OS/390®

Security Server
LDAP Server
Administration and Usage Guide

SC24-5861-04

OS/390®

Security Server
LDAP Server
Administration and Usage Guide

SC24-5861-04

Note

Before using this information and the product it supports, be sure to read the general information under Appendix G, “Notices” on page 399.

Acknowledgement

Some of the material contained in this document is a derivative of LDAP documentation provided with the University of Michigan LDAP reference implementation (Version 3.3). Copyright © 1992-1996, Regents of the University of Michigan, All Rights Reserved.

Fifth Edition (December 1999)

This edition, SC24-5861-04, applies to Version 2 Release 8 of OS/390 Security Server LDAP Server (5647-A01), and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC24-5861-03.

Contents

Figures

Tables

About This Book

This book explains the LDAP Server of the OS/390 Security Server. The LDAP Server supports Lightweight Directory Access Protocol (LDAP) and runs as a stand-alone daemon. It is based on a client/server model that provides client access to an LDAP Server. The LDAP Server provides an easy way to maintain directory information in a central location for storage, updating, retrieval, and exchange.

Who Should Use This Book

This document is intended to assist system administrators. System administrators should be experienced and have previous knowledge of directory services. It is also intended for anyone that will be implementing the directory service.

How This Book Is Organized

This book is divided into the following parts:

Conventions Used in This Book

This book uses the following typographic conventions:

Bold	**Bold** words or characters represent API names, attributes, status codes, environment variables, parameter values, and system elements that you must enter into the system literally, such as commands, options, or path names.
Italic	*Italic* words or characters represent values for variables that you must supply.
`Example font`	Examples and information displayed by the system appear in `constant width type style`.
[ ]	Brackets enclose optional items in format and syntax descriptions.
{ }	Braces enclose a list from which you must choose an item in format and syntax descriptions.
|	A vertical bar separates items in a list of choices.
< >	Angle brackets enclose the name of a key on the keyboard.
...	Horizontal ellipsis points indicate that you can repeat the preceding item one or more times.
\	A backslash is used as a continuation character when entering commands from the shell that exceed one line (255 characters). If the command exceeds one line, use the backslash character **** as the last nonblank character on the line to be continued, and continue the command on the next line.

Where to Find More Information

Where necessary, this book references information in other books, using shortened versions of the book title. For complete titles and order numbers of the books for all products that are part of OS/390, see the *OS/390 Information Roadmap*, GC28-1727. For a list of titles and order numbers of the books that are useful for the LDAP Server, see “Bibliography” on page 407.

Online Books

All the books belonging to the OS/390 Security Server library are available as online publications. They are included in the *IBM OS/390 Collection*, SK2T-6700.

All the books in the Online Library are viewable, without change, on these IBM operating platforms: OS/390, VM, OS/2®, DOS, and AIX/6000®. The same book can be viewed on any of these platforms using the IBM BookManager® Library Readers™ for OS/2, Windows, and DOS, or any of the IBM BookManager READ licensed programs for OS/390, VM, OS/2, Windows, DOS, or AIX/6000.

The booklet included with the Online Library provides details on accessing the OS/390 DCE online publications.

How to Send Your Comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book or any other OS/390 documentation:

- Visit the home page at: `http://www.ibm.com/s390/os390`
- Fill out one of the forms at the back of the book and return it by mail, by fax, or by giving it to an IBM representative.

Summary of Changes

Summary of Changes
for SC24-5861-04
OS/390 Version 2 Release 8

This book contains information previously presented in *OS/390 Security Server LDAP Server Administration and Usage Guide*, SC24-5861-03, which supports OS/390 Version 2 Release 8.

The following summarizes the changes to that information.

New Information

- Information about password encryption has been added which includes:
  - OCSF and ICSF requirements for password encryption
  - Configuration information regarding **userPassword** encryption
  - The **pwEncryption** option for the configuration file
  - The **db2pwden** utility
  - Password encryption information for **ldif2db** and **db2ldif**
  - New option, **-t**, for the **db2ldif** command

Changed Information

- Updates to the sample JCL for the LDAP Server, **ldif2db**, and **db2ldif**.

This book includes terminology, maintenance, and editorial changes that are not marked. Technical changes or additions to the text and illustrations, however, are indicated by a vertical line to the left of the change.

Summary of Changes
for SC24-5861-03
OS/390 Version 2 Release 8

This book contains information previously presented in *OS/390 Security Server LDAP Server Administration and Usage Guide*, SC24-5861-02, which supports OS/390 Version 2 Release 7.

The following summarizes the changes to that information.

New Information

- Information for LDAP Version 3 protocol support, which includes support for root DSE, controls, internationalization (UTF-8), SASL external bind and client and server authentication, and additional referral support.

  Note: The following elements of the LDAP Version 3 protocol, as defined by IETF RFCs 2251-2256, are not supported by the OS/390 LDAP Server in Release 8:

  - Schema discovery (publication)
  - Schema update through the LDAP protocol
  - Modify name
  - Attribute tagging

- Migration information which contains instructions for migrating existing OS/390 (Releases 5, 6, and 7) LDAP Server data to Release 8.

- The following configuration file options have been added: **altServer**, **extendedGroupSearching**, **masterServer**, **sendV3stringsoverV2as**, **sslAuth**, **sslKeyRingPWStashFile**, **verifySchema**, and **validateincomingV2strings**.
- A greatly expanded set of LDAP schema information is now shipped with the LDAP Server to simplify and speed up the deployment and exploitation of Directory Services. These files are named **schema.system.at**, **schema.system.oc**, **schema.IBM.at**, **schema.IBM.oc**, **schema.user.at**, and **schema.user.oc**.
- The **ldapadd**, **ldapdelete**, **ldapmodify**, **ldapmodrdn**, and **ldapsearch** utilities have a new **-M** parameter to control the use of the **manageDSAIT** control.

Changed Information

- The LDAP client and server are both packaged in the OS/390 Security Server and are always enabled.
- The following information has been reorganized within the book:
  - The steps that were previously in the "Getting the LDAP Server Started" chapter, are now part of Chapter 3, "Installing" on page 11.
  - The "Securing Your LDAP Server with SSL" chapter is now part of Chapter 5, "Configuring" on page 31.
  - The "Command Line Utilities" chapter is now part of Chapter 6, "Running LDAP Utilities and Programs" on page 73.
  - All of the sample configuration files are now contained in Appendix A, "Configuration Files" on page 241.
  - All of the sample JCL is now contained in Appendix B, "Sample JCL" on page 363.
  - The example programs are now contained in Part 4, Appendixes.
  - The "Setting Up Your Server to Run with SDBM" section is now contained in Chapter 5, "Configuring" on page 31, and Chapter 12, "Accessing RACF Information" on page 149 has been moved to Part 2, Usage.
  - Chapter 13, "Using Access Control" on page 159 has been moved to Part 2, Usage.
- The chapter "Using the Make Key File Utility (MKKF)" has been removed.

This book includes terminology, maintenance, and editorial changes that are not marked. Technical changes or additions to the text and illustrations, however, are indicated by a vertical line to the left of the change.

Summary of Changes
for SC24-5861-02
OS/390 Version 2 Release 7

This book contains information previously presented in *OS/390 Security Server LDAP Server Administration and Usage Guide*, SC24-5861-01, which supports OS/390 Version 2 Release 6.

The following summarizes the changes to that information.

New Information

- Chapter 5, “Configuring” on page 31 has new information about how to configure your LDAP server to run in multi-server mode with or without dynamic workload management enabled.
- Chapter 12, “Accessing RACF Information” on page 149 is a new chapter discussing how the LDAP server can provide LDAP access to the user and group information stored in RACF.
- Chapter 4, “Migrating” on page 21 contains new instructions for migrating from Release 5 or Release 6.
- “Securing Your LDAP Server with SSL” on page 65 and Chapter 4, “Migrating” on page 21 have new information on using System SSL with LDAP.

Summary of Changes
for SC24-5861-01
OS/390 Version 2 Release 6

This book contains information previously presented in *OS/390 Security Server LDAP Server Administration and Usage Guide*, SC24-5861-00, which supports OS/390 Version 2 Release 5.

The following summarizes the changes to that information.

New Information

- Chapter 4, “Migrating” on page 21 is a new chapter discussing migration issues if you have the Release 5 LDAP Server installed and want to move to Release 6.
- “Dynamic Debugging” on page 76 is a new section describing how to turn the debugging facility on and off.
- “Input Modes” on page 91 is a new section describing the type of input accepted by the **ldapmodify** and **ldapadd** utilities.
- “Creating ACLs and Owners Using LDIF-Format Input to ldif2db” on page 166 is a new section showing several examples of how to add entries, with ACL attributes and owner attributes, represented in LDIF format.
- Chapter 7, “Using the ldapcp Command” on page 115 includes new flags for the **ldapcp** command and a new section “Using ldapcp to Administer Remote Server Data” on page 117.

Part 1. Administration

Chapter 1. Introducing the OS/390 LDAP Server

The OS/390 Lightweight Directory Access Protocol (LDAP) Server (part of the OS/390 Security Server) is based on a client/server model that provides client access to an LDAP Server. An LDAP directory provides an easy way to maintain directory information in a central location for storage, update, retrieval, and exchange.

The LDAP Server provides the following functions:

- Interoperability with other LDAP clients
- Access controls on directory information
- Access Control List and Group Administration Utility (**ldapcp**)
- Secure Sockets Layer (SSL) communication
- Certificate management
- Password encryption
- LDAP Version 2 and Version 3 protocol support
- Client and server authentication using SSL
- Replication
- Referrals

This book describes how to install, configure, and run the stand-alone LDAP daemon (SLAPD) and other LDAP programs. It is intended for newcomers and experienced administrators alike. This section provides a basic introduction to directory service, and the directory service provided by SLAPD in particular.

The *OS/390 LDAP Client Application Development Guide and Reference* describes the LDAP client application programming interfaces (APIs) you can use to develop LDAP applications.

What Is a Directory Service?

A directory is like a database, but tends to contain more descriptive, attribute-based information. The information in a directory is generally read much more often than it is written. As a consequence, directories do not usually implement the complicated transaction or rollback schemes regular databases use for doing high-volume complex updates. Directory updates are typically simple all-or-nothing changes, if they are allowed at all. Directories are tuned to give quick-response to high-volume lookup or search operations. They may have the ability to replicate information widely in order to increase availability and reliability, while reducing response time. When directory information is replicated, temporary inconsistencies between the replicas may be alright, as long as they get in sync eventually.

There are many different ways to provide a directory service. Different methods allow different kinds of information to be stored in the directory, place different requirements on how that information can be referenced, queried and updated, how it is protected from unauthorized access, and so on. Some directory services are local, providing service to a restricted context (for example, the finger service on a single machine). Other services are global, providing service to a much broader context (for example, the entire Internet). Global services are usually distributed, meaning that the data they contain is spread across many machines, all of which cooperate to provide the directory service. Typically a global service defines a uniform namespace which gives the same view of the data no matter where you are in relation to the data itself.

What Is LDAP?

SLAPD's model for directory service is based on a global directory model called LDAP, which stands for the Lightweight Directory Access Protocol. LDAP Version 2 (V2) and LDAP Version 3 (V3), both supported in OS/390, are directory service protocols that run over TCP/IP. The details of LDAP V2 are defined in Internet Engineering Task Force (IETF) Request for Comments (RFC) 1777 *The Lightweight Directory Access Protocol* and the details of LDAP V3 are defined in IETF RFC 2251 *The Lightweight Directory Access Protocol (V3)*.

This section gives an overview of LDAP from a user's perspective.

What Kind of Information Can Be Stored in the Directory?

The LDAP directory service model is based on *entries*. An entry is a collection of attributes that has a name, called a *distinguished name* (DN). The DN is used to refer to the entry unambiguously. Each of the entry's attributes has a type and one or more values. The types are typically mnemonic strings, like `cn` for common name, or `mail` for e-mail address. The values depend on what type of attribute it is. For example, a mail attribute might contain an e-mail address in an attribute value `thj@vnet.ibm.com`. A **jpegPhoto** attribute would contain a photograph in binary JPEG or JFIF format.

How Is the Information Arranged?

In LDAP, directory entries are arranged in a hierarchical tree-like structure that reflects political, geographic or organizational boundaries. Entries representing countries appear at the top of the tree. Below them are entries representing states or national organizations. Below them might be entries representing people, organizational units, printers, documents, or just about anything else you can think of. Figure 1 on page 5 shows an example LDAP directory tree, which should help make things clear.

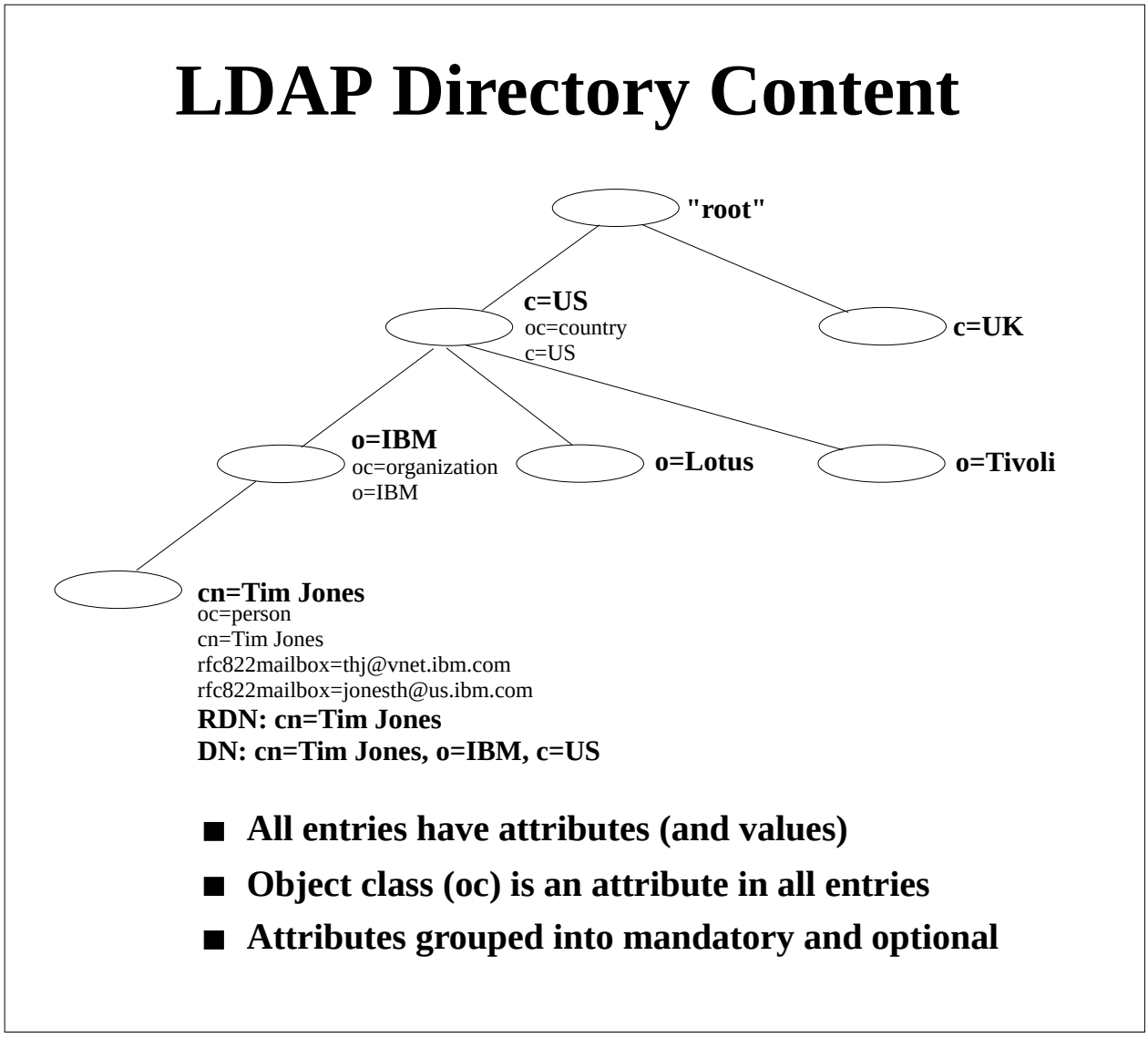


Figure 1. Directory Hierarchy Example

In addition, LDAP allows you to control which attributes are required and allowed in an entry through the use of a special attribute called *object class*. The values of the **objectClass** attribute determine the schema rules the entry must obey.

How Is the Information Referenced?

An entry is referenced by its distinguished name, which is constructed by taking the name of the entry itself (called the *relative distinguished name*, or RDN) and concatenating the names of its ancestor entries. For example, the entry for Tim Jones in the example above has an RDN of `cn=Tim Jones` and a DN of `cn=Tim Jones, o=IBM, c=US`. The full DN format is described in IETF RFC 2253, *LDAP (V3): UTF-8 String Representation of Distinguished Names*.

How Is the Information Accessed?

LDAP defines operations for interrogating and updating the directory. Operations are provided for adding and deleting an entry from the directory, changing an existing entry, and changing the name of an entry. Most of the time, though, LDAP is used to search for information in the directory. The LDAP search operation allows some portion of the directory to be searched for entries that match some criteria specified by a search filter. Information can be requested from each entry that matches the criteria.

For example, you might want to search the entire directory subtree below IBM for people with the name Tim Jones, retrieving the e-mail address of each entry found. LDAP lets you do this easily. Or you might want to search the entries directly below the `c=US` entry for organizations with the string `Acme` in their name, and that have a FAX number. LDAP lets you do this too. The next section describes in more detail what you can do with LDAP and how it might be useful to you.

How Is the Information Protected from Unauthorized Access?

The Access Control List (ACL) provides a means to protect information stored in an LDAP directory. Administrators use ACLs to restrict access to different portions of the directory, specific directory entries, or information within an entry. Access control can be specified for individual users or groups.

How Does LDAP Work?

LDAP directory service is based on a client/server model. One or more LDAP Servers contain the data making up the LDAP directory tree. An LDAP client application connects to an LDAP Server using LDAP APIs and asks it a question. The server responds with the answer, or with a pointer to where the application can get more information (typically, another LDAP Server). With a properly constructed namespace, no matter which LDAP Server an application connects to, it sees the same view of the directory; a name presented to one LDAP Server references the same entry it would at another LDAP Server. This is an important feature of a global directory service, like LDAP.

What About X.500?

LDAP was originally developed as a front end to X.500, the OSI directory service. X.500 defines the Directory Access Protocol (DAP) for clients to use when contacting directory servers. DAP has been characterized as a heavyweight protocol that runs over a full OSI stack and requires a significant amount of computing resources to run. LDAP runs directly over TCP and provides most of the functionality of DAP at a much lower cost.

The stand-alone LDAP daemon, or SLAPD, is meant to remove much of the burden from the server side just as LDAP itself removed much of the burden from clients. If you are already running an X.500 service and you want to continue to do so, you can probably stop reading this guide, which is all about running LDAP through SLAPD, without running X.500. If you are not running X.500, want to stop running X.500, or have no immediate plans to run X.500, read on.

What Are the Capabilities of the OS/390 LDAP Server?

The name of the LDAP Server program in the Hierarchical File System (HFS) is **slapd** to be consistent with other UNIX® implementations. You can use it to provide a directory service of your very own. Your directory can contain just about anything you want to put in it. Some of the OS/390 LDAP Server's more interesting features and capabilities include:

- **Robust database**: The OS/390 LDAP Server comes with an RDBM backend database based on DB2®.
- **Multiple database instances**: The OS/390 LDAP Server can be configured to serve multiple databases at the same time. This means that a single OS/390 LDAP Server can respond to requests for many logically different portions of the LDAP tree.
- **Access control**: The OS/390 LDAP Server provides a rich and powerful access control facility, allowing you to control access to the information in your database or databases. You can control access to entries based on LDAP authentication information, including users and groups. Access control is configurable down to sets of attributes within entries. See Chapter 13, “Using Access Control” on page 159 for more information.
- **Threads**: The OS/390 LDAP Server is threaded for high performance. A single multithreaded OS/390 LDAP Server process handles all incoming requests, reducing the amount of system overhead required.
- **Replication**: The OS/390 LDAP Server can be configured to maintain replica copies of its database. This master/slave replication scheme is vital in high-volume environments where a single OS/390 LDAP Server just does not provide the necessary availability or reliability. See Chapter 14, “Replication” on page 169 for more information.
- **Referrals**: The OS/390 LDAP Server provides the ability to refer clients to additional directory servers. Using referrals you can distribute processing overhead, distribute administration of data along organizational boundaries, and provide potential for widespread interconnection beyond an organization's own boundaries. See Chapter 15, “Referrals” on page 177 for more information.
- **Configuration**: The OS/390 LDAP Server is highly configurable through a single configuration file which allows you to change just about everything you would ever want to change. Configuration options have reasonable defaults, making your job much easier. See Chapter 5, “Configuring” on page 31 for more information.
- **Secure communications**: The OS/390 LDAP Server can be configured to encrypt data to and from LDAP clients using the OS/390 Cryptographic Services System SSL. It has a variety of ciphers for encryption to choose from, all of which provide server and optionally client authentication through the use of X.509 certificates. See “Securing Your LDAP Server with SSL” on page 65 for more information.
- **Multiple concurrent servers**: The OS/390 LDAP Server can be configured to permit multiple instances to serve the same RDBM database at the same time. The multiple server instances may run on the same OS/390 image, and they may run on multiple OS/390 images in a Parallel Sysplex(®). This improves availability and may offer improved performance in certain configurations. See Chapter 5, “Configuring” on page 31 for more information.
- **Dynamic workload management**: The OS/390 LDAP Server can be configured to participate in dynamic workload management in a Parallel Sysplex by exploiting TCP/IP connection optimization. With multiple concurrent server instances configured in this way, availability is improved, as is resource utilization. In addition, performance improvements may be experienced as sysplex resource utilization is more evenly balanced across OS/390 systems in the sysplex. See Chapter 5, “Configuring” on page 31 for more information.
- **Access to RACF® data**: The OS/390 LDAP Server can be configured to provide access to RACF user and group profiles using the LDAP protocol. If the RACF data is shared across the sysplex, then users and groups in the sysplex can be managed using LDAP. The LDAP Server's access to RACF is managed by an additional configurable backend called SDBM. See Chapter 12, “Accessing RACF Information” on page 149 for more information.
- **LDAP Version 3 Protocol Support**: The OS/390 LDAP Server provides support for Version 3 protocol. This includes implicit bind, certificate (or SASL) bind, Version 3 referrals, handling of controls, and root DSE support.

Note: The following elements of the LDAP Version 3 protocol, as defined by IETF RFCs 2251-2256, are not supported by the OS/390 LDAP Server in Release 8:

- Schema discovery (publication)
- Schema update through the LDAP protocol
- Modify name
- Attribute tagging

- **Internationalization (UTF-8) Support** The OS/390 LDAP Server allows storage update and retrieval, through LDAP operations, of national language data using LDAP Version 3 protocol. See "UTF-8 Support" on page 137 for more information.
- **SASL External Bind and Client and Server Authentication**: The OS/390 LDAP Server allows client applications to use a certificate when communicating with the server using SSL communications. In order to use a certificate on bind, the server must be configured to perform both client and server authentication. This ensures both entities are who they claim to be. See "Securing Your LDAP Server with SSL" on page 65 for more information.
- **Support for Root DSE**: The OS/390 LDAP Server supports search operations against the Root of the Directory tree as described in IETF RFC 2251, *The Lightweight Directory Access Protocol (V3)*. The so-called Root DSE can be accessed using LDAP V3 protocol operations. See "ldapsearch Utility" on page 103 for more information.
- **Extended Group Membership Searching**: The OS/390 LDAP Server supports extended group membership searching which allows the LDAP Server to find a DN that may be a member of a group in a backend where the DN does not reside. The LDAP Server can find the group memberships for the DNs in the other backends it services. See the **extendedGroupSearching** configuration file option on page 38 for more information.
- **Supported Server Controls**: The OS/390 LDAP Server supports the **manageDsaIT** and **authenticateOnly** server controls. See Appendix F, "Supported Server Controls" on page 397 for more information.
- **Password Encryption**: The OS/390 LDAP Server allows prevention of unauthorized access to user passwords stored in the RDBM backend. See "userPassword Encryption" on page 44 for more information.

