentally read or destroyed:

Ownership:

informix

Group ID:

same as \$INFORMIXDIR/aaodir

Permissions:

660

Important: The AAO should be careful when selecting the directory in which the audit files are stored (ADTPATH). The directories in the path must have adequate ownership and access permissions for the level of risk that the AAO allows. The default directory (**/tmp**) does not have adequate protection.

The following examples show the security configuration for UNIX audit files with no role separation:

aaodir

Ownership:

informix

Group ID:

informix

Permissions:

775

aaodir/adtcfg.std

Ownership:
informix

Group ID:
informix

Permissions:
644

The following examples show the UNIX security configuration with role separation:

aaodir

Ownership:
informix

Group ID:
<aao_group>

Permissions:
775

aaodir/adtcfg.std

Ownership:
informix

Group ID:
<aao_group>

Permissions:
644

Important: Because any account with the group ID of **informix** or superuser (**root**) ownership, or both, can access the audit trail, you must exercise care to protect these accounts and their passwords.

Access to Audit Records on Windows

The following characteristics control access to the Windows audit file and protect it from accidental viewing or deletion:

Ownership:
informix

Group ID:
same as %INFORMIXDIR%\aaodir

The following examples show how to control access to the Windows audit file:

aaodir

Ownership:
informix

Group ID:
Administrator

aaodir\adtcfg.std

Ownership:
database server administrator

Group ID:
Administrator

Audit Analysis Overview

The AAO performs audit analysis. This section explains the importance of audit analysis, how to prepare for it, some strategies for audit analysis, and how to react to a perceived security problem.

Importance of Audit Analysis

The database server audit mechanism is designed to both deter and reveal attempted, as well as successful, security violations. However, the audit data it generates is only as useful as the analysis and reviews performed on it. Never reviewing or analyzing the audit data is equivalent to disabling auditing altogether (and is, in fact, worse because auditing might reduce database server performance).

If, on the other hand, you routinely analyze and review the audit data, you might discover suspicious activity before a successful violation occurs. The first step to terminate any security violation is to detect the problem. If a database server violation should occur, the audit trail permits you to reconstruct the events that lead up to and include this violation.

Tip: To play the greatest role in the security of your database server, watch the database server activity regularly.

Become accustomed to the types of activity that occur at various times of day at your site. You become the expert on types of user activity when you perform the following actions:

- Review the database server security audit trail on a daily basis, or more frequently, if necessary.
- Note the types of activity that each user performs.

Periodically check the types of events that are audited versus the data that actually appears in the security audit trail to ensure that the audit facility is operating properly.

Your continual observance of the audit trail might be the only way to determine if some users browse through the database server. You might catch a user performing an unusual amount of activity at 2 A.M., a time of day when that user is not even at work. Once you identify a potential security anomaly, you can then investigate further to determine if anyone on the database server attempts to obtain unauthorized information, if a user misuses the database server, or if a user becomes lenient in self-regulated security enforcement.

Preparation for Audit Analysis

This section describes two methods to analyze database server audit records:

- The first method is simply to display audit data as it appears in the audit trail, which you can subject to your own audit-analysis tools. This method guarantees accuracy because no processing is done on the raw audit records.
- The second method converts the audit records into a form that can be uploaded into a table that the database server manages. You can then use SQL to generate

reports based on this data. With the SQL-based method, you can create and use customized forms and reports to manipulate and selectively view audit data, which provides a flexible and powerful audit-analysis procedure. Be sure, however, that records are not deleted or modified from either the intermediate file or from the database prior to analysis.

Important: The SQL-based procedure is more convenient but remains untrusted because users can use SQL data-manipulation statements to tamper with the records that are copied into a table.

Both methods rely on a utility called **onshowaudit**, which Chapter 9, “Audit Analysis,” on page 9-1 and Chapter 10, “Utility Syntax,” on page 10-1 describe. For either method, you can extract audit events for specific users, database servers, or both.

To perform audit analysis, first have audit records in your database server. The **onshowaudit** utility does not remove data from the audit trail. It only reads records from the audit trail and allows them to be viewed or manipulated with standard SQL utilities.

To clear or remove audit logs, delete the files that contain the audit trail.

Strategies for Audit Analysis

The primary threat to database server security is unauthorized disclosure or modification of sensitive information. This section discusses those and other threats that might be discovered through audit analysis.

Event Failure

The audit records that indicate that an attempted database server operation failed are particularly important in audit analysis. The audit record could indicate, for example, that a user is attempting to give sensitive data to another user who does not have the correct UNIX permissions or Windows access privileges to access the data.

Event Success

Failed operations are the most common indicators of a security problem in the audit trail. Somewhat harder to find, but of equal security importance, is any successful but unusual activity for a particular user.

For example, a user who repeatedly creates and drops databases might be attempting to discover and exploit a covert channel to relay sensitive information to an unauthorized process or individual. Watch for a marked increase in the occurrence of database server events that would typically occur infrequently during normal database server use.

Perhaps a particular user who has never granted privileges suddenly shows a great deal of activity in this area, or perhaps a user who has never written large amounts of data into a database begins to generate hundreds of new records. You must determine the extent of the abnormalities (for example, the number of objects that this user accessed) and the possible severity of the compromise (for example, the importance of the accessed objects).

Insider Attack

An *insider attack* occurs when an authorized user with malicious intent obtains sensitive information and discloses it to unauthorized users. An unscrupulous user of this sort might not exhibit immediately recognizable signs of system misuse. Auditing is a countermeasure for this threat. Careful auditing might point out an attack in progress or provide evidence that a specific individual accessed the disclosed information.

Browsing

Users who search through stored data to locate or acquire information without a legitimate need are *browsing*. Browsers do not necessarily know of the existence or format of the information for which they are looking. Browsers usually execute a large number of similar queries, many of which might fail because of insufficient privileges. Auditing is a countermeasure for this threat. The behavior pattern makes browsers relatively easy to identify in the audit trail.

Aggregation

An *aggregate* is an accumulation of information that results from a collection of queries. An aggregate becomes a security threat when it comprises queries to objects that have little significance themselves but as a whole provide information that is considered more important than any component piece. The higher sensitivity of the aggregate results from the sensitivity of the associations among the individual pieces. Auditing is a countermeasure for this threat. As with browsing, careful auditing might point out an attack in progress or provide evidence that a specific individual accumulated the disclosed information.

Responses to Identified Security Problems

After you identify the user or users who are responsible for irregularities in the security audit trail, refer to your site security procedures. If your site has no security procedures regarding potential security breaches, you might consider the following actions:

- Enable additional auditing to further identify the problem.
- Shut down the database server to halt any unauthorized information flow.
- Develop a plan with the supervisor of the user to address the problem.
- Confront the specific individual.

In some cases, you might find that an otherwise authorized user is browsing a bit too widely on the database server. After some observation, you might want to talk with the supervisor of the user. It might not be wise to talk directly with an individual whose actions are being monitored.

You must ascertain whether a particular problem that is identified through the audit trail is actually someone attempting to breach security or just, for example, a programming error in a newly installed application.

The exact type of security irregularity that might occur and the specific action to take in response to it are not within the scope of this manual.

DBMS Security Threats

This section discusses responses to various kinds of security threats to the DBMS. For more information on various roles, see “Administrative Roles and Role Separation” on page 8-1.

Primary Threats

Primary threats to the security of a database server involve unauthorized disclosure or modification of sensitive information. To counter these measures, the DBSSO, DBSA, and OSA must ensure that all users of the DBMS are identified and authenticated before they are able to use or access the software or data.

Users must belong to the correct group to access the database server. They must also have a valid login ID in the operating-system password file.

In addition, all users who attempt to access data must satisfy Discretionary Access Control (DAC) restrictions before access is granted. DAC uses SQL statements to specify which users can and cannot access data in the database. Access can be allowed or revoked at the following levels:

- Database level
- Table level
- SPL routine level
- Column level
- Role level
- Fragmentation level

These countermeasures are adequate for legitimate use of the product when users attempt to access the data directly. They cannot, however, counter threats of confidentiality or modification to the data posed by illegitimate use of the product, such as if a privileged user abuses his or her permissions or access privileges.

Privileged Activity Threats

Improper or unchecked activity by users with privileged roles (DBSSO, AAO, DBSA, or OSA) can introduce security vulnerabilities and possible threats to the database server. Dynamic Server is carefully designed to give the DBSSO, AAO, and DBSA only the abilities needed to do their jobs. Nevertheless, these roles, as well as those of operating-system administrators, impart sufficient power that careless use of such power could result in breaches of security.

Database Server Administrator

The DBSA controls and monitors the database server and can configure role separation during database server installation. The countermeasure to a threat from the DBSA is independent scrutiny of the DBMS audit trail. The DBSSO can enable auditing of all DBSA actions, and the AAO can review DBSA actions in the audit trail.

Database System Security Officer

The DBSSO sets up DBMS audit masks for individual users. The countermeasure to a threat from the DBSSO is independent scrutiny of the DBMS audit trail because auditing DBSSO actions are enabled by the AAO.

Operating-System Administrator

A malicious OSA also poses a serious security threat because the OSA can violate the assumptions about the product environment and the methods that underpin its security functions. As with a DBSSO, the countermeasure to an OSA threat is independent scrutiny of the activities of the OSA, as recorded in the audit trail.

Audit Analysis Officer

The AAO reviews the DBMS audit trail. The countermeasure to this threat is to ensure that an AAO is authorized to view information that might be yielded when the database audit trail is reviewed. It is also important that the output of the **onshowaudit** utility be accessible only to an AAO and that manipulation of this output also be audited in the audit trail.

Shared-Memory Connection Threats on UNIX

A shared-memory connection provides fast access to a database server if the client and the server are on the same computer, but it poses some security risks. False or nontrusted applications could destroy or view message buffers of their own or of other local users. Shared-memory communication is also vulnerable to programming errors if the client application explicitly addresses memory or over-indexes data arrays.

The OSA ensures that the shared-memory connection method is not specified in the configuration file for client/server connections. If the client and the server are on the same computer, a client can connect to a server with a stream-pipe connection or a network-loopback connection.

The default pathname for the UNIX configuration file is **\$INFORMIXDIR/etc/sqlhosts**.

For more information on shared-memory connections, see the *IBM Informix Administrator's Guide*.

Threats from Malicious Software

Database users can easily and unknowingly download malicious or unauthorized software. This is a security threat that can come from not only server machines that host the databases, but also computers used to access the databases.

To protect the database server from malicious software:

- Keep the database server on a different computer from the clients that need to connect to it
- Restrict access to the computer hosting the database server
- Monitor the software installed on the database server computers (for example, by running a checksum process periodically)
- Keep a record of all the files and permissions on the database server computer
- Institute a strict security policy
- Make all users aware of the dangers of starting software of unknown or untrusted origin

Malicious software can defeat security controls in many ways. For example, such software can copy data for subsequent access by an unauthorized user or grant database access privileges to an unauthorized user.

Remote-Access Threats

When a user is granted database access privileges, the host computer of the user is not specified. Therefore, the user can gain access to the privileged data from any computer that is configured to connect to the host computer. As a result, a user might not be aware of having remote access to privileged data when the user grants another user direct access to that data. This situation could lead to data that is inappropriately accessed remotely.

Make sure that all users are aware that access privileges are granted to user names, with no dependencies on the origin of the remote connection.

Obsolete-User Threats

A user is identified by an operating-system user name or user ID or both. The data access privileges and individual user audit masks of the software are based on the user name. At the operating-system level, a user account might be removed and this user name might become unassigned.

If any of the access privileges of the software or the individual user audit mask associated with that user name are not removed before the same user name is allocated to a new user, the new user inadvertently inherits the privileges and audit mask of the previous user.

To avoid this problem, have the OSA notify the DBSA when a user account is removed from the operating system. The DBSA can then perform the actions necessary to eliminate references to this name in the DBMS. These actions might involve revoking access privileges and removing an individual audit mask.

Untrusted Software Used in a Privileged Environment

Problems might occur if DBSAs or OSAs run untrusted software. Untrusted software can use the privileges of the DBSA or the OSA to perform actions that bypass or disable the security features of the product or that grant inappropriate access privileges.

The primary countermeasure to this vulnerability is to make sure that DBSAs and OSAs do not run software of unknown or untrusted origin. We further recommend that the operating-system access controls protect all software that DBSAs and OSAs run against unauthorized modification.

Distributed Database Configuration Threats

When you set up a distributed database, you configure two or more software installations. The configurations of these software installations could be incompatible.

A distributed database user might be able to gain access to data on a remote system with an incompatible configuration when that data would not be accessible to the same user directly on the remote system. In the worst case, the software could connect two systems that have an account with the same user name but are owned by a different user. Each user is granted the privileges of the other user at access of the database that resides on the host computer of the other user.

When two UNIX workstations are connected, the OSA must ensure that accounts with user names in common are owned by the same user.

Chapter 8. Audit Administration

This chapter explains how to set up and administer auditing on your database server after the database server is installed and functioning properly.

Administrative Roles and Role Separation

This section describes the main administrative roles involved in secure auditing:

- The database server administrator (DBSA)
- Audit administrator roles:
 - The database system security officer (DBSSO)
 - The audit analysis officer (AAO)

This section also touches on the roles and responsibilities of database administrators (DBAs), operating-system administrators (OSAs), system users, and privileged users. It tells how to set up role separation and provides guidelines on how to assign roles.

Database Server Administrator

The DBSA configures, maintains, and tunes the database server. The DBSA becomes involved with the security of a database server during installation. Your *IBM Informix Administrator's Guide* defines the overall role of the DBSA.

Someone who has the appropriate UNIX permissions or Windows access privileges to view all the data on a database server should perform this role. It is supported by a designated account and software designed to support DBSA tasks.

To use the administrative software designed for this role, the person who performs the role of the DBSA must log in to one or more designated accounts and meet access-control requirements.

If the DBSA group is not group **informix**, the permissions on **oninit** must be modified to 6755 (granting others execute permission) so that members of the new DBSA group can start the database server

The DBSA is responsible for granting or revoking the EXTEND role to restrict users who can register DataBlade modules or external user-defined routines (UDRs).

Database System Security Officer

The DBSSO is a system administrator who performs all the routine tasks related to maintaining the security of a database server. These tasks include the following actions:

- Maintaining the audit masks
- Responding to security problems
- Educating users

The DBSSO performs these tasks with the **onaudit** utility. For information, see "The onaudit Utility" on page 10-1

The DBSSO role is supported by a designated account and software. To use the audit tools, the users who fill the DBSSO role must log into the designated account and meet access-control requirements. After the DBSSO users meet the access-control requirements and use the administrative software, their actions can be audited.

Tip: A DBSSO on UNIX is any user who belongs to the group that owns **\$INFORMIXDIR/dbssodir**. On Windows, the administrator uses registry settings, through the **Role Separation** dialog box that appears during installation, to specify DBSSO users.

Important: The **onaudit** utility can create a potential threat to the security of the database server. An unscrupulous user can abuse a DBSSO account, for example, by turning off auditing for a specific user. To reduce this risk, all actions taken through **onaudit** should be audited.

Audit Analysis Officer

The AAO configures auditing and reads and analyzes the audit trail. The AAO can specify whether and how auditing is enabled, how the system responds to error conditions, and who is responsible for managing the audit trail.

For database server-managed auditing on UNIX, the AAO also determines the directory for the audit trail and the maximum size of each audit file.

The AAO can load the audit-trail data into a database server and use SQL to analyze it, either through a utility such as DB–Access or a customized application developed with an IBM Informix SQL API or application development tool.

The AAO performs these tasks with the **onaudit** and **onshowaudit** utilities, which Chapter 10, “Utility Syntax,” on page 10-1 describes. If the AAO uses **onaudit** to change the audit configuration parameters during a database server session, the new values are written to the **adtcfg.servnum** file for that instance of the database server.

The installation script for the database server creates a **\$INFORMIXDIR/aaodir** UNIX directory or a **%INFORMIXDIR%\aaodir** Windows directory, which contains files that the AAO uses. These files include the **adtcfg** audit configuration file as well as the **adtcfg.std** file, both of which contain examples of valid definitions for audit configuration parameters.

The AAO needs appropriate UNIX permissions or Windows access privileges to view all the data in the database server to analyze events that might involve sensitive information. The AAO decides whether to audit all actions of the DBSSO and the DBSA.

Tip: On UNIX, an AAO is any user who belongs to the group that owns **\$INFORMIXDIR/aaodir**. On Windows, the administrator uses registry settings, through the **Role Separation** dialog box that appears during installation, to specify AAO users.

Other Administrative Roles and Users

A number of other, more minor, roles might be involved in database server secure auditing.

Database Administrator

A DBA manages access control for a specific database. A DBA cannot change database system modes, add or delete space, or maintain or tune the system. For information on the role and responsibilities of a DBA, see the *IBM Informix Guide to SQL: Tutorial*. For information on this and other database server roles and users, see your *IBM Informix Administrator's Guide*.