Chapter 2. Planning

Before configuring and populating your database, determine:

- What type of data you are going to store in the directory.

  You should decide on what sort of schema you need to support the type of data you want to keep in your directory. The directory server is shipped with a standard set of attribute type definitions and object class definitions.

  Before you begin adding entries to the directory, you might want to add new attribute type and object class definitions that are customized to your data.

  Note: Schema additions may be made after the directory is already populated with data, but schema changes may require unloading and reloading your data.

  Refer to Chapter 11, “Directory Schema” on page 145 for more information.

- How you want to structure your directory data.

  Refer to Chapter 10, “Distinguished Names” on page 143 for more information.

- Data communications security.

  Refer to “Securing Your LDAP Server with SSL” on page 65 for more information.

- A set of policies for access permissions.

  Refer to Chapter 13, “Using Access Control” on page 159 for more information.

Note: If you already have the LDAP Server installed on your system, see Chapter 4, “Migrating” on page 21.

Chapter 3. Installing

This chapter discusses how to install, set up, and run the LDAP Server product and OS/390 DB2.

See Chapter 4, “Migrating” on page 21 if you already have the LDAP Server installed on your system.

Installing the LDAP Server Product

For complete instructions for installing the LDAP Server product, see the *OS/390 Program Directory* which comes with the LDAP Server tape or cartridge. Be sure to read the license agreement in the *Licensed Program Specifications*, which is also included in the box.

Important

Before you proceed to the next section, review the *Memo to Users*, which describes any late changes to the procedures in this book. A printed copy is included with the LDAP Server tape or cartridge.

The LDAP Server searches for and loads a number of dynamic load libraries (DLLs) during its startup processing. All DLLs for the LDAP server are shipped in PDS format only. In order for these DLLs to be located by the LDAP server at runtime, the PDS which contains these DLLs (*GLDHLQ*.SGLDLNK) must either be in the LINKLIST, referenced in a **STEPLIB DD** card (if the LDAP server is started from JCL), or listed in the **STEPLIB** environment variable (if the LDAP server is started from the OMVS command prompt). Any of these methods can be used, and the choice of the best method is dependent on the way you will most often be running the LDAP Server.

Access to RACF information through the LDAP Server is available. The PDS which contains the LDAP Server and the DLLs must now be APF-authorized to allow the LDAP Server to make the RACF calls necessary to provide this access. Also, if program control is active on your system, the PDS which contains the LDAP Server and the DLLs, and the PDS that contains the C runtime libraries and SYS1.LINKLIB must be program controlled. Also, if using both the DB2 data store and RACF access in the same server instance, the PDS containing the DB2 CLI (Call Level Interface) *DB2HLQ*.SDSNLOAD must also be APF-authorized and program controlled.

Preparing the LDAP Server

This section describes the optional and required actions that must be completed before running the LDAP Server.

1. Defining the user ID that runs the LDAP Server (optional)
2. Getting DB2 installed and set up for CLI and ODBC (optional)
3. Creating the LDAP Server DB2 database (optional)
4. Configuring the LDAP Server for SSL (optional)
5. Installing OCSF and ICSF for password encryption (optional)

Note that if your LDAP Server will be used **only** for accessing RACF information, it is not necessary to install DB2 or set up a DB2 database. See “Setting Up Your Server to Run with SDBM” on page 64 for information on configuring the LDAP Server for accessing RACF information.

Some of the examples and descriptions reflect assumptions that may not apply to your environment. Following are descriptions of these assumptions, with guidance on how to use this information if they do not apply to your environment:

- Some examples use Resource Access Control Facility (RACF). You can use any OS/390 external security manager that has equivalent support. You must substitute the appropriate procedures for any examples that use RACF.
- The default name **/usr/lpp/ldap** is used for the directory in which you installed the LDAP Server product. If you used a different name, substitute that name in the examples and descriptions where applicable.
- The language setting **En_US.IBM-1047** is used for the locale in which you are running the LDAP Server. This setting is used in the names of several directories that are referred to in this information. If you are using a different language setting, substitute that setting in the examples and descriptions where applicable. You must also specify this setting as the value of the **LANG** parameter in the environment variable file as described in Chapter 8, "Internationalization Support" on page 137. The default environment variable file already sets **LANG** to **En_US.IBM-1047**.
- The name **LDAPSRV** is used for the user ID that runs the LDAP Server. If you use a different name, substitute that name in the examples and descriptions where applicable.
- The **/etc** is used as the name of a production directory. If you use a different name, you must symbolic link the names of the appropriate files in your directory to the **/etc** directory.

Defining the User ID that Runs the LDAP Server: To define the user ID that runs the LDAP Server, perform the following steps:

1. Create a user ID with the required attributes
2. Define the started task for the LDAP Server

If you are going to set up more than one LDAP Server, set up a separate user ID for each one.

Creating the User ID: You must create a user ID for the LDAP Server that has the following attributes:

- A UID of 0 so that it always runs with superuser authority
- Read access, if you defined the BPX.DAEMON facility class
- Update access, if you defined the BPX.SERVER facility class
- Read access to the data sets defined in the startup procedure

To create **LDAPSRV**, you can use the RACF commands in the following example:

```
ADDGROUP LDAPGRP SUPGROUP(SYS1) OMVS(GID(2))
ADDUSER LDAPSRV DFLTGRP(LDAPGRP) OMVS(UID(0) PROGRAM ('/bin/sh'))
PERMIT BPX.DAEMON CLASS(FACILITY) ID(LDAPSRV) ACCESS(READ)
PERMIT BPX.SERVER CLASS(FACILITY) ID(LDAPSRV) ACCESS(UPDATE)
```

For information on setting superuser authority, see the topic about defining superusers in *OS/390 UNIX System Services Planning*, SC28-1890.

Defining the Started Task for the LDAP Server: After you create the **LDAPSRV** user ID, define the **LDAPSRV** started task. The examples and the sample startup procedure use the name **LDAPSRV** for this task, but you can use any name for it.

To define the started task for the user ID you just created, you can use the following RACF commands.

```
RDEFINE STARTED LDAPSRV.** STDATA(USER(LDAPSRV))
SETROPTS RACLIST(STARTED) REFRESH
```

Setting Up DB2 and Getting the LDAP Server Running

This section describes how to get OS/390 DATABASE 2™ (DB2) running and how to run the LDAP Server using the RDBM (DB2) backend. You should also have or have access to the *DB2 for OS/390 Call Level Interface Guide and Reference* and the *DB2 for OS/390 Application Programming and SQL Guide*. There is an additional set of example files shipped in **/usr/lpp/ldap/examples/sample_server** that can be used to understand how to configure and run the LDAP Server. The following list shows the files shipped in that directory.

- **dsnaoini.db2ini**
- **dsntijcl.jcl**
- **ldapspfi.spufi**
- **sample.ldif**
- **slapd.at.conf**
- **slapd.at.system**
- **slapd.oc.system**
- **slapd.oc.conf**
- **slapd.conf**
- **README**

See the **README** file for step-by-step instructions for getting a sample SLAPD server running.

Getting DB2 Installed and Setup for CLI and ODBC

Following are the steps to get DB2 installed:

1. Have your database system administrator install DB2 Version 5. If you will be running your LDAP server in multi-server mode on multiple images in a Parallel Sysplex, your administrator must configure a DB2 data sharing group with members on each of the OS/390 images on which an LDAP server instance will run. (See Chapter 5, “Configuring” on page 31 for a description of the various operating modes in which the LDAP server may run.)

   Make sure the that SMP/E jobs are a part of the DB2 installation. See the section about installing DB2 CLI in the *DB2 for OS/390 Call Level Interface Guide and Reference*. Also, specify the user ID (for example, su*xxxx*) that should be granted database system administrator authority. You need to find out the following information from your database administrator:

   - DB2 subsystem name. For example, DSN5.
   - DB2 server location (or data source). For example, LOC1.

   In order to use a local or remote DB2 database, you must include a DDF record in your Bootstrap Data Set (BSDS). That DDF record must include a LOCATION keyword and an LUNAME keyword. If you are using a DB2 database that is on the local system (including a database that is setup for DB2 data sharing) the DDF component need not be started. If you are using a DB2 database that is on a remote system, the DDF component of DB2 must be configured and started on systems using the DB2 Call Level Interface (CLI). CLI is used by the LDAP server for requesting services from DB2. (The DB2 Call Level Interface is IBM's callable SQL interface used by the DB2 family of products, based on the ISO Call Level Interface Draft International Standard specification and the Microsoft® Open Database Connectivity specification.)

   The LDAP Server uses 32KB table spaces. Have your database system administrator define TEMP 32K SPACE and TEMP 32K DATA SETS. Your database system administrator must also define 32K buffer pools in your DB2 configuration. The sizes of the buffer pools (both 4K and 32K) should also be examined by your database system administrator to ensure they are large enough to meet the additional needs of the LDAP Server, once you have loaded data into its database.

2. Enter:

-*dsn* start db2

from the image console and wait for DB2 to finish the DB2 initialization. The *dsn* is the DB2 subsystem name.

You can stop DB2 by entering:

-*dsn* stop db2

from the console.

Note: This may already be done when the system is re-ipled.

3. Edit and Submit DSN*HLQ*.SDSNSAMP(DSNTIJCL) where DSN*HLQ* is the high-level qualifier used during DB2 installation. See the section on setting up DB2 CLI runtime environment in the *DB2 for OS/390 Call Level Interface Guide and Reference*. You must run this from the user ID that has been granted the appropriate database authorities.
4. Create (Allocate) DB2 CLI Initialization File. The file that you create must be a sequential file (a DASD data set and not an HFS file). A sample of the CLI initialization file can be found at DSN*HLQ*.SDSNSAMP(DSNAOINI). Create your own CLI initialization files and copy DSN*HLQ*.SDSNSAMP(DSNAOINI) into it. Refer to the section on the DB2 CLI initialization File in the *DB2 for OS/390 Call Level Interface Guide and Reference* for more information on the contents of this file. Figure 2 shows a sample file.

```
;This is a comment line...
; Example COMMON stanza
[COMMON]
MVSDEFAULTSSID=yoursubsystemname

; Example SUBSYSTEM stanza for your DB2 subsystem name
[yoursubsystemname]
MVSATTACHTYPE=yourmvsattachtype
PLANNAME=yourCLIplanname

; Example DATA SOURCE stanza for your data source
[yourdatasourcename]
AUTOCOMMIT=0
CONNECTTYPE=1
```

Figure 2. Sample DSNAOINI File

Creating the LDAP Server DB2 Database and Table Spaces

The LDAP Server DB2 database must be created by running a SPUFI (SQL Processor Using File Input) script from DB2 Interactive (DB2I). DB2I is a DB2 facility that provides for the running of SQL statements, DB2 (operator) commands, and utility invocation. For details on how to use DB2I and SPUFI, see the *DB2 for OS/390 Application Programming and SQL Guide*. A sample DB2I SPUFI script to create the LDAP Server DB2 database is provided. To use it, do the following:

1. Copy the SPUFI script over to your SPUFI input data set.

   The SPUFI script for creating the database and table spaces can be found in *GLDHLQ*.SGLDSAMP(LDAPSPFI) where *GLDHLQ* refers to the high-level qualifier that was used to install the LDAP Server data sets.

   Create a database and table spaces for your LDAP Server. Use the DB2 SPUFI (SQL Processor Using File Input) facility to create the database and table spaces. Figure 3 on page 15 shows an example of the file to edit and run in the SPUFI facility.

```
--//*************************************************************//
--//*
--//* Licensed Materials - Property of IBM
--//* 5647-A01
--//* (C) Copyright IBM Corp.  1997, 1998
--//*
--//*************************************************************//

-- Use the following statements to create your LDAP Server DB2 database
-- and tablespaces in SPUFI.  The database and tablespace names you
-- create will be used to update the database section of the LDAP
-- Server configuration file.
--
-- Change DDDDDDDD to the name of the LDAP database name you want to create.
-- Be sure this name is updated to match what is defined for databasename in
-- the server configuration file.
-- Change the AAAAAAAA to the LDAP entry tablespace name you want to create.
-- Be sure this name is updated to match what is defined for tbspaceentry in
-- the server configuration file.
-- Change the BBBBBBBB to the LDAP 4K tablespace name you want to create.
-- Be sure this name is updated to match what is defined for tbspace4k in
-- the server configuration file.
-- Change the CCCCCCCC to the LDAP 32K tablespace name you want to create.
-- Be sure this name is updated to match what is defined for tbspace32k in
-- the server configuration file.
-- Change the EEEEEEEE to another LDAP 4K tablespace name you want to
-- create
-- Be sure this name is updated to match what is defined for tbspacemutex
-- in the server configuration file.
-- Change the SSSSSSSS to the storage group you want to contain the
-- LDAP tablespaces.  Use SYSDEFLT to choose the default storage group.
--
-- NOTE: The values provided below for PRIQTY and SECQTY may need
--       to be modified depending on the projected size of the
--       Directory information to be stored.

CREATE DATABASE DDDDDDDD STOGROUP SSSSSSSS;

CREATE LARGE TABLESPACE AAAAAAAA IN DDDDDDDD
       NUMPARTS 1
       USING STOGROUP SSSSSSSS PRIQTY 14400 SECQTY 7200
       BUFFERPOOL BP32K;

CREATE TABLESPACE BBBBBBBB IN DDDDDDDD
       SEGSIZE 4
       USING STOGROUP SSSSSSSS PRIQTY 14400 SECQTY 7200
       BUFFERPOOL BP0;
CREATE TABLESPACE CCCCCCCC IN DDDDDDDD
       SEGSIZE 4
       USING STOGROUP SSSSSSSS PRIQTY 14400 SECQTY 7200
       BUFFERPOOL BP32K;

CREATE TABLESPACE EEEEEEEE IN DDDDDDDD
       LOCKSIZE TABLESPACE
```

Figure 3 (Part 1 of 2). ldapspfi.spufi File

```
        USING STOGROUP SSSSSSSS
        BUFFERPOOL BP0;


-- Use the following statements if you need to delete your LDAP Server DB2
-- database and tablespaces in SPUFI.  You need to remove the '--'
-- from each line before you can run these statements.
-- Change the AAAAAAAA to the LDAP entry tablespace name you want to delete.
-- Change the BBBBBBBB to the LDAP 4K tablespace name you want to delete.
-- Change the CCCCCCCC to the LDAP 32K tablespace name you want to delete.
-- Change the EEEEEEEE to the LDAP 4K tablespace name you want to delete.
-- Change the DDDDDDDD to the LDAP database name you want to delete.

--DROP TABLESPACE DDDDDDDD.AAAAAAAA;
--DROP TABLESPACE DDDDDDDD.BBBBBBBB;
--DROP TABLESPACE DDDDDDDD.CCCCCCCC;
--DROP TABLESPACE DDDDDDDD.EEEEEEEE;
--DROP DATABASE   DDDDDDDD;
```

Figure 3 (Part 2 of 2). ldapspfi.spufi File

If you are going to be running in multi-server mode on multiple images in a Parallel Sysplex and you have an existing Release 5 or Release 6 DB2 database, see "Preparing an Existing DB2 Database for Multi-server" on page 22 for specific information on migrating your database using the **ldapspfi.spufi.migrate** file. The **ldapspfi.spufi** file you are using must be updated with the information in the **ldapspfi.spufi.migrate** file.

2. Run the script from DB2I SPUFI under a user ID with DB2 **SYSADM** authority. When the script completes running, scan the output data set to ensure that it ran successfully.

3. **Grant appropriate DB2 resource authorizations**.

   In order to run the **ldif2db** tool and the LDAP server, certain minimum DB2 resource authorizations must be granted to the user ID or user IDs that will be running these programs. Following are the suggested minimums which should be granted to those user IDs, where *xxx* is the user ID, *yyy* is the database name identified in the **slapd.conf** file for the **databasename** option and *zzz* is the CLI plan name as specified in your DB2 CLI initialization file. Run the following statements through SPUFI (DB2 Interactive):

   ```
   grant execute on plan zzz to xxx;
   grant select on sysibm.systables to xxx;
   grant dbadm on database yyy to xxx;
   ```

   These privileges may be granted by any user ID with **SYSADM** authority. The commands above can be run using the DB2 SPUFI facility.

   The LDAP Server requires **SELECT** access to the SYSIBM.SYSCOLUMNS table in DB2. If **SELECT** access to this table is tightly controlled in your DB2 installation, then it may be necessary to grant this access to the user ID under which the LDAP Server runs by performing the following operation (either through SPUFI or another means of issuing SQL commands):

   ```
   grant select on sysibm.syscolumns to xxx;
   ```

   where *xxx* is the user ID under which the LDAP Server runs. If this authority is not granted to the user ID under which the LDAP Server runs, the LDAP Server will fail during start-up with an SQL -551 return code.

4. **Make a configuration file.** Create a file called `myslapd.conf` and enter the lines from Figure 28 on page 241. This is the **slapd.conf** file that is provided in the **/usr/lpp/ldap/examples/sample_server** directory. See Chapter 5, "Configuring" on page 31 for more information about this file.

Notes:

a. Do not remove the `suffix "cn=localhost"` line.
b. Be sure to specify **adminDN**. You can specify the **adminPW** here or in a database entry. See "Establishing the Administrator DN and Password" on page 43 for more information. Note that the use of the **adminPW** option is strongly discouraged. Instead, an existing entry in the directory should be designed as the **adminDN**.
c. The **slapd.conf** file in the **/usr/lpp/ldap/examples/sample_server** directory is different from the **slapd.conf** file in the **/usr/lpp/ldap/etc** directory referred to in these installation instructions.

Make sure the values in the general section of this file are set correctly and be sure to replace the following fields in the RDBM Database Definitions section:

`yourDBserver` The name of your database server. (This is also the name from step 1 on page 13.)

`yourDBname` The name of your database. (This is from step 1 on page 14.)

`yourDBuserid` The database user ID used to create the database tables.

`yourEntryTablespace` The entry tablespace name. (This was done in step 1 on page 14.)

`your32KTablespace` The 32K tablespace name. (This was done in step 1 on page 14.)

`your4KTablespace` The 4K tablespace name. (This was done in step 1 on page 14.)

`your4KMutexTablespace`
The 4K mutex tablespace name. (This was done in step 1 on page 14.)

`yourCLIInitializationFile`
The DB2 CLI initialization file. (This was done in step 4 on page 14.)

Note that the sample **slapd.conf** file assumes a simple configuration of a single RDBM backend without multiserver. The **slapd.conf** must be tailored to make use of other capabilities, including use of multiserver, use of sysplex support, use of an SDBM backend for access to RACF, and use of server and client SSL authentication. The configuration file changes needed for these options are discussed in Chapter 5, "Configuring" on page 31.

5. **Run the ldif2db tool to create entries in the SLAPD database.**

   In order to run the **ldif2db** tool, you need to:

   - See "Running the LDAP DB2 Backend Utilities" on page 77 for setup instructions.
   - Create an LDIF file. A sample LDIF file is shown in Appendix C, "Sample LDIF Input File" on page 369 and **/usr/lpp/ldap/examples/sample_server** includes a complete `sample.ldif` file.
   - Add an additional suffix line in the **slapd.conf** file. For example, the root suffix for `sample.ldif` is `"o=Your Company,c=US"`.

   The following example shows you how to use the **ldif2db** tool to create the entries.

   ```
   /usr/sbin/ldif2db -f myslapd.conf -i sample.ldif
   ```

   The `myslapd.conf` file is the configuration file you made in step 4 on page 16, and `sample.ldif` is the LDIF file you just created.

6. **Start the LDAP Server (SLAPD).** The following example shows you how to start SLAPD from the HFS.

   ```
   /usr/sbin/slapd -f myslapd.conf
   ```

   To run SLAPD as a started task, customize the **LDAPSRV** JCL and start the job from the console or SDSF. Also, be sure to have **STEPLIB** set up to reference *GLDHLQ*.SGLDLNK before running **ldif2db** and SLAPD. If your runtime libraries for DB2 are not in LINKLIB or LPA on the system, make

sure you specify the DB2 high-level qualifier for your DB2 installation in a STEPLIB DD card in the LDAPSRV started task, GLDLD2DB batch job, or GLDDB2LD batch job.

When SLAPD has been started and is ready, the message

```
GLD0122I SLAPD is ready for requests.
```

is displayed.

7. **See if it works.** You can use any LDAP client to do this, but our example uses the **ldapsearch** tool.

```
ldapsearch -h 127.0.0.1 -b "o=Your Company,c=US" "objectclass=*"
```

This command assumes a default port of 389. If your port is not 389, use the **-p** parameter to specify the correct port. The **slapd.conf** file from **/usr/lpp/ldap/examples/sample_server** uses port 1800, therefore `-p 1800` should be added to the command above if you are using this configuration file.

This command searches for and retrieves every entry in the database. Be sure to substitute the correct TCP/IP host name or TCP/IP address for the `127.0.0.1` after the **-h** parameter. The **-b** parameter specifies the starting point for the search. The use of the quotation marks around the filter prevents the asterisk (*) from being interpreted by the shell.

Note that this can be done from TSO as well, substituting **LDAPSRCH** for **ldapsearch**. See "Running the LDAP Operation Utilities" on page 85 for setup instructions.

You are now ready to add more entries (for example, using **ldapadd** or another LDAP client), experiment with various configuration options, backend arrangements, and so on. Note that by default, the SLAPD database grants read access to everybody. Therefore, if you want to add or modify entries over LDAP, you will have to bind as the **adminDN** specified in the configuration file (see "Configuration File Global Options" on page 33), or change the default access control (see Chapter 13, "Using Access Control" on page 159).

Preparing the LDAP Server for SSL

In order for your LDAP Server to provide SSL support, you must install OS/390 Cryptographic Services System SSL and use STEPLIB, LPALIB, or LINKLIST to make their libraries available. See "Securing Your LDAP Server with SSL" on page 65 and the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for more information regarding SSL.

Installing OCSF and ICSF for Password Encryption

The LDAP Server uses the Open Cryptographic Services Facility (OCSF) to provide MD5 and SHA hashing of user passwords. The LDAP Server uses both OCSF and the Integrated Cryptographic Service Facility (ICSF) to provide DES encryption and decryption of user passwords. The LDAP Server does not require OCSF or ICSF to provide crypt() level encryption of user passwords. If you plan to encrypt passwords using MD5 hashing, SHA hashing, or DES encryption, you must install and configure the appropriate facility, or facilities, along with the LDAP Server.

OCSF

To install and configure OCSF, refer to the configuration information in the *OS/390 Open Cryptographic Services Facility Application Developer's Guide and Reference*. This contains instructions on how to set up the necessary security authorizations using RACF to use OCSF. OCSF must be configured so that the user ID under which the LDAP Server runs can use OCSF services. It also contains information on Program Control if RACF Program Control is activated. This documentation also contains instructions on how to run the installation scripts necessary to use OCSF.

Note: Although both OCSF and ICSF come with the base feature of OS/390, in the United States and Canada, an additional OCSF Security Level 3 feature must be ordered. There is no charge for this feature.