Operating-System Administrator

The OSA carries out responsibilities and tasks that the database server requires from the operating system. The OSA enables role separation, grants and revokes access to and from the database server if role separation is enforced, and adds new AAO, DBSSO, and DBSA accounts as necessary. In addition, the OSA coordinates with the DBSSO and AAO to perform various security-related functions of the database server, such as periodic reviews of the operating-system audit trail.

No special account exists for the operating-system needs of the database server, and no special database server protection mechanisms are associated with OSA tasks. For more information, refer to your operating-system documentation.

System Users

All operating-system accounts, including those for the DBSA, DBSSO, AAO, and the account called **informix**, potentially can use the database server. All users with accounts who want to use the database server must explicitly be granted access to the database server if role separation is configured to enforce access control on database server users. The DBSA can revoke that access at any time, whether or not role separation is enabled. For more information on granting or revoking access, see “Configuring and Enforcing Role Separation” on page 8-4.

Privileged Users

Privileged users are those users whom the database server recognizes as having additional privileges and responsibilities. These privileged users include the DBSA, DBSSO, AAO, and DBA. In addition, the users **informix** and **root** can also operate as any privileged user on database servers configured without role separation. Even with role separation, **root** can be a privileged user.

Using Role Separation

Role separation is a database server option that allows users to perform different administrative tasks. Role separation is based on the principle of separation of duties, which reduces security risks with a checks-and-balances mechanism in the system. For example, the person who determines what to audit (DBSSO) should be different from the person who monitors the audit trail (AAO), and both should be different from the person who is responsible for the operations of the database server (the DBSA).

Assigning Roles

This section provides general guidelines on how to assign people to accounts and give them access to perform roles. These guidelines should be amended to fit the resources and security policies of your site.

- Have one account for each person who performs a role.

For example, if you have multiple users who perform the DBSA role, have each person work from a separate account. Establish a one-to-one mapping between accounts and users to make it easier to trace audit events to a single user.

- Have as few DBSA and DBSSO accounts as possible.

The DBSA and DBSSO accounts can compromise the security of the database server. Limit the number of accounts that can disrupt the database server to lower the chance that an unscrupulous user can abuse a privileged account.

- Keep the DBSA and DBSSO roles separate.

You might not have the resources or see the need to have different users perform the DBSA and DBSSO roles, nor does Dynamic Server strictly require this role separation. When you keep the DBSA and DBSSO roles separate, however, you constrain them to perform only those tasks that their duties specify and limit the risk of compromising security.

- Keep the AAO role separate from the DBSA and DBSSO roles.

The AAO determines whether to audit all DBSA or DBSSO actions in the system. It is essential that someone with a role different from that of the DBSA or DBSSO be in charge of auditing configuration, so that all users, including the DBSA and DBSSO, are held accountable for their actions in the system. This constrains users to perform only those tasks that their duties specify and limits the risk of compromising security.

- Limit access to the account **informix** because it can bypass role-separation enforcement and other database server access-control mechanisms.

Configuring and Enforcing Role Separation

The DBSA, or the person who installs the database server, enforces role separation and decides which users will be the DBSSO and AAO. To find the group for the DBSA, DBSSO, or AAO, look at the appropriate subdirectory of **\$INFORMIXDIR** on UNIX or **%INFORMIXDIR%** on Windows.

On Windows, role separation is configured only during installation. On UNIX, you normally configure role separation during installation, but you can also configure it after the installation is complete or after the database server is configured. The OSA who installs the software enforces role separation, and decides which users (Windows) or groups (UNIX) will be the DBSSO and AAO. On UNIX, the group that owns **\$INFORMIXDIR/aaodir** is the AAO group; the group that owns **\$INFORMIXDIR/dbssodir** is the DBSSO group. By default, group **informix** is the DBSSO, AAO, and DBSA group.

On UNIX, if you use the InstallShield MultiPlatform (ISMP) installer in GUI or terminal mode to install the database software, you will be asked if you want to configure role separation. If instead you use the scripted bundle installer, then the environment variable **INF_ROLE_SEP** controls whether you will be asked to set up separate roles. If the **INF_ROLE_SEP** environment variable exists (with or without a value) role separation is enabled and you will be asked to specify the DBSSO and AAO groups. (You will not be asked about the DBSA group.) If the **INF_ROLE_SEP** environment variable is not set, then the default group **informix** is used for all these roles.

You do not need to set **INF_ROLE_SEP** to a value to enable role separation. For example, in a C shell, issuing **setenv INF_ROLE_SEP** is sufficient.

After the installation is complete, **INF_ROLE_SEP** has no effect. You can establish role separation manually by changing the group that owns the **aaodir**, **dbssodir**, or **etc** directories. You can disable role separation by resetting the group that owns

these directories to **informix**. You can have role separation enabled for the AAO without having role separation enabled for the DBSSO.

Role separation control is through the following group memberships:

- Users who can perform the DBSA role are group members of the group that owns the directory **\$INFORMIXDIR/etc**.
- Users who can perform the DBSSO role are group members of the group that owns the **\$INFORMIXDIR/dbssodir** directory.
- Users who can perform the AAO role are group members of the group that owns the **\$INFORMIXDIR/aaodir** directory.

Note: For each of the groups, the default group is the group **informix**.

The **ls -lg** UNIX command produces the output that Figure 8-1 shows.

```
total 14
drwxrwx--- 2 informix      ix_aao    512 Nov 21 09:56 aaodir/
drwxr-xr-x 2 informix      informix  1536 Nov 30 18:35 bin/
drwxrwx--- 2 informix      ix_dbssso 512 Nov 30 10:54 dbssodir/
drwxr-xr-x 10 informix     informix  512 Nov 21 09:55 demo/
drwxrwxr-x 2 informix     informix  1024 Nov 30 11:37 etc/
.
.
.
```

Figure 8-1. Example Output Showing Role Separation

In Figure 8-1, the AAO belongs to the group **ix_aao**, the DBSSO belongs to the group **ix_dbssso**, and the DBSA belongs to the group **informix**.

Users must belong to the correct group to access the database server. To find the group for database users, you must look at the contents of the **\$INFORMIXDIR/dbssodir/seccfg** file. For example, the contents of a typical **seccfg** file might be **IXUSERS=***. This group setting means that all users are allowed to connect to the database server. If the file contains a specific name such as **IXUSERS=engineer**, then only members of the group **engineer** can gain access to the database server.

For Windows, role separation control is through the **Role Separation** dialog box, which appears during installation, and through registry settings. If the **Enable Role Separation** check box is checked in the **Role Separation** dialog box, the DBSA can specify different roles.

For more information on environment variables, see the *IBM Informix Guide to SQL: Reference*. For more information on configuring role separation, see your *IBM Informix Administrator's Guide*.

Auditing Setup

Auditing does not start automatically when the database server is first installed. Before any user actions are audited, the DBSSO or AAO must perform the following tasks to configure the database server for auditing:

- Specify events to audit in the default, user, and global audit masks (DBSSO)
- Specify how the database server should behave if an auditing error occurs when an audit record is written (AAO)

- Determine the desired level of auditing (AAO)
- Turn on auditing (AAO)
- Specify the directory where audit files are located (AAO)

Setting Up the Default and Global Masks

Before setting up default and global masks, the DBSSO needs to understand how the various masks work and what the implications are for different auditing instructions. Also, the DBSSO must understand which auditing events to place in which masks. For details, see Chapter 7, “Overview of Auditing,” on page 7-1.

Use the **onaudit** utility to add audit events to audit masks. Chapter 11, “Audit Event Mnemonics and Fields,” on page 11-1 lists the audit events and their mnemonics. Chapter 10, “Utility Syntax,” on page 10-1 shows the complete syntax for **onaudit**.

The following command shows how the Update Audit Mask and Delete Audit Mask audit events are added to the **_default** mask by their four-letter event codes, or mnemonics:

```
onaudit -m -u _default -e +UPAM,DRAM
```

You can add audit events to the **_require** and **_exclude** masks in the same way. For specifics, see Chapter 10, “Utility Syntax,” on page 10-1.

All users who initiate a database session after this command is run (and auditing is turned on) are audited for the specified events.

Specifying a Directory for the Audit Trail (UNIX)

The database server stores audit files in a file-system directory. You can specify the directory with the **onaudit** utility. For example, the following command specifies **/work/audit** as the UNIX file system in which the database server is to store audit files:

```
onaudit -p /work/audit
```

Note: The **onaudit -p /work/audit** command works only if logging is enabled or if **-1 N** options are included in the command line.

You can change the audit directory at any time. You can also set up the type of auditing and specify the directory with the **ADTCFG** file, which is described in Chapter 12, “The ADTCFG File,” on page 12-1.

For more information about the **onaudit** utility, see Chapter 10, “Utility Syntax,” on page 10-1.

Setting the Error Mode

As Chapter 7, “Overview of Auditing,” on page 7-1 describes, the database server has three actions that it can perform if an error occurs when writing to the audit trail: a *continue* error mode, and two levels of severity of *halt* error mode. Be sure that you, as the AAO, understand the implications of each error mode before you select one.

Use the **onaudit** utility or the **ADTCFG** file to set the error mode. For the **onaudit** syntax, see Chapter 10, “Utility Syntax,” on page 10-1. For the **ADTERR** configuration parameter, see Chapter 12, “The ADTCFG File,” on page 12-1.

The following **onaudit** command sets the error mode to *continue*. The database server processes the thread and notes the error in the message log.

```
onaudit -e 0
```

The following command sets the error mode to the most severe level of *halt*, in which the database server shuts down:

```
onaudit -e 3
```

Setting the Audit Level

The AAO or DBSSO configures the level of auditing in the system. The AAO monitors the audit trail and handles all audit-record management.

The DBSSO has significant leeway regarding the auditing level of the database server. For example, a minimal audit configuration might involve auditing only DBSSO actions, database server utilities, and the start of each new database server user session. A maximal audit configuration involves auditing all security-relevant database server events for all users.

The AAO and DBSSO should coordinate efforts to determine the auditing level. For instance, to audit the DBSA actions, the DBSSO would use masks for the DBSA accounts, and the AAO would set the audit mode with the **onaudit** utility or the **ADTCFG** file.

To ensure that the appropriate database server activities are monitored, review the audit records that are stored in the operating-system audit trail, database server audit files, or Windows event log. You must configure the database server to monitor these events.

You can reconfigure auditing as usage changes and potential security threats are identified. For the **onaudit** syntax, see Chapter 10, “Utility Syntax,” on page 10-1. For information on the ADTMODE configuration parameter, see Chapter 12, “The ADTCFG File,” on page 12-1.

Important: Although database server audit-record generation might have a negative effect on database server performance and resources, you should perform more than the minimal database server audit. This additional audit improves the likelihood that you will detect security violations and any attempts to circumvent security mechanisms.

If you perform minimal or no auditing for database server users, it is virtually impossible to detect creative attempts to circumvent the database server security policy. If someone suspects a security violation or a particular user exhibits unusual behavior, you should enable full auditing of the suspect user to get a complete picture of the user’s activities.

Balance the security needs of your site and the performance and resource effect of different auditing levels. The auditing level at any given time has a direct effect on both the operating-system resources and the database server performance. The effect depends on the following factors:

- Number of users or events audited, or both
- Processor configuration
- System load (number of processes and users)
- Disk space
- Work load (types of processes performed)

Tip: To specify disk space, use the Windows Event Viewer administration tool.

For more information on database server performance considerations, refer to your *IBM Informix Performance Guide*.

Activating Auditing

Auditing is turned off by default when you install the database server. Use the **onaudit** utility to turn on auditing at runtime or set the ADTMODE configuration parameter in the **ADTCFG** file. If you use the **ADTCFG** file, the setting takes effect when the database server is initialized.

The following **onaudit** command turns on auditing:

```
onaudit -l 1
```

After you turn on auditing, auditing changes take effect immediately for all sessions.

The AAO can configure the database server to turn on auditing when the server starts when the ADTMODE configuration parameter is set to the numbers 1, 3, 5, or 7 in the **ADTCFG** file. For details on ADTMODE parameter values, see “Changing the Audit Configuration” on page 10-7 and Chapter 12, “The ADTCFG File,” on page 12-1.

When the database server is initialized with auditing turned on, all user sessions generate audit records according to the individual, default, or global (**_require**, **_exclude**) mask in effect for each user.

To turn off auditing after it starts, see “Turning Off Auditing” on page 8-14.

Audit Mask Maintenance

You might want to change the auditing instructions as your auditing needs change. This section explains the following procedures, which you use to change audit masks:

- Creating audit masks
- Displaying audit masks
- Modifying audit masks
- Deleting audit masks

These tasks, which the DBSSO performs, apply whether the database server or your operating system administers the audit records.

Creating Audit Masks

You can create masks that more closely match the types of activities that individual users perform than do default and global masks.

- To create individual user masks, specify user IDs as mask names.
- To create template masks, preface the name of a mask with an underscore (_). Chapter 7, “Overview of Auditing,” on page 7-1 describes template masks and user masks.

You specify events in the mask when you create it, using the audit events from the alphabetical listing in the table “Audit-Event Mnemonics” on page 11-1. You

specify events for customized (template and user) audit masks the same way that you do for the **_default**, **_require**, and **_exclude** audit masks.

For example, you might want to create three template masks with different levels of security: **_low**, **_medium**, and **_high**. Alternatively, you might need just two templates for familiar and unfamiliar users that you copy to individual user masks: **_guest** and **_trusted**.

Creating a Template Mask

To create a template audit mask:

Use the **onaudit** utility. The Chapter 10, “Utility Syntax,” on page 10-1 shows the syntax. The following example shows how to create a template mask called **_guest** with the audit events Create Database, Grant Database Access, and Grant Table Access:

```
onaudit -a -u _guest -e +CRDB,GRDB,GRTB
```

Creating a User Mask from a Template Mask

A mask that is used as the foundation for one or more other masks is referred to as a *base mask*.

To create a user mask from a template mask:

Create the template mask. After you create a template mask for a given user category, you can use it as the basis of masks for individual users, adding or removing only the audit events that differ for each user.

The following example creates a user mask for the user **terry**, based on the **_guest** template mask:

```
onaudit -a -u terry -r _guest -e -CRDB
```

The **terry** mask has the same audit events as the **_guest** mask, except for the CRDB (Create Database) audit event, which was removed.

Instead of template masks, you can also use existing user **_default**, **_require**, and **_exclude** masks as base masks.

Tip: If you use a template or user mask as a base mask for another mask, the new mask inherits the events in the base mask. The new mask does not refer to the base mask dynamically. Future changes to the base mask are not reflected in other masks that might have been created or modified with that mask as a base.

Creating a User Mask Without a Template Mask

To create user masks without a template mask:

Use events as the basis for the user mask. The following example creates a mask for the user **pat** with the Show Table Statistics event and the failed attempts of the Alter Table event:

```
onaudit -a -u pat -e +SSTB,FALTB
```

For the syntax for creating a user mask and another example, see Chapter 10, “Utility Syntax,” on page 10-1.

Adding One or More Masks Using an Input File

To add one or more masks using an input file:

Use the **onaudit** utility to add one or more masks to the mask table with instructions from a file that has the same format as the output of **onaudit -o**. The following command reads a file in **/work/audit_up** and adds audit masks to the mask table according to the instructions in that file:

```
onaudit -f /work/audit_up
```

Figure 8-2 shows an example of an input file. The syntax for the input file is explained in Chapter 10, “Utility Syntax,” on page 10-1.

The example input file in Figure 8-2 includes the following information:

kickt	_secure1	
jacks	-	+ADCK,SRDRW,GRDB,OPDB
pat	_secure2	+ALTB -CRTB,CRIX,STSN
jaym	-	
johns	akee	-SALIX

Figure 8-2. Example Input File

- In the first line, the instructions specify auditing for user **kickt** in the new template **_secure1**.
- The second line creates a new mask called **jacks**, which contains the events Add Chunk (ADCK), successful attempts at Read Row (SRDRW), and all attempts at Grant Database Access (GRDB) and Open Database (OPDB).
- In the third line, the user **pat** is audited for all events that are specified in the template **_secure2**, and also for all attempts at Alter Table (ALTB), but not for attempts at Create Table (CRTB), Create Index (CRIX), and Start New Session (STSN).
- No template is specified for the target mask **jaym** in the fourth line, and no events are indicated; the mask is empty. (This prevents the **_default** mask from being applied to **jaym**.)
- In the fifth line, the target mask **johns** audits the same events as the mask **akee**, minus all successful attempts at Alter Index (SALIX).

Important: Future changes to a base mask are not reflected in other masks that might have been created or modified with that mask as a base.

An example of an audit mask input file, **adtmaskstd**, is provided in the **\$INFORMIXDIR/aaodir** UNIX directory or in the **%INFORMIXDIR%\aaodir** Windows directory. The **adtmaskstd** file is intended only to serve as a guide to the DBSSO for how to set up an audit mask.

Audit masks do not work the same way as audit configuration parameters during initialization of the database server. (See “Audit Configuration and the ADTCFG File” on page 7-11.) Specifically, audit masks are not automatically read from a file and initialized.

Displaying Audit Masks

To display all the audit masks and the audit events that each mask contains:

Use the **-o** option of the **onaudit** utility. When you issue the **onaudit -o -y** command, the output (mask name, base mask, audit events) appears as follows:

<code>_default</code>	-	UPAM,DRAM
<code>_require</code>	-	
<code>_exclude</code>	-	
<code>_guest</code>	-	CRDB,GRDB,GRTB
<code>terry</code>	-	-CRDB

You can specify a mask as an argument to the **-o** option. The following example displays only the mask for user **terry**:

```
onaudit -o -u terry
```

A list of audit masks is helpful when you need to modify them. You can use the modified output as an input file to modify a single mask or groups of masks in a single batch. For more information, see “Modifying Audit Masks.” For the complete syntax of the **onaudit -o** option and a description of the output, see Chapter 10, “Utility Syntax,” on page 10-1.

Tip: If you use a base mask to create or modify a mask, the base mask itself does not appear in the **onaudit -o** output for the new mask. If a mask is created or modified with a base mask, it does not refer to the base mask.

Modifying Audit Masks

The DBSSO can modify masks individually from the command line.

To modify audit masks:

Use the **-m** option of the **onaudit** utility to modify a single mask. This option lets you use another mask as a base to add or remove individual audit events. To modify several masks at a time, you can create a new input file, change the appropriate masks, and reload them in the mask table.

The following example shows how to modify the user mask **pat**. The **_guest** template mask forms a base from which a complete set of audit events is drawn. Settings for specific events from that file are then superseded by the events listed as arguments to the **-e** option.

```
onaudit -m -u pat -r _guest -e +ALTB,USTB
```

When you supply a base mask with the **-r** option, you replace all the audit events in the initial mask. When you change only a few events in a mask, you might not want to specify a base mask. For the syntax and another example of how to modify a mask, see Chapter 10, “Utility Syntax,” on page 10-1.

Deleting Audit Masks

To delete a single mask or all masks at once:

Use the **-d** option of the **onaudit** utility. The following example deletes the individual user mask for user **terry**:

```
onaudit -d -u terry
```

For the syntax of the **onaudit** utility, see Chapter 10, “Utility Syntax,” on page 10-1.

Audit Configuration Maintenance

The AAO normally performs the following tasks to maintain the audit configuration:

- Displaying the audit configuration
- Changing the audit mode (including auditing specific roles)
- Changing the audit error mode
- Turning off auditing
- Starting a new audit file (including specifying a directory and maximum file size).

This section describes how to use **onaudit** to perform these tasks. For the syntax of the **onaudit** utility, see Chapter 10, “Utility Syntax,” on page 10-1.

Displaying the Audit Configuration

To display the current audit configuration use the **-c** option of the **onaudit** utility.

- On UNIX, the Figure 8-3 shows output from the **onaudit -c** command.

```
onaudit -c

Onaudit -- Audit Subsystem Control Utility
Copyright (c) IBM Corp., 1998 - 2008

Current audit system configuration:
  ADTMODE =          1
  ADTERR =           0
  ADTPATH =          /tmp
  ADTSIZE =         20000
  Audit file =        64
```

Figure 8-3. Example of Output from the **onaudit -c** Command on UNIX

In Figure 8-3, the current audit system is configured as follows:

- ADTMODE is set to 1, which indicates that database server-managed auditing is on.
- ADTERR is set to 0, which indicates a continue error mode.
- ADTPATH shows the default directory for audit files.
- ADTSIZE, which represents the maximum size of the audit file, is specified as 20,000 bytes.
- The number of the current audit file in the current audit directory is 64.

If you are user **informix**, you can also retrieve this information from the SMI **sysadinfo** table in the **sysmaster** database. For details, see the *IBM Informix Administrator's Reference*.

- On Windows, the Figure 8-4 on page 8-13 shows output from the **onaudit -c** command.

```
onaudit -c

Onaudit -- Audit Subsystem Control Utility
Copyright IBM Corporation 1996, 2008 All rights reserved

Current audit system configuration:
  ADTMODE =          1
  ADTERR =           0
  ADTPATH =          %informixdir%/aaodir
  ADTSIZE =          50000
  Audit file =         0
```

Figure 8-4. Example Output from the `onaudit -c` Command on Windows

In Figure 8-4, the current audit system is configured as follows:

- `ADTMODE` is set to 1, which indicates that database server-managed auditing is on.
- `ADTERR` is set to 0, which indicates a continue error mode.
- `ADTPATH` shows the default directory for audit files.
- `ADTSIZE`, which represents the maximum size of the audit file, is specified as 50,000 bytes.
- The number of the current audit file in the current audit directory is 0, meaning that no other audit file exists in the current series.

Starting a New Audit File

You use the **onaudit** command to start a new audit file. For the **onaudit** syntax to start a new audit file, change the audit-file size, or change the pathname of the audit directory, see “The `onaudit` Utility” on page 10-1.

You can use more than one flag at a time in an **onaudit** command.

You can start a new audit file in one of the following ways:

- Use **onaudit -s** to change the maximum size of an audit file. If the audit file is already larger than the new size that you specify, the utility saves the current file and starts to write to a new one.

The following example changes the default size to 20,000 bytes:

```
onaudit -s 20000
```

- Use **onaudit -n** to start a new audit file without changing the maximum size.

This option, which the following example shows, saves the current audit log to another file whenever you run it:

```
onaudit -n
```

- Use **onaudit -p** to change the directory in which the database server writes audit files.

The following example specifies `/work/audit` as the UNIX file system where the audit files are to be kept:

```
onaudit -p /work/audit
```

The directory that you specify must exist.

- Start database-server- managed auditing. A new audit file starts every time that you start database-server- managed auditing.

Changing Audit Levels

- use the **onaudit** utility to change levels of auditing by the database server and to change the mandatory auditing of the DBSA.

For example, to start basic auditing, enter the following command:

```
onaudit -l 1
```

- To start auditing and automatically audit the actions of the DBSA, enter the following command:

```
onaudit -l 5
```

Changing the Audit Error Mode

As Chapter 7, “Overview of Auditing,” on page 7-1 and “Setting the Error Mode” on page 8-6 explain, the database server behaves in one of three ways if it encounters an error when it writes to the current audit file.

To change the audit error mode:

Use the **onaudit** utility.

The following example directs the database server to suspend processing of the current thread and continue the write attempt until it succeeds:

```
onaudit -e 1
```

Turning Off Auditing

To turn off auditing:

Use the **onaudit** utility.

The following example shows the command that turns off auditing:

```
onaudit -l 0
```

Warning: Although auditing might be properly configured to audit the execution of a particular utility by a particular user, audit records might not be generated if the utility fails to execute for any of the following reasons:

- The user does not have the correct UNIX permissions or Windows access privileges to execute the utility.
- The user incorrectly specifies the command syntax of the utility.
- The utility cannot connect to shared memory.

Chapter 9. Audit Analysis

The importance of audit analysis cannot be stressed enough. This chapter explains the following:

- The format of audit records that the database server produces
- How to perform audit analysis with or without SQL
- How to extract audit information from the audit trail for quick viewing
- How to load that data into a database for analysis with SQL
- How best to perform audit analysis on the extracted audit information

This chapter applies whether you use the database server or your operating system to store and maintain the audit trail. An overview of the audit analysis process is in Chapter 7, “Overview of Auditing,” on page 7-1

Audit-Record Format

The database server generates the second part of the audit record, with fields that depend on the audit event.

Table 9-1 shows the format of the database server audit records.

Table 9-1. Audit-Record Format

ONLN	date and time	hostname or hostname. domain.ext	pid	database server name	user name	errno	event mnemonic	Additional Fields
ONLN	2008-07-28 15:43:00.000	turk	4549	khan	jazt	0	CRDB	dbsch
ONLN	2008-07-28 15:43:18.000	turk	4549	khan	jazt	0	ACTB	dbsch;jazt:v1:103
ONLN	2008-07-28 15:43:19.000	turk	4549	khan	jazt	0	CLDB	dbsh
ONLN	2008-07-28 15:43:21.000	turk	4549	khan	jazt	0	ALFR	local:109::-:4:4: db1,db2,db3, rootdbs
ONLN	2008-07-28 15:43:28.000	turk	4549	khan	jazt	0	ALFR	local:109:aa5x:-: 32:4: db1,db2
ONLN	2008-07-28 15:43:29.000	turk	4549	khan	jazt	0	STDS	2:-
ONLN	2008-07-28 15:43:29.000	turk	4549	khan	jazt	0	STPR	100
...

ONLN

A fixed field used to identify Dynamic Server events

date and time

Indicates when the audit event was recorded

hostname

The name of the UNIX host computer of the client application that executes the audit event

hostname.domain.ext

The name of the Windows host computer, domain, and extension of the client application that executes the audit event

pid

The process ID of the client application that causes the database server to execute the audit event

database server name

The name of the database server on which the audit event is executed

user name

The login name of the user who requests the event

errno

The event result that contains the error number that the event returns, indicating success (0) or failure

event mnemonic

Database server audit event that the database server executed, such as ALFR (Alter Fragment)

additional fields

Any fields that identify databases, tables, and so on. These additional fields are audit-event fields that contain information captured in tabular form by the **onshowaudit** utility for audit analysis.

For operating-system-managed auditing on UNIX, the database server audit record is an additional field for the operating-system audit record. Chapter 11, "Audit Event Mnemonics and Fields," on page 11-1 lists the audit-event fields.

Audit Analysis Without SQL

Use the **onshowaudit** utility to extract data for audit analysis. This utility can perform some basic filtering such as user or database server name. You can then send the extracted data to standard output (for example, your screen) and use UNIX utilities such as **grep**, **sed**, and **awk** or Windows utilities to analyze it. You can also choose to put the data in a database and analyze it with SQL, as the next section describes.

Only the AAO can execute **onshowaudit**. If role separation is not enabled, user **informix** will be the AAO. (Superuser **root** on UNIX is always an AAO.) Because disclosure of audit records represents a security threat, only the AAO should read the extracted records.

For example, the following command extracts audit records for the user **pat** from an audit file named **laurel.12**, on UNIX, and sends the audit records to standard output:

```
onshowaudit -I -f laurel.12 -u pat
```

The command-line syntax for how to extract information with **onshowaudit** is explained in Chapter 10, "Utility Syntax," on page 10-1.

Audit Analysis with SQL

You can also use the **onshowaudit** utility to reformat the extracted data and redirect it to a data file and then use the **dbload** utility to load that data into a database table. This section explains this process.

Planning for SQL Audit Analysis

When you plan audit analysis with the database server, consider that the audit-analysis process itself might generate audit records, depending on how the audit is configured. One way to avoid generating unwanted audit records as a result of audit analysis is to use a separate unaudited instance of the database server.

To perform audit analysis with SQL, you must use a program to access the database and table that you created. Use the DB–Access utility to construct and execute SQL statements or develop an application with an IBM Informix application development tool or an SQL API, such as Informix ESQL/C.

Whether you perform analysis with DB–Access or build a customized application, remember the advice given for audit review in “Audit Analysis Overview” on page 7-14. To view audit events for specific objects, select rows based on their value in the **dbname**, **tabid**, or **row_num** column.

If you discover suspicious activity based on initial analysis of the audit table in the database server, you might increase the scope of your collection of audit events to pinpoint the problem. If you feel certain you have a security problem, see “DBMS Security Threats” on page 7-17.

Revoking and Granting Privileges to Protect Audit Data

When you create a database as described in the following sections, make sure that the database is protected against unauthorized access.

Tables that you create in non-ANSI compliant databases have privileges that allow all users access. Although the default database permissions or access privileges prevent access to the tables, proper security practice protects the audit-analysis table in a database that is not ANSI-compliant by revoking access from all other users as soon as that table is created.

You can use the following SQL statements to control access:

```
REVOKE ALL ON table FROM PUBLIC
GRANT ALL ON table TO informix
```

After table privileges are revoked, generally with the REVOKE statement, you can grant individual users (for example, user **informix**) access to the tables with the GRANT statement. For information on SQL statements, see the *IBM Informix Guide to SQL: Syntax*.

Tables created in ANSI-compliant databases have privileges that allow access only by the owner, which is the appropriate security measure.

You can also use the **NODEFDAC** environment variable to control access. When set to yes, **NODEFDAC** does not allow default table privileges (Select, Insert, Update, and Delete) to be granted to PUBLIC when a new table is created in a database that is not ANSI-compliant. For details, see the *IBM Informix Guide to SQL: Reference*.

Preparing Audit Analysis Records for SQL Access

Take the following steps to prepare audit records for SQL analysis:

1. Create a data file to use with **dbload**.
2. Create a database and table in which to store the audit data.
3. Create a command file to use with **dbload**.
4. Load the audit data into the table.

Creating a Data File for dbload

The first step to prepare for SQL-based audit analysis is to use **onshowaudit -l** to extract selected audit records in **dbload** format and put them in an output file. The following example extracts audit records for the user **pat** from the database server-managed audit file **laurel.11** and directs the records to the **records_pat** output file:

```
onshowaudit -l -f laurel.11 -u pat -l > records_pat
```

Important: You must remove the six header lines that appear in the output file before you use the file as input for the **dbload** utility because **dbload** cannot process the header lines.

The command-line syntax to extract information with **onshowaudit** is explained in Chapter 10, “Utility Syntax,” on page 10-1.

Creating a Database for Audit Data

To load data files into a database with **dbload**, a database to receive the data must already exist.

Create a database to hold copies of audit records with the **CREATE DATABASE** statement. By default, the **CREATE DATABASE** statement creates the database with privileges that allow access only to the owner, which is the appropriate security measure. It is not necessary to use logging within a database created strictly for audit analysis because the data should not be modified.

The following SQL statement creates a database called **auditlogs97**:

```
CREATE DATABASE auditlogs97
```

You can also create an ANSI-compliant database. Although an ANSI-compliant database has the additional overhead of logging, its treatment of table permissions or access privileges makes it attractive in a secure environment. For more information about UNIX permissions or Windows access privileges, refer to “Revoking and Granting Privileges to Protect Audit Data” on page 9-3.

The following SQL statement creates an ANSI-compliant database:

```
CREATE DATABASE auditlogs97 WITH LOG MODE ANSI
```

Creating a Table for Audit Data

To load data files into a database with **dbload**, a table to receive the data must already exist.

Create a table to hold audit data with the **CREATE TABLE** statement. The order and data types of the columns is important.

Use the order shown in the example in Figure 9-1 on page 9-5. The sample schema reflects the format of the **dbload** data file that **onshowaudit** created.

The sample **CREATE TABLE** statement in Figure 9-1 on page 9-5 creates an audit table with the name **frag_logs**. For information about the contents of each column,

see “Interpreting Data Extracted from Audit Records” on page 9-6. The sample CREATE TABLE statement in Figure 9-1 does not include the WITH CRCOLS option, which is for conflict resolution during database replication. To replicate the audit database, use WITH CRCOLS in the CREATE TABLE statement. The table that the statement in Figure 9-1 creates does not have any indexes. To

```
CREATE TABLE frag_logs (  
    adttag CHAR(4),  
    date_time DATETIME YEAR TO FRACTION(3),  
    hostname CHAR(18),  
    pid INT,  
    server CHAR(18),  
    username CHAR(8),  
    errno INT,  
    code CHAR(4),  
    dbname CHAR(18),  
    tabid INT,  
    objname CHAR(18),  
    extra_1 INT,  
    partno INT,  
    row_num INT,  
    login CHAR(8),  
    flags INT,  
    extra_2 VARCHAR(160,1));
```

Figure 9-1. Sample CREATE TABLE Statement for a Dynamic Server Audit Table

improve audit-analysis performance, you can place indexes on columns within the table, depending on the type of analysis that you perform. For guidance on indexing columns, see your *IBM Informix Performance Guide*.

Creating a Command File for dbload

To load the audit information into the table that you created:

First create an ASCII command file for the **dbload** utility. This command file must specify the number of columns and the field delimiter that are used in the data file that **onshowaudit** created. For a description of command files and their use with **dbload**, see the *IBM Informix Migration Guide*.

Include the following information when you create the command file for **dbload**:

Delimiter

|

Number of columns

17

Table name

Table you created to receive the data

Data file name

Output file you create (to serve as input for **dbload**)

The following example uses the **FILE** statement to create a command file for **dbload**. The example includes the **records_pat** data file created in “Creating a Data File for dbload” on page 9-4 and the **frag_logs** table created in “Creating a Table for Audit Data” on page 9-4.

```
FILE records_pat DELIMITER '|' 17;  
INSERT INTO frag_logs;
```

You now have the tools necessary to load a data file into the table that you created.

Loading Audit Data into a Database

After you have the database, table, data, and command files for audit analysis:

Use the **dbload** command to load the audit data into the table.