If the LDAP Server uses an SDBM backend to access RACF, then the OCSF libraries also need to be APF-authorized. The APF-authorized extended attribute must be turned on for the OCSF DLLs. The DLLs (`.dll` and `.so` files) in the **/usr/lpp/ocsf/lib** and **/usr/lpp/ocsf/addins** directories must have their APF-authorized extended attribute turned on by using the **extattr +a** command. Use the following commands:

```
RDEFINE FACILITY BPX.FILEATTR.APF UACC(NONE)
PERMIT BPX.FILEATTR.APF CLASS(FACILITY) ID(userid) ACCESS(UPDATE)
SETROPTS RACLIST(FACILITY) REFRESH
```

where *userid* is the ID from which the **extattr** command will be run.

Use the following commands from an OMVS command prompt:

```
$   cd /usr/lpp/ocsf/lib
$   extattr +a *.dll
$   cd /usr/lpp/ocsf/addins
$   extattr +a *.so
```

Refer to *OS/390 UNIX System Services Planning*, SC28-1890, for more details.

If program control is active, then the OCSF DLLs must also be program controlled. See the configuration information in the *OS/390 Open Cryptographic Services Facility Application Developer's Guide and Reference* for more information.

ICSF

To install, configure, and activate ICSF, your processor must have hardware cryptographic support. All new processors have hardware cryptographic support, while some older processors optionally provided this support.

Two other services of ICSF needed for DES encryption in the LDAP Server are the Key Generator Utility Program (KGUP) and the Cryptographic Key Data Set (CKDS). These are needed to generate and store the key and key label needed for DES encryption. Refer to the information about managing cryptographic keys and using the Key Generator Utility Program in the *OS/390 ICSF Administrator's Guide* for instructions on how to generate and store into CKDS a data-encrypting key (also referred to as data key) for DES encryption and how to set up the necessary security authorizations using RACF to use the key. It is important to remember to refresh both CKDS and RACF after you make the changes. ICSF must be configured so that the user ID under which the LDAP Server runs can use ICSF services.

Other parts of the ICSF book may be useful for general background information about ICSF and Cryptographic Keys.

Chapter 4. Migrating

This chapter provides important migration information. Table 1 shows which sections in this chapter apply to your LDAP Server installation.

Table 1. Migrating Your LDAP Server

Migrating From	Sections to Read
Release 5	"Migrating from Release 5" "Migrating from Release 5 or 6" on page 22 "Migrating to Release 8" on page 24
Release 6	"Migrating from Release 5 or 6" on page 22 "Migrating to Release 8" on page 24
Release 7	"Migrating to Release 8" on page 24

Migrating from Release 5

This section contains additional information that pertains to your installation if you are migrating from Release 5 of the LDAP Server to Release 8.

Moving Configuration Files

You need to move the following SLAPD configuration files to the **/etc/ldap** directory from the **/etc** directory:

slapd.conf
slapd.at.system
slapd.at.conf
slapd.oc.system
slapd.oc.conf
slapd.envvars

The following command moves all of these files:

```
cp /etc/slapd.* /etc/ldap/
```

Changing Configuration File Include Statements

Be sure to change the path of the **include** statements in your configuration files from the **/etc** directory to the **/etc/ldap** directory. For example, in the **slapd.conf** file the **include** statements should be changed to:

```
include   /etc/ldap/slapd.at.system
include   /etc/ldap/slapd.oc.system
```

Also, the **slapd.conf** configuration file was changed in Release 6 to add the **include** statements that were previously in the **slapd.at.system** and **slapd.oc.system** files. Figure 28 on page 241 shows the file with these changes. You can leave the **include** statements in the **slapd.oc.system** and **slapd.at.system** files or you can move the **include** statements to the **slapd.conf** file. Either setup will work correctly.

The sample configuration files are shown in Appendix A, "Configuration Files" on page 241.

Migrating from Release 5 or 6

| This section contains additional information that pertains to your installation if you are migrating from
| Release 5 or 6 of the LDAP Server to Release 8.

SLAPD and Libraries Only in PDS

| Beginning in Release 7, SLAPD and its libraries reside only in PDS. It becomes necessary to set **STEPLIB** or place the *GLDHLQ*.SGLDLNK data set in LINKLIST or LPA to run SLAPD or other tools.

Using SSL

Beginning in Release 7, the LDAP Server relies on OS/390 Cryptographic Services System SSL to provide Secure Socket Layer (SSL) technology. If you want to use SSL, OS/390 Cryptographic Services System SSL must be installed and its libraries must be made available to SLAPD and other LDAP programs using STEPLIB, LPALIB, or LINKLIST. See the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for more information.

| With Release 8, all of your certificates should be in the same key database file. If you want to run a server to accept connections over SSL and also replicate to a replica server over SSL, you must put all of your certificates in one key database file.

| With Release 8, the **replKeyRingFile** and **replKeyRingPW** configuration file options are no longer necessary or evaluated by the LDAP Server. These options should be removed from the configuration file. Use the **sslKeyRingFile** and **sslKeyRingFilePW** options to specify the key database file and password, respectively.

Migrating Your MKKF Key Ring Files

If you have existing MKKF-generated key ring files from your Release 6 LDAP Server, you need to convert them to key database format using the **gskkyman** utility. See the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference*, for instructions on how to convert these files.

Preparing an Existing DB2 Database for Multi-server

If you have an existing Release 6 LDAP Server DB2 database and you are going to run in multi-server mode with or without dynamic workload management enabled, you must follow the steps outlined in the **ldapspfi.spufi.migrate** file shown below to migrate your database to the Release 8 schema.

```
--//*************************************************************//
--//*
--//* Licensed Materials - Property of IBM
--//* 5647-A01
--//* (C) Copyright IBM Corp.  1997, 1998
--//*
--//*************************************************************//

-- Use the following statements to migrate your EXISTING Release 5 or
-- Release 6 LDAP Server DB2 database to the Release 7 schema.
-- After running this script in SPUFI you should be able to run the
-- Release 7 LDAP Server using your newly-migrated database, with no
-- loss of existing data.  The newly-created tablespace name will
-- be used to update the database section of the LDAP Server
-- configuration file.  Additional migration steps may be needed for
-- later releases of the LDAP server.  Refer to the latest copy
-- of the OS/390 Security Server LDAP Server Administration and
-- Usage Guide for complete migration instructions for each LDAP
-- release.
--
-- First, to determine whether you need to migrate your existing
-- database, uncomment the SQL query following this paragraph
-- and run the query under SPUFI. Change the uuuuuuuu to the
-- name of the user which created your original tables. This
-- will be the name associated with "dbuserid" in your server
-- configuration file.  If the result of the query
-- is '5.0', you must uncomment and run the remainder of this SPUFI
-- script; if the result of the query is '5.1' or higher, your
-- database has already been migrated and no further changes are
-- needed.
--
-- select DB_VERSION from uuuuuuuu.LDAP_NEXT_EID;
--
-- If the result of the preceding query was '5.0', uncomment the
-- remaining statement in this script, change the dddddddd to the
-- name of the database in which your LDAP tables reside.  This
-- will be the name associated with "databasename" in your server
-- configuration file. Change tttttttt to a name of your choosing
-- which will be assigned to the newly-created tablespace.
-- This name will be associated with the "tbspacemutex" keyword
-- in the database section of the server configuration file.
-- Run it under SPUFI.  Note that this statement MUST be run by the
-- userid used to create your database originally.
-- This will be the name associated with the "dbuserid" in your
-- server configuration file (slapd.conf).
--
-- create tablespace tttttttt in dddddddd locksize tablespace
--              bufferpool BP0;
--
--
```

Figure 4. ldapspfi.spufi.migrate File

Note that the lines associated with the new tablespace should be merged into any existing SPUFI scripts for LDAP databases.

Also note that replication is not supported in multi-server mode.

Using Referrals with Multi-server: Within a server migrating from single-server mode to multi-server mode, the default referral or referrals defined in the configuration file and the referral objects defined in the RDBM backend do not need to change. In other servers that presently point to this server, the default referral objects should have their *host*:*port* updated. See “Operating in Multi-server Mode Without Dynamic Workload Management Enabled” on page 48 “Operating in Multi-server Mode With Dynamic Workload Management Enabled” on page 49 for more information.

Migrating to Release 8

The information in this section pertains to your installation if you are migrating from Release 5, 6, or 7 of the LDAP Server to Release 8.

Coexistence and Migration with Previous Releases

With OS/390 Release 8 and support for the LDAP Version 3 protocol, the LDAP Server now contains the ability to store and retrieve string-syntax attribute types which contain values that span the full range of UTF-8 characters. Thus, international characters can be used in the formulation of distinguished names and in attribute values for attributes which have **caseignorestring** or **caseexactstring** syntax. In order to provide support for the full UTF-8 character set, a number of changes have been necessary in the use and format of data stored in the underlying DB2 tables which are used and managed by the LDAP Server. The changes are localized to attribute type values and distinguished names which have been entered into the LDAP Server using the LDAP Version 2 protocol and contain characters outside of the IA5 character set.

The IA5 character set is defined to consist of 128 single-byte characters in the range X'00'-X'7F'. IA5 is commonly referred to as “7-bit ASCII”. In the range X'00' - X'7F', the ISO8859-1 character set is identical to IA5. In fact, the UTF-8 character set is identical to IA5 for the range of values represented by IA5. The format of data transferred over the LDAP Version 2 protocol was not as well-defined as for the LDAP Version 3 protocol. Indeed, for string-syntax attribute types, use of characters outside of the IA5 character set was undefined.

In previous releases of the OS/390 Security Server LDAP Server, ISO8859-1 was used to perform ASCII/EBCDIC conversions on data transferred to the LDAP Server from LDAP clients. For data that was constrained to the IA5 character set, the data was merely represented in the local codepage of the LDAP Server, stored into DB2 tables, and retrieved on requests. No filtering was performed by the LDAP Servers to limit incoming data to only the IA5 character set. As such, it is possible that existing DB2 tables contain distinguished names and attribute values which contain characters outside of the IA5 character set (that is, incoming strings sent to the LDAP Server contained character data in the range X'80' - X'FF').

With the introduction of support for LDAP Version 3, the format of data for character data unrepresentable by the IA5 character set is now well-defined. However, while the LDAP Server made the assumption, by using ISO8859-1 for ASCII/EBCDIC conversion, that incoming data over the LDAP Version 2 protocol was ISO8859-1, it may or may not have been. We have found that many LDAP clients now send UTF-8 data over the LDAP Version 2 protocol.

Limiting the Format of Incoming LDAP Version 2 Protocol Data: In order to be sure to represent incoming information over both the LDAP Version 2 and LDAP Version 3 protocols in their intended format, and support the coexistence of LDAP Servers running in the sysplex environment, we have found it necessary to now limit LDAP Version 2 strings to the IA5 character set. In order to allow this to be done, an APAR is available and required for coexistence with the OS/390 Release 8 LDAP Server. See the *OS/390 Program Directory* for APAR information.

The compatibility APAR, by default, limits the format of incoming string data sent over the LDAP Version 2 protocol to the IA5 character set. By setting configuration option **validateincomingV2strings** to **no**, this data filtering can be disabled. However, it is not recommended that the option be disabled since moving to LDAP Version 3 support at a later date could be made more difficult if the data filtering is turned off.

In order to operate the OS/390 Release 8 LDAP Server in the same sysplex with the OS/390 Release 7 LDAP Server, where the two servers are exploiting DB2 data sharing to operate against the same DB2 tables, the OS/390 Release 7 APAR **must** be applied. In addition, in order for OS/390 R7 LDAP Servers and OS/390 R8 LDAP Servers to access and update the same DB2 tables, the data filtering of incoming data sent over the LDAP Version 2 protocol must be enabled. This will be the default behavior with the compatibility APAR applied. When running in this environment, it is possible for UTF-8 encoded information to be stored in the LDAP directory. This can be done using the LDAP Version 3 protocol and communicating with the OS/390 R8 LDAP Server. Thus, information can be stored into the LDAP directory that is outside the IA5 character set, but only over the LDAP Version 3 protocol. When this situation exists, the OS/390 R8 LDAP Server must be able to send these attribute values in some format to LDAP Version 2 clients. An additional configuration option, **sendV3stringsoverV2as**, which has possible values **ISO8859-1** or **UTF-8**, can be used to indicate which output format to use when sending this information over the LDAP Version 2 protocol.

Converting Existing Data Outside the IA5 Character Set: In order to support the storage and retrieval of information over the LDAP Version 3 protocol which contains UTF-8 characters outside of the IA5 character set, it may be necessary to unload the information contained in the directory, possibly modify some of the attribute values and distinguished names which were contained in the directory, rebuild the DB2 tables used to store the information, and reload the DB2 tables. These migration steps will be necessary if data has been stored into the directory that is outside of the IA5 character set.

To determine if table migration is necessary, the OS/390 Release 8 **db2ldif** program has been modified to evaluate character string data as it is read from the DB2 tables and placed into LDAP Data Interchange Format (LDIF). For data that it finds that is outside of the IA5 character set, comments will be placed in the LDIF file indicating the nature of the data that was found. (Refer to “db2ldif Program” on page 81 for details on these comment formats.) Because of the differing behavior of various LDAP clients which support the LDAP Version 2 protocol, the data may have been entered in UTF-8, ISO8859-1, or possibly even the local character set of the client application at the time the data was entered. The LDIF file will contain comments with our best interpretation of the data and a candidate, UTF-8 encoded, distinguished name or attribute value. If the name or value chosen by the **db2ldif** program is acceptable, the LDIF file can remain unchanged for that entry. If a change is necessary, the LDIF format of the attribute value should be corrected prior to reloading the DB2 tables from the LDIF file.

If the **db2ldif** program found no data which requires modification (refer to “db2ldif Program” on page 81 for details on interpreting the output for migration purposes), then migration of the DB2 tables can be performed using the following SQL operation entered through SPUFI:

```
UPDATE USERID.LDAP_NEXT_EID SET DB_VERSION = '8.0';
```

where *USERID* is the user ID specified by the **dbuserid** configuration file option found in the configuration file used by the LDAP Server.

Note: Once the SPUFI script above has been run, the OS/390 Release 7 LDAP Server will always run with **validateincomingV2strings** set to on. This implies that data sent over the LDAP Version 2 protocol

| and handled by an OS/390 Release 7 LDAP Server which is sharing the DB2 tables with an OS/390 Release 8 LDAP Server will be limited to the IA5 character set. The limitation of IA5 data over LDAP Version 2 protocols can be lifted once all LDAP Servers in the sysplex are running at the OS/390 Release 8 level. By default, however, all LDAP Version 2 string data will be limited to IA5 strings.

| If the **db2ldif** program found data which requires modification, the DB2 tables must be migrated. The OS/390 Release 8 LDAP Server can be run with the existing DB2 tables prior to migrating the tables. There is a limitation that if distinguished names contain characters outside of the IA5 character set, the OS/390 Release 8 LDAP Server will be unable to operate against these entries. Coexistence of OS/390 R8 LDAP Servers and OS/390 Release 7 LDAP Servers sharing DB2 tables in a sysplex is limited to supporting the IA5 character set for distinguished names and character string attribute types sent and retrieved over the LDAP Version 2 protocol.

| **Migrating the DB2 Tables to the New Format:** To migrate the DB2 tables to the new database format if data was found by the **db2ldif** command that requires modification, follow these steps:

| 1. Create a backup of the directory data using native DB2 utilities.
| 2. Unload the directory information using **db2ldif** and specifying the distinguished name of the root of the directory tree that is stored in DB2. As an example:

   ```
   db2ldif -f /etc/ldap/slapd.conf  -o /tmp/directorydata.unload -s "o=IBM, c=US"
   ```

   assuming this command was run from the OS/390 shell prompt.
| 3. Evaluate all comments (lines beginning with a pound sign (#)) that begin with AF, AP, DF, or DP in the LDIF file created in the previous step. If the attribute value or distinguished name described in this comment is acceptable to you, no further updates are necessary for that attribute value. If the attribute value is not acceptable to you, modify the uncommented LDIF line below the comments to contain an attribute value acceptable to you. (Refer to “db2ldif Program” on page 81 command for details on interpreting the output for migration purposes.)
| 4. Remove the old DB2 tables. To do this, you must remove the DB2 database by running the following SPUFI command:

   ```
   DROP DATABASE databasename;
   ```

   where *databasename* is the name of the original database used to hold the original DB2 tables.
| 5. Re-create the database and table spaces in DB2. Use the SPUFI script defined in Figure 3 on page 15 to re-create the database and table spaces. Be sure to modify the **create tablespace** command as shown in “Increasing Your Table Space Size” on page 63 prior to running the SPUFI script.
| 6. Adjust the **slapd.conf** configuration file or files, if necessary. If you are using Release 7 and Release 8 and data sharing, the Release 7 LDAP Server's configuration file may also need to be updated.
| 7. Reload the directory information into the DB2 tables using the **ldif2db** command. An example of invoking **ldif2db** is:

   ```
   ldif2db -f /etc/ldap/slapd.conf -i /tmp/directorydata.unload
   ```

   Note: The **ldif2db** command sets the **DB_VERSION** value in the LDAP_NEXT_EID table to `'8.0'`.

| With the DB2 tables recreated in this way, both the OS/390 Release 7 and OS/390 Release 8 LDAP Servers can continue to share the DB2 tables. However, the OS/390 Release 7 LDAP Server will always operate with **validateincomingV2strings** set to **yes**. This will ensure that all data stored into the DB2 tables is in UTF-8 format and thus usable over both the LDAP Version 2 and LDAP Version 3 protocols.

| Once all LDAP Servers that are sharing the DB2 tables are running at the OS/390 Release 8 level, **validateincomingV2strings** can be set to **off** in the LDAP Server configuration file. All incoming strings

sent over the LDAP Version 2 protocol will be interpreted as UTF-8. Note that this is a change from previous versions of the LDAP Server which assumed ISO8859-1. Previous versions of the LDAP client for OS/390 also assumed ISO8859-1 when communicating over the LDAP Version 2 protocol. An APAR is available for the OS/390 LDAP client to allow the client to use UTF-8 encodings for data sent over the LDAP Version 2 protocol. Refer to the *OS/390 Program Directory* for APAR information.

In order to send and receive information over the LDAP Version 2 protocol in UTF-8 format between the OS/390 LDAP client and the OS/390 Release 8 LDAP Server, the APAR must be applied. In addition, **validateincomingV2strings** must be set to no in the OS/390 Release 8 LDAP Server. Refer to the notes earlier in this section which indicate that this is not supported if the OS/390 Release 8 LDAP Server is sharing DB2 tables with an OS/390 Release 7 LDAP Server.

Using the New LDAP Schema

Beginning with OS/390 Release 8, a greatly enhanced set of LDAP schema is shipped along with the LDAP Server. While the schema defined for an LDAP Server has always been extendable, it is sometimes easier to use attribute type and object class definitions which are already available in the directory service. The new schema files are not placed into the environment by default as using them causes extra DB2 operations to be performed during start up of the server which may be incompatible with existing production environments. The new schema files are, however, installed into the installation directories of the OS/390 LDAP Server installation for customer use. The new schema files are in the **/usr/lpp/ldap/etc** directory and are named **schema.***.

Before enabling the new schema files, you should determine if any of the attribute types or object classes that are noted in the files as requiring migration are used in data that is already stored in your directory service. The following table indicates the attribute types and object classes which will require migration if entries exist in your directory which use these definitions:

Table 2. LDAP Schema File Changes

Object Class or Attribute Type	Change
oc: document	lastModifiedBy deleted lastModifiedTime deleted
oc: dSA	presentationAddress deleted cn deleted
oc: residentialPerson	l (locality) moved from **allows** to **requires**
oc: organizationalPerson	userPassword deleted
oc: pilotObject	lastModifiedBy deleted lastModifiedTime deleted
at: cn	length changed to 256 (was 128)
at: generationQualifier	length changed to 20 (was 10)
at: houseIdentifier at: knowledgeInformation	length changed to 32700 (was 32768)
at: labeledURI	length changed to 32700 (was 100)
at: registeredAddress	length changed to 5000 (was 500)
at: lastModifiedTime	length changed to 30 (was 20)

When looking for these definitions in the schema files, look for the following comments in the file:

```
# NOTE: Before using this attribute, See the Migration
# section of the documentation.
```

The `generalizedTime`, `boolean`, and `integer` syntaxes are not supported by name in the new schema definitions. Instead, `cis 30`, `cis 5`, and `cis 11`, respectively, should be used. Use Table 3 on page 28 to determine how to specify attribute type definitions in the new schema files.

Table 3. OS/390 Release 8 Attribute Configuration Line Values

RFC 2252 Syntax Name	Syntax	Length
generalizedTime	cis	30
boolean	cis	5
integer	cis	11

Note that the new schema files represent a superset of what has been developed and shipped with the current LDAP Server on OS/390. Thus, once the new files are used, the old attribute type and object class definition configuration files must not be included in the configuration. To use the new schema files, copy the files to **/etc/ldap** and change the **include** lines in the main configuration file **slapd.conf** to:

```
include /etc/ldap/schema.system.at
include /etc/ldap/schema.IBM.at
include /etc/ldap/schema.user.at
include /etc/ldap/schema.system.oc
include /etc/ldap/schema.IBM.oc
include /etc/ldap/schema.user.oc
```

Note: The original schema shipped with the LDAP Server results in the creation of approximately 100 DB2 tables. The new schema definition files result in the creation of approximately 600 tables. You must ensure that DB2 is configured to allow over 600 DB2 tables to be created and used before you attempt to use the new schema files. See “Increasing Your Table Space Size” on page 63 for information on adjusting your table space size.

Updating Existing Directory Information

If existing directory information makes use of these attribute types and object classes, there are two options for resolution.

1. Continue to work with the existing definitions for these attribute types and object classes. If this option is chosen, the **schema.*** files, as shipped, can be used. The definitions of object classes and attribute types in the **schema.oc.*** and **schema.at.*** files will use the old definitions by default.
2. Unload the directory information, adjust the schema definitions, remove the existing DB2 tables, and reload the directory information. In order to do this, follow these steps:
   a. Create a backup of the directory data using native DB2 utilities.
   b. Unload the directory information using **db2ldif** and specifying the distinguished name of the root of the directory tree that is stored in DB2. As an example:

      ```
      db2ldif -f /etc/ldap/slapd.conf  -o /tmp/directorydata.unload -s "o=IBM, c=US"
      ```

      assuming this command was run from the OS/390 shell prompt.
   c. Adjust the schema files by finding all the noted object classes and attribute types in the **schema.*.*** files, adding comment characters (pound sign (#)) to the existing definitions, and removing the comment characters that precede the new definitions.
   d. Remove the old DB2 tables. To do this, you must remove the DB2 database by running the following SPUFI command:

      ```
      DROP DATABASE databasename;
      ```

      where *databasename* is the name of the original database used to hold the original DB2 tables.

e. Re-create the database and table spaces in DB2. Use the SPUFI script defined in Figure 3 on page 15 to re-create the database and table spaces. Be sure to modify the **create tablespace** command as shown in "Increasing Your Table Space Size" on page 63 prior to running the SPUFI script.

f. Adjust the **slapd.conf** configuration file to use the new schema files. The schema files are included in the configuration using the **include** configuration option.

g. Reload the directory information into the DB2 tables using the **ldif2db** command.

   An example of invoking **ldif2db** is:

   ```
   ldif2db -f /etc/ldap/slapd.conf -i /tmp/directorydata.unload
   ```

   Note: The **ldif2db** command sets the **DB_VERSION** value in the LDAP_NEXT_EID table to '8.0'.