The following example executes the commands specified in the **user_records** command file to load data into the **auditlogs97** database created in “Creating a Database for Audit Data” on page 9-4:

```
dbload -d auditlogs97 -c user_records
```

After the data is loaded, begin your audit analysis with SQL.

Interpreting Data Extracted from Audit Records

When you create a database table to contain audit records that you extract from audit files, you provide a column for each field in the audit record. Figure 9-2 on page 9-7 lists recommended column names that are used in Figure 9-1 on page 9-5 and describes the information that each column contains.

Column Name	Description
adttag	ONLN
date_time	The date and time of the audited event
hostname	The database servername
pid	The process ID
server	The database server name
username	The username associated with the audited event
errno	The error number, if any
code	The error code, if any
dbname	The name of the database
tabid	The ID number of the affected table
objname	The index name and the table name, or similar identifier (Not in audit tables created with Informix database servers prior to Version 7.0)
extra_1	Information specific to the object and event, as shown in “Audit-Event Fields” on page 11-5
partno	Fragmentation information (Not in audit tables created with Informix database servers prior to Version 7.0)
row_num	The physical row number in the affected table, which combines the row ID and the old row ID and identifies each row for the events Read Row (RDRW), Insert Row (INRW), Update Current Row (UPRW), and Delete Row (DLRW)
login	The user login name
flags	The flag set for the event, as shown in “Audit-Event Fields” on page 11-5
extra_2	Information determined by the flag.

Figure 9-2. Audit-Event Columns in Database Table for SQL Access

Chapter 10. Utility Syntax

This chapter contains syntax and usage information for the following utilities:

- The **onaudit** utility performs the following operations on both UNIX and Windows:
 - Displays audit masks
 - Creates audit masks
 - Modifies audit masks
 - Deletes audit masks
 - Shows the audit configuration
 - Changes global auditing activities
 - Enables and disables auditing
 - Sets the error mode
 - Establishes mandatory auditing for various administrative roles
 - Starts a new audit file in the audit trail
 - Sets the directory in which audit files reside
 - Specifies the maximum size for each audit file
- The **onshowaudit** utility performs the following operations on both UNIX and Windows:
 - Extracts audit information from the audit trail
 - Prepares extracted audit data for the **dbload** utility
- On UNIX, the **onaudit** utility also performs the following operations:
Determines whether the database server or the operating system manages the audit trail
- The **onaudit** utility also performs the following operation on Windows:
Establishes auditing that the database server manages

The onaudit Utility

The **onaudit** utility manages audit masks and auditing configuration. You can perform the following tasks with **onaudit**:

- “Showing Audit Masks” on page 10-2
- “Creating or Adding an Audit Mask” on page 10-3
- “Modifying an Audit Mask” on page 10-2
- “Deleting an Audit Mask” on page 10-5
- “Starting a New Audit File” on page 10-6
- “Showing the Audit Configuration” on page 10-6
- “Changing the Audit Configuration” on page 10-7

You must perform each of these tasks individual in separate **onaudit** commands. If you run the **onaudit** command without any options, it displays a usage summary.

If your system has role separation, only the DBSSO or AAO can run the **onaudit** utility. The DBSSO can perform only **onaudit** functions that involve audit masks,

and the AAO can perform only **onaudit** functions that involve audit configuration parameters. Without role separation, the user **informix** or **root** can perform all these tasks.

The DBSSO can change audit masks dynamically. Changes to user, default, template, and global masks become effective immediately for user sessions.

Showing Audit Masks



Element	Purpose	Key Considerations
-o	Outputs audit masks.	None.
-u mask	Names a specific mask to display.	Can be any existing audit mask.
-y	Automatically responds yes to the confirmation prompt.	None.

The **-o** option of the **onaudit** utility sends the mask display to standard output, as follows:

- If the **-u mask** option is omitted, all masks are displayed.
- If the **-y** and **-u** options are omitted, **onaudit** requests confirmation before it displays all the masks, which can amount to a lot of data.

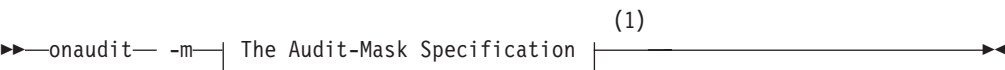
The following example illustrates the format of the output file. The format is the same as that of an input file for **onaudit**, as “Modifying an Audit Mask” describes.
maskname basemask audit_events

Because the database server keeps no record of the base mask that is used to create or modify a mask, a single hyphen (-) always appears in the *basemask* placeholder.

The following example shows output for the command **onaudit -o -u pat**. It indicates that the individual user mask **pat** contains the Lock Table (LKTB), Create Table (CRTB), and failed attempts of Add Chunk (ADCK) audit events.

```
pat      -      LKTB,CRTB,FADCK
```

Modifying an Audit Mask



Notes:

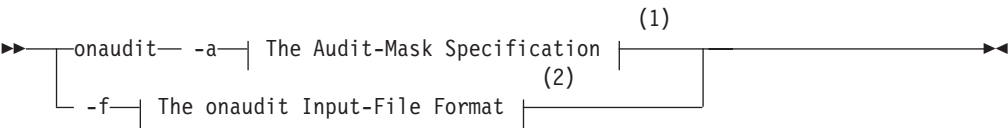
- 1 see “The Audit-Mask Specification” on page 10-3

Element	Purpose	Key Considerations
-m	Modifies an existing audit mask.	None.

The following example modifies an audit mask for the user **pat**. The modified mask audits the events specified in the **_Hsecure** template mask, with the addition of all attempts of Lock Table (LKTB) and only failed attempts of Alter Table (ALTB).

```
onaudit -m -u pat -r _Hsecure -e +LKTB,FALTB
```

Creating or Adding an Audit Mask



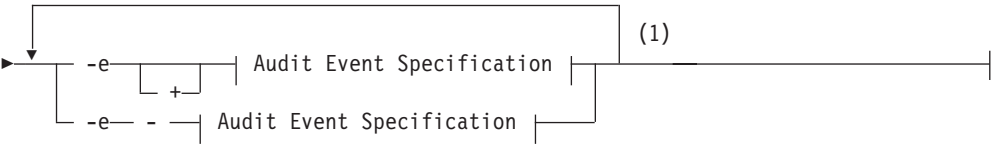
Notes:

- 1 see “The Audit-Mask Specification”
- 2 see “The onaudit Input-File Format” on page 10-4

Element	Purpose	Key Considerations
-a	Adds a new audit mask.	None.
-f	Names a file that can include instructions to add any or all of the audit masks to the mask table.	References: The syntax for the input file is described in “The onaudit Input-File Format” on page 10-4.

The Audit-Mask Specification

The Audit-Mask Specification:



Audit Event Specification:



Notes:

- 1 Only one occurrence of each choice is allowed. However, multiple options are allowed on the same invocation

Element	Purpose	Key Considerations
+	Events that follow are to be added to <i>targetmask</i> list of audit events	The + is the default and thus is optional.
–	Events that follow are to be dropped from <i>targetmask</i> list of audit events	None.
-e	Indicates that audit events are to be added or removed from <i>targetmask</i>	Events specified as arguments to -e override events listed in any base mask specified with the -r option.
-r <i>basemask</i>	Name of an existing audit mask. Events currently listed in <i>basemask</i> are applied to <i>targetmask</i> .	Subsequent changes to <i>basemask</i> are not reflected in masks for which <i>basemask</i> has been used as a base. If no <i>basemask</i> is specified and no events are specified with the -e flag, onaudit creates an empty target mask.
-u <i>targetmask</i>	Names a user, template, _default , _require , or _exclude mask to be created or modified.	The <i>targetmask</i> identifier must have 32 or fewer characters.
Fevent	Specifies that only failed event attempts are to be audited.	The <i>event</i> can include the event code (mnemonic) for any event listed in the table “Audit-Event Mnemonics” on page 11-1.
Sevent	Specifies that only successful event attempts are to be audited.	Same as for Fevent
<i>event</i>	An event to audit, whether the event execution succeeds or fails.	Same as for Fevent

Important: Do not include any spaces in the events list. You might get unpredictable results.

The following example creates a new audit mask named **pat** for the user **pat**. The new mask audits the events specified in the **_secureL** template mask, but excludes Read Row (RDRW) and includes Lock Table (LKTB), successful attempts at Add Chunk (ADCK), and all attempts at Create Table (CRTB).

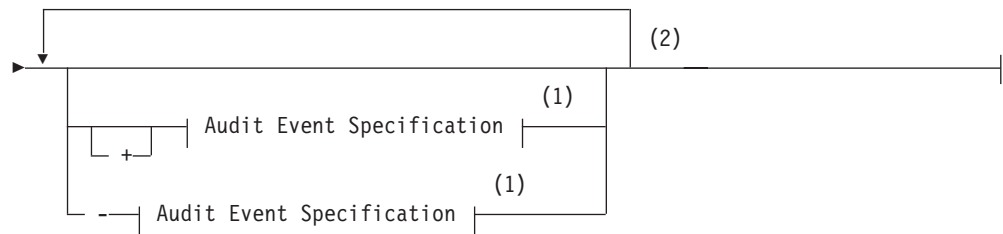
```
onaudit -a -u pat -r _secureL -e -RDRW, -e +LKTB,SADCK,CRTB
```

A user mask is only one of the three masks that specify auditing for an individual. Auditing instructions are read from the user mask first, followed by the **_require** and **_exclude** masks. For details, refer to Chapter 7, “Overview of Auditing,” on page 7-1.

The onaudit Input-File Format

The onaudit Input-File Format:





Notes:

- 1 See page “The Audit-Mask Specification” on page 10-3
- 2 Only one occurrence of each choice is allowed. However, multiple options are allowed on the same invocation

Element	Purpose	Key Considerations
+	Events that follow are to be added to the list of audit events in <i>targetmask</i> .	None.
–	Used before an event, it indicates that the events that follow are to be removed from the list of audit events in <i>targetmask</i> . Used alone, it creates an empty mask.	None.
<i>basemask</i>	Name of an existing audit mask to use as a base.	The auditing instructions of the base mask are copied to the target mask, in addition to (or except for) the audit events that follow.
<i>targetmask</i>	Identifies the user, template, _default , _require , or _exclude mask to add.	Mask names must not exceed eight characters, and template mask names must begin with an underscore (_) symbol.

The following example uses a modified output file, created by the **onaudit -o** option, as the input file for **onaudit -f**:

```
onaudit -f /work/masks_feb.97
```

For an example of an **onaudit** input file, see Chapter 8, “Audit Administration,” on page 8-1.

Deleting an Audit Mask



Element	Purpose	Key Considerations
-d	Deletes an audit mask.	None.
-u mask	Names a specific mask to delete.	<i>Mask</i> can be any existing mask.
-y	Automatically responds yes to the confirmation prompt.	None.

The **-d** option of the **onaudit** utility deletes audit masks, as the following list describes:

- If the **-u** *mask* option is omitted, all masks are deleted, including **_default**, **_require**, and **_exclude**.
- Because of the potential to make a significant mistake, the **onaudit** utility prompts you for confirmation before it deletes all masks. Thus, if the **-y** and **-u** options are omitted, **onaudit** requests confirmation.

Starting a New Audit File

►► onaudit — -n ◀◀

Element	Purpose	Key Considerations
-n	Starts a new audit file.	None.

Storing Database Server Audit Files

For database server-managed auditing, the **-n** option to the **onaudit** utility closes the current database server audit file, stores it in a specified directory, and opens a new audit file named *servername.integer*. The *servername* value is the name of the database server being audited, and *integer* is the next available integer. For example, if the last audit file saved for the **maple** database server was named **maple.123**, the next audit file is saved in a file called **maple.124**.

Storing Operating-System Audit Files

For operating-system-managed files, the **-n** option to **onaudit** closes the current operating-system audit file, stores it as part of the operating-system audit trail, and opens a new audit file. For the naming conventions for files in the audit trail, see your operating-system documentation.

Showing the Audit Configuration

►► onaudit — -c ◀◀

Element	Purpose	Key Considerations
-c	Shows the current audit configuration.	None.

The **-c** option directs **onaudit** to display the current state of auditing.

Figure 10-1 on page 10-7 shows an example audit-configuration output on UNIX.

```

onaudit -c

Onaudit -- Audit Subsystem Control Utility
Copyright (c) IBM Corp., 1998-2008

Current audit system configuration:
    ADTMODE          = 1
    ADTERR           = 0
    ADTPATH          = /tmp
    ADTSIZE          = 20000
    Audit file       = 64

```

Figure 10-1. Sample Audit-Configuration Output on UNIX

Figure 10-2 shows an example of audit-configuration output on Windows.

```

onaudit -c

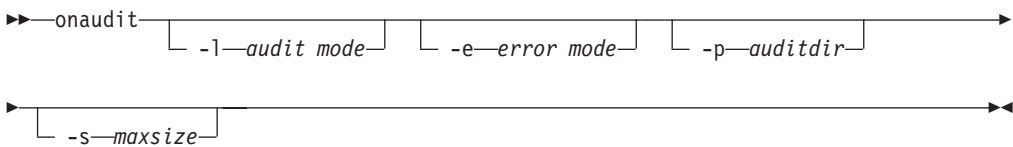
Onaudit -- Audit Subsystem Control Utility
Copyright (c) IBM Corp., 1998-2008

Current audit system configuration:
    ADTMODE          = 1
    ADTERR           = 0
    ADTPATH =          %informixdir%/aaodir
    ADTSIZE =          50000
    Audit file =          0

```

Figure 10-2. Sample Audit-Configuration Output on Windows

Changing the Audit Configuration



Element	Purpose	Key Considerations
-e <i>error mode</i>	Specifies the error-handling method for auditing when a record cannot be written to the audit file or event log.	Restrictions: The <i>error mode</i> parameter can have one of the following values: 0, 1, 3. Additional Information: This option pertains to the value set for the ADTERR configuration parameter in the ADTCFG file. The value can be changed only when auditing is on. For details of the valid <i>error mode</i> values, see “Using the -e Option” on page 10-8.
-l <i>audit mode</i>	Specifies the audit mode.	Restrictions: The <i>audit mode</i> parameter can have one of the following values: 0, 1, 3, 5, 7. Additional Information: This option pertains to the value set for the ADTMODE configuration parameter in the ADTCFG file. For details of the valid <i>audit mode</i> values and prerequisites for using this option, see “Using the -l Option” on page 10-8.

Element	Purpose	Key Considerations
-p <i>auditdir</i>	Specifies the directory in which the database server creates audit files.	Restrictions: You can change the <i>auditdir</i> value only for database server-managed auditing and only when auditing is in effect. Additional Information: This option pertains to the value set for the ADTPATH configuration parameter in the ADTCFG file. The change occurs with the next write attempt. The database server starts a new audit file in the new directory, beginning with the first available number that is equal to or greater than 0.
-s <i>maxsize</i>	Specifies the maximum size (in bytes) of an audit file.	Restrictions: The <i>maxsize</i> can be any value between 10,240 bytes and approximately 2 gigabytes (the maximum value of a 32-bit integer). If you specify a size that is less than the minimum, it will be set automatically at the minimum. You can specify the <i>maxsize</i> value only for database server-managed auditing and only when auditing is in effect. Additional Information: This option pertains to the value set for the ADTSIZE configuration parameter in the ADTCFG file. When an audit file reaches or exceeds <i>maxsize</i> , the database server closes the current file and starts a new audit file.

For information on the audit configuration parameters in the **ADTCFG** file, see Chapter 12, “The ADTCFG File,” on page 12-1.

Changes made to the audit configuration with **onaudit** take effect immediately for all user sessions, including existing sessions. For information on how audit-configuration changes interact with the **ADTCFG** file, see Chapter 7, “Overview of Auditing,” on page 7-1.

Using the -e Option

This section discusses the values that you can enter for the **-e** *error mode* option of **onaudit**.

To specify *continue mode*, enter 0 as the argument to the **-e** option. In continue mode, the database server continues processing the thread and notes the error in the message log. Errors for subsequent attempts to write to the audit file are also sent to the message log. For information about the message log, see your *IBM Informix Administrator's Guide*.

To specify one of the *halt modes*, which suspend processing or shut the database server down, enter one of the following arguments to the **-e** option:

- Enter 1 to suspend processing a thread when the database server cannot write a record to the current audit file and should continue the write attempt until it succeeds.
- Enter 3 to shut down the database server.

Using the -l Option

This section discusses the values that you can enter for the **-l** *audit mode* option of **onaudit**.

The value 0 turns *off* auditing. The database server stops auditing for all existing sessions, and new sessions are not audited.

To turn *on* auditing with the **-l** option, the **ADTPATH** configuration parameter must be set to an existing directory and corresponding permissions must be set up correctly.

The following values turn on auditing, as follows:

- 1 turns on auditing for all sessions but does not automatically audit DBSSO and the DBSA actions.
- 3 turns on auditing and automatically audits DBSSO actions.
- 5 turns on auditing and automatically audits DBSA actions.
- 7 turns on auditing and automatically audits all DBSSO and DBSA actions, including `_exclude` events.