As an alternative to the two steps above, a partial move to the new schema definitions can be done. If an object class is not used to describe any entries in the directory, then the new object class definition can be enabled. In addition, if there is information in the directory that uses these object class definitions but the current information conforms to the new schema definition, then the new object class definition can be enabled. This latter situation can happen when optional attributes listed in the new schema definition are not contained in the existing entry data in the directory. In order to use any of the attribute type changes defined in the new schema, the directory must be unloaded, the DB2 tables recreated, and then reloaded.

Using GLDCLDAP.x and GLDCLDAP

Release 7 of the LDAP Server was the last release in which **EUVCLDAP.x**, **EUVCLDAP**, **ldap.x**, and **ldap.dll** were available. Applications should use **GLDCLDAP.x** (equivalent to **EUVCLDAP.x** and **ldap.x**) and **GLDCLDAP** (equivalent to **EUVCLDAP** and **ldap.dll**) during link-edit and at run time.

Password Encryption

There are three ways that an administrator can update the RDBM backend to take advantage of password encryption. The first option is the preferred way, especially if your database is large.

1. Update the configuration file with **pwEncryption** and a valid value, start the LDAP Server, and then run **db2pwden**. This reads all the **userPassword** attribute values, encrypts all clear text **userPassword** attribute values, and stores them into the RDBM backend.
2. Update the configuration file with **pwEncryption** and a valid value, start the LDAP Server, and then let the passwords be updated as modify operations are processed in order to reset or add **userPassword** attribute values. This will slowly change clear text **userPassword** attribute values in the RDBM backend to encrypted values.
3. Update the configuration file with **pwEncryption** and a valid value. Then, run **db2ldif** to unload the RDBM backend and run **ldif2db** to reload the RDBM backend. Do not use the **-t** option of **db2ldif**.

If the LDAP Server is running in a sysplex, then all LDAP Servers must have the R8 PTF for APAR OW41326 applied or **pwEncryption** cannot be used.

LDAP Servers without this Release 8 PTF applied can only support clear text passwords correctly (such as for bind purposes).

LDAP Servers with this Release 8 PTF applied have a limitation on clear text passwords that start with *tag*. A *tag* in this context is defined as a string of characters that starts with a left brace ({) and ends with a right brace (}). This includes all the algorithm tags including {none}. These clear text passwords are interpreted by the LDAP Server to be tagged, and as such are stored in DB2 as if already encrypted.

Since binds are not accepted from a client with a password value that starts with *tag*, these password values are useless. Therefore, clear text user passwords should not begin with *tag*. Additional tags may be defined in the future. It is recommended that no passwords be defined which begin with a left brace ({) and also contain a right brace (}).

Withdrawal of Support for inheritOnCreate Attribute

Support for the **inheritOnCreate** attribute has been withdrawn from the LDAP Server in Release 8. Operations against the RDBM database which attempt to add or modify the **inheritOnCreate** attribute will ignore such additions or changes for this attribute. Also, entry object additions to the RDBM database will ignore the **inheritOnCreate** attribute when determining whether the bound user is authorized to perform the requested operation. While the database column pertaining to **inheritOnCreate** remains defined for the database schema for backward compatibility with previous releases of the LDAP Server, they will no longer be used in Release 8. LDIF-format files which are used as input to the **ldif2db** utility program may contain instances of the **inheritOnCreate** attribute, but that attribute will be ignored when loading the entry into the RDBM database.

Replication

Beginning with Release 8 of the OS/390 LDAP Server, the LDAP Version 3 protocol is supported. When running this release of the LDAP Server as a replication master, all replicated updates will be attempted using the LDAP Version 3 protocol. If the replica server does not support the Version 3 protocol, the master server will retry the update operation using the Version 2 protocol. For Version 3 specific update operations (for example, updates to referral objects requiring the Version 3 **manageDsaIT** control), a Version 3 master replicating to a Version 2 replica will fail and the master/replica will then be out of sync. Therefore, it is recommended that all replica servers be upgraded to a Version 3 capable release prior to upgrading the master server.

ldif.h Code is Replaced

The **ldif.h** example code in **/usr/lpp/ldap/examples** has been replaced with **line64.h**.

Chapter 5. Configuring

Note: If you already have the LDAP Server installed, see Chapter 4, “Migrating” on page 21.

Once the software has been installed, you are ready to configure it for use at your site. The LDAP server may be configured to run in one of several operational modes:

- **Single-server mode**

  In this operational mode, only a single instance of the LDAP server may use a given RDBM database to store directory data. This server may perform replication (see Chapter 14, “Replication” on page 169) of database changes to other servers (on the same host system or on another host system) which store data in physically distinct RDBM databases. Multiple concurrent instances of the LDAP server may run in single-server mode on the same host system **only** if they operate against different RDBM databases.

  See “Operating in Single-server Mode” on page 46 for more information.

- **Multi-server mode without dynamic workload management enabled**

  In this operational mode, multiple concurrent instances of the LDAP server using the same database to store directory data may run on a given host system, as well as on different host systems when those hosts are coupled in a Parallel Sysplex. A *Parallel Sysplex* is a collection of MVS systems that cooperate, using certain hardware and software products, to process work. A Parallel Sysplex enables high-performance, multisystem data sharing across multiple Central Processor Complexes and OS/390 images, as well as dynamic workload balancing across constituent systems in the sysplex. For additional information, see *Parallel Sysplex Overview: Introducing Data Sharing and Parallelism in a Sysplex*, GC28-1860.

  Multi-server mode is intended for use in an environment where high transactional volume is common, or where maximum availability is required. This mode provides benefits of improved availability, fault tolerance, improved resource utilization, and improved performance. These benefits are achieved by enabling concurrent running of multiple servers which are functionally equivalent and which provide access to the same LDAP data.

  Note: In this operational mode, replication of RDBM database changes to other servers is not supported.

  See “Operating in Multi-server Mode Without Dynamic Workload Management Enabled” on page 48 for more information.

- **Multi-server mode with dynamic workload management enabled**

  This operational mode augments the benefits of the previously described mode with dynamic workload management. This mode may only be used when all host systems on which instances of the LDAP server, all using the same RDBM database, are coupled in the same Parallel Sysplex. The dynamic workload management for LDAP servers is provided through the use of an OS/390 TCP/IP feature called “connection optimization”. Connection optimization uses Domain Name Services (DNS) for distributing connections among server applications within a sysplex domain. Connection optimization achieves workload balancing by distributing connections to systems with the most available resources and by avoiding unavailable sysplex resources. See *OS/390 TCP/IP Update Guide*, GC31-8553, for information on connection optimization.

  Note: In this operational mode, replication of RDBM database changes to other servers is not supported.

  See “Operating in Multi-server Mode With Dynamic Workload Management Enabled” on page 49 for more information.

In any of these modes, combinations of both RDBM (DB2) and SDBM (RACF) backends are possible, including:

- RDBM only
- SDBM only
- RDBM and SDBM

Note that in any of these configurations, the configuration backend, CDBM, is always loaded and available.

All LDAP Server runtime configuration is accomplished through the configuration file **slapd.conf**, installed in the **/etc/ldap** directory. If this is your first time installing the LDAP Server, create a new copy of **slapd.conf** with:

```
cp /usr/lpp/ldap/etc/slapd.conf /etc/ldap/slapd.conf
```

and edit **/etc/ldap/slapd.conf**.

An alternate configuration file can be specified through a command-line option to the LDAP Server (SLAPD) and other LDAP programs.

The initial configuration contains default versions of:

- Object class definitions
- Attribute definitions
- Configuration settings

It does not contain a database suffix.

This chapter describes the general format of the configuration file and a detailed description of each configuration file option. These descriptions are followed by sections describing configuration specifics, dependencies, compatibility issues, and restrictions of each of the three operating modes outlined above.Appendix A, “Configuration Files” on page 241 contains all of the LDAP Server configuration files.

Configuration File Format

The **slapd.conf** file consists of a series of global configuration options that apply to SLAPD as a whole (including all backends), followed by zero or more database backend definitions that contain information specific to a backend instance. The configuration file or files must be in code page IBM-1047.

For options that appear more than once, the last appearance in the **slapd.conf** file is used. Blank lines and comment lines beginning with the pound sign character (#) are ignored. If a line begins with one or more blank spaces, it is considered a continuation of the previous line. Figure 5 shows the general format of **slapd.conf**.

```
# comment - these options apply to every database
<global configuration options>

# database definition and configuration options
database <backend type> <load library>
<configuration options specific to backend>

# subsequent database definitions and configuration options
```

Figure 5. General Format of slapd.conf

Configuration line arguments are separated by blank spaces. If an argument contains one or more blank spaces, the argument should be enclosed in double quotation marks (for example, `"argument one"`). If an argument contains a double quotation mark or a backslash character (\), the double quotation mark or backslash character should be preceded by a backslash character (\).

A sample configuration file, **slapd.conf** (shown in Figure 28 on page 241), is provided and is installed in the **/usr/lpp/ldap/etc** directory.

Configuration File Global Options

Options described in this section apply to all backends.

adminDN *dn*

The distinguished name (DN) of the administrator for this LDAP Server. This DN will have unrestricted access to all entries in the local directory. The name that is chosen should be descriptive of the person or group of people that will have knowledge of and administer the LDAP Server. The format of the name must be in DN format which is described in Chapter 10, "Distinguished Names" on page 143. It is recommended, though not necessary, that the DN have the same suffix as one of the **suffix** option values in the configuration file.

"Establishing the Administrator DN and Password" on page 43 describes how to set up your administrator DN.

With LDAP V3 support, UTF-8 characters can be used for textual attributes stored in the directory. It is also desirable to allow any UTF-8 character to appear in distinguished names, and in particular, the **adminDN** distinguished name. Since the LDAP configuration files are defined to hold information in only the IBM-1047 character set, a solution is required to allow entering distinguished names into the configuration file which contain UTF-8 characters but only using the IBM-1047 character set. To solve this problem, an escape mechanism has been introduced for purposes of entering either the **adminDN** or the **masterServerDN**. This escape mechanism allows the entry of UTF-8 characters while keeping the input string value to within the IBM-1047 character set. The escape mechanism employed requires that you express UTF-8 characters which are not within the X'00' - X'7F' range (7-bit ASCII which is the single-byte form of UTF-8 characters) in the form of a set of four character representations. This representation has the form "&*nmm*" where $0 \le n \le 3$ and $0 \le m \le 7$. You might recognize *nmm* as being an octal value for a byte of information. Thus, if you want to create an **adminDN** which was the following distinguished name:

```
cn=Peter <U umlaut>nger, o=Widgets, c=DE
```

enter the **adminDN** into this option value as:

```
cn=Peter &nmm&nmmnger, o=Widgets, c=DE
```

Because the <U umlaut> is not within the 7-bit ASCII range, the value must be escaped to the octal representation of the UTF-8 multi-byte character. In the case of <U umlaut>, the Unicode code point is X'00DC'. Converted to UTF-8, this character is a multi-byte sequence: X'C3BC'. (Refer to "UTF-8 Support" on page 137 for conversion information.) Converted to the escaped form for input into the **adminDN** field, this character is represented as "&303&234" since X'C3' is octal 303 and X'BC' is octal 234. Thus, the **adminDN** above would be entered as:

```
cn=Peter &303&234nger, o=Widgets, c=DE
```

If there is a case where you need to enter an **adminDN** string which contains the string "&*nmm*" where 0≤*n*≤3 and 0≤*m*≤7, then you must escape the ampersand by using its octal representation which is "&046".

adminPW *string*
The password of the administrator (**adminDN**) for this server.

"Establishing the Administrator DN and Password" on page 43 describes how to set up your administrator password.

Note: Use of the **adminPW** configuration option is strongly discouraged in production environments. Instead, specify your **adminDN** as the distinguished name of an existing entry in the directory information tree. This will eliminate passwords from the configuration file

altServer *ldap_URL*
Specifies an equivalent server to this LDAP Server. It may or may not be a replica, but should contain the same naming contexts. The alternate server is specified as an LDAP URL.

attribute *name* **[***name2* **...** *namen***] {bin|ces|cis|tel|dn}** *colname maxlen* **{normal|sensitive|critical}**
This option defines an attribute type. The definition specifies the *name* of the attribute and associates a syntax with it. One or more optional alternate names (*name2*...*namen*) can be given for the attribute. The possible syntaxes and their meanings are:

bin (binary)	Specifies binary.
ces (caseexactstring)	Specifies that the case must match during comparisons.
cis (caseignorestring)	Specifies that the case is ignored during comparisons.
tel (telephone)	Specifies that the case, blanks, and dashes (-) are ignored during comparisons.
dn	Specifies the distinguished name.

The *colname* is the desired database column name. Specify a maximum of 14 characters and be sure the column name is unique. The column name must begin with an alphabetic character.

In the case of the access to RACF support (see Chapter 12, "Accessing RACF Information" on page 149), the attribute types and object classes required for that support are defined. However, there is no need to create entries in the DB2-backed directory tree which use this information. Since it is undesirable to allocate DB2 resources which will never be used, the configuration processing code will accept a special value for the *colname* portion of the attribute configuration option to avoid allocating DB2 tables to handle the attribute. If **_nocreate** is used for the *colname* portion of the attribute configuration specification, then DB2 resources will not be allocated to handle this attribute as it will not appear in the DB2-backed directory tree.

The *maxlen* is a maximum length.

An access class of **normal**, **sensitive**, or **critical** must be specified. The access class of each attribute implies differing levels of sensitivity of the data in an attribute. See "Attribute Access Classes" on page 160 for more information about these access classes.

include *filename*
Specifies the path and file name of a file to be included as a part of the LDAP Server configuration. This will be used for inclusion of the schema definition, in a file such as **slapd.conf**.

maxConnections *int* Absolute maximum number of client connections which SLAPD should accept.

Default = operating system default.
Range = 1 - operating system maximum

The **maxConnections** option should use the same value as the **maxThreads** option. It should be set to the maximum number of expected concurrent client connections.

maxThreads *int* Absolute maximum number of threads which can be created by the SLAPD daemon for processing client requests.

Default = operating system maximum
Range = 10 - operating system maximum

The **maxThreads** option should use the same value as the **maxConnections** option.

objectclass *name* **[requires** *attrs***] [allows** *attrs***]**
This option defines an object class. The definition specifies the name of the object class and the rules for attribute content. The rules specify which attributes are required to be present in an entry with the given object class and additionally which attributes are allowed to be present. The *attrs* are specified as comma-separated attribute names. These names must have already been defined with attribute statements. Typically all object class statements come after all the attribute statements.

port *int* TCP/IP port used for non-SSL communications.

Default = 389
Range = 1 - 65535

If the **sysplexGroupName** and **sysplexServerName** options are present in the configuration file, the port number specified for this server instance must be the same as the port number specified for all other members of the same *group_name* in the sysplex for dynamic workload balancing to function properly. Note that the port number may be established in the configuration file, or it may be established using the optional startup parameter for port (see “Running the LDAP Server in the OS/390 Shell” on page 73).

It is advisable to reserve the port number chosen here in your TCP/IP profile data set. Consult the *TCP/IP for MVS: Customization and Administration Guide*, SC31-7134 for further information.

referral *ldap_URL* Specifies the referral to pass back when the local database cannot handle the request. It is also known as the default referral. The **referral** option can appear multiple times and should list equivalent servers.

securePort *int* TCP/IP port used for SSL communications.

Default = 636
Range = 0 - 65535

It is advisable to reserve the port number chosen here in your TCP/IP profile data set. Consult the *TCP/IP for MVS: Customization and Administration Guide*, SC31-7134 for further information.

security {ssl|sslonly|none|nossl}
Specifies what type of communications will be accepted. The **ssl** indicates that the server will listen on the SSL port as well as the non-SSL port. The **sslonly** option means that the server will listen only on the SSL port. The **none** or **nossl** indicates that the server will listen only on the non-SSL port.

Default = none

sendV3stringsoverV2as {UTF-8|ISO8859-1}
Specifies the output data format to use when sending UTF-8 information over the LDAP Version 2 protocol.

Default = UTF-8

See “Migrating to Release 8” on page 24 for more detailed information on the use of this setting.

sslAuth {serverAuth|serverClientAuth}
Specifies the SSL authentication method. The **serverAuth** method allows the LDAP client to validate the LDAP Server on the initial contact between the client and the server. The **serverAuth** method is the default.

The **serverClientAuth** method allows the LDAP client to validate the LDAP Server and the LDAP Server to validate the LDAP client if the client sends its digital certificate on the initial contact between the client and the server.

Note: To allow clients to SASL bind to the LDAP Server, it is necessary to configure the server with **sslAuth serverClientAuth**.

See “Securing Your LDAP Server with SSL” on page 65 for more SSL information.

sslCipherSpecs *int*
Specifies the SSL cipher specifications that will be accepted from clients. Table 4 shows a list of supported ciphers. Refer to the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for a description of supported cipher specifications.

Table 4. Supported Ciphers

Cipher	Hexadecimal value	Decimal value
SLAPD_SSL_RC4_MD5_US	0x0800	2048
SLAPD_SSL_TRIPLE_DES_SHA_US	0x0100	256
SLAPD_SSL_DES_SHA_US	0x0200	512
SLAPD_SSL_RC2_MD5_EXPORT	0x1000	4096
SLAPD_SSL_RC4_MD5_EXPORT	0x2000	8192
SLAPD_SSL_RC2_MD5_EXPORT	0x1000	4096
SLAPD_SSL_RC4_MD5_EXPORT	0x2000	8192

The integer value used with the **sslCipherSpecs** keyword is the decimal representation of the ORed bitmask defined by the hexadecimal values above. For example, to use only Triple DES, the value should be 256. In this case, only clients that also support Triple DES would be able to establish an SSL connection with the server. As another example, to use all the available ciphers, the value should be 12288. In this case, clients that also support these ciphers would be able to establish an SSL connection with the server.

See “Securing Your LDAP Server with SSL” on page 65 for more SSL information.

sslKeyRingFile *filename*
Specifies the path and file name of the SSL key database file for the server. The file name must match the key database file name using the **gskkyman** utility. If you have an existing MKKF-generated key ring file, you need to convert it to key database format using the **gskkyman** utility. See the

OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference for information about the **gskkyman** utility. Also, see "Securing Your LDAP Server with SSL" on page 65 for more SSL information.

The LDAP Server supports the use of a RACF key ring. Specify the RACF key ring name for the **sslKeyRingFile** and specify NULL for both the **sslKeyRingFilePW** and **sslKeyRingPWStashFile**. See "Support of RACF Key Rings" on page 70 for more information on using RACF key rings.

sslKeyRingFilePW *string*
Specifies the password protecting access to the SSL key database file. The password string must match the password to the key database file that was created using the **gskkyman** utility (see the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference*). Also, see "Securing Your LDAP Server with SSL" on page 65 for more SSL information.

Note: Use of the **sslKeyRingFilePW** configuration option is strongly discouraged. As an alternative, use either the RACF certificate management support or the **sslKeyRingPWStashFile** configuration option. This will eliminate this password from the configuration file.

sslKeyRingPWStashFile *filename*
Specifies a file name where the password for the server's key database file is stashed. If this option is present, then the password from this stash file overrides the **sslKeyRingFilePW** configuration option, if present. Use the **gskkyman** utility with the **-s** option to create a key database password stash file. See "Securing Your LDAP Server with SSL" on page 65 for more SSL information.

validateincomingV2strings {on|off}
Specifies whether the incoming strings are validated. If set to **on**, this setting limits the format of incoming string data sent over the LDAP Version 2 protocol to the IA5 character set (X'00'-X'7F' or "7-bit ASCII"). With this setting, textual data received on operations outside of the IA5 character set causes the operations to fail with **LDAP_PROTOCOL_ERROR**.

Default = on

Note that while supported, it is not recommended to run with this data filtering disabled. Refer to "Migrating to Release 8" on page 24 for more detailed information on the use of this setting.

verifySchema {on|off}
Specifies whether the schema, as defined by the **attribute** and **objectclass** lines in the configuration, is checked. The check ensures that all attributes referenced by object class settings are defined by attribute settings. Note that the check stops after encountering an error, therefore, only the first error encountered is reported.

Default = on

waitingThreads *int*
Number of threads which can be pooled to wait for incoming client requests.

Default = operating system maximum
Range = 10 - operating system maximum

Set **waitingThreads** to a number between 10 and the **maxThreads** setting. Keeping **waitingThreads** low frees up resources such as threads, some storage, and DB2 connections, during idle times, but costs more when the server starts to enter high volume periods since those resources must be acquired dynamically.

The **replKeyRingFile** and **replKeyRingPW** options are no longer necessary or evaluated by the LDAP Server. These options should be removed from the configuration file. Use the **sslKeyRingFile** option to specify the key database file and use the **sslKeyRingPWStashFile** configuration option or the RACF certificate management support for the password.

Configuration File Global Backend Options

The following options apply to all backends, unless specifically overridden in a backend definition. Specifying these prior to a **database** line in the configuration sets the option for all backends. Specifying it after a **database** line sets the option just for the backend defined by the **database** line.

sizeLimit *int*
: Specifies the maximum number of entries to return from a search operation, regardless of any size limit that may have been specified on the search request.

  0 = no limit
  Default = 500
  Range = 0 - 2147483647

timeLimit *int*
: Specifies the maximum number of seconds (in real time) SLAPD will spend answering a search request. If a request cannot be processed within this time, a result indicating an exceeded time limit is returned.

  0 = no limit
  Default = 3600
  Range = 0 - 2147483647

Note that the following behavior is used when referring to the **sizeLimit** and **timeLimit** parameters. These two parameters are also valid on an **ldapsearch** from the client.

- If a client has passed a limit, then the smaller value of the client value, and the value read from **slapd.conf** will be used.
- If the client has not passed a limit, and has bound as the **adminDN**, then the limit will be considered unlimited.
- If the client has not passed a limit, and has not bound as the **adminDN**, then the limit will be that which was read from the **slapd.conf** file.

Configuration File Backend Options

These options apply only to the backend in which they are defined.

database *dbtype dblibpath* Marks the beginning of a new database section.

- For *dbtype*:
  - IBM supports **rdbm** (DB2) and **sdbm** (RACF). The type **config** is reserved by the LDAP Server and should not appear as *dbtype* in your configuration files.
- For *dblibpath*:
  - This is the file name of the shared library (DLL) containing the backend database code.

extendedGroupSearching {on|off}
: Specifies whether a backend participates in extended group membership searching on a client bind request. If this option is **on**, group memberships are gathered from this backend in addition to the backend in which the bind

DN resides. If this option is **off**, group memberships are not gathered from this backend unless the bind DN resides in this backend. The default is **off**.

The group memberships gathered on a client's bind request are used for authorization checking of the client's request. The administrator should know in general which backends may contain group information so they can be marked for **extendedGroupSearching**. Group memberships are necessary for complete authorization checking of a client request.

The server control **authenticateOnly** is supported by the LDAP Server so that a client can override both **extendedGroupSearching** and group membership gathering from the backend where the DN resides. See Appendix F, "Supported Server Controls" on page 397 for more information.

suffix *dn_suffix* Denotes the root of a subtree in the namespace managed by this server within this backend. At least one suffix is required. SDBM specifies a format for this suffix. See Chapter 12, "Accessing RACF Information" on page 149.

Note that the SDBM backend requires only the **database** and **suffix** options.

Configuration File RDBM Backend Specific Options

These options must follow a **database rdbm** line and come before any other **database** *dbtype* line.

databasename *dbname* The name of the database this backend will use to store directory data.

dbuserid *userid* An OS/390 user ID that will be the owner of the tables.

dsnaoini *dataset* The name of the CLI Initialization sequential data set you created in step 4 on page 14. For example, if the CLI initialization data set name is `myuserid.dsnaoini`, then this would be specified as:

```
dsnaoini myuserid.dsnaoini
```

Refer to *DB2 for OS/390 Call Level Interface Guide and Reference* for information about the DB2 CLI initialization file.

index *attrlist* **[eq|none]** This option specifies the indexing for fast retrieval to be maintained for the specified attribute or attributes. The *attrlist* is a comma-separated list of attribute names. If only an *attrlist* is given, all possible indexes are maintained.

eq Specifies an index is used for these attributes.

none Specifies that no index is used for these attributes.