The onshowaudit Utility

The **onshowaudit** utility lets you extract information from an audit trail. You can direct this utility to extract information for a particular user or database server or both. This information enables you to isolate a particular subset of data from a potentially large audit trail.

The records are formatted for output. By default, **onshowaudit** displays the extracted information on the screen. You can redirect the formatted output to a file or pipe and can specify that **onshowaudit** reformat the output so you can load it into an Informix database table.

The **onshowaudit** utility extracts data from an audit trail but does not process the records or delete them from the audit trail. Access the audit trail only with the **onshowaudit** utility, which has its own protection:

- With role separation off, only user **informix** (and user **root** on UNIX) can run **onshowaudit**.
- With role separation on, only the AAO can run **onshowaudit**.

Syntax

The UNIX command-line syntax for **onshowaudit** follows.

```
▶▶ onshowaudit [-I] [-n servernumber] [-f path] [-u username]
               [-s servername] [-l loadfile]
```

The Windows command-line syntax for **onshowaudit** follows.

```
▶▶ onshowaudit [-n servernumber] [-f path] [-ts] [-tf]
               [-u username -s servername] [-l loadfile] [-d]
```

Important: *Windows only:* If you include the **-l** option in your **onshowaudit** command, you must remove the six header lines that appear in the output file before you use that file as input for **dbload** or for an external file.

Elements

The following table identifies the syntax terms that can appear in an **onshowaudit** command line.

Any command-line options that you specify determine which part of the audit trail the **onshowaudit** utility uses.

Element	Purpose	Key Considerations
-d	On Windows, assumes the default values for the user (<i>current user</i>) and the database server (INFORMIXSERVER)	None
-f path	Specifies a specific audit trail to examine, only for database server-managed auditing	If this option is omitted, or if <i>path</i> is only a <i>filename</i> , see the notes that immediately follow this table.
-I	On UNIX, uses the Informix database server audit trail	None
-l	Directs onshowaudit to extract information with delimiters so that it can be redirected to a file or pipe and loaded into a database table or other application that accepts delimited data. On Windows, you must enter a load file name argument for the -l option. On UNIX, this file name argument is optional. If you do not specify a file name on UNIX, the output is placed in a standard output file.	On Windows, you must remove the six header lines that appear in the output file before you use that file as input for dbload or for an external file. For information on the file format, see Chapter 9, "Audit Analysis," on page 9-1. For information on the dbload utility, see the <i>IBM Informix Migration Guide</i> . For information on loading data with external tables, see the <i>IBM Informix Administrator's Reference</i> .
-n servernumber	Extracts audit records from the ADTPATH location specified in the adtcfg.servernumber file.	If the adtcfg.servernumber file does not exist, the contents of adtcfg are used for audit configuration.
-tf	On Windows, shows only <i>failure</i> audit records	None
-ts	On Windows, shows only <i>success</i> audit records	None
-s servername	Specifies the database server about which to extract audit information	None
-u username	Specifies the login name of a user about which to extract audit information	None

If **-f** is omitted, **onshowaudit** searches for audit files in the **ADTPATH** directory specified in the default **adtcfg** file. The **-f path** option specifies the directory and filename of the audit files. The audit directory and filename must conform to minimum security levels. The directory should be owned by user **informix**, belong to that AAO group, and should not allow public access (0770 permission). The files should have comparable permissions (0660 permission). The files should not be symbolic links to other locations. The directory, however, can be a symbolic link. If the audit directory and files are not secure, **onshowaudit** returns an error message and does not display the audit results.

| If you modify the audit configuration with the **onaudit** utility, the
| **adtcfg.servernumber** file stores the changed configuration. If the server's audit
| configuration is modified, use the **-n** option to specify the server number for
| **onshowaudit**. Using the **-n** option allows **onshowaudit** to read the right
| **ADTPATH** stored in **adtcfg.servernumber** file. The **onshowaudit** utility extracts
| data from all the audit files it finds that are in sequence, starting with the lowest
| integer.

| If an *incomplete pathname* (nothing but a filename) is specified, the **onshowaudit**
| utility searches the **ADTPATH** directory for that file and extracts audit data from
| it.

| If a *complete pathname* is specified, the **onshowaudit** utility extracts audit data from
| the named file.

| For information on the auditing configuration parameters in the **ADTCFG** file, see
| Chapter 12, "The ADTCFG File," on page 12-1.

| The database server does not audit the execution of the **onshowaudit** utility.

| **Important:** Version 7.2 and later versions of the **onshowaudit** utility can parse and
| process the new and updated record structures for fragmented tables and indexes,
| which can span multiple partitions. If you use Version 7.2 or a later version of
| **onshowaudit** to analyze records that a database server prior to Version 7.0 created,
| you might receive inaccurate results. Version 7.2 and later versions of **onshowaudit**
| expect to find an additional field for fragmentation (**partno**) in certain audit
| records, but this field is absent in audit records prior to Version 7.0.

Chapter 11. Audit Event Mnemonics and Fields

This chapter contains a table listing auditable events for each database server (listed alphabetically by event mnemonic), as well as a table listing audit-event records and their fields.

Important: The Dynamic Server secure-auditing facility audits only the events that this chapter lists. You might encounter additional SQL statements that the secure-auditing facility does not audit.

Audit-Event Mnemonics

This table contains an alphabetical list of audit-event mnemonics (event codes) mapped to the name of the event.

Mnemonic	Event Name
ACTB	Access Table
ADCK	Add Chunk
ADLG	Add Transaction Log
ALFR	Alter Fragment
ALIX	Alter Index
ALLC	Alter Security Label Component
ALME	Alter Access Method
ALOC	Alter Operator Class
ALOP	Alter Optical Cluster
ALSQ	Alter Sequence
ALTB	Alter Table
BGTX	Begin Transaction
CLDB	Close Database
CMTX	Commit Transaction
CRAG	Create Aggregate
CRAM	Create Audit Mask
CRBS	Create Storage Space
CRBT	Create Opaque Type
CRCT	Create Cast
CRDB	Create Database
CRDM	Create Domain
CRDS	Create Dbspace
CRDT	Create Distinct Type
CRIX	Create Index
CRLB	Create Security Label
CRLC	Create Security Label Component
CRME	Create Access Method

Mnemonic	Event Name
CROC	Create Operator Class
CROP	Create Optical Cluster
CRPL	Create Security Policy
CRPT	Decryption failure or attempt
CRRL	Create Role
CRRT	Create Named Row Type
CRSN	Create Synonym
CRSP	Create SPL Routine
CRSQ	Create Sequence
CRTB	Create Table
CRTR	Create Trigger
CRVW	Create View
CRXD	Create XA Data Source
CRXT	Create XA Data Source Type
DLRW	Delete Row
DNCK	Bring Chunk Off-line
DNDM	Disable Disk Mirroring
DRAG	Drop Aggregate
DRAM	Delete Audit Mask
DRBS	Drop Storage Space
DRCK	Drop Chunk
DRCT	Drop Cast
DRDB	Drop Database
DRDM	Drop Domain
DRDS	Drop Dbospace
DRIX	Drop Index
DRLB	Drop Security Label
DRLC	Drop Security Label Component
DRLG	Drop Transaction Log
DRME	Drop Access Method
DROC	Drop Operator Class
DROP	Drop Optical Cluster
DRPL	Drop Security Policy
DRRL	Drop Role
DRRT	Drop Named Row Type
DRSN	Drop Synonym
DRSP	Drop SPL Routine
DRSQ	Drop Sequence
DRTB	Drop Table
DRTR	Drop Trigger
DRTY	Drop Type

Mnemonic	Event Name
DRVW	Drop View
DRXD	Drop XA Data Source
DRXT	Drop XA Data Source Type
EXSP	Execute SPL Routine
GRDB	Grant Database Access
GRDR	Grant Default Role
GRFR	Grant Fragment Access
GRLB	Grant Security Label
GRRL	Grant Role
GRSA	Grant DBSECADM
GRSS	Grant SETSESSIONAUTH
GRTB	Grant Table Access
GRXM	Grant Exemption
INRW	Insert Row
LGDB	Change Database Log Mode
LKTB	Lock Table
LSAM	List Audit Masks
LSDB	List Databases
MDLG	Modify Transaction Logging
ONAU	onaudit
ONBR	onbar
ONCH	oncheck
ONIN	oninit
ONLG	onlog
ONLO	onload
ONMN	onmonitor
ONMO	onmode
ONPA	onparams
ONPL	onpload
ONSP	onspaces
ONST	onstat
ONTP	ontape
ONUL	onunload
OPDB	Open Database
RDRW	Read Row
RLOP	Release Optical Cluster
RLTX	Rollback Transaction
RMCK	Clear Mirrored Chunks
RNDB	Rename Database
RNDS	Rename dbspace
RNIX	Rename Index

Mnemonic	Event Name
RNLB	Rename Security Label
RNLC	Rename Security Label Component
RNPL	Rename Security Policy
RNSQ	Rename Sequence
RNTC	Rename Table/Column
RSOP	Reserve Optical Cluster
RVDB	Revoke Database Access
RVDR	Revoke Default Role
RVFR	Revoke Fragment Access
RVLB	Revoke Security Label
RVRL	Revoke Role
RVSA	Revoke DBSECADM
RVSS	Revoke SETSESSIONAUTH
RVTB	Revoke Table Access
RVXM	Revoke Exemption
SCSP	SYSTEM Command, SPL Routine
STCN	Set Constraint
STCO	Set Collation
STDF	Set Debug File
STDP	Set Database Password
STDS	Set Dataskip
STEP	Set Encryption Password
STEV	Set Environment
STEX	Set Explain
STIL	Set Isolation Level
STLM	Set Lock Mode
STNC	Set No Collation
STOM	Set Object Mode
STOP	Stop Violations
STPR	Set Pdqpriority
STRL	Set Role
STRS	Set Resident
STRT	Start Violations
STSA	Set Session Authorization
STSC	Set Statement Cache
STSN	Start New Session
STTX	Set Transaction Mode
SVXD	Save External Directives
TCTB	Truncate Table
TMOP	Time Optical Cluster
ULTB	Unlock Table

Mnemonic	Event Name
UPAM	Update Audit Mask
UPCK	Bring Chunk Online
UPDM	Enable Disk Mirroring
UPRW	Update Row
USSP	Update Statistics, SPL Routine
USTB	Update Statistics, Table

Audit-Event Fields

The following table contains audit-event information in alphabetic order by mnemonic code.

The table shows the audit-event information that is captured in tabular form by the **onshowaudit** utility for audit analysis:

- The **Event** column shows the event name.
- The **Mnemonic** column has the acronym that database server utilities use to identify audit events.
- The **Variable Contents** column has other categories of **onshowaudit** information that appear for the event on that row. These categories of information are the following: **dbname**, **tabid**, **objname**, **extra_1**, **partno**, **row_num**, **login**, **flags**, and **extra_2**.

For some events, the **onshowaudit** utility puts two different pieces of information in the **extra_2** field. In this case, the two parts are separated by a semicolon.

- The Notes section after the table provides more information about some of the entries in the **Variable Contents** column.

Tip: Granted lists can be long for SQL statements such as GRANT and REVOKE. If the list for an event to be audited does not fit into a single record, the database server creates several audit records to carry the complete information.

Event	Mnemonic	Variable Contents
Access Table	ACTB	dbname: dbname tabid: owner name, tabid
Chunk, Add	ADCK	dbname: dbspace, name extra_1: offset flags: mirror status ¹ extra_2: path and size
Transaction Log, Add	ADLG	dbname: dbspace, name extra_1: log size

Event	Mnemonic	Variable Contents
Alter Fragment	ALFR	dbname: dbname tabid: tabid objname: idxname extra_1: operation type ¹⁸ login: owner flags: frag flags ¹⁵ extra_2: dbspaces
Index, Alter	ALIX	dbname: dbname tabid: tabid login: owner ¹⁴ flags: cluster flag ^{9,14} extra_2: index name ¹⁴
Security Label Component, Alter	ALLC	dbname: dbname objname: component name extra_2: component type
Access Method, Alter	ALME	dbname: dbname tabid: access, method ID objname: access method, name login: access method, owner
Operator Class, Alter	ALOC	dbname: dbname extra_1: cluster size login: owner extra_2: cluster name
Optical Cluster, Alter	ALOP	dbname: dbname extra_1: cluster size login: owner extra_2: cluster name
Sequence, Alter	ALSQ	dbname: dbname tabid: tabid
Table, Alter	ALTB	dbname: dbname tabid: old tabid extra_1: new tabid ¹⁴ partno: frag_id extra_2: new part-nolist ¹⁴

Event	Mnemonic	Variable Contents
Transaction, Begin	BGTX	
Database, Close	CLDB	dbname: dbname
Transaction, Commit	CMTX	
Aggregate, Create	CRAG	dbname: dbname objname: aggregate name login: owner
Audit Mask, Create	CRAM	login: user id
Storage Space, Create	CRBS	dbname: storage, space name login: owner flags: mirror status ¹ extra_2: media
Opaque Type, Create	CRBT	dbname: dbname objname: opaque type name login: opaque type, owner
Cast, Create	CRCT	dbname: dbname tabid: type ID of from type objname: function name or "-" extra_1: xid of the from type partno: type ID of the to type row_num: xid of the to type login: function owner or "-"
Database, Create	CRDB	dbname: dbspace extra_2: dbname
Domain, Create	CRDM	
Dbspace, Create	CRDS	dbname: dbspace, name flags: mirror status ¹
Distinct Type, Create	CRDT	dbname: dbname objname: distinct type name login: distinct type, owner
Index, Create	CRIX	dbname: dbname tabid: tabid objname: idxname login: owner flags: frag flags ¹⁵ extra_2: dbspacelist

Event	Mnemonic	Variable Contents
Security Label, Create	CRLB	dbname: dbname objname: policy.label name
Security Label Component, Create	CRLC	dbname: dbname objname: component name
Access Method, Create	CRME	dbname: dbname tabid: access method ID objname: access method name login: access method owner
Operator Class, Create	CROC	dbname: dbname tabid: operator class ID objname: operator class name login: owner
Optical Cluster, Create	CROP	dbname: dbname tabid: tabid extra_1: cluster size login: owner extra_2: cluster name
Security Policy, Create	CRPL	dbname: dbname objname: policy name
Encryption/Decryption	CRPT	dbname: dbname objname: statement
Create Role	CRRL	dbname: dbname objname: rolename
Named Row Type, Create	CRRT	dbname: dbname tabid: xid of row type objname: named row type name login: named row type owner
Synonym, Create	CRSN	dbname: dbname tabid: syn. tabid extra_1: base tabid login: owner flags: syn. type ⁷ extra_2: synonym name

Event	Mnemonic	Variable Contents
SPL Routine, Create	CRSP	dbname: dbname tabid: proc. id login: owner extra_2: procedure name
Sequence, Create	CRSQ	dbname: dbname tabid: tabid objname: owner
Table, Create	CRTB	dbname: dbname tabid: tabid objname: owner login: tabname flags: frag flags ¹⁵ extra_2: dbspacelist
Trigger, Create	CRTR	dbname: dbname tabid: tabid row_num: trigger id ¹⁴ login: owner ¹⁴ extra_2: trigger name ¹⁴
View, Create	CRVW	dbname: dbname tabid: view tabid login: owner extra_2: view name
XADatasource, Create	CRXD	dbname: dbname objname: owner objname: name of XA data source
XADatasource Type, Create	CRXT	dbname: dbname objname: owner objname: name of XA data source type
Row, Delete	DLRW	dbname: dbname tabid: tabid extra_1: partno partno: frag_id row_num: row-num ¹⁴

Event	Mnemonic	Variable Contents
Chunk, Bring Off-line	DNCK	extra_1: chunk number flags: mirror status ¹
Disk Mirroring, Disable	DNDM	extra_1: dbspace number
Display Page	DPPG	tabid: page-num
Aggregate, Drop	DRAG	dbname: dbname objname: aggregate name login: owner
Audit Mask, Delete	DRAM	login: user id
Storage Space, Drop	DRBS	dbname: storage space name
Chunk, Drop	DRCK	dbname: dbspace name flags: mirror status ¹ extra_2: path
Cast, Drop	DRCT	dbname: dbname tabid: type ID of from type extra_1: xid of the from type partno: type of the to type row_num: xid of the to type
Database, Drop	DRDB	dbname: dbname
Domain, Drop	DRDM	
Dbspace, Drop	DRDS	dbname: dbspace name
Index, Drop	DRIX	dbname: dbname tabid: tabid login: owner extra_2: index name
Security Label, Drop	DRLB	dbname: dbname objname: policy.label name
Security Label Component, Drop	DRLC	dbname: dbname objname: component name
Transaction Log, Drop	DRLG	extra_1: log number
Access Method, Drop	DRME	dbname: dbname tabid: access method ID objname: access method name login: access method owner
Operator Class, Drop	DROC	dbname: dbname objname: operator class name login: owner

Event	Mnemonic	Variable Contents
Optical Cluster, Drop	DROP	dbname: dbname login: owner extra_2: cluster name
Security Policy, Drop	DRPL	dbname: dbname objname: policy name
Role, Drop	DRRL	dbname: dbname objname: rolename
Named Row Type, Drop	DRRT	dbname: dbname tabid: xid of dropped type
Synonym, Drop	DRSN	dbname: dbname tabid: syn. tabid login: owner extra_2: synname
SPL Routine, Drop	DRSP	dbname: dbname login: owner extra_2: spname
Sequence, Drop	DRSQ	dbname: dbname tabid: tabid
Table, Drop	DRTB	dbname: dbname tabid: tabid objname: tablename login: owner flags: drop-flags ²¹ extra_2: partnolist
Trigger, Drop	DRTR	dbname: dbname row_num: trigger id login: owner extra_2: trigname
Type, Drop	DRTY	dbname: dbname objname: type name login: type owner
View, Drop	DRVW	dbname: dbname tabid: view tabid flags: drop-flags ²¹

Event	Mnemonic	Variable Contents
XADatasource, Drop	DRXD	dbname: dbname objname: owner objname: name of XA data source
XADatasource Type, Drop	DRXT	dbname: dbname objname: owner objname: name of XA data source type
SPL Routine, Execute	EXSP	dbname: dbname tabid: proc. id
Grant Database Access	GRDB	dbname: dbname extra_1: privilege ⁵ extra_2: grantees ⁴
Grant Default Role	GRDR	
Grant Fragment Access	GRFR	dbname: dbname tabid: tabid objname: fragment extra_1: privilege ^{5, 14} login: grantor extra_2: grantees ^{4, 14}
Grant Security Label	GRLB	dbname: dbname objname: policy.label name login: grantee ⁴ extra_2: access type
Grant Role	GRRL	dbname: dbname objname: rolename login: grantor extra_2: grantees ⁴
Grant DBSECADM	GRSA	login: grantee
Grant SETSESSIONAUTH	GRSS	dbname: dbname login: grantee extra_2: surrogate user list

Event	Mnemonic	Variable Contents
Grant Table Access	GRTB	dbname: dbname tabid: tabid extra_1: privilege ^{5, 14} login: grantor extra_2: grantee ^{4, 14} , update columns, select columns ^{4, 14}
Grant Exemption	GRXM	dbname: dbname objname: policy name login: grantee extra_2: rule
Row, Insert	INRW	dbname: dbname tabid: tabid partno: frag_id row_num: rowid
Database Log Mode, Change	LGDB	dbname: dbname flags: log status ⁶
Table, Lock	LKTB	dbname: dbname tabid: tabid flags: lock mode ⁸
Audit Masks, List	LSAM	
Databases, List	LSDB	
Modify Transaction Logging	MDLG	flags: bufferedlog flags ²
onaudit	ONAU	extra_2: command line
onbar	ONBR	extra_2: command line
oncheck	ONCH	extra_2: command line
oninit	ONIN	extra_2: command line
onlog	ONLG	extra_2: command line
onload	ONLO	extra_2: command line
onmonitor	ONMN	extra_2: command line
onmode	ONMO	extra_2: command line
onparams	ONPA	extra_2: command line
onpload	ONPL	extra_2: command line
onspaces	ONSP	extra_2: command line
onstat	ONST	extra_2: command line
ontape	ONTP	extra_2: command line
onunload	ONUL	extra_2: command line

Event	Mnemonic	Variable Contents
Database, Open	OPDB	dbname: dbname flags: exclusive flag extra_2: dbpassword
Row, Read	RDRW	dbname: dbname tabid: tabid extra_1: partno partno: frag_id row_num: rowid ¹⁴
Optical Cluster, Release	RLOP	dbname: family name row_num: volume number
Transaction, Rollback	RLTX	
Chunks, Clear Mirrored	RMCK	extra_1: dbspace number
Rename Database	RNDB	dbname: dbname objname: new dbname login: user id
Rename dbspace	RNDS	dbname: dbspace name objname: new dbspace name
Rename index	RNIX	dbname: index name objname: new index name
Security Label, Rename	RNLB	dbname: dbname objname: old policy.label name extra_2: new label name
Security Label Component, Rename	RNLC	dbname: dbname objname: old component name extra_2: new component name
Security Policy, Rename	RNPL	dbname: dbname objname: old policy name extra_2: new policy name
Sequence, Rename	RNSQ	dbname: dbname tabid: tabid

Event	Mnemonic	Variable Contents
Table/ Column, Rename	RNTC	dbname: dbname tabid: tabid objname: new tab/ colname extra_1: colno(*) login: owner extra_2: tabname(**)
Optical Cluster, Reserve	RSOP	dbname: family name row_num: volume number
Revoke Database Access	RVDB	dbname: dbname extra_1: privilege ⁵ extra_2: revokees ⁴
Revoke Default Role	RVDR	
Revoke Fragment Access	RVFR	dbname: dbname tabid: tabid objname: fragment extra_1: privilege ^{5, 14} login: revoker extra_2: revokees ^{4, 14}
Revoke Security Label	RVLB	dbname: dbname objname: policy.label name login: grantee extra_2: access type
Revoke Role	RVRL	dbname: dbname objname: rolename login: revoker extra_2: revokees ⁴
Revoke DBSECADM	RVSA	login: grantee
Revoke SETSESSIONAUTH	RVSS	dbname: dbname login: grantee extra_2: surrogate user list

Event	Mnemonic	Variable Contents
Revoke Table Access	RTVB	dbname: dbname tabid: tabid extra_1: privilege ^{5, 14} login: revoker flags: drop-flags ²¹ extra_2: revokees ^{4, 14}
Revoke Exemption	RVXM	dbname: dbname objname: policy name login: grantee extra_2: rule
SPL Routine, System Command	SCSP	extra_2: command string
Collation, Set	STCO	dbname: dbname objname: locale name
Constraint, Set	STCN	dbname: dbname flags: constraint mode ¹¹ extra_2: constraint names
Set Debug File	STDF	dbname: dbname extra_2: file path
Set Database Password	STDP	dbname: dbname login: user id
Set Dataskip	STDS	flags: skip flags ¹⁶ extra_2: dbspacelist
Set Encryption Password	STEP	dbname: dbname
Set Environment	STEV	objname: environment variable and value
Set Explain	STEX	flags: explain flags ¹²
Isolation Level, Set	STIL	extra_1: isolation level ³
Set Lock Mode	STLM	flags: wait flags ¹³
Set No Collation	STNC	dbname: dbname objname: locale name
Set Object Mode	STOM	dbname: dbname tabid: tabid extra_1: command mode flag ²³ flags: object typeflag ^{2 4} extra_2: object names

Event	Mnemonic	Variable Contents
Stop Statement	STOP	dbname: dbname tabid: tabid
Set Pdqpriority	STPR	flags: prlevel ¹⁷
Set Role	STRL	dbname: dbname objname: rolename
Set Resident	STRS	
Start Statement	STRT	dbname: dbname tabid: tabid extra_1: Vio_tid flags: Dia_tid
Set Session Authorization	STSA	dbname: dbname login: new username
Set Statement Cache	STSC	objname: statement name
Start New Session	STSN	
Set Transaction Mode	STTX	extra_1: operation ²⁰ flags: mode flags ¹⁹ extra_2:
Save External Directives	SVXD	dbname: dbname objname: active/inactive/test objname: directive text
Truncate Table	TCTB	dbname: dbname tabid: tabid objname: tabname
Optical Cluster, Time	TMOP	flags: time flag ¹³
Table, Unlock	ULTB	dbname: dbname tabid: tabid
Audit Mask, Update	UPAM	login: user id
Chunk, Bring Online	UPCK	extra_1: chunk number flags: mirror status ¹
Disk Mirroring, Enable	UPDM	extra_1: dbspace number
Row, Update Current	UPRW	dbname: dbname tabid: tabid extra_1: old partno row_num: old rowid ¹⁴ flags: new rowid extra_2: new partno

Event	Mnemonic	Variable Contents
SPL Routine, Update Statistics	USSP	dbname: dbname tabid: proc. id
Table, Update Statistics	USTB	dbname: dbname tabid: tabid

Notes

1. Mirror Status:
 - 0 Not mirrored
 - 1 Mirrored
2. Buffered Log Flag:
 - 0 Buffering turned off
 - 1 Buffering turned on
3. Isolation Level:
 - 0 No transactions
 - 1 Dirty Read
 - 2 Committed Read
 - 3 Cursor Stability
 - 5 Repeatable Read
4. Grantees, Revokees, Select Columns, Update Columns:

These can be lists of comma-separated names. If longer than 166 characters, the audit processing described in "Audit Analysis with SQL" on page 9-3 truncates the lists to 166 characters.
5. Database Privileges:

Table-Level Privileges:

 - 1 Select
 - 2 Insert
 - 4 Delete
 - 8 Update
 - 16 Alter
 - 32 Index
 - 64 Reference
 - 4096 Execute Procedure (When Grant privilege is executed. tabid refers to the procedure ID.)

Database-Level Privileges:

 - 256 Connect
 - 512 DBA
 - 1024 Resource
6. Log Status:
 - 1 Logging on
 - 2 Buffered logging
 - 4 ANSI-compliant
7. Synonym Type:
 - 0 Private
 - 1 Public
8. Lock Mode:
 - 0 Exclusive
 - 1 Shared
9. Cluster Flag:
 - 0 Not cluster

- 1 Cluster
- 10. Chunk Flag:
 - 0 Check root reserve size
 - 1 Check entire chunk
 - <0 Check silently
- 11. Constraint Mode:
 - 0 Deferred
 - 1 Immediate
- 12. Explain Flag:
 - 0 Explain turned off
 - 1 Explain turned on
- 13. Wait Flag:
 - 1 Wait forever
 - 0 Do not wait
 - >0 Waiting period (in seconds)
- 14. If the user request is turned down because of the authorization, those fields are either 0 or blank, depending on the data type.
- 15. Fragmentation (*frag*) Flag:
 - 0 Not fragmented
 - 1 In dbspace
 - 2 Fragment by round robin
 - 4 Fragment by expression
 - 8 Fragment same as table
- 16. Skip Flag:
 - 0 DATASKIP for all the dbspaces is turned OFF
 - 1 DATASKIP for the following dbspaces is turned ON
 - 2 DATASKIP for all the dbspaces is turned ON
 - 3 DATASKIP is set to the default
- 17. Priority Level:
 - 1 PDQPRIORITY is set to the default
 - 0 PDQPRIORITY is turned OFF
 - 1 PDQPRIORITY is LOW
 - 100 PDQPRIORITY is HIGH
 - n any other positive integer less than 100 that the user entered in the SET PDQPRIORITY statement
- 18. Operation Type:
 - 4 Add a new fragment
 - 8 Modify fragmentation
 - 16 Drop a fragment
 - 32 Initialize fragmentation
 - 64 Attach table(s)
 - 128 Detach fragment
- 19. Mode Flag:
 - 0 Read/Write if operation is Set Access Mode; Dirty Read if operation is Set Isolation Level
 - 1 Read-only if operation is Set Access Mode; Committed Read if operation is Set Isolation Level
 - 2 Cursor Stability
 - 3 Repeatable Read
- 20. Operation:
 - 0 Set Access Mode
 - 1 Set Isolation Level
- 21. Dropflags:

- 0 Cascade
- 1 Restrict
- 22. Command Mode Flag:
 - 1 Disabled
 - 2 Filtering without error
 - 4 Filtering with error
 - 8 Enabled
- 23. Object Type Flag:
 - 1 Constraint
 - 2 Index
 - 3 Constraints and indexes
 - 4 Trigger
 - 5 Triggers and constraints
 - 6 Triggers and indexes
 - 7 All

Chapter 12. The ADTCFG File

ADTCFG refers to the audit configuration file. This chapter contains a list of the configuration parameters in the ADTCFG file and a short discussion of each configuration parameter.

Note: When any changes are made to the audit configuration, the server stores the changed configuration settings to the **adtcfg.servernumber** file. The server then reads the parameters in the **adtcfg.servernumber** file instead of the **adtcfg** file.

Each configuration parameter has one or more of the following attributes (depending on their relevance):

default value

Default value that appears in the **adtcfg.std** file

if not present

Value that is supplied if the parameter is missing from your ADTCFG file

units Units in which the parameter is expressed

separators

Separators that can be used when the parameter value has several parts.
Do not use white space within a parameter value

range of values

Valid values for this parameter

takes effect

Time at which a change to the value of the parameter actually affects the operation of the database server

utility Name of the command-line utility that you can use to change the value of the parameter

refer to Cross-reference to further discussion

ADTCFG File Conventions

The UNIX file **\$INFORMIXDIR/aaodir/adtcfg** or the Windows file **%INFORMIXDIR%\aaodir\adtcfg** is called the ADTCFG configuration file or simply the ADTCFG file. In the ADTCFG file, each parameter is on a separate line. The file can also contain blank lines and comment lines that start with a pound (#) symbol. The syntax of a parameter line is as follows:

```
PARAMETER_NAME           parameter_value           # comment
```

Parameters and their values in the ADTCFG file are case sensitive. The parameter names are always in uppercase letters. You must put white space (tabs, spaces, or both) between the parameter name, parameter value, and optional comment. Do not use any tabs or spaces within a parameter value.

For information about additional Dynamic Server configuration parameters, see the *IBM Informix Administrator's Reference*.

ADTERR

ADTERR specifies how the database server behaves when it encounters an error while it writes an audit record.

default value
0

range of values
0, 1, 3

0 = *continue* error mode

When it encounters an error as it writes an audit record, the database server writes a message of the failure into the message log. It continues to process the thread.

1 = *halt* error mode: suspend thread processing

When the database server encounters an error as it writes an audit record, the database server suspends processing of the thread until it successfully writes a record.

3 = *halt* error mode: shut down system

When the database server encounters an error as it writes an audit record, the database server shuts down.

takes effect

When **onaudit** is run to change the value or after shared memory is initialized. ADTMODE must be nonzero (auditing is on).

utility **onaudit** (onaudit -e *errormode*)

ADTMODE

ADTMODE controls the level of auditing.

default value
0

range of values
0, 1, 3, 5, 7

0 = auditing disabled

1 = auditing on; starts auditing for all sessions

3 = auditing on; audits DBSSO actions

5 = auditing on; audits database server administrator actions

7 = auditing on; audits DBSSO and database server administrator actions

takes effect

When **onaudit** is run to change the value or after the server is started

utility **onaudit** (onaudit -l *auditmode*)

ADTPATH

ADTPATH specifies the directory in which the database server saves audit files. Make sure that the directory that you specify has appropriate access privileges to prevent unauthorized use of audit records.

To change the ADTPATH value with **onaudit**, database server-managed auditing must be on.

The ADTPATH values are:

default value

/usr/informix/aaodir (on UNIX), **%informixdir%\aaodir** (on Windows)

range of values

Any valid directory path

takes effect

When **onaudit** is run to change the value or after shared memory is initialized

utility **onaudit** (onaudit -p auditdir)

ADTSIZE

ADTSIZE specifies the maximum size of an audit file. When a file reaches the maximum size, the database server saves the audit file and creates a new one. This parameter applies only to database server-managed auditing.

default value

10, 240

units Bytes

range of values

Between 10,240 bytes and approximately 2 gigabytes (the maximum value of a 32-bit integer)

takes effect

When **onaudit** is run to change the value or after shared memory is initialized

utility **onaudit** (onaudit -s maxsize)

Part 3. Appendixes

Appendix. Accessibility

IBM strives to provide products with usable access for everyone, regardless of age or ability.

Accessibility features for IBM Informix Dynamic Server

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use information technology products successfully.

Accessibility Features

The following list includes the major accessibility features in IBM Informix Dynamic Server. These features support:

- Keyboard-only operation.
- Interfaces that are commonly used by screen readers.
- The attachment of alternative input and output devices.

Tip: The IBM Informix Dynamic Server Information Center and its related publications are accessibility-enabled for the IBM Home Page Reader. You can operate all features using the keyboard instead of the mouse.

Keyboard Navigation

This product uses standard Microsoft® Windows navigation keys.

Related Accessibility Information

IBM is committed to making our documentation accessible to persons with disabilities. Our publications are available in HTML format so that they can be accessed with assistive technology such as screen reader software. The syntax diagrams in our publications are available in dotted decimal format. For more information about the dotted decimal format, go to “Dotted Decimal Syntax Diagrams.”

You can view the publications for IBM Informix Dynamic Server in Adobe Portable Document Format (PDF) using the Adobe Acrobat Reader.

IBM and Accessibility

See the *IBM Accessibility Center* at <http://www.ibm.com/able> for more information about the commitment that IBM has to accessibility.

Dotted Decimal Syntax Diagrams

The syntax diagrams in our publications are available in dotted decimal format, which is an accessible format that is available only if you are using a screen reader.