This option may appear multiple times in the RDBM database section. For example:

```
index cn none
index sn,uid eq
```

This example causes no indexes to be maintained for the **cn** attribute; an index for the **sn** and **uid** attributes; and no indexes for all other attributes.

Note: Specifying **index default** creates an index for all attributes and **index default none** creates no indexes. These two specifications are mutually exclusive.

masterServer *ldap_URL* Specifies the location of this replica's master server in LDAP URL format.

masterServerDN *dn*
Specifies the DN allowed to make changes to the replica. The format of the name must be in DN format which is described in Chapter 10, “Distinguished Names” on page 143. The presence of this option indicates that the server instance using this configuration file is a slave.

In order to enter characters in this distinguished name which are outside of the 7-bit ASCII range when expressed in Unicode (or UTF-8), then you must escape these characters. See the description under the **adminDN** configuration option (page 33) for details on how to do this.

masterServerPW *string*
Specifies the password for the **masterServerDN** that will be allowed to make updates. This option is only applicable for a slave SLAPD. See “Establishing the Administrator DN and Password” on page 43 for additional information about the master server password.

Note: Use of the **masterServerPW** configuration option is strongly discouraged in production environments. Instead, specify your **masterServerDN** as the distinguished name of an existing entry in the directory information tree. This will eliminate passwords from the configuration file.

pwEncryption {none|crypt|MD5|SHA|DES:*keylabel***}**
Specifies what encryption method to use when storing the **userPassword** attribute values in the RDBM backend of the directory.

none
Specifies no encryption. The **userPassword** attribute values are stored in clear text format. Note that these clear text passwords are stored in RDBM prefixed with the tag {none}.

crypt
Specifies that **userPassword** attribute values are encoded by the UNIX crypt hash algorithm before they are stored in the directory. These passwords are stored in RDBM prefixed with the tag {crypt}.

MD5
Specifies that **userPassword** attribute values are encoded by the MD5 hash algorithm using OCSF before they are stored in the directory. These passwords are stored in RDBM prefixed with the tag {MD5}.

SHA
Specifies that **userPassword** attribute values are encoded by the SHA hash algorithm using OCSF before they are stored in the directory. These passwords are stored in RDBM prefixed with the tag {SHA}.

DES:*keylabel*
Specifies that **userPassword** attribute values are encrypted by the DES algorithm using the specified key label using OCSF and ICSF before they are stored in the directory, and can be retrieved as part of an entry in the original clear text format. These passwords stored in RDBM are prefixed with the tag and key label {DES:*keylabel*}. Retrieval will continue to be limited by the access controls in effect on the entry that contains the **userPassword** attribute values.

The *keylabel* must refer to a valid data-encrypting key, also called a data key, generated by KGUP of ICSF and stored in the CKDS. The *keylabel* maximum length is 64 characters. See the information on managing cryptographic keys and using the Key Generator Utility Program in the *OS/390 ICSF Administrator's Guide* for instructions on how to generate, store into CKDS, and authorize a data key and key label for DES encryption. It is important to remember to refresh both CKDS and RACF after you enter and authorize a key.

Notes:

1. When an encrypted password is stored in the RDBM backend, it is prefixed with the appropriate encryption tag so that when a clear text password is sent on an LDAP API simple bind it can be encrypted in that same method for password verification.
2. The UNIX crypt() algorithm, implemented across all of the UNIX platforms, accepts only the first eight characters of a password. As a result, any password supplied on an **ldap_simple_bind** operation or **ldap_compare** operation that matches the first eight characters of a **userPassword** attribute value encrypted with the crypt() algorithm in the directory will match.
3. The values returned by the crypt() algorithm are not portable to other X/Open-conformant systems. This means that user password values encoded by the crypt() algorithm and unloaded as tagged output using **db2ldif -t** are not portable when loaded by another platform's load utility.

Default = none

If **pwEncryption** is not specified in the configuration file, no password encryption takes place. User passwords are stored in clear text format and no tag is prefixed.

multiserver {Y|y|N|n} Indicates the operating mode in which this server will run. Specifying either **y** or **Y** indicates the server runs in multi-server mode with or without dynamic workload management enabled (see page 31 for a description of multi-server operating modes). Specifying either **n** or **N** indicates the server runs in single-server mode.

If **n** or **N** is specified, and both **sysplexGroupName** and **sysplexServerName** options are present in the configuration file, the **multiserver** option value is overridden and the server operates in multi-server mode.

If **y** or **Y** is specified, the **tbspacemutex** option must also be present in the configuration file. If not specified, a server startup error message is displayed and the server stops. See “Configuration File RDBM Backend Specific Options” on page 39 for a description of the **tbspacemutex** option.

readOnly {on|off} Any attempt to modify the database will fail if **readOnly** is turned **on**.

Default = off

servername *string*
The name of the DB2 server location that manages the tables for the LDAP Server. This value must match the name of one of the `DATA SOURCE` stanzas that must be specified in the ODBC initialization data set which is specified by the **dsnaoini** option in the configuration file. See the *DB2 for OS/390 Call Level Interface Guide and Reference* for a description of the DSNAOINI ODBC initialization data set contents. Using the example DSNAOINI file in Figure 2 on page 14 the value of *string* for **servername** would be `yourdatasourcename`.

sysplexGroupName *group_name*
Specifies the name of the application group within which this server instance becomes a member for purposes of dynamic workload balancing. All concurrently-running LDAP servers which participate in dynamic workload balancing within the same Parallel Sysplex (see page 31 for the description of a Parallel Sysplex) must use the same *group_name* in their server configuration file. This name may be up to 18 characters in length, and must be unique among all application groups using Workload Manager services. When the **sysplexGroupName** option is present in the server configuration file, the **sysplexServerName** option must also be present among the configuration file global options. If the **sysplexGroupName** option is present, the **tbspacemutex** option must also be present in the configuration file. See "Configuration File RDBM Backend Specific Options" on page 39 for a description of the **tbspacemutex** option.

sysplexServerName *server_name*
Specifies the name of the server within the *group_name* sysplex group which participates in dynamic workload balancing. (See page 31 for the description of a sysplex.) All concurrently-running LDAP servers which participate in dynamic workload balancing within the same Parallel Sysplex are identified by *server_name*s within a given *group_name* used in the sysplex. This name may be up to 8 characters in length, and must be unique within a given *group_name* using Workload Manager services. When the **sysplexServerName** option is present in the server configuration file, the **sysplexGroupName** option must also be present among the configuration file global options. If the **sysplexServerName** option is present, the **tbspacemutex** option must also be present in the configuration file. See "Configuration File RDBM Backend Specific Options" on page 39 for a description of the **tbspacemutex** option.

tbspaceentry *name*
The partitioned table space name which was used when creating the LDAP entry table.

tbspacemutex *name*
Specifies the nonsegmented table space name that was used when creating the LDAP 4K mutex table space. (Figure 7 on page 53 shows an example that specifies the **tbspacemutex** option.)

tbspace32k *name*
The segmented table space name that was used when creating the LDAP 32K tables.

tbspace4k *name*
The segmented table space name that was used when creating the LDAP 4K tables.

The **sysplexGroupName** and **sysplexServerName** keywords are corequisites, and sysplex Workload Management features will only be enabled if both are present with non-null arguments. When sysplex Workload Management features are enabled, the server is automatically assumed to be in multiserver mode, and replication will be disabled in this server.

The **multiserver** keyword may be present without the **sysplexGroupName** and **sysplexServerName** keywords, in which case replication is disabled in the server and sysplex Workload Management features are also disabled.

If **sysplexGroupName**, **sysplexServerName**, and **multiserver** keywords are all omitted from the server configuration file, the server will operate in single-server mode and replication will be enabled.

Establishing the Administrator DN and Password

There are three ways that the administrator DN and password or the master server DN and password can be configured. One of the three ways must be used, since an administrator DN and password are required for the LDAP Server and some other LDAP programs to operate. The administrator DN must be present in the configuration file using the **adminDN** option (see page 33). The administrator DN password can optionally (this is not recommended) be placed in the configuration file using the **adminPW** option (see page 34) or can be held in the namespace managed by this instance of the LDAP Server. If a replica is being established, the **masterServerDN** option must be present in the configuration file. The **masterServerPW** option can optionally be present. (This is also not recommended.) All of the options described below are applicable for either **adminDN** or **masterServerDN**.

- Administrator DN and password in configuration file

  The simplest but least secure method is to select an administrator DN that is outside of the scope of suffixes managed by this server (see the **suffix** option on page 39). In other words, choose an administrator DN such that it does not fall within the portion or portions of the namespace managed by this server. Selection of this type of administrator DN requires that the password be placed in the configuration file using the **adminPW** option (see page 34).

  For example, you might choose a simple DN, such as `"cn=Admin"` for the administrator DN and a simple password such as `secret`. The configuration file options would then be established this way:

  ```
  adminDN "cn=Admin"
  adminPW secret
  ```

  Note: Do not use the example above without changing the password value, as well as the actual distinguished name.

  When a program or user binds using this administrator DN, the LDAP Server verifies that the password supplied on the request matches the value provided in the configuration file for the **adminPW** option.

- Administrator DN and password as an RDBM entry

  In this method, the administrator DN is established as an entry managed by the RDBM backend. The **userPassword** attribute is used to hold the password for the administrator DN in this case. Use of this method requires that **ldif2db** be used to load an entry into the database for the administrator DN prior to starting the LDAP Server.

  For example, if the RDBM database is managing the portion of the namespace `"o=Your Company,c=US"`, one administrator DN that could be selected would be `"cn=LDAP Admin,o=Your Company,c=US"`.

  The configuration file would include the following options:

  ```
  adminDN "cn=LDAP Admin,o=Your Company,c=US"
  ...
  database rdbm GLDBRDBM
  ...
  suffix "o=Your Company,c=US"
  ```

  The LDIF-format entry to be added to the database through **ldif2db** might be:

```
dn: cn=LDAP Admin,o=Your Company,c=US
objectclass: person
cn: LDAP Admin
description: Administrator DN for o=Your Company,c=US server
userpassword: secret
```

Note: Do not use the example above without changing the password value, as well as the actual distinguished name.

If this entry is contained in an HFS file called `admin.ldif`, it can be loaded using **ldif2db**:

```
ldif2db -f my.slapd.conf -i admin.ldif
```

Note that the LDIF-format entry can alternately be stored in a dataset and loaded through a batch job using the sample JCL for **ldif2db** (GLDLD2DB).

When a program or user binds using this administrator DN, the LDAP Server verifies that the password supplied on the request matches the value of the **userPassword** attribute stored in the entry in DB2.

- Administrator DN and password in RACF

  This method requires that the LDAP Server be configured to use the RACF support provided in the SDBM backend. The administrator DN can be established as a RACF-style DN based upon a RACF user ID. (See "RACF-style Distinguished Names" on page 144 for more information.) In this case, the password for the administrator DN is the RACF user ID's password, and is stored and verified by RACF.

  For example, if you configure the LDAP Server with RACF support where the portion of the namespace held by RACF is `"sysplex=Sysplex1,o=Your Company,c=US"`, and the RACF user ID that is used for the administrator is `g1admin`, the configuration file would include these options:

  ```
  adminDN "racfid=g1admin,profiletype=user,sysplex=Sysplex1,o=Your Company,c=US"
  ...
  database sdbm GLDBSDBM
  suffix "sysplex=Sysplex1,o=Your Company,c=US"
  ```

  When a program or user binds using this administrator DN, the LDAP Server makes a request to RACF to verify that the password supplied on the request matches the RACF password for RACF user ID `g1admin`.

userPassword Encryption

The LDAP Server allows prevention of unauthorized access to user passwords in the RDBM backend. The **userPassword** attribute values can be encoded when stored in the directory, which prevents clear text passwords from being accessed by any users, including the system administrators. In the current implementation, only the **userPassword** attribute values are encrypted. Use of the terms "user password" and "password" refer to the **userPassword** attribute. Use of the term "user entry" refers to an entry in RDBM that contains a **userPassword** attribute.

The administrator may configure the server to encode **userPassword** attribute values in either a one-way hash format or a two-way, symmetric, encryption format.

After the server is configured and started, any new passwords for new user entries, or modified passwords for existing user entries are encoded before they are stored in the RDBM backend. The encoded passwords are tagged with the encoding algorithm name so that passwords encoded in different formats can coexist in the directory. When the encoding configuration is changed, existing encoded passwords remain unchanged and continue to be usable.

When the **ldif2db** program is used to load an RDBM backend, all clear text user passwords in new entries are encrypted by the method specified in the configuration file.

If there are encrypted **userPassword** values in the LDAP database, the **db2ldif** program unloads the RDBM backend to LDIF format with the password in the binary format of:

```
userPassword::  base64encoded_and_tag_encryptedvalue
```

This format can be loaded by the **ldif2db** program, for example at a replica server, and preserves the encrypted passwords.

The **-t** option on **db2ldif** unloads the user passwords in an encrypted format that might be more appropriate for loading by non-OS/390 LDAP Servers. The LDIF format unloaded by the **-t** option can be called encryption "tag visible" format and looks like:

```
userPassword: {tag}base64encoded_and_encryptedvalue
```

where *tag* is **none**, **crypt**, **MD5**, **SHA**, or **DES:***keylabel*.

In this format the tag is visible, and only the password itself is encrypted and base64 encoded.

Notes:

1. The tag is enclosed in a left brace and a right brace. One colon is used between the **userPassword** keyword and the value, as opposed to two colons in the standard LDIF format of **userPassword** unloaded by **db2ldif**. This format cannot be read by the OS/390 **ldif2db** program. It is intended for other LDAP platforms or tools that may be able to interpret this LDIF format with the encryption tag visible.
2. The values returned by the crypt() algorithm are not portable to other X/Open-conformant systems. This means that user password values encoded by the crypt() algorithm and unloaded as tagged output using **db2ldif -t** are not portable when loaded by another platform's load utility.

If the RDBM backend is already loaded and the LDAP Server is running, the **db2pwden** utility is provided to encrypt all user passwords in the method configured on the LDAP Server. The **db2pwden** utility is similar to the LDAP operation utilities, such as **ldapsearch**, in that it acts like a client to the LDAP server and has similar command line options. See Chapter 6, "Running LDAP Utilities and Programs" on page 73 for information about **db2pwden** and other LDAP operation utilities. The **db2pwden** utility must be run by the LDAP Server administrator using the **adminDN** and password configured on the server. Be aware that once a password is encrypted in a one-way hash, its clear text value can no longer be retrieved or displayed.

- **One-way hash formats:**

  crypt
  MD5
  SHA

  A crypt, MD5, or SHA hashed password can be used for password matching on an LDAP simple bind, but it cannot be decrypted. During simple bind, the bind password is hashed and compared with the stored **userPassword** attribute values for matching verification.

  MD5 and SHA hashing require the OS/390 Open Cryptographic Services Facility (OCSF) be installed and configured, and the necessary security authorizations to be set up for the LDAP Server user ID in RACF. See the configuring information in the *OCSF Application Developer's Guide and Reference* for instructions on how to do this.

  For applications which require retrieval of clear passwords, such as middle-tier authentication agents, the directory administrator must configure the LDAP Server to perform either a two-way encoding or

no encryption of user passwords. Clear passwords stored in the directory can be protected by the directory access control mechanism.

- **Two-way encryption format:**

  DES

  The DES algorithm is provided to allow values of the **userPassword** attribute to be encoded in the RDBM backend and retrieved as part of an entry in the original clear format. Some applications such as middle-tier authentication servers require passwords to be retrieved in clear text, however, corporate security policies might prohibit storing clear passwords in a secondary permanent storage. This option satisfies both requirements.

  A DES encrypted password can be used for password matching on a simple bind and can be decrypted to be returned as clear text on a search request when the client is authorized to do so. During simple bind, the bind password is encrypted and compared with the stored version for matching verification. During a search, if the client is authorized through directory access controls to see the **userPassword** attribute value, then it is decrypted and returned as clear text.

  DES encryption requires OCSF be installed and configured, and the necessary security authorizations to be set up for the LDAP Server user ID in RACF. See the configuring information in the *OCSF Application Developer's Guide and Reference* for instructions on how to do this.

  The DES algorithm also requires a key label and the data key it refers to for password matching and decryption to take place. The key label is stored in RDBM along with the tag and encrypted user password. The Integrated Cryptographic Service Facility (ICSF), Key Generator Utility Program (KGUP) and Cryptographic Key Data Set (CKDS) are used to generate and store the key label and RACF is used to limit access to this DES encryption key to only the LDAP Server. See the information on managing cryptographic keys and using the Key Generator Utility Program in the *ICSF Administrator's Guide* for information on generating, storing into CKDS, and authorizing a Data-Encrypting Key. Remember to refresh CKDS and RACF after entering and authorizing a key.

  The DES key label must correspond to the same DES keys across a sysplex and be accessible to all LDAP Servers that are using the same RDBM backend. It is recommended that you use one DES key label. If multiple DES key labels are used by different servers in the sysplex, for example, then all the servers in the sysplex need to have access to all the keys.

A simple bind will succeed if the password provided in the bind request matches with any of the multiple values of the **userPassword** attribute. Note that depending on when **userPassword** values are stored in the directory, different attribute values can be encoded using different encoding methods.

Note: The UNIX crypt() algorithm, implemented across all of the UNIX platforms, accepts only the first eight characters of a password. As a result, any password supplied on an **ldap_simple_bind** operation or **ldap_compare** operation that matches the first eight characters of a **userPassword** attribute value encrypted with the crypt() algorithm in the directory will match.

Operating in Single-server Mode

For the LDAP Server to operate in single-server mode, the server configuration file may contain any of the previously documented options except the **sysplexGroupName** and **sysplexServerName** options (the presence of which causes the LDAP server to operate in multi-server mode with dynamic workload management enabled). If the **multiserver** option is present, its value must be set to either **n** or **N**.

Restrictions

If one LDAP server instance using a given RDBM database to store directory information is operating in single-server mode, it must be the *only* instance of the LDAP server using that RDBM database. Although it is possible for a given RDBM database to be accessible from more than one host system in a Parallel Sysplex, only one instance of the LDAP server may be running on *any* host system in the Parallel Sysplex using a given RDBM database if it is configured to operate in single-server mode. Configuring more than one LDAP server instance to use the same RDBM database may yield unpredictable results if one or more of those server instances is configured in single-server mode. If it is desired to access the same RDBM database from more than one server instance, all server instances using the same RDBM database must be configured to operate in multi-server mode.

If you are going to set up more than one LDAP Server, set up a separate user ID for each one. It is necessary to have a separate **dbuserid**, and thus TSO user ID, associated with each LDAP Server instance. This is necessary because the LDAP Server creates tables on behalf of the **dbuserid**, and because DB2 uses only the **dbuserid** to qualify those table names, not the table space or database names. Attempts to use the same **dbuserid** for more than one instance of the server may result in configuration errors. In addition, the SPUFI that is run to establish the table spaces should be run under the TSO user ID associated with that server instance.

In the context of the LDAP server, RDBM databases are uniquely identified by the pair *servername*/*dbuserid*. For any two RDBM databases to be considered unique to the LDAP server, those databases must not have identical values for both of the above options in their respective server configuration files. It should be noted that the RDBM **databasename** option does not influence RDBM database uniqueness.

For example, the following combinations represent pairs of uniquely-identified RDBM databases:

```
servername loc1   databasename ldap1   dbuserid user1
servername loc2   databasename ldap1   dbuserid user1
```

```
servername loc1   databasename ldap1   dbuserid user1
servername loc2   databasename ldap1   dbuserid user2
```

```
servername loc1   databasename ldap1   dbuserid user1
servername loc2   databasename ldap2   dbuserid user2
```

```
servername loc1   databasename ldap1   dbuserid user1
servername loc2   databasename ldap2   dbuserid user1
```

```
servername loc1   databasename ldap1   dbuserid user1
servername loc1   databasename ldap1   dbuserid user2
```

```
servername loc1   databasename ldap1   dbuserid user1
servername loc1   databasename ldap2   dbuserid user2
```

However, the following pair does not represent unique RDBM databases:

```
servername loc1   databasename ldap1   dbuserid user1
servername loc1   databasename ldap2   dbuserid user1
```

Compatibility Issues

An LDAP server operating in single-server mode is compatible with RDBM databases which were created using the LDAP server or the **ldif2db** utility program from OS/390 Release 5 and Release 6.

Operating in Multi-server Mode Without Dynamic Workload Management Enabled

For the LDAP Server to operate in multi-server mode without dynamic workload management enabled, the server configuration file may contain any of the previously documented options *except* the **sysplexGroupName** and **sysplexServerName** options. If either of these keywords are present, they cause the LDAP server to operate in multi-server mode with dynamic workload management enabled. The **multiserver** mode option must be present with a value of **y** or **Y**. All instances of the LDAP server using the same RDBM database (see the definition of what makes an RDBM database unique in "Restrictions" on page 47) must have the **multiserver** option present in the configuration file used to start each server instance when operating in this mode.

In addition to the **multiserver** option, the **tbspacemutex** option must also be present in the configuration file.

When you are using referrals without dynamic workload management enabled, multiple default referrals can be set up to point to each of the multiple server instances. Similarly, any referral objects which point to it can have multi-valued **ref** attributes set up, each of which is an LDAP URL pointing to the corresponding server instances.

Dependencies

The LDAP server may operate in multi-server mode without dynamic workload management enabled while running multiple concurrent server instances configured to use the same RDBM database on the same OS/390 image, and/or while running multiple concurrent server instances on multiple OS/390 images coupled in a Parallel Sysplex. If multiple instances will be run on multiple OS/390 images in a Parallel Sysplex, the DB2 subsystems to which each server instance will attach (see "Getting DB2 Installed and Setup for CLI and ODBC" on page 13) must be configured on each of the images as members of the same DB2 data sharing group. (See *DB2 Data Sharing: Planning and Administration*, SC26-8961, for information on planning, installing, and enabling DB2 data sharing, and *MVS Setting Up a Sysplex*, GC28-1779, for information on planning and installing a Parallel Sysplex.)

Restrictions

If multiple LDAP server instances are using the same RDBM database to store directory information, all LDAP server instances using that database must be operating in multi-server mode, either with or without dynamic workload management enabled.

If you are going to set up more than one LDAP Server, set up a separate user ID for each one. It is necessary to have a separate **dbuserid**, and thus TSO user ID, associated with each LDAP Server instance. This is necessary because the LDAP Server creates tables on behalf of the **dbuserid**, and because DB2 uses only the **dbuserid** to qualify those table names, not the table space or database names. Attempts to use the same **dbuserid** for more than one instance of the server may result in

configuration errors. In addition, the SPUFI that is run to establish the table spaces should be run under the TSO user ID associated with that server instance.

LDAP server instances operating in multi-server mode, either with or without dynamic workload management enabled, will not perform replication (see Chapter 14, “Replication” on page 169), even if replication objects are present in the RDBM database. If replication is required, single-server mode must be used.

Compatibility Issues

An LDAP server operating in multi-server mode is compatible with RDBM databases which were created using the LDAP server or the **ldif2db** utility program from OS/390 Release 7 or higher only. Databases which were created with the LDAP server or **ldif2db** from Releases 5 and 6 must be migrated to the current (Release 7 or higher) DB2 schema. The migration may be performed by using the SQL Processor Using File Input (SPUFI) script, located in *GLDHLQ*.SGLDSAMP(LDAPSPMG), from DB2 Interactive (DB2I) and following the directions in that script. (For details on how to use DB2I SPUFI, see the *DB2 for OS/390 Application Programming and SQL Guide*, SC26-8958.)

No data movement is necessary for this migration to be performed; however, it is recommended that you make a current backup of your LDAP RDBM database before using this migration script.

Operating in Multi-server Mode With Dynamic Workload Management Enabled

For the LDAP Server to operate in multi-server mode with dynamic workload management enabled, the server configuration file may contain any of the previously documented options. The **sysplexGroupName** and **sysplexServerName** options must both be present in the server configuration file. If the **multiserver** option is present with a value of **n** or **N**, the option value is overridden and treated as though it were set to **Y**.

In addition to **sysplexGroupName** and **sysplexServerName** options, the **tbspacemutex** option must also be present in the configuration file.

To exploit the dynamic workload management feature, the OS/390 images on which the LDAP server runs must be coupled in a Parallel Sysplex. Name servers must be configured for connection optimization and started in the same sysplex in which the LDAP server instances are running. See “Dependencies” on page 50 for more information.

When multiple concurrent instances of the LDAP server are operating in multi-server mode with dynamic workload management enabled, server instances within the same group are considered to provide equivalent service. In essence, the servers are treated as clones of each other. With this in mind, it should be noted that the port on which the server is started should be the same for each instance.

To connect to any unspecified server instance in **sysplexGroupName** *group_name*, the client specifies a target host name of *group_name.sysplex_domain_name*, where *group_name* is the name of the application group in which servers are configured using the **sysplexGroupName** option in the server configuration file used to start each respective LDAP server instance, and *sysplex_domain_name* is the name or alias for the sysplex domain. The client will be connected to a server instance in the group on the system in the sysplex with the most available resources, at the port specified by the client which must agree with the port configured when the server instances in the group were started. While it is possible to specify a particular server instance using a fully-qualified server name, doing so reduces the effectiveness of dynamic workload management through connection optimization.

Also note that it is possible to run multiple concurrent LDAP servers using the same RDBM database with a mix of servers enabled or not enabled for dynamic workload management, but doing so also reduces the effectiveness of dynamic workload management through connection optimization.

When you are using referrals with dynamic workload management enabled, a single default referral is used where the *host* is specified as **sysplexGroupName**.*sysplexDomainName* and similarly, referral objects use this as the host within a single-valued **ref** attribute.

Dependencies

The LDAP server may operate in multi-server mode with dynamic workload management enabled while running multiple concurrent server instances configured to use the same RDBM database on multiple OS/390 images coupled in a parallel sysplex. The DB2 subsystems to which each server instance will attach (see "Getting DB2 Installed and Setup for CLI and ODBC" on page 13) must be configured on each of the images as members of the same DB2 data sharing group. (See *DB2 Data Sharing: Planning and Administration*, SC26-8961, for information on planning, installing, and enabling DB2 data sharing, and *MVS Setting Up a Sysplex*, GC28-1779, for information on planning and installing a Parallel Sysplex.)

Name servers must be configured for connection optimization and started in the same sysplex in which the LDAP server instances are running. Proper distribution of server application addresses for connection optimization to function properly requires DNS queries to be answered by the name server within the sysplex. For this reason, name servers located outside the sysplex cannot be configured as primary or secondary servers for the sysplex domain.

The port numbers on which the LDAP server instances will listen, both secure and unsecure, must be the same for all servers in the same **sysplexGroupName** for the dynamic workload management to be effective.

Also, the Workload Management Services on each host in the sysplex must be configured in "goal mode". (See *OS/390 TCP/IP Update Guide*, GC31-8553, for information on configuring a sysplex domain for connection optimization and for information on how to configure Workload Management Services in goal mode.)

Restrictions

If multiple LDAP server instances are using the same RDBM database to store directory information, all LDAP server instances using that database must be operating in multi-server mode, either with or without dynamic workload management enabled.

If you are going to set up more than one LDAP Server, set up a separate user ID for each one. It is necessary to have a separate **dbuserid**, and thus TSO user ID, associated with each LDAP Server instance. This is necessary because the LDAP Server creates tables on behalf of the **dbuserid**, and because DB2 uses only the **dbuserid** to qualify those table names, not the table space or database names. Attempts to use the same **dbuserid** for more than one instance of the server may result in configuration errors. In addition, the SPUFI that is run to establish the table spaces should be run under the TSO user ID associated with that server instance.

LDAP server instances operating in multi-server mode, either with or without dynamic workload management enabled, will not perform replication (see Chapter 14, "Replication" on page 169), even if replication objects are present in the RDBM database.

Compatibility Issues

An LDAP server operating in multi-server mode is compatible with RDBM databases which were created using the LDAP server or the **ldif2db** utility program from OS/390 Release 7 or higher only. Databases which were created with the LDAP server or **ldif2db** from Releases 5 and 6 must be migrated to the current (Release 7 or higher) DB2 schema. The migration may be performed by using the SQL Processor Using File Input (SPUFI) script, located in *GLDHLQ*.SGLDSAMP(LDAPSPMG), from DB2 Interactive (DB2I) and following the directions in that script. (For details on how to use DB2I SPUFI, see the *DB2 for OS/390 Application Programming and SQL Guide*, SC26-8958.)

No data movement is necessary for this migration to be performed; however, it is recommended that you make a current backup of your LDAP RDBM database before using this migration script.

Example of Configuring and Using Multiple Concurrent Servers in a Sysplex

Following is a simplified example scenario to demonstrate the configuration and usage of the multi-server operation mode with dynamic workload balancing enabled. While the user's operational environment may be more complex than this example, the lessons demonstrated apply in a similar fashion.

Example Scenario

ABC Company is running a 3-way Parallel Sysplex with DB2 data sharing available on each host in the sysplex. Configure an instance of the LDAP server on each of the OS/390 systems in the sysplex, serving the same LDAP directory RDBM database, such that the three instances are functional equivalents of each other, permitting users to exploit dynamic workload balancing across these LDAP servers in the sysplex.

Assume that at this point, the system administrator has installed and configured a DB2 subsystem on each host in the sysplex to be part of the same data sharing group, and that the system administrator has configured at least one Domain Name Service (DNS) server for the sysplex. In addition, the TCP/IP stacks which serve each host are registered with Workload Manager (WLM). See “Dependencies” on page 50 for more information.

The operational environment in which the LDAP servers will run is configured as indicated in the following diagram:

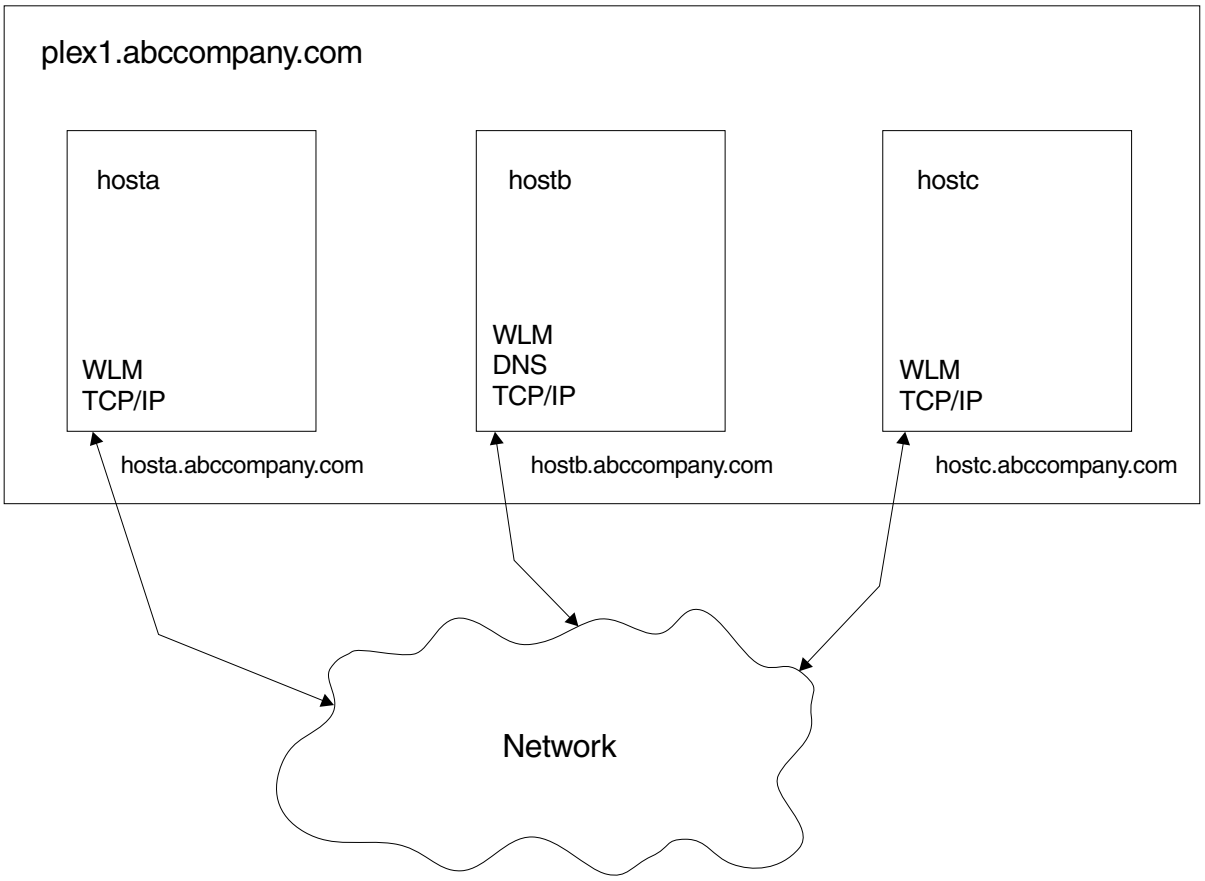


Figure 6. Multi-server Sample Configuration (Phase 1)

The three host systems are named `hosta`, `hostb`, and `hostc`, and are coupled in sysplex `plex1` in internet sub-domain `abccompany.com`. Each of the host systems is configured with a single network adapter, and these systems are accessible with these names:

```
hosta.abccompany.com
hostb.abccompany.com
hostc.abccompany.com
```

The sysplex domain name for this sysplex is `plex1.abccompany.com`. The primary DNS server for the sysplex domain runs on `hostb`.

Three server instances will be started, one on each host in the sysplex. The server configuration file for each of the three servers follows and differences among the files are highlighted.

```
#/*****************************************************************
#* This file is shipped in code page IBM-1047 and must remain in     *
#* code page IBM-1047.                                               *
#*****************************************************************/
#/*************************************************************
# *
# * Licensed Materials - Property of IBM
# * 5647-A01
# * (C) Copyright IBM Corp.  1997, 1998
# *
# *************************************************************/
#/*************************************************************
# * Filename slapd.conf
# *
# * This file is the LDAP Server configuration file for OS/390.
# *************************************************************/

#referral ldap://ldap.itd.umich.edu
include                 /etc/ldap/slapd.at.system
include                 /etc/ldap/slapd.at.conf
include                 /etc/ldap/slapd.oc.system
include                 /etc/ldap/slapd.oc.conf
port                    389
securePort              636
security                none
| sslKeyRingFile         /keyringFilePath/key.kdb
| sslKeyRingPWStashFile  /stashFilePath/key.sth
sslCipherSpecs          12288
maxthreads              0
maxconnections          0
waitingthreads          0
timelimit               3600
sizelimit               500

sysplexGroupName  ldapgrp1
sysplexServerName servera

# The following adminDN option should be updated with the
# appropriate value.  Remove the '#' to uncomment this option.
#adminDN

#####################################################################
# rdbm database definitions
```

Figure 7 (Part 1 of 2). Configuration File for Server A on hosta

```
#####################################################################

database      rdbm GLDBRDBM

# The following options must be filled in with appropriate values
# for your DB2 setup, prior to attempting to run with the DB2 backend.

multiserver   y
servername    loc1
databasename  abcdb1
dbuserid      dbu01
tbspaceentry  ldapentry
tbspace32k    ldap32k
tbspace4k     ldap4k
tbspacemutex  ldapmutx
dsnaoini      ABCCO.DB2CLI.CLIINIA


suffix        "cn=localhost"

index         cn              eq
index         ou              eq
index         sn              eq
index         telephoneNumber eq
index         title           eq
readOnly      off
```

Figure 7 (Part 2 of 2). Configuration File for Server A on hosta

Figure 8 shows the contents of ABCCO.DB2CLI.CLIINIA:

```
;This is a comment line...
; Example COMMON stanza
[COMMON]
MVSDEFAULTSSID=DB1G

; Example SUBSYSTEM stanza for your DB2 subsystem name
[DB1G]
MVSATTACHTYPE=RRSAF
PLANNAME=DSNACLI

; Example DATA SOURCE stanza for your data source
[LOC1]
AUTOCOMMIT=0
CONNECTTYPE=1
```

Figure 8. Contents of ABCCO.DB2CLI.CLIINIA

```
#/*****************************************************************
#* This file is shipped in code page IBM-1047 and must remain in     *
#* code page IBM-1047.                                               *
#*****************************************************************/
#/*************************************************************
# *
# * Licensed Materials - Property of IBM
# * 5647-A01
# * (C) Copyright IBM Corp.  1997, 1998
# *
# *************************************************************/
#/*************************************************************
# * Filename slapd.conf
# *
# * This file is the LDAP Server configuration file for OS/390.
# *************************************************************/

#referral ldap://ldap.itd.umich.edu
include           /etc/ldap/slapd.at.system
include           /etc/ldap/slapd.at.conf
include           /etc/ldap/slapd.oc.system
include           /etc/ldap/slapd.oc.conf
port              389
securePort        636
security          none
| sslKeyRingFile         /keyringFilePath/key.kdb
| sslKeyRingPWStashFile  /stashFilePath/key.sth
sslCipherSpecs    12288
maxthreads        0
maxconnections    0
waitingthreads    0
timelimit         3600
sizelimit         500

sysplexGroupName  ldapgrp1
sysplexServerName serverb

# The following adminDN option should be updated with the
# appropriate value.  Remove the '#' to uncomment this option.
#adminDN

#####################################################################
# rdbm database definitions
```

Figure 9 (Part 1 of 2). Configuration File for Server B on hostb

```
#########################################################################

database     rdbm GLDBRDBM

# The following options must be filled in with appropriate values
# for your DB2 setup, prior to attempting to run with the DB2 backend.

multiserver        y
servername         loc1
databasename       abcdb1
dbuserid           dbu01
tbspaceentry       ldapentry
tbspace32k         ldap32k
tbspace4k          ldap4k
tbspacemutex       ldapmutx
dsnaoini           ABCCO.DB2CLI.CLIINIB


suffix             "cn=localhost"

index              cn              eq
index              ou              eq
index              sn              eq
index              telephoneNumber eq
index              title           eq
readOnly           off
```

Figure 9 (Part 2 of 2). Configuration File for Server B on hostb

Figure 10 shows the contents of ABCCO.DB2CLI.CLIINIB:

```
;This is a comment line...
; Example COMMON stanza
[COMMON]
MVSDEFAULTSSID=DB2G

; Example SUBSYSTEM stanza for your DB2 subsystem name
[DB2G]
MVSATTACHTYPE=RRSAF
PLANNAME=DSNACLI

; Example DATA SOURCE stanza for your data source
[LOC1]
AUTOCOMMIT=0
CONNECTTYPE=1
```

Figure 10. Contents of ABCCO.DB2CLI.CLIINIB

```
#/*****************************************************************
#* This file is shipped in code page IBM-1047 and must remain in     *
#* code page IBM-1047.                                               *
#*****************************************************************/
#/*************************************************************
# *
# * Licensed Materials - Property of IBM
# * 5647-A01
# * (C) Copyright IBM Corp.  1997, 1998
# *
# *************************************************************/
#/*************************************************************
# * Filename slapd.conf
# *
# * This file is the LDAP Server configuration file for OS/390.
# *************************************************************/

#referral ldap://ldap.itd.umich.edu
include           /etc/ldap/slapd.at.system
include           /etc/ldap/slapd.at.conf
include           /etc/ldap/slapd.oc.system
include           /etc/ldap/slapd.oc.conf
port              389
securePort        636
security          none
| sslKeyRingFile         /keyringFilePath/key.kdb
| sslKeyRingPWStashFile  /stashFilePath/key.sth
sslCipherSpecs    12288
maxthreads        0
maxconnections    0
waitingthreads    0
timelimit         3600
sizelimit         500

sysplexGroupName  ldapgrp1
sysplexServerName serverc

# The following adminDN option should be updated with the
# appropriate values.  Remove the '#' to uncomment this option.
#adminDN

#####################################################################
# rdbm database definitions
```

Figure 11 (Part 1 of 2). Configuration File for Server C on hostc

```
#####################################################################

database        rdbm GLDBRDBM

# The following options must be filled in with appropriate values
# for your DB2 setup, prior to attempting to run with the DB2 backend.

multiserver  y
servername   loc1
databasename abcdb1
dbuserid     dbu01
tbspaceentry ldapentry
tbspace32k   ldap32k
tbspace4k    ldap4k
tbspacemutex ldapmutx
dsnaoini     ABCCO.DB2CLI.CLIINIC


suffix       "cn=localhost"

index        cn              eq
index        ou              eq
index        sn              eq
index        telephoneNumber eq
index        title           eq
readOnly     off
```

Figure 11 (Part 2 of 2). Configuration File for Server C on hostc

Figure 12 shows the contents of ABCCO.DB2CLI.CLIINIC:

```
;This is a comment line...
; Example COMMON stanza
[COMMON]
MVSDEFAULTSSID=DB3G

; Example SUBSYSTEM stanza for your DB2 subsystem name
[DB3G]
MVSATTACHTYPE=RRSAF
PLANNAME=DSNACLI

; Example DATA SOURCE stanza for your data source
[LOC1]
AUTOCOMMIT=0
CONNECTTYPE=1
```

Figure 12. Contents of ABCCO.DB2CLI.CLIINIC

The DB2 subsystem IDs must be different from the DB2 on each system and those subsystem IDs must all be defined into the same DB2 data sharing group.

Several things should be noted in the server configuration files above:

- All three configuration files use the same RDBM database (which is accessible through a DB2 data sharing group which contains the subject database) and database resources, as indicated by identical values for the parameters **servername**, **databasename**, **dbuserid**, **tbspaceentry**, **tbspace32k**,

tbspace4k, and **tbspacemutex**. Each server configuration file points to the DB2 Call Level Interface (CLI) initialization file for that respective server. The MVSDEFAULTSSID defined in the CLI initialization file for each server must be the name of the DB2 subsystem on the host on which that server instance will be started which is a member of the DB2 data sharing group which all three systems in the sysplex share, and which contains the RDBM database of interest.

- All three configuration files use the same name for their **sysplexGroupName** option; this is required to ensure all three servers are recognized as functional equivalents of each other for purposes of dynamic workload management. In addition, the **sysplexServerName** for each of the three must be unique within this **sysplexGroupName** in this sysplex.
- The ports on which the servers will listen (both secure and unsecure) must be the same for all servers in the same **sysplexGroupName** for the dynamic workload management to be effective.

With the additional configuration information just outlined, we can extend the diagram in Figure 6 on page 52 to look like this:

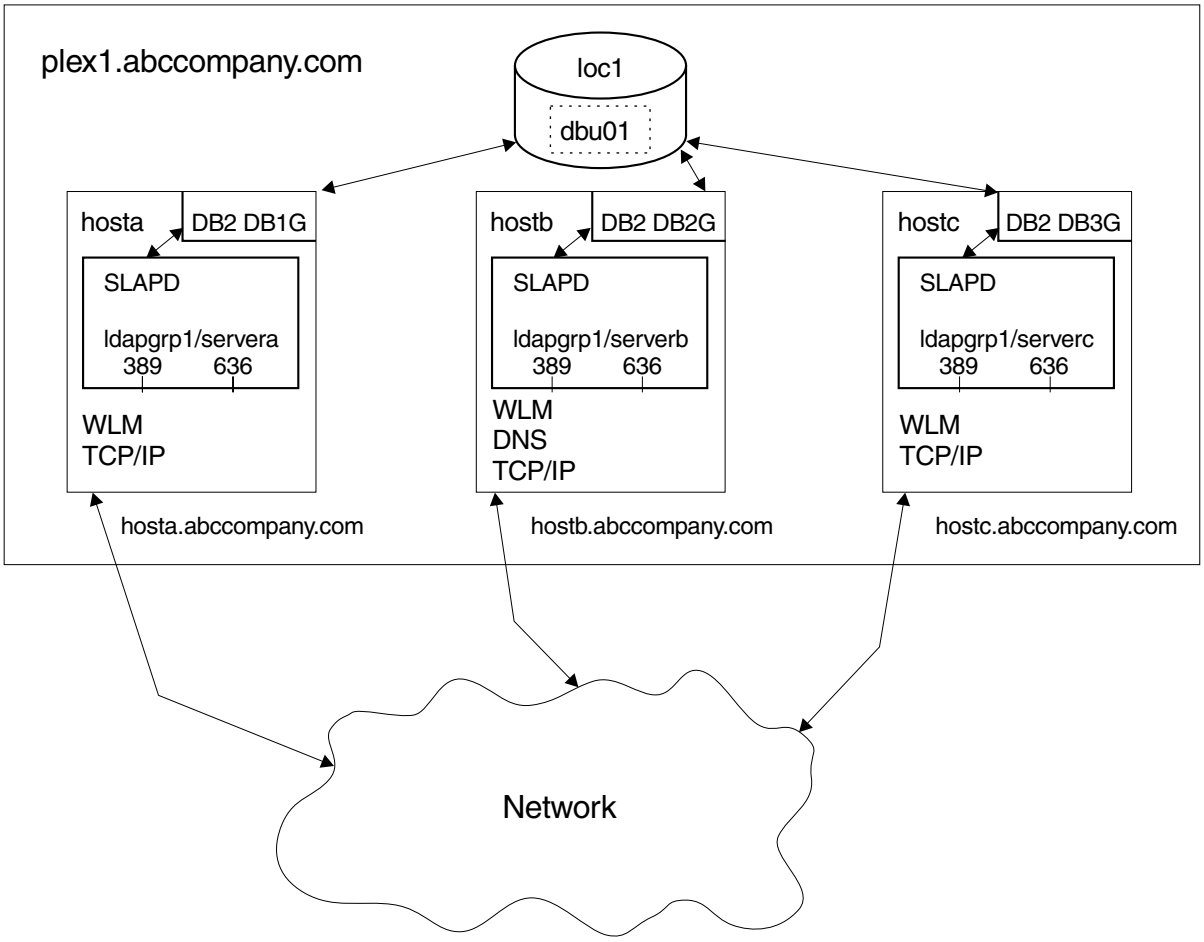


Figure 13. Multi-server Sample Configuration (Phase 2)

When the LDAP servers are started, the following messages will be printed to STDERR:

```
GLD0115I Workload Manager enablement initialization successful for
 group=ldapgrp1, server=servera on host HOSTA.

GLD0115I Workload Manager enablement initialization successful for
 group=ldapgrp1, server=serverb on host HOSTB.

GLD0115I Workload Manager enablement initialization successful for
 group=ldapgrp1, server=serverc on host HOSTC.
```

At this point, a client on `hostd` might send a search request to any LDAP server instance which will provide service for the RDBM database of interest:

```
ldapsearch -h ldapgrp1.plex1.abccompany.com -p 389 -D "cn=admin" -w secret
-b "o=ABCCOMPANY,c=US" objectclass=person cn
```

With the host specified as `ldapgrp1.plex1.abccompany.com`, the request will be routed to the server instance in **sysplexGroupName** `ldapgrp1` in sysplex `plex1.abccompany.com` which has the most available resources with which to perform the work (see Figure 14). Although requests could be directed at a specific server instance (that is, `serverc.ldapgrp1.plex1.abccompany.com` or `hostc.abccompany.com`), doing so defeats the use of the dynamic workload management features by bypassing the TCP/IP connection optimization.