In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), the elements can appear on the same line, because they can be considered as a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that your screen reader is set to read punctuation. All syntax elements that have the same dotted decimal number (for example, all syntax elements that have the number 3.1) are mutually exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, the word or symbol is preceded by the backslash (\) character. The * symbol can be used next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is read as 3 * FILE. Format 3* FILE indicates that syntax element FILE repeats. Format 3* * FILE indicates that syntax element * FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol that provides information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, this identifies a reference that is defined elsewhere. The string following the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you should refer to a separate syntax fragment OP1.

The following words and symbols are used next to the dotted decimal numbers:

- ? Specifies an optional syntax element. A dotted decimal number followed by the ? symbol indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element (for example, 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that syntax elements NOTIFY and UPDATE are optional; that is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.
- ! Specifies a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicates that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the same dotted decimal number can specify a ! symbol. For example, if you hear the lines

2? FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In this example, if you include the FILE keyword but do not specify an option, default option KEEP is applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1! (KEEP), and 2.1.1 (DELETE), the default option KEEP only applies to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.

- * Specifies a syntax element that can be repeated zero or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data-area, you know that you can include more than one data area or you can include none. If you hear the lines 3*, 3 HOST, and 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

Notes:

1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you could write HOST STATE, but you could not write HOST HOST.
3. The * symbol is equivalent to a loop-back line in a railroad syntax diagram.

- + Specifies a syntax element that must be included one or more times. A dotted decimal number followed by the + symbol indicates that this syntax element must be included one or more times. For example, if you hear the line 6.1+ data-area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. As for the * symbol, you can only repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loop-back line in a railroad syntax diagram.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

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

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

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

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

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

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.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>.

Adobe, Acrobat, Portable Document Format (PDF), PostScript, and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

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

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

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

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

Index

Special characters

\$INFORMIXDIR
permissions 1-1

A

aaodir directory 7-10, 8-2
access control
 authentication 4-12
Access privileges, Windows 7-15, 8-1, 8-2
Access to audit trail, controlling 7-12, 7-13, 9-3
accessibility A-1
 keyboard A-1
 shortcut keys A-1
Accessibility A-1
 dotted decimal format of syntax diagrams A-1
 syntax diagrams, reading in a screen reader A-1
Adding audit masks 8-8
Administrative roles
 audit analysis officer 8-2
 database administrator 8-2
 database server administrator 8-1
 database system security officer 8-1
 listed 7-4
 operating-system administrator 8-2
Administrator
 audit analysis officer 8-2
 database 8-2
 database server 8-1
 database system security officer 8-1
 operating system 8-2
ADTCFG file
 aaodir directory 8-2
 adtcfg.std file 7-9
 ADTMODE configuration parameter 7-9
 audit configuration
 UNIX 7-11, 10-7
 Windows 7-11, 10-7
 configuration parameters 8-12
 conventions used 12-1
 description of 12-1
 UNIX audit file size 7-9
 white space 12-1
ADTERR configuration parameter 8-12, 12-2
adtmasks.std file 8-10
ADTMODE configuration parameter 7-9, 8-12, 12-2
ADTPATH configuration parameter 7-9, 8-12, 12-3
ADTSIZE configuration parameter 8-12, 12-3
Advanced Encryption Standard 2-1, 2-2
AES. 2-1
Aggregation 7-15
ALTER SECURITY LABEL COMPONENT statement 6-9
ALTER TABLE statement 6-12, 6-14
Application Event log, Windows 7-10, 7-11
archchecker utility 6-19
ARRAY
 see security label component 6-5
Audit
 features 7-1
 performance 7-6

Audit (*continued*)
 process for 7-3
 reasons for 7-1
 record format 9-1
 turning on auditing 8-8
Audit administrator
 audit analysis officer 7-4, 8-1
 audit configuration 7-3, 7-11
 audit instructions 7-6
 audit masks 7-1, 7-5
 audit-trail analysis 7-1
 auditing on or off 7-5, 7-9
 database system security officer 7-4, 8-1
 roles 7-4, 8-1
 security risk 7-5
Audit analysis
 creating a data file 9-4
 importance of 7-14
 loading audit data into a database 9-6
 overview 7-14
 preparing for 7-14
 records indicating event failure 7-15
 records indicating event success 7-15
 strategies for 7-15
 with SQL
 creating a command file 9-5
 creating a database 9-4
 creating a table 9-4
 description 9-3
 performing 9-3
 preparing for 9-4
 without database 9-2
 without SQL 9-2
Audit analysis officer (AAO)
 audit administrator 7-4, 8-1
 role description 8-2
 security threats 7-17
 UNIX 8-2
 Windows registry settings 8-2
Audit configuration
 ADTCFG file 7-11
 changing from a command line 10-7
 showing
 from a command line 8-12, 10-6
 with onshowaudit 10-6
 UNIX onaudit output 10-6
 Windows
 ADTCFG file 10-7
 onaudit output 10-6
Audit data
 controlling access to 9-3
 creating a table for 9-4
 loading into database 9-6
 privileges to protect 9-3
Audit error mode
 and onaudit 10-7
 changing 8-14
 in ADTCFG file 12-2
 setting 8-6
Audit events
 alphabetical listing of codes 11-1

- Audit events *(continued)*
 - displaying 8-10
 - fields shown 11-5
 - listed 11-5
- Audit files, UNIX
 - controlling access to 7-12
 - directory
 - specifying with ADTPATH 12-3
 - specifying with onaudit 10-7
 - error modes when writing to 7-11
 - extracting information with onshowaudit 10-9
 - location of 7-9
 - naming 7-9
 - properties of 7-9
 - specifying maximum size
 - with ADTSIZE 12-3
 - with onaudit 10-7
 - starting
 - new file 7-9
 - with onaudit 10-6
 - storage
 - in database server 10-6
 - in operating system 10-6
 - write errors 10-8
- Audit instructions
 - resource and performance implications 7-6
 - who sets 7-6
- Audit level, setting 8-7
- Audit masks
 - _default mask 7-5
 - _exclude mask 7-5
 - _require mask 7-5
 - adding 8-8
 - base mask 8-9
 - compulsory masks 7-5
 - conflict in audit instructions 7-5, 11-1
 - creating a template 8-9
 - creating a user mask from a template mask 8-9
 - creating from a command line 10-3
 - deleting 8-11, 10-5
 - displaying 8-10
 - how to use 7-8
 - individual user mask 7-5
 - maintaining 8-8
 - modifying
 - command syntax for 10-2
 - from a command line 8-11
 - from an input file 8-10
 - instructions 8-11
 - restricted names 7-6
 - setting up default and compulsory 8-6
 - showing 10-2
 - specification with onaudit 10-3
 - templates 7-6
 - types, listed 7-4
 - user mask 7-5
- Audit records
 - controlling access to 7-12
 - interpreting extracted information 9-6
- Audit trail
 - administration 8-8, 8-10
 - controlling access to 7-12, 7-13
 - extracting information with onshowaudit 10-9
 - operating-system, UNIX 7-3
 - reviewing 7-4
 - starting a new UNIX file 8-13
 - starting auditing from a command line 10-6

- Audit trail *(continued)*
 - UNIX file permissions 7-12, 7-13
 - UNIX files 7-12
 - Windows access privileges 7-13
 - Windows Application Event log 7-12
- Audit trail, controlling access to 7-13
- Auditing
 - ADTCFG file
 - UNIX 7-11, 10-7
 - Windows 7-11, 10-7
 - creating user masks from template masks 8-9
 - displaying fragmentation information 7-7
 - error mode levels 10-7
 - granularity 7-7
 - setting the level 8-7
 - setting up 8-5
 - specifying UNIX directory
 - with ADTPATH 12-3
 - with onaudit 10-7
 - turning off 7-9, 8-14, 10-8
 - turning on 7-9, 8-8, 10-8
- authentication
 - modules 4-12
 - single sign-on 4-12, 4-13
 - and Kerberos layer 4-14
 - and keytab file 4-17
 - types 4-12
- Authentication
 - modules 4-1
 - single sign-on 4-1

B

- backup and restore 6-19
- Base mask, defined 8-9
- Blowfish 2-1
- Browsing 7-15

C

- Changing the audit error mode 8-14
- Changing the system audit configuration 10-7
- Cipher
 - AES 2-2, 2-5
 - Blowfish 2-2
 - defined 2-1
 - DES 2-2, 2-5
 - in encryption 2-2
 - supported types 2-2, 2-5
- cipher encryption tag 2-6
- ciphers
 - switch frequency 2-5
- client software
 - single sign-on 4-14
- clients
 - single sign-on 4-18, 4-19
- column level data protection 6-12, 6-14
- column-level data protection 6-1
- Command files
 - creating for dbload 9-5
 - use with dbload 9-5
- Communication Support Module 2-1, 4-9
 - concsn.cfg entry 2-8, 4-12
 - configuration file 2-2, 4-10
 - for single sign-on (GSSCSM) 4-12

- Compulsory audit masks
 - setting up 8-6
 - when applied 7-5
- concsn.cfg file 2-1, 2-2, 4-9, 4-10
 - building SMI tables 4-11
 - entry for network data encryption 2-8
 - entry for password encryption 4-12
 - entry for single sign-on 4-16
 - location 2-2, 4-10
 - single sign-on
 - configuration 4-16
- confidentiality service 4-12, 4-16
- configuration parameters
 - PLCY_HASHSIZE 6-16
 - PLCY_POOLSIZE 6-16
 - USRC_HASHSIZE 6-16
 - USRC_POOLSIZE 6-16
- Configuration parameters
 - ADTERR 8-12, 12-2
 - ADTMODE 8-12, 12-2
 - ADTPATH 8-12, 12-3
 - ADTSIZE 8-12, 12-3
 - DB_LIBRARY_PATH 1-5
 - DBCREATE_PERMISSION 5-2
 - described 12-1
 - IFX_EXTEND_ROLE 5-3
 - listed 8-12
 - LISTEN_TIMEOUT 4-22
 - MAX_INCOMPLETE_CONNECTIONS 4-22
 - SECURITY_LOCALCONNECTION 4-21
 - UNSECURE_ONSTAT 5-3
- Configuration, audit
 - displaying 8-12
 - maintaining 8-12
 - overview 7-8
 - tasks listed 7-8
- Configuring role separation 8-4
- Continue error modes 7-11
- Controlling access to audit trail 7-12, 7-13, 9-3
- Coserver 7-9, 12-3
- CREATE ROLE statement 5-1
- CREATE SECURITY LABEL COMPONENT statement 6-8
- CREATE SECURITY LABEL statement 6-10
- CREATE SECURITY POLICY statement 6-9
- CREATE TABLE statement 6-12, 6-14
- Creating a data file 9-4
- Creating a database for audit data 9-4
- Creating a table for audit data 9-4
- Creating a user mask from a template mask 8-9
- Creating an audit mask from a command line 10-3
- credentials 4-13
 - single sign-on 4-17, 4-18

D

- Data
 - audit, loading into database 9-6
 - creating a file for dbload 9-4
 - extracting with onshowaudit 9-2
 - transmission encryption 2-1
- data classifications 6-4
- data definition language (DDL)
 - with label-based access control (LBAC) 6-19
- Data encryption 2-1
- Data Encryption Standard 2-1
- data labels 6-1
- data loading and unloading 6-19

- Data replication
 - security option 4-20
- Database administrator (DBA) 8-2
- database security administrator
 - see DBSECADM 6-1
- database server administrator (DBSA) 6-4
- Database server administrator (DBSA)
 - administrative role 8-1
 - in label-based access control 6-1, 6-3
 - role description 8-1
 - security threats 7-17
- Database servers
 - audit log 10-9
 - auditing 7-9, 10-9
 - groups 1-3
 - managing auditing
 - with ADTMODE 12-2
 - with onaudit 10-8
 - monitoring events and users 8-7
 - naming convention 7-9
 - Password Communication Support Module 4-11
 - quiescent mode 7-9
- Database system security officer (DBSSO)
 - audit administrator 7-4, 8-1
 - role description 8-1
 - security threats 7-17
 - UNIX 8-1
 - Windows registry settings 8-1
- Databases
 - creating for Dynamic Server audit records 9-4
 - sysmaster 8-12
- DataBlade
 - restricting access for registering UDRs 5-3
- DB_LIBRARY_PATH configuration parameter 1-5
- DBCREATE_PERMISSION configuration parameter 5-2
- dbexport utility 6-19
- dbimport utility 6-19
- dbload utility 6-19
 - creating a command file for 9-5
 - creating a data file for 9-4
 - creating a database for 9-4
 - creating a table for 9-4
 - creating onshowaudit output files for 10-9
 - loading audit data into a database 9-6
 - redirecting onshowaudit output 10-9
- DBMS security threats 7-17
- dbschema utility 6-19
- DBSECADM 6-1, 6-3, 6-4, 6-9, 6-10, 6-11, 6-15, 6-17
- dbserveralias
 - for single sign-on 4-15
- dbservername
 - for single sign-on 4-15
- dbssodir directory 8-1
- Default audit mask 7-5
 - setting up 8-6
 - when applied 7-5
- Defaults
 - role
 - creating 5-2
 - granting privileges 5-2
 - using 5-2
- Deleting audit masks 8-11, 10-5
- Denial-of-service flood attacks 4-21
- DES. 2-1
- DES3. 2-1
- digital certificates 2-12

- Directories
 - aaodir 8-2
 - specifying for UNIX audit files
 - with ADTPATH 12-3
 - with onaudit 8-6, 10-7
- Disabilities, visual
 - reading syntax diagrams A-1
- disability A-1
- discretionary access control 5-1
- discretionary access control (DAC) 6-1
- Discretionary Access Control (DAC) 7-17
- Displaying
 - audit configuration 8-12, 10-6
 - audit masks 8-10, 10-2
- Distributed database configuration threats 7-19
- distributed queries 6-22
- Distributed transactions
 - configuring SSL connections for server groups 2-17
- Dotted decimal format of syntax diagrams A-1
- DRDA communications
 - with an SSL connection 2-12, 2-15
- DROP SECURITY LABEL COMPONENT 6-17
- DROP SECURITY LABEL statement 6-17
- DROP SECURITY POLICY statement 6-17
- dropping
 - security label components 6-17
 - security labels 6-17
 - security policies 6-17
- Dynamic Server
 - audit record format for 9-1
 - extracting and loading audit records for 9-4

E

- elements
 - for label-based access control 6-2
- Enable Role Separation check box 8-4
- ENCCSM
 - Communication Support Modules, encryption 2-1
- ENCCSM_CIPHERS encryption parameter 2-9
- ENCCSM_MAC encryption parameter 2-9, 2-10
- ENCCSM_MACFILES encryption parameter 2-9, 2-10
- ENCCSM_SWITCH encryption parameter 2-9, 2-10
- ENCRYPT function 3-1
- Encrypting
 - column data 3-3
- encryption
 - in single sign-on 4-12
 - password 4-9
 - switch frequency 2-5
- Encryption
 - column storage consideration 3-1
 - column-level 3-1, 3-3
 - credentials in SSO 4-16
 - data transmissions 2-1, 4-9
 - defined 2-1
 - example
 - encrypting a column 3-4
 - querying encrypted data 3-4
 - size of an encrypted column 3-3
 - in single sign-on 4-13
 - modes 2-1
 - of data in a column 3-1, 3-3
 - of network data transmissions 2-1
 - of passwords 2-1, 4-9
 - Users
 - user IDs 4-13

- encryption parameter
 - ENCCSM_CIPHERS 2-9
- encryption parameters
 - example 2-10
 - overview 2-9
- encryption tags 2-5
 - cipher tag 2-6
 - example 2-8
 - mac tag 2-7
 - switch tag 2-8
- Enforcing role separation 8-4
- Enterprise Replication 4-20, 6-19
 - configuring SSL connections 2-17
- Environment variables
 - INF_ROLE_SEP 8-4
 - INFORMIXCONCSMCFG 2-2, 4-10
 - NODEFDAC 9-3
- Error messages
 - for server utilities security check 1-2
- Error messages log, size of 7-11
- Error mode
 - and ADTERR 12-2
 - and onaudit 10-7
 - changing 8-14
 - continue 7-11
 - halt 7-11
 - implications of 8-6
 - setting 8-6
 - when writing to an audit file 7-11
- Event codes, alphabetical listing 11-1
- Event failure 7-15
- Event success 7-15
- Events
 - defined 7-1
 - fields shown 11-5
 - level of auditing for specified 7-7
 - mnemonics listed 11-1
- Exclude audit mask 7-5
- exemptions 6-15
- External modules
 - security for loading 1-5
- External routines
 - security for 5-3

F

- Fields for audit events 11-5
- FILE statement 9-5
- Files
 - ADTCFG 7-9
 - data, creating for dbload 9-4
 - input
 - for modifying masks 8-10
 - for onaudit 10-4
 - UNIX audit
 - controlling access to 7-12
 - location of 7-9
 - naming 7-9
 - starting new file 7-9
 - starting with onaudit 10-6
 - storage in database server 10-6
 - storage in operating system 10-6
- Format
 - for audit records 9-1
 - for dbload data file 9-4
 - for onaudit input file 10-4
- fragmentation 6-22