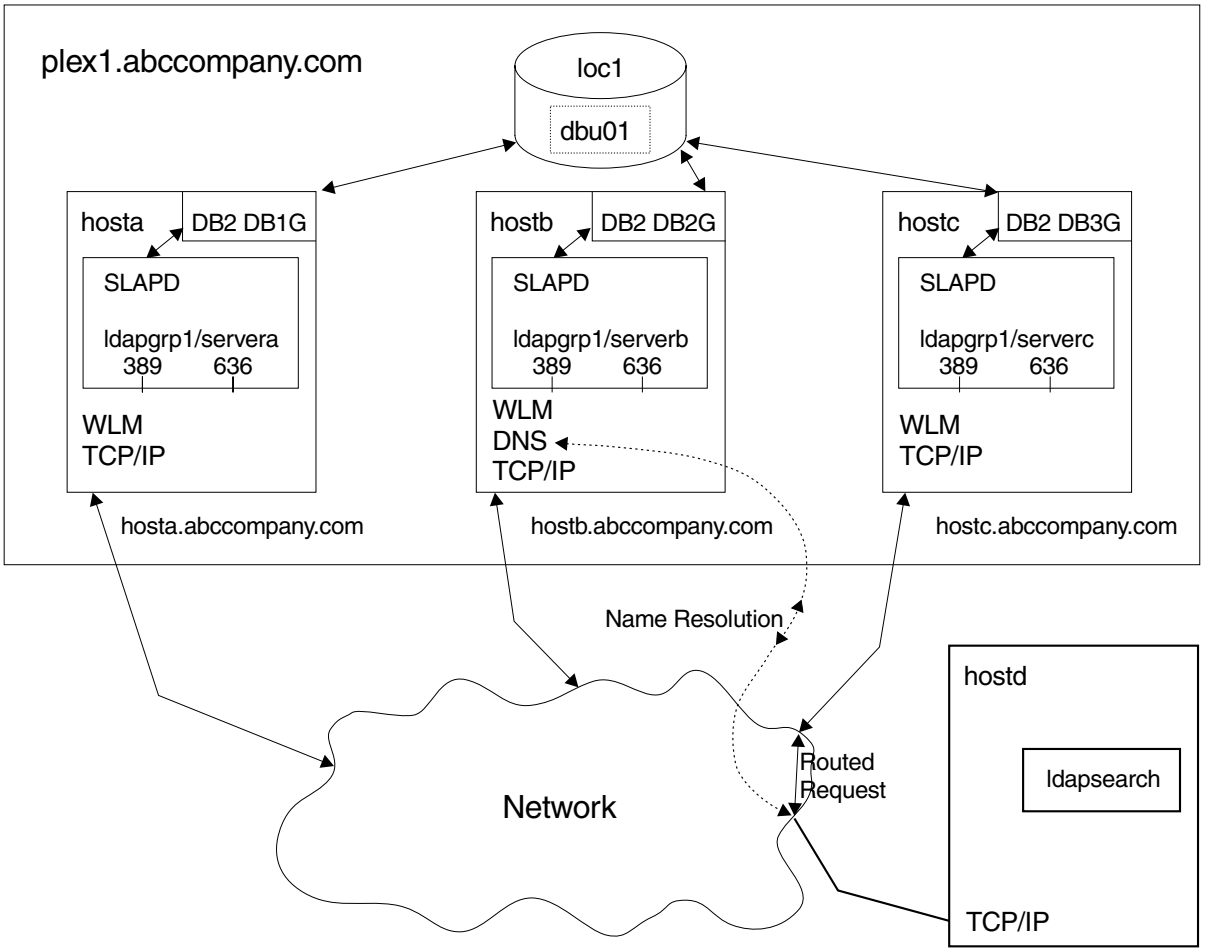


Figure 14. Multi-server Sample Configuration (Phase 3)

It should be noted that for proper workload management using TCP/IP connection optimization, DNS queries must be answered by the name server within the sysplex. For this reason, name servers that are

located outside the sysplex cannot be configured as primary or secondary servers for the sysplex domain. Thus, in this example, `hostd` must be configured to resolve names through the sysplex name server.

Using the Configuration Files

This section describes the **slapd.at.system**, **slapd.at.conf**, **slapd.oc.system**, and **slapd.oc.conf** files. It is recommended that the definition of the schema for the LDAP Server be defined in a separate file or files from the definition of the global options. This allows the schema to be maintained separately from the server operation. An effective way to do this is to use the **include** directive and store the attribute definitions in one file and the object class definitions in another file.

It is recommended that these configuration files shipped by IBM should not be modified. Rather, create a separate file or set of files to hold attribute types or object classes that you need to add. These files can then be included by the server configuration file in **/etc/ldap/lpp/slapd.conf** using the **include** keyword.

The following information applies to the configuration files:

- All configuration files are in code page IBM-1047.
- Attributes must be defined before their use in object classes.
- All attributes and object classes must be defined prior to any database lines in the configuration file. The default setup is for **slapd.conf** to include **slapd.at.system**, **slapd.oc.system**, **slapd.at.conf**, and **slapd.oc.conf**.
- Attribute column names must begin with an alphabetic character.

If you are building your directory starting with OS/390 Release 8, it is recommended that you use the newly shipped **schema.*** files. These are not used by default to support existing installations which already have schema files which conform to the configuration file naming conventions used in previous releases. Refer to more detailed information on these new schema files in “Using IBM Schema Configuration Files” on page 63.

Attribute type names and object class names in an LDAP directory are always treated as case-insensitive strings. This means that if an attribute is defined as `commonName` then the same attribute can be referenced as `COMMONNAME` or `commonName`. In addition, the X.500 standard requires that attribute type names and object class names be constrained to a subset of the set of printable characters, namely a-z, A-Z, 0-9, and hyphen (-). Note that if hyphens are used in the attribute name, they should not appear in the *colname* field of the attribute definition, as hyphens are not allowed in DB2 table definitions or column names. The attribute type or object class must begin with a letter.

For compatibility purposes, the LDAP Server supports underscores in attribute type and object class names. However, since these values are strictly outside the X.500 defined set of characters for attribute type and object class names, the use of underscores (_) in these names is strongly discouraged. In the IBM schema files (shown in Appendix A, “Configuration Files” on page 241) alias names have been introduced which either eliminate the underscores from existing attribute type and object class names or replace the underscores with hyphens.

The slapd.at.system File

The **slapd.conf** file includes the **slapd.at.system** file. It contains many commonly used attribute definitions. Figure 29 on page 245 shows the default **slapd.at.system** file. Page 34 shows how the **attribute** option is used to define an attribute type.

The slapd.at.conf File

The **slapd.conf** file includes the **slapd.at.conf** file. It contains many commonly used attribute definitions. Figure 31 on page 249 shows the default **slapd.at.conf** file. Page 34 shows how the **attribute** option is used to define an attribute type.

The slapd.oc.system File

The **slapd.conf** file includes the **slapd.oc.system** file. It contains many commonly used object class definitions used by the LDAP Server. Figure 30 on page 247 shows the default **slapd.oc.system** file.

The slapd.oc.conf File

The **slapd.conf** file includes the **slapd.oc.conf** file. It contains many commonly used object class definitions. Figure 32 on page 253 shows the default **slapd.oc.conf** file.

Specifying the Configuration Files as Data Sets

It is possible to create and use data set versions of all of the configuration files. These files are only shipped in the HFS, but data set versions can be easily created using **OGET**. See *OS/390 DCE Command Reference* for instructions on the use of the **OGET** command.

Once data set versions of these files are created, the data sets can be specified using either data set names or DD names on the **include** lines of the configuration files.

For example, if a data set called `MYUSERID.SLAPD.AT.CONF` has been created, it can be included in the configuration file this way:

```
include //'MYUSERID.SLAPD.AT.CONF'
```

It is also possible to set up a DD name in the **LDAPSRV** procedure, the **ldif2db** JCL or the **db2ldif** JCL for this configuration file. For example, the following modification has been made to the JCL:

```
SLAPDAT  DD  DSN="MYUSERID.SLAPD.AT.CONF",DISP=SHR
```

Now, this file can be included by specifying the following line in the configuration file:

```
include //DD:SLAPDAT
```

The same approach can be taken with all of the other configuration files. A mix of the approaches is also supported.

The LDAP Server configuration file can be contained in a data set and specified on the LDAP Server command line as well. An example of this is:

```
slapd -f "//'myuserid.slapd.conf'"
```

This format is also accepted by the **ldif2db** and **db2ldif** commands.

Using IBM Schema Configuration Files

A set of six schema files is included in **/usr/lpp/ldap/etc**:

- **schema.system.at**
- **schema.system.oc**
- **schema.IBM.at**
- **schema.IBM.oc**
- **schema.user.at**
- **schema.user.oc**

These schema files represent the latest developments in the set of attributes and object classes used by various IBM and vendor applications. Using these schema files can help you to set up the LDAP directory to support as many applications as possible. These files can be included into the LDAP Server configuration file in the same way that the previous **slapd.at.system** and **slapd.oc.system** files are included.

Increasing Your Table Space Size

If the LDAP Server is started using the new attribute and object class definitions contained in these schema files, you may have to increase the size of the table spaces which are created prior to LDAP Server startup. This can be done using the **priqty** and **secqty** sub-options of the **using stogroup** option which is specified on the **create tablespace** command issued through SPUFI prior to starting up the LDAP Server. When using the IBM schema, approximately 600 tables will be created during LDAP Server startup (or **ldif2db** if it is used to prime the directory tree prior to starting up the LDAP Server). An example of using the **stogroup** option on the **create tablespace** line is as follows:

```
create large tablespace eeeeeeee in ddddddddd
   numparts 1 bufferpool BP32K
   using stogroup sysdeflt priqty 720 secqty 720;
```

Setting Up to Use the IBM Schema Files

All attributes and object classes that are included in the default set of configuration files that are shipped are also included in the configuration files which represent the IBM schema. Attributes which are defined in **slapd.at.system** can be found in **schema.system.at**. Attributes which are defined in **slapd.at.racf** and **slapd.at.cb.conf** can be found in **schema.IBM.at**. Attributes which are defined in **slapd.at.conf** can be found in **schema.user.at**. Object classes which are defined in **slapd.oc.system** can be found in **schema.system.oc**. Object classes which are defined in **slapd.oc.racf** and **slapd.oc.cb.conf** can be found in **schema.IBM.oc**. Object classes which are defined in **slapd.oc.conf** can be found in **schema.user.oc**. If you are considering using the IBM schema files with an existing directory tree of information, please refer to “Using the New LDAP Schema” on page 27 for important information regarding changes made to some existing attribute types and object classes.

The structure of the files is set up to encourage you to make additions to the shipped schema in either your own schema file (or files) or into the **schema.user.at** and **schema.user.oc** files. To ease migration in the future, it is recommended that you place your additions to the schema into a separate set of files which are then included by the LDAP Server configuration file. Doing this will keep your organization's specific attribute types and object classes separated from the default files shipped with the LDAP Server and avoid the possibility of merging files at a later date should the shipped schema files require changes.

Additional schema files can be included in the LDAP Server configuration file using the **include** configuration option.

It is strongly recommended that no modifications are made to the existing set of attribute types and object classes that are defined in the shipped IBM schema files. There are cases where this has been done in the past and to support existing information, changes may be required but this should be discouraged for new development of information stored in the directory. Alternatively, a new object class should be defined which is a superset of the existing object class which was to be changed. The new object class should be considered to be a descendent of the existing object class and applications should list both object class names when adding entries into the directory tree using the new object class. Modifications to the IBM schema files should only be made to existing attribute types and object classes in the **schema.user.at** and **schema.user.oc** files to support the needs of existing directory data found in your installation. For new information formats, it is strongly recommended that new object classes be defined rather than modifying existing definitions.

Setting Up Your Server with Access Control

If you want to set up an initial server with Access Control Lists (ACLs), see Chapter 13, “Using Access Control” on page 159.

Setting Up Your Server to Run with SDBM

The LDAP Server can provide LDAP access to the user and group information stored in RACF. See Chapter 12, “Accessing RACF Information” on page 149 for details about how you can use this RACF information.

In order to configure your LDAP Server to run with the SDBM database of the LDAP Server:

- Copy the configuration files containing RACF information from the **/usr/lpp/ldap/etc** directory to the **/etc/ldap** directory. The **slapd.at.racf** configuration file contains the LDAP Server RACF attribute type definitions. The IBM schema configuration file **schema.IBM.at** also contains these RACF attribute type definitions. See “Using IBM Schema Configuration Files” on page 63 for more information.

  The **slapd.oc.racf** configuration file contains the LDAP Server object class definitions for access to RACF data through LDAP. The IBM schema configuration file **schema.IBM.oc** also contains these object class definitions. See “Using IBM Schema Configuration Files” on page 63 for more information.

  Use one of the following commands:

  ```
  cp /usr/lpp/ldap/etc/slapd.*.racf /etc/ldap

  cp /usr/lpp/ldap/etc/schema.IBM.* /etc/ldap
  ```

- Add the following lines to the common section of your existing **slapd.conf** file in **/etc/ldap**.

  ```
  include /usr/lpp/ldap/etc/slapd.at.racf
  include /usr/lpp/ldap/etc/slapd.oc.racf
  ```

Note: If you are using the IBM schema files, the attribute types and object classes needed to support RACF access are contained in **schema.IBM.at** and **schema.IBM.oc**, respectively. In this case, the **slapd.at.racf** and **slapd.oc.racf** files should not be included by the LDAP Server configuration file. Instead, the **schema.IBM.at** and **schema.IBM.oc** files should be included.

You also need to create a new database section in your **slapd.conf** file containing the following lines:

```
database sdbm GLDBSDBM
suffix "sysplex=YourSysplex"
```

Note that the keyword **sysplex** is required to be present in the suffix for SDBM. The keyword **sysplex** is meant to identify the system or sysplex whose RACF data store will be accessible.

The **slapd.at.racf**, **slapd.oc.racf**, **schema.IBM.at**, and **schema.IBM.oc** configuration files should not be modified. They document the interface to RACF through SDBM.

Be sure to set **STEPLIB** to point to the *GLDHLQ*.SGLDLNK data set before running SLAPD with SDBM, otherwise an error will be returned. (*GLDHLQ* refers to the high-level qualifier that was used to install the LDAP Server data sets.)

Note: Only one SDBM database can be defined in any given LDAP Server.

Running SDBM With RDBM

You can configure your LDAP Server to run with both SDBM and RDBM. RDBM will not create attribute tables in DB2 for each RACF attribute because the column name for these attributes in the **slapd.at.racf** and **schema.IBM.at** files is **_nocreate**.

Running SDBM Without RDBM

If you are running SDBM but not RDBM, be sure to comment out the RDBM database definitions in the **slapd.conf** file. Prefix the line to comment with a # (pound sign). When running only an SDBM backend, replication and referrals are not supported.

Securing Your LDAP Server with SSL

The LDAP Server contains the ability to protect LDAP access with Secure Sockets Layer (SSL) security. When using SSL to secure communications with the LDAP Server, the server is configured to provide server and, optionally, client authentication.

With server authentication, the LDAP Server must have a digital certificate (based on the X.509 standard). This digital certificate is used to authenticate the LDAP Server to the LDAP client application. The LDAP Server supplies the client with the LDAP Server's X.509 certificate during the initial SSL handshake. If the client validates the server's certificate, then a secure, encrypted communication channel is established between the LDAP Server and the LDAP client application.

In addition, if the LDAP Server is configured to use server and client authentication, and the client sends a digital certificate on the initial SSL handshake, it must be validated by the LDAP Server before the secure encrypted communication channel is established between them.

Client authentication by the LDAP Server facilitates the use of certificate bind (SASL mechanism of **EXTERNAL**) by an LDAP client. The bind identity becomes the distinguished name in the client digital certificate.

Note: If the LDAP Server is configured for both server and client authentication, but a client does not send a digital certificate, then the server will act as if configured for server authentication only. This provides backward compatibility of the LDAP Server.

The following high-level steps are required to enable SSL support for LDAP. These steps assume you have already installed and configured the LDAP directory server and installed OS/390 Cryptographic Services System SSL and set STEPLIB, LPALIB, or LINKLIST.

1. Configure the LDAP Server to listen for LDAP requests on the SSL port for server and, optionally, client authentication (see “Setting Up the Security Options” on page 66 below).

2. Generate the LDAP Server private key and server certificate and mark it as the default in the key database (see “Using SSL Protected Communications and the gskkyman Utility” on page 66).
3. Restart the LDAP Server.

Setting Up the Security Options

The following options for SSL can be set in the **slapd.conf** file. They are described in detail in “Configuration File Global Options” on page 33.

- **security**
- **securePort**
- **sslAuth**
- **sslCipherSpecs**
- **sslKeyRingFile**
- **sslKeyRingFilePW**
- **sslKeyRingPWStashFile**

Note: The **replKeyRingFile** and **replKeyRingPW** options are no longer necessary or evaluated by the LDAP Server. These options should be removed from the configuration file.

LDAP can be configured in the following ways:

- LDAP without SSL. Set **security none** or **security nossl** in the **slapd.conf** file. Also, specify **port** *int* in the **slapd.conf** file.
- LDAP using SSL only. Set **security sslonly** and **securePort** *int* in the **slapd.conf** file.
- Both LDAP without SSL and LDAP using SSL. Set **security ssl**, **port** *int*, and **securePort** *int* in the **slapd.conf** file.

Using SSL Protected Communications and the gskkyman Utility

The TCP/IP socket connection used to carry the LDAP protocol between an LDAP client and a LDAP Server can be protected using the Secure Sockets Layer (SSL) protocol. This protocol uses public-key infrastructure (PKI) algorithms to establish and maintain an encrypted communications path between a client and server. In OS/390, the ability to set up and communicate over SSL-protected communications links is provided by the LDAP Server with a set of services provided in OS/390 - the OS/390 Cryptographic Services System SSL set of services.

The LDAP client and server implementations take advantage of the System SSL services. If you have used SSL-protected LDAP communications in OS/390 Releases 5 or 6, you must perform a migration step (“Migrating an Existing Key Ring File” on page 67) prior to using SSL-protected LDAP communications. If you are using SSL-protected LDAP communications for the first time, then you can skip the steps pertaining to migrating an existing key ring file.

In order for the LDAP client to communicate with an LDAP Server over an SSL-protected TCP/IP socket connection, the LDAP Server must transmit a certificate to the LDAP client and, optionally, the client can transmit its certificate to the LDAP Server. The LDAP client and server must verify that the certificates they received are valid. Once the LDAP client and LDAP Server have determined the validity of the certificates provided to them, SSL-protected communications is between the LDAP client and LDAP Server.

The LDAP client and server verify the certificates sent to them by using public-key digital signatures. The LDAP client and server send their certificates to each other when the TCP/IP socket is opened. The LDAP client and server take these certificates and compare the digital signature in the certificates with a signature that it computes based on having the public-key of the signer of the certificate. In order to do this, the LDAP client and server must have the public-key of the signer of the certificates. The LDAP

client and server obtain this by reading a file that contains these public-keys. This file is called a key database.

A key database contains the public-keys that are associated with signers of certificates. These public-keys are, in reality, contained in certificates themselves. Thus, verifying one certificate requires the use of a different certificate, the signer's certificate. In this fashion, a chain of certificates is established, with one certificate being verified by using another certificate and that certificate being verified by yet another certificate, and so on. A certificate, and its associated public key, can be defined as a *root* certificate. A root certificate is *self-signed*, meaning that the public-key contained in the certificate is used to sign the certificate. Using a root certificate implies that the user *trusts* the root certificate.

The key databases used by the LDAP client and server must contain enough certificates in order to verify the certificates sent by the LDAP client and server during the start-up of the SSL connection. If either certificate is self-signed, then that certificate must be stored in the other's key database. If the certificates are signed by some other certificate signer, then the signer's certificate and any certificates that this certificate depends upon must be stored in the key databases. The key databases used by the LDAP client and server must also contain the certificates that they will transmit to each other during the startup of the SSL-protected communications.

The LDAP client and server use the System SSL functions provided in OS/390 to set up SSL-protected communications. The System SSL capability requires a key database to be set up before SSL-protected communications can begin. The key database is a password-protected file stored in the hierarchical file system (HFS). This file is created and managed using a utility program provided with System SSL called **gskkyman**.

Migrating an Existing Key Ring File: In OS/390 Releases 5 and 6, SSL-protected communications could be established between LDAP clients and servers. In these releases, the file containing all certificates used by the LDAP client and server was called a key ring file and was managed using a program called MKKF. If you have used SSL-protected communications in OS/390 Release 5 or 6, you must migrate the key ring file managed by MKKF to a key database file managed by **gskkyman**. If you have worked with a key ring file in OS/390 Release 5 or 6, you can run the following command:

```
gskkyman -m keyringfilename
```

which creates a key database containing the same set of certificates that were contained in the key ring file. This allows you to use SSL-protected LDAP communications in OS/390. This step is only required if you have an existing key ring file from Release 5 or 6 that you want to use for protected LDAP communications in OS/390. See the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for details on using the **-m** option of the **gskkyman** utility.

Creating and Using a Key Database File: In OS/390, the LDAP client and server use the System SSL services to provide the SSL-protected communications. System SSL provides the **gskkyman** utility program to create and maintain the key database file. The key database file is an encrypted (password-protected) file stored in the hierarchical file system (HFS). Before starting SSL-protected communications, both the LDAP Server and an LDAP client must create a key database file. See the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for details on using the **gskkyman** utility.

Example of Using the gskkyman Utility to Create a Key Database File: Start the **gskkyman** utility by invoking **gskkyman** from a shell prompt (OMVS or **rlogin** session):

```
$ gskkyman
```

The **gskkyman** utility provides a menu-based interface. To perform a function, choose the option you wish to perform by entering its number at the command prompt.

To create a new key database, use option **1**:

```
            IBM Key Management Utility

Choose one of the following options to proceed.

   1  - Create new key database
   2  - Open key database
   3  - Change database password

   0  - Exit program

Enter your option number: 1
```

You will be prompted for the key database file name (**key.kdb** is the default).

```
Enter key database name or press ENTER for "key.kdb": mykey.kdb
```

You will be prompted for a password to protect the key database:

```
Enter password for the key database.......>
```

You will be prompted to re-enter the password for verification:

```
Enter password again for verification.....>
```

You will be prompted about whether the password should expire. Choose either **0** or **1**:

```
Should the password expire? (1 = yes, 0 = no) [1]: 0
```

At this point, the key database will be created. You will receive a message indicating the success or failure of this operation and a prompt about whether you would like to work with the database:

```
The database has been successfully created, do you want to continue to work with
the database now? (1 = yes, 0 = no) [1]: 1
```

Choosing option 1 (yes) brings up a menu of operations. This menu of operations is the same menu that is presented if you choose option 2 from the first set of options presented by **gskkyman**.

LDAP Server: At this point, you should follow the directions for using the **gskkyman** utility which can be found in the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* in order to get the correct certificates into the key database for the LDAP Server. The LDAP Server uses the **gskkyman** utility to obtain a certificate, store the certificate into the key database using a label, and specify this certificate as the default certificate for the LDAP Server's key database.