Fragmentation, information in audit events 7-7
functions
 security label support 6-12

G

Generic Security Services
 API 4-12
 communications support module 4-12
Global Security Kit 2-14
GRANT DBSECADM statement 6-3
GRANT EXEMPTION statement 6-15
GRANT SECURITY LABEL statement 6-11
GRANT statement 5-1
 when granting privileges to DataBlade users 5-3
Group
 database server 1-3
GSKit 2-14
Guidelines for assigning roles 8-3

H

Halt modes 7-11, 10-8
hierarchical tables 6-1
High-Availability Data Replication
 configuring SSL connections 2-17
high-availability solutions 6-22
High-Performance Loader 6-19

I

IBM Data Server Security Blueprint vii
IBM Global Security Kit 2-15
IDSLBACREADARRAY rule 6-5
IDSLBACREADSET rule 6-6
IDSLBACREADTREE rule 6-7
IDSLBACWRITEARRAY rule 6-5
IDSLBACWRITESET rule 6-6
IDSLBACWRITETREE rule 6-7
IDSSECURITYLABEL data type 6-12
IFX_EXTEND_ROLE configuration parameter 5-3
iKeyman utility 2-14, 2-15
INF_ROLE_SEP environment variable 8-4
informix user account 7-11, 8-2, 10-9
INFORMIXCONCSMCFG environment variable 2-2, 4-10
INFORMIXDIR directory
 respective permissions 1-4
Input file for onaudit utility 10-4
Insider attack 7-15
integrity service 4-12, 4-16
ipload utility 6-19
IXUSERS seccfg setting 8-4

J

Java Cryptography Extension 2-14

K

Kerberos 4-13, 4-14
 authentication
 single sign-on 4-18
 testing setup for SSO 4-18
Key Distribution Center 4-13
keystores 2-12

keytab file 4-14, 4-17

L

label-based access control
 and other IDS features 6-22
 configuration parameters 6-16
 configuring 6-1
 DBSECADM role 6-1
 maintaining 6-16
 overview 6-1
 planning 6-4
 read and write access 6-2
 row and column protection on one table 6-12
 security precautions 6-19
LDAP Authentication Support
 configuration file 4-4
 installing 4-4
LDAP Authentication Support on Windows 4-4
LDAP module
 application development 4-6
 authentication 4-5
 client APIs 4-8
 compatibility issues 4-9
 configuring 4-4
 configuring Dynamic Server 4-5
 distributed transactions 4-8
 implicit connections 4-5
LDAP server 4-4
Level of auditing, determining 8-7
LISTEN_TIMEOUT configuration parameter 4-21, 4-22
Listener threads 4-21
Loading onshowaudit data into a database table 9-6

M

mac encryption tag 2-7
MAC key files
 generating new 2-4
 generation levels 2-4
 overview 2-4
Malicious software security threats 7-18
Malware
 see Malicious software security threats 7-18
mandatory access control (MAC) 6-1
Mask
 _default 7-5
 _exclude 7-5
 _require 7-5
 creating
 template 8-9
 user mask from a template mask 8-9
 user mask without a template mask 8-9
 with onaudit 10-3
 deleting 8-11, 10-5
 displaying 8-10
 how to use 7-8
 modifying
 from an input file 8-10
 from the command line 8-11
 with onaudit 10-2
 onaudit input-file format 10-4
 setting up compulsory 8-6
 setting up default 8-6
 showing with onaudit 10-2
 specification with onaudit 10-3

- Mask (*continued*)
 - template 7-6
 - types, listed 7-4
 - user 7-5
- MAX_INCOMPLETE_CONNECTIONS configuration
 - parameter 4-21, 4-22
- Message log 10-8
- Message Server service 7-11
- Mnemonics, alphabetical listing for events 11-1
- Modifying audit masks 8-11, 10-2
- mutual authentication 4-12

N

- Named pipes interprocess communications 7-11
- Network
 - data encryption 2-8
- Network data transmissions
 - encryption of 2-1
- NODEFDAC environment variable 9-3

O

- Obsolete user security threats 7-19
- onaudit utility
 - ADTERR parameter 12-2
 - ADTMODE parameter 12-2
 - ADTPATH parameter 12-3
 - ADTSIZE parameter 12-3
 - audit events, adding to audit masks 8-6
 - audit file location 7-9
 - audit masks
 - creating 10-3
 - deleting 8-11, 10-5
 - described 7-6
 - displaying 8-10
 - showing from command line 10-2
 - auditing mode levels 10-7
 - auditing on or off 7-9
 - changing the audit error mode 8-14
 - changing the system audit configuration 10-7
 - description of 10-1
 - displaying the audit configuration 8-12
 - error modes 7-11
 - error-mode levels 10-7
 - fragmentation information 7-7
 - HDR limitations 7-3
 - input-file format 10-4
 - level of auditing for certain events 7-7
 - masks, modifying 8-10, 10-2
 - overview 10-1
 - setting the error mode 8-6
 - showing the audit configuration 10-6
 - specifying a directory for UNIX audit files 8-6
 - starting a new UNIX audit file 10-6
 - storage of audit records 10-6
 - template mask
 - creating 8-9
 - creating a user mask from 8-9
 - creating a user mask without 8-9
 - turning off auditing 8-14
 - turning on auditing 8-8
 - UNIX operations 10-1
 - used by AAO 8-2
 - used by DBSSO 8-1
 - who can run 10-1

- onaudit utility (*continued*)
 - Windows operations 10-1
- onbar utility 6-19
- oncheck utility 6-19
- ONCONFIG file 7-9, 7-11
- Online mode 7-9
- onload utility 6-19
- onlog utility 6-19
- onpload utility 6-19
- onshowaudit utility
 - audit trail access 7-12
 - data extraction from audit trail 7-4, 7-14
 - description of 10-1
 - extracting data for audit analysis 9-2
 - listing of audit events for analysis 11-5
 - output accessible by AAO 7-17
 - role separation 7-12
 - syntax 10-9
 - used by AAO 8-2
 - using dbload with 9-4
 - who can run 10-9
- ontape utility 6-19
- onunload utility 6-19
- Operating system
 - coordinating auditing between AAO and OSA 8-2
 - protected subsystem for audit trail 7-14
- Operating-system administrator (OSA)
 - administrative role 8-2
 - role defined 8-2
 - security threats 7-17
- Operating-system audit trail, UNIX 7-3

P

- Parameters, configuration
 - ADTERR 8-12, 12-2
 - ADTMODE 8-12, 12-2
 - ADTPATH 8-12, 12-3
 - ADTSIZE 8-12, 12-3
 - described 12-1
 - listed 8-12
- Password Communication Support Module 4-12
- Password encryption
 - CSM configuration file 2-2, 4-10, 4-11
 - database server initialization 4-10
- Path, specifying for auditing
 - with ADTPATH 12-3
 - with onaudit 10-7
- Performance implications of auditing 7-6
- Performing SQL audit analysis 9-3
- Permissions
 - for creating databases 5-2
- Permissions, UNIX 7-15, 8-1, 8-2
- Pluggable Authentication Module 4-2
 - application development 4-6
 - authentication mode 4-3
 - client APIs 4-8
 - compatibility issues 4-9
 - configuring the server 4-3
 - defined 4-2
 - distributed transactions 4-8
 - implicit connections 4-5
 - required stack size 4-3
 - service name 4-2
 - supported platforms 4-2
- Preparing for audit analysis 7-14, 9-4
- Primary security threats 7-17

- principals 4-13, 4-14
 - validating 4-18
- privacy policy
 - label-based access control 6-4
- Privileged activity security threats 7-17
- Privileged environment, security threat from untrusted software 7-19
- Privileged users 8-2
- Privileges to protect audit data 9-3
- protected data 6-10, 6-12

Q

- Queries by browsers 7-15
- Quiescent mode 7-9

R

- Raw audit records 7-14
- read access
 - label-based access control 6-2
- referential integrity scans 6-22
- Registry settings, Windows
 - for AAO 8-2
 - for DBSSO 8-1
 - for role separation 8-4
- Remote access to data, security threat 7-19
- RENAME SECURITY LABEL COMPONENT statement 6-18
- RENAME SECURITY LABEL statement 6-18
- RENAME SECURITY POLICY statement 6-18
- renaming
 - security objects 6-18
- Require audit mask 7-5
- Resource implications of auditing 7-6
- Responding to security problems 7-16
- REVOKE DBSECADM statement 6-4
- REVOKE DEFAULT ROLE statement 5-2
- REVOKE EXEMPTION statement 6-15
- REVOKE SECURITY LABEL statement 6-11
- REVOKE statement
 - when granting privileges to DataBlade users 5-3
- Role
 - creating 5-2
 - default 5-2
 - defined 5-2
 - GRANT DEFAULT ROLE statement 5-2
 - overview 5-1
 - separation 5-1
- Role separation and onshowaudit 7-12
- Role Separation dialog box 8-1, 8-4
- Roles
 - administrative, listed 7-4
 - assigning 8-3
 - audit analysis officer 8-2
 - configuring and enforcing 8-4
 - database administrator 8-2
 - database server administrator 8-1
 - database system security officer 8-1
 - no separation, security configuration for 7-12
 - operating-system administrator 8-2
 - separation 7-12, 8-3, 8-4
- root user account 8-2, 10-9
- row and column protection on one table 6-12
- row level data protection 6-12, 6-14
- row-level data protection 6-1

S

- Screen reader
 - reading syntax diagrams A-1
- seccfg file 8-4
- SECLABEL functions 6-9
 - see security label support functions 6-12
- Secured Socket Layer protocol
 - configuring client connections 2-16
 - configuring Enterprise Replication Nodes 2-17
 - configuring High-Availability Data Replication connections 2-17
 - configuring IDS for connections 2-15
 - configuring server-to-server connections 2-17
 - overview 2-12
- security
 - overview vii
- Security
 - disabling server utilities check 1-2
 - encryption options 2-1
 - for DataBlade user-defined routines 5-3
 - for external routines 5-3
 - for loading external modules 1-5
 - INFORMIXDIR directory permissions 1-4
 - Pluggable Authentication Module 4-2
 - preventing denial-of-service flood attacks 4-21
 - resetting directory permissions 1-2
 - server utilities check before starting on UNIX 1-1
 - through LDAP Authentication Support 4-4
 - through roles 5-2
 - using column-level encryption 3-1
 - using Communication Support Modules 2-1, 4-9
- Security configuration for audit files 7-12
- Security Event log, Windows 7-10
- security label component
 - ARRAY 6-5
 - SET 6-6
- security label components 6-1, 6-4
 - altering 6-9
 - creating 6-8
 - in exemptions 6-15
 - TREE 6-7
- security label support functions 6-12
- security labels 6-1, 6-10
 - creating 6-10
 - functioning 6-2
 - granting 6-11
 - revoking 6-11
- security option 4-20
- security policies 6-1, 6-9
 - creating 6-9
- Security threats
 - aggregation 7-15
 - audit analysis officer 7-17
 - browsing 7-15
 - database server administrator 7-17
 - database system security officer 7-17
 - DBMS 7-17
 - distributed databases configuration 7-19
 - granting remote access to data 7-19
 - insider attack 7-15
 - malicious software 7-18
 - obsolete user 7-19
 - operating-system administrator 7-17
 - primary 7-17
 - privileged activity 7-17
 - responses to 7-16
 - setting the auditing level 8-7

- Security threats (*continued*)
 - shared-memory connection 7-18
 - untrusted software in privileged environment 7-19
- SECURITY_LOCALCONNECTION configuration
 - parameter 4-21
- SERVERNUM configuration parameter 7-11
- Session, effects of errors 7-11
- SET
 - see security label component 6-6
- SET ENCRYPTION PASSWORD statement 3-1
- SET ROLE DEFAULT statement 5-2
- SET SESSION AUTHORIZATION statement 6-19
- SET statement 5-1
- setenv utility 8-4
- Shared-memory connection 7-18
- shortcut keys
 - keyboard A-1
- Showing
 - audit configuration 10-6
 - audit masks 10-2
- Simple Password Communication Support Module
 - CSM configuration file 4-12
- single sign-on 4-12, 4-13
 - and Kerberos protocol 4-13
 - clients with 4-18, 4-19
 - configuring the database server 4-15
- Size, specifying maximum for UNIX audit files
 - with ADTSIZE 12-3
 - with onaudit 10-7
- SMI sysadinfo table 8-12
- SMI tables
 - conscsm.cfg 4-11
- Specification, audit mask 10-3
- SPWDSCSM 2-1, 4-9
- SQL statements
 - CREATE DATABASE 9-4
 - CREATE ROLE 5-1
 - GRANT 5-1, 9-3
 - GRANT DEFAULT ROLE 5-2
 - REVOKE 9-3
 - REVOKE DEFAULT ROLE 5-2
 - SET ROLE 5-1
 - SET ROLE DEFAULT 5-2
- sqlhosts
 - for single sign-on 4-15
- sqlhosts directory 7-18
- SSL protocol
 - configuring client connections 2-16
 - configuring IDS connections 2-15
 - overview 2-12
- SSL_KEYSTORE_LABEL configuration parameter 2-12, 2-15
- Storage of UNIX audit files
 - in database server 10-6
 - in operating system 10-6
 - new file (-n) option 10-6
- Strategies for audit analysis 7-15
- Superuser (root) 7-12
- switch encryption tag 2-8
- switch frequency
 - encryption 2-5
- synonyms 6-22
- Syntax
 - onshowaudit utility 10-9
- Syntax diagrams
 - reading in a screen reader A-1
- sysadinfo table 8-12
- sysaudit table 7-6

- sysmaster database 7-6
- sysmaster database, sysadinfo table 8-12
- sysroleauth table 5-3
- sysuser database 4-8

T

- Table
 - creating for audit data 9-4
 - sysadinfo 8-12
- Template audit masks 7-6
 - base mask 8-9
 - creating from user masks 8-9
 - creating with onaudit 8-9
 - description 7-6
 - naming rules 7-6
- temporary (TEMP) tables 6-1, 6-19
- Threads, suspended 7-11
- tickets
 - in Kerberos authentication 4-13
- Triple Data Encryption Standard 2-1
- typed tables 6-1

U

- UNIX
 - ADTCFG file 7-11, 10-7
 - audit configuration 7-11, 10-7
 - audit files
 - data extraction 10-9
 - directory 8-6, 10-7, 12-3
 - error modes when writing to 10-8
 - location 7-9
 - naming 7-9
 - new 7-9, 8-13, 10-6
 - properties 7-9
 - size 7-9, 10-7, 12-3
 - storage in database server 10-6
 - storage in operating system 10-6
 - audit-trail files 7-12
 - machine notes file 7-14
 - onaudit output 10-6
 - operating-system audit trail 7-3
 - operations with onaudit 10-1
 - permissions 8-1, 8-2
 - workstations 7-19
- Unscrupulous user 7-5, 7-15, 8-1, 8-3
- UNSECURE_ONSTAT configuration parameter 5-3
- Untrusted software 7-19
- User informix
 - audit files owner 7-12
 - retrieving audit configuration information 8-12
 - running onaudit 10-1
 - running onshowaudit 7-12, 10-9
- user labels 6-1
- User mask
 - and _default mask 7-5
 - creating from a template mask 8-9
 - creating without a template mask 8-9
- user operations
 - in label-based access control 6-2
- user roles
 - default roles 5-2
 - overview 5-1
 - role separation 5-1
- user-defined routines 6-19

- User-defined routines
 - external 5-3
 - registering 5-3
- Users
 - accounts with same name 7-19
 - auditing 7-5, 10-9
 - privileged 8-2
 - system 8-2
 - user IDs 4-14
- Utilities
 - setenv 8-4

V

- values
 - for label-based access control 6-2
- views 6-22
- violations tables 6-22
- virtual-index interface (VII) tables 6-1
- virtual-table interface (VTI) tables 6-1
- Visual disabilities
 - reading syntax diagrams A-1

W

- Warning messages
 - for server utilities security check 1-2
- White space in ADTCFG file 12-1
- Windows
 - access privileges 8-1, 8-2
 - access privileges for audit trail 7-13, 7-15
 - ADTCFG file 7-11
 - Application Event log 7-11
 - description 7-10
 - audit configuration 7-11, 10-7
 - audit trail in Application Event log 7-12
 - onaudit output 10-6
 - operations with onaudit 10-1
 - registry settings
 - for AAO 8-2
 - for DBSSO 8-1
 - for role separation 8-4
 - Security Event log 7-10
- write access
 - label-based access control 6-2

Z

- Zero (0) 8-12
 - ADTERR setting 8-12
 - ADTMODE default value 12-2
 - continue error code 7-11
 - onaudit error mode 8-6



Printed in USA

SC23-7754-02



Spine information:

IBM Informix **Version 11.50**

IBM Informix Security Guide