For testing purposes, the LDAP Server can use a self-signed certificate. In this case, the certificate of the LDAP Server must also be stored into the key database of the LDAP client in order for SSL-protected LDAP communications to work between the client and server. To create a self-signed certificate with the **gskkyman** utility:

1. Use option 5 (Create a self-signed certificate).
2. Copy the certificate from the key database of the LDAP Server into a file. Send it through any appropriate method, such as FTP, to the client.
3. Store the LDAP Server's self-signed certificate into the LDAP client's key database using option 6 (Store a CA certificate).

Once the LDAP Server has a certificate and has this certificate stored as the **default** certificate in a key database file and the LDAP client has created a key database file that contains the appropriate certificates, SSL-protected communications between an LDAP client and LDAP Server can begin.

The **sslKeyRingFile**, **sslKeyRingFilePW**, and **sslKeyRingPWStashFile** options of the LDAP Server configuration file are used to indicate what HFS file contains the key database to use in the LDAP Server for setting up SSL-protected communications. The **replKeyringFile** and **replKeyRingPW** configuration options which were required for setting up SSL-protected replication links in previous releases, are no longer required and will be ignored. Instead, the LDAP Server requires only one key database file which is specified using the **sslKeyRingFile**, **sslKeyRingFilePW**, and **sslKeyRingPWStashFile** configuration options. Because the replicating server may be acting as both a replica server and an LDAP Server, the replica server's certificate (or CA's certificate) must be contained in the replicating server's key database file. In OS/390, the files used by the LDAP Server must be key database files managed by the **gskkyman** utility which is part of the System SSL services. The **-s** option of the **gskkyman** utility is used to create a key database password stash file. More information about SSL, key database files, and the **gskkyman** utility program can be found in the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference*.

Obtaining a Certificate: The LDAP Server or client can obtain a certificate by contacting a certificate authority (CA) and requesting a certificate. The **gskkyman** utility can be used to formulate a certificate request. This certificate request is usually passed to the CA by means of an electronic mail message or by an HTML form which is filled out using a web browser. The contents of the certificate request generated by using option 3 of **gskkyman** must be cut-and-pasted into either the mail message or HTML form. The certificate request will be stored in a file (default name is `certreq.arm`). Once the CA verifies the information for the LDAP client or server, a certificate is returned to the requester, usually by an electronic mail message. The contents of the mail message are cut-and-pasted into a file placed into the HFS. This file is then supplied as the file name when using option 4 of the **gskkyman** utility to receive the certificate created by the CA.

Support of RACF Key Rings: The LDAP Server supports the use of a RACF key ring. See the Certificate/Key Management section in the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for instructions on how to migrate a key database to RACF and how to use the **RACDCERT** command to protect the certificate and key ring.

The user ID under which the LDAP Server runs must also be authorized by RACF to use RACF key rings. To authorize the LDAP Server, you can use the RACF commands in the following example:

```
RDEFINE FACILITY IRR.DIGTCERT.LIST UACC(NONE)
RDEFINE FACILITY IRR.DIGTCERT.LISTRING UACC(NONE)
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(LDAPSRV) ACCESS(CONTROL)
PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY) ID(LDAPSRV) ACCESS(CONTROL)
```

Remember to refresh RACF after doing the authorizations.

```
SETROPTS RACLIST(FACILITY) REFRESH
```

Once the RACF key ring is set up and authorized, specify the RACF key ring name for the **sslKeyRingFile** and specify NULL for both the **sslKeyRingFilePW** and **sslKeyRingPWStashFile** in the LDAP Server configuration file.

LDAP Client: You should follow the directions for using the **gskkyman** utility which can be found in the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* in order to get the correct certificates into the key database for the LDAP client to verify the certificate used by the LDAP Server you wish to communicate with.

If the LDAP Server you are going to contact is using a self-signed certificate (which is done frequently while testing SSL-protected communications between LDAP client and server), then the self-signed certificate of the LDAP Server must be stored into the LDAP client's key database. To do this:

1. Copy the certificate from the key database of the LDAP Server into a file. Send it through any appropriate method, such as FTP, to the client.
2. Store the LDAP Server's self-signed certificate into the LDAP client's key database using option 6 (Store a CA certificate) of the **gskkyman** utility.

If the LDAP Server you are going to contact is using a certificate which is signed by a certificate authority (CA), you must ensure that the certificate for the CA is contained in the key database. Use whatever means is provided by the CA for obtaining the CA certificate. The certificate should be obtainable in a format that is acceptable to the **gskkyman** utility. Use option 6 (Store a CA certificate) to store the certificate of the CA into the LDAP client's key database.

If the LDAP Server is configured for server and client authentication and the client wants client authentication to occur, the LDAP client must obtain its own certificate from a CA and store it in the clients own key database and mark it as the **default**.

The LDAP client uses the key database when establishing SSL-protected communications between the LDAP client and LDAP Server. The key database file name is passed to the LDAP client on the **ldap_ssl_client_init** API. This API also requires the password of the key database file.

Once the key database file is created and contains the proper certificates, then the LDAP client is ready to perform SSL-protected communications with an LDAP Server.

Using Your LDAP Client to Access LDAP Using SSL

The **ldap_ssl_client_init** and **ldap_ssl_init** APIs can be used to start a secure connection to an LDAP Server. A description of these APIs can be found in the *OS/390 LDAP Client Application Development Guide and Reference*.

In addition, the command line utilities (for example, **ldapsearch**) can be used to communicate securely with the LDAP Server. These utilities are explained in "Using the Command Line Utilities" on page 86.

Support of Certificate Bind

The SASL bind mechanism of **EXTERNAL** is supported by the LDAP Server. This means that the authentication on the bind is performed by the SSL client authentication that was performed on the initial connection from the client.

To use SASL bind, the following steps must occur:

- The LDAP Server must be configured and started with **sslAuth** set to **serverClientAuth** so that the server can authenticate the client.
- The client chooses SSL communication with the LDAP Server through the use of **ldap_ssl_client_init** and **ldap_ssl_init**, and sends its certificate on **ldap_ssl_client_init** by having a client certificate with a private key. The private key can be marked as the **default** in its key database, or the private key's label can be specified on the **ldap_ssl_init** call.
- The LDAP Server authenticates the client's certificate (the LDAP Server's key database contains and trusts the CA certificate of the signer of the client's certificate, or if the client is using a self-signed certificate the server contains and trusts the client's certificate).
- The client chooses to SASL bind with the LDAP Server through the use of **ldap_sasl_bind** with the mechanism of **EXTERNAL**.

At this point, the LDAP server will consider the bind DN of the client for authorization purposes to be the client's DN as transmitted in the client's certificate on the **ldap_ssl_client_init**.

Chapter 6. Running LDAP Utilities and Programs

This chapter discusses how to run:

- LDAP Server
- LDAP DB2 backend utilities
- LDAP operation utilities
- LDAP password encryption utility
- **ldapcp** program

Detailed information about how to use the utilities is also included.

Running the LDAP Server

The LDAP Server (SLAPD) can be run as a process in the OS/390 shell, or can be run as a started task, using JCL.

In order to restart the LDAP Server, it is necessary to stop the server and then start it again.

Running the LDAP Server in the OS/390 Shell

In order to start the LDAP Server (SLAPD) in the shell, some environment variables need to be set properly. Ensure that **/usr/sbin** is added to the **PATH** environment variable. Also, make sure **STEPLIB** is set to *GLDHLQ*.SGLDLNK, if this data set is not in your link list. Then, start SLAPD by issuing:

```
slapd
```

SLAPD has the following optional parameters. One or more of these may be specified when starting SLAPD.

-f *pathname* Name of configuration file to be read. Default is **/etc/ldap/slapd.conf**.

-p *port* Port number where SLAPD will listen for nonsecure communications. Default is 389.

-s *secureport* Port number where SLAPD will listen for secure communications. Default is 636.

-d *debug_level* Start SLAPD with debug on. Table 5 lists the debug levels you can specify.

Table 5 (Page 1 of 2). Debug Levels

Decimal	Hexadecimal	Value	Description
0	0x00000000	**LDAP_DEBUG_OFF**	No debugging.
1	0x00000001	**LDAP_DEBUG_TRACE**	Entry and exit from routines.
2	0x00000002	**LDAP_DEBUG_PACKETS**	Packet activity.
4	0x00000004	**LDAP_DEBUG_ARGS**	Data arguments from requests.
8	0x00000008	**LDAP_DEBUG_CONNS**	Connection activity.
16	0x00000010	**LDAP_DEBUG_BER**	Encoding and decoding of data, including ASCII and EBCDIC translations, if applicable.
32	0x00000020	**LDAP_DEBUG_FILTER**	Search filters.
64	0x00000040	**LDAP_DEBUG_MESSAGE**	Messaging subsystem activities and events.

Table 5 (Page 2 of 2). Debug Levels

Decimal	Hexadecimal	Value	Description
128	0x00000080	**LDAP_DEBUG_ACL**	Access Control List activities.
256	0x00000100	**LDAP_DEBUG_STATS**	Operational statistics.
512	0x00000200	**LDAP_DEBUG_THREAD**	Threading activities.
1024	0x00000400	**LDAP_DEBUG_REPL**	Replication activities.
2048	0x00000800	**LDAP_DEBUG_PARSE**	Parsing activities.
4096	0x00001000	**LDAP_DEBUG_PERFORMANCE**	Relational backend performance statistics.
8192	0x00002000	**LDAP_DEBUG_RDBM**	Relational backend activities.
16384	0x00004000	**LDAP_DEBUG_REFERRAL**	Referral activities.
32768	0x00008000	**LDAP_DEBUG_ERROR**	Error conditions.
65536	0x00010000	**LDAP_DEBUG_SYSPLEX**	Sysplex/WLM activities.
131072	0x00020000	**LDAP_DEBUG_MULTISERVER**	Multi-server activities.
262144	0x00040000	**LDAP_DEBUG_LDAPBE**	Connection between a front end and a back end.
524288	0x00080000	**LDAP_DEBUG_STRBUF**	UTF-8 support activities.
2147483647	0x7fffffff	**LDAP_DEBUG_ANY**	All levels of debug.

Following is an example of how to start SLAPD with debugging for threading activity.

```
slapd -d 512
```

You can also use the debug levels additively. For example, if you want to see debug levels for packet and connection activity only, add together the two numbers and the result is 10 (2 + 8). For performance and error debug levels only, specify 36864 (4096 + 32768).

All informational and error messages will be printed to the screen from which SLAPD was started.

Debugging can also be turned on by setting the **LDAP_DEBUG** environment variable.

Note the low port numbers for the default ports. Use of port numbers in this low range will require that the SLAPD process run under a user ID that has UID 0.

It is also possible to define a separate user ID that will be used to run the LDAP Server. See “Defining the User ID that Runs the LDAP Server” on page 12 for instructions on defining a separate user ID to run SLAPD.

When started, SLAPD will read an environment variable file. The default file is **/etc/ldap/slapd.envvars**. This default can be changed by setting the environment variable **LDAP_SLAPD_ENVVARS_FILE** to the full path name of the desired environment variable file.

When SLAPD has been started and is ready, the message

```
GLD0122I SLAPD is ready for requests.
```

is displayed.

To stop SLAPD in the OS/390 shell, it is necessary to know its process ID. On any user ID that has a UID of 0, enter:

```
ps -ef | grep slapd
```

This will provide the process ID for SLAPD. Next, enter:

```
kill -15 process-ID
```

where *process-ID* is the process ID of SLAPD from the **ps -ef** command. This command will cause the LDAP Server to shut down.

Running the LDAP Server as a Started Task

The JCL needed to run the LDAP Server (SLAPD) as a started task is provided with the product as a procedure. (The sample JCL is shown in Figure 43 on page 363.) This JCL procedure can be started in SDSF or from the operator's console, once the sample JCL has been placed into the installation-specific library for procedures. This JCL must be tailored before it can be run.

To start SLAPD in SDSF, enter:

```
/s ldapsrv
```

To start SLAPD from the operator's console, enter:

```
s ldapsrv
```

The same parameters described in “Running the LDAP Server in the OS/390 Shell” on page 73 can be provided to SLAPD when starting it from the JCL procedure.

“Running the LDAP Server Using Data Sets” on page 76 discusses using data sets for the configuration file. In order to specify the configuration file as either a data set name or a DD name in SDSF, some special syntax is necessary.

In order to specify a full data set name, it may be necessary to be in the expanded input screen for SDSF. This is accomplished by entering a slash (/) in **sdsf.log**. On the expanded screen, it is then possible to specify a data set name for the configuration file. Assuming that the configuration file has been established in data set `MYUSERID.SLAPD.CONF`, the start command for the LDAP Server in expanded **sdsf.log** would be:

```
s ldapsrv,parms='-f ////'''MYUSERID.SLAPD.CONF'''''
```

or, if additional parameters are desired:

```
s ldapsrv,parms='-f ////'''MYUSERID.SLAPD.CONF'''' -p 999'
```

If a **DD** name, **SLAPCONF**, was established in the **LDAPSRV PROC**, as follows:

```
SLAPCONF DD DSN=MYUSERID.SLAPD.CONF,DISP=SHR
```

the LDAP Server could be started from expanded **sdsf.log** by entering:

```
s ldapsrv,parms='-f //DD:SLAPCONF'
```

When SLAPD has been started and is ready, the message

```
GLD0122I SLAPD is ready for requests.
```

is displayed.

To stop SLAPD in SDSF, enter:

```
/p ldapsrv
```

To stop SLAPD from the operator's console, enter:

```
p ldapsrv
```

This command causes the LDAP Server to shut down.

Running the LDAP Server Using Data Sets: The LDAP Server, when run as a started task, accepts several of its files as data sets. Data set versions of the configuration files and **envvars** file are not shipped with the LDAP Server, but can be created using the **OGET** command to copy the HFS versions of the files into data sets. (See *OS/390 DCE Command Reference* for information on the use of the **OGET** command.)

The default data set characteristics for record format and record length (V 255) which **OGET** will use when creating a new data set are not acceptable for JCL when submitting for batch processing. In order to avoid this, allocate the MYUSER.DSNTIJCL sequential data set to be fixed block 80 prior to performing the **OGET** operation.

A data set version of the DSNAOINI file needed for the DB2 backing store can be created by copying and editing the default file provided by DB2. See step 4 on page 14. The DSNAOINI file can be specified either in the configuration file or in a **DSNAOINI DD** statement. The DSNAOINI file needs to be a sequential data set and should also be pre-allocated as fixed block 80 prior to using **OGET**. The **DD** statement takes precedence.

Once the data set versions of these files are available, they can be specified in the **LDAPSRV** procedure. The configuration file can be specified using the **CONFIG DD** statement, the **envvars** file can be specified using the **ENVVAR DD** statement, and the **DSNAOINI** file can be specified using the **DSNAOINI DD** statement.

To use data set versions of the **slapd.at.conf**, **slapd.at.system**, **slap.oc.conf**, and **slapd.oc.system** files, create data set versions of these files using the **OGET** command, then edit the **slapd.conf** file, or data set, to use DD names to include the other configuration files. See Chapter 5, “Configuring” on page 31 for a description of the configuration files and the syntax for using DD names on the include statements.

Dynamic Debugging

When the LDAP Server is running as a started task or from the OS/390 shell, it is possible to dynamically turn the debugging facility on and off. The following command can be sent to the LDAP Server from the ISPF **sdsf.log**, or from the operator console. Note that if the command is entered from **sdsf.log**, it must be preceded by a slash (/). In the command:

```
f ldapsrv,appl=debug=nnnnn
```

the *nnnnn* is the decimal value of the desired debug level. To send the same command to the LDAP Server in the OS/390 shell, it is necessary to know the job name assigned to the process by performing

```
/d a,l
```

in **sdsf.log** and determining the name, which includes the user ID under which the LDAP Server is running and a suffix. Once this name is found, use it to replace `ldapsrv` in the command above. See “Running the LDAP Server in the OS/390 Shell” on page 73 for an explanation of the debug level values.

Debug information will be added to the output associated with the LDAP Server.

To turn the debug tracing off, enter the same command providing the value zero (0) for *nnnnn*.

Running the LDAP DB2 Backend Utilities

Two utility programs are provided to assist in initializing and backing up the data managed by the LDAP Server (SLAPD). The **ldif2db** utility is used to load data into the SLAPD backend. The **db2ldif** utility is used to unload a copy of the data in the SLAPD backend to another file. Both of these programs can be run in the OS/390 shell, as jobs using JCL and procedures, or from TSO.

Running the LDAP DB2 Backend Utilities in the OS/390 Shell

In order to run either **ldif2db** or **db2ldif** in the shell, some environment variables need to be set properly. Ensure that **/usr/sbin** is added to the **PATH** environment variable. Also, be sure **STEPLIB** is set to *GLDHLQ*.SGLDLNK. Refer to "Using the Command Line Utilities" on page 86 for information about using the **ldif2db** and **db2ldif** utilities.

An example of running **ldif2db** is:

```
ldif2db -f pathname -i pathname
```

An example of running **db2ldif** is:

```
db2ldif -f pathname -o pathname
```

When started, both **ldif2db** and **db2ldif** will read an environment variable file. The default file is **/etc/ldap/slapd.envvars**. This default can be changed by setting the environment variable **LDAP_SLAPD_ENVVARS_FILE** to the full path name of the desired environment variable file.

Running the LDAP DB2 Backend Utilities from JCL

Sample JCL for running both **ldif2db** and **db2ldif** from batch is provided with the LDAP Server. The JCL includes an inline procedure, which will need to be modified by each installation to ensure that the **ldif2db** and **db2ldif** load modules can be found. It may also be necessary to modify the **JOB** card for installation-specific requirements. See the Figure 44 on page 365 and Figure 45 on page 367 for specific instructions. These jobs can be run by editing the JCL member and entering the **submit** command.

The sample JCL also contains instructions for passing parameters into **ldif2db** or **db2ldif**.

| The sample LDIF2DB JCL that is shipped with the LDAP Server is misleading. Input to the **ldif2db** program (GLDLD2DB), whether run from TSO, batch, or shell environment, must be stored in the hierarchical file system (HFS). This is true whether the input LDIF information is taken from the standard input (SYSIN in batch) or from a file or DD specified using the **-i** command line option (PARMS setting in JCL). The HFS file can be referenced using a DD card in the JCL, but the DD card must use the **PATH** value to point to an HFS file. The following line in the sample LDIF2DB JCL:

```
| //*SYSIN    DD DSN=<INPUT.LDIF.DATASET>,DISP=SHR
```

| should be changed to:

```
| //*SYSIN    DD PATH=<INPUT.HFS.FILE>
```

| before attempting to use a DD card to specify the input LDIF file for **ldif2db**. Here is an example of setting the **SYSIN DD** card. Assume that a file exists in the HFS called `/tmp/ldif.1`. To specify this for input through the input stream to the **ldif2db** program using the **SYSIN DD** card in JCL, specify the following line in your LDIF2DB JCL:

```
| //*SYSIN    DD PATH='/tmp/ldif.1'
```

| The **ldif2db** program will read the `/tmp/ldif.1` HFS file as if it was supplied on the standard input stream.

Running the LDAP DB2 Backend Utilities in TSO

The **ldif2db** and **db2ldif** utilities can be run from TSO. Following are the steps to do this:

1. Specify the PDS (*GLDHLQ*.SGLDLNK) where the LDAP Server load modules are installed:

   ```
   tsolib act dsn('GLDHLQ.SGLDLNK')
   ```

   If your runtime libraries for DB2 are not in LINKLIB or LPA on the system, make sure you specify the DB2 high-level qualifier for your DB2 installation in a STEPLIB DD card in the LDAPSRV started task, GLDLD2DB batch job, or GLDDB2LD batch job. The LDAP Server and utilities require the following DB2 dataset:

   ```
   <DB2HLQ>.SDSNLOAD
   ```

2. Make the PDS (*GLDHLQ*.SGLDEXEC) containing the CLISTs needed to run the utilities available in SYSEXEC:

   ```
   concatd f(SYSEXEC) da('LDAP.SGLDEXEC')
   ```

If you want to specify a configuration file that is a data set, enter:

```
-f "//'datasetname'"
```

Alternately, to specify the configuration file by associating it with a DD name, enter:

```
alloc da('datasetname') fi(config) shr
```

Once this setup is complete, running these utilities follows the same syntax as would be used if running in the OS/390 shell. See “Running the LDAP DB2 Backend Utilities in the OS/390 Shell” on page 77.

Note that when using an LDIF file that is in a data set, the parser is sensitive to the format of blank lines in the data set. Blank lines must be created as a line which contains a single space character X'40'. This implies that the data set must be defined with variable block record format. In addition, if the data set is created by using **OGET** to take a file from the HFS and place it into a data set, you must edit and save the file in order to transform the blank lines (which appear as zero-length lines in the data set after the **OGET**) into lines which contain a single space character. This can be done in the ISPF editor by finding the first blank line, adding a space, and then saving the data set.

ldif2db Program

Purpose

This program is used to load entries specified in text LDAP Data Interchange Format (LDIF) into a directory stored in a relational database. The database must already exist. The **ldif2db** may be used to add entries to an empty directory database, or to a database that already contains entries.

Format

ldif2db [-i *input_file*] **[-f** *config_file*] **[-d** *debug_level*]

Parameters

-i *input_file*	Specify the input file containing directory entries in LDIF format. If the file is not in the current directory, a full path and file name must be specified. If the **-i** option is not specified, then the input to the program is read from **stdin**.
-f *config_file*	Specify the configuration file. Default is **/etc/ldap/slapd.conf**.
-d *debug_level*	Specify the debug level. See Table 5 on page 73 for a listing of the specific levels.

All other command line inputs will result in a syntax error message, after which the proper syntax will be displayed.

Notes

When running the **ldif2db** program concurrently with an LDAP Server instance which is using the same RDBM database as the **ldif2db** program (the server configuration file, **slapd.conf**, specifies the same arguments for **SERVERNAME** and **DBUSERID** keywords), the server configuration files must request or imply LDAP Server multi-server operating mode. This restriction applies to both the **ldif2db** program and the LDAP Server for Releases 7 and higher; the **ldif2db** program and the LDAP Server from Releases 5 and 6 may not be run concurrently against the same RDBM database being used by another instance of the **ldif2db** program or another instance of the LDAP Server (even if the other instances are Release 7 or higher programs designated through the server configuration file to operate in multi-server mode). See "Operating in Single-server Mode" on page 46 and "Operating in Multi-server Mode Without Dynamic Workload Management Enabled" on page 48 for more information on server operating modes.

If one or more replication objects are present in an existing database to which entries are added by the **ldif2db** program and the configuration file requests or implies LDAP server single-server operating mode, the necessary data will be created in the database to permit replication of these entries when the master LDAP server is started in single-server mode. If the configuration file requests or implies LDAP server multi-server operating mode, no data will be saved for replication purposes, and these entries will never be replicated by the master LDAP server. See "Operating in Single-server Mode" on page 46 and "Operating in Multi-server Mode Without Dynamic Workload Management Enabled" on page 48 for more information on server operating modes.

The **ldif2db** program encrypts clear text **userPassword** attribute values for new entries loaded into the RDBM backend with the **pwEncryption** method specified in the configuration file. The **ldif2db** program can load the LDIF format of an encrypted password unloaded by the **db2ldif** program. The **ldif2db** program cannot load the LDIF format unloaded by the **db2ldif** program with the **-t** option. The **-t** option unloads encryption "tag visible" format passwords for use with non-OS/390 LDAP Servers.

| The presence of the `version: 1` tag (see “db2ldif Program” on page 81 for details) indicates that all
| textual data contained within the LDIF file is portable and of UTF-8 origin. If this tag is absent, **ldif2db**
| validates (and converts to UTF-8, if necessary) all textual values prior to writing them to the database.
| The **ldif2db** program runs much slower if the `version: 1` tag is not present because of this additional
| validation. However, if database load time is a concern, you can do the following to achieve better
| performance results:

| 1. Be sure your LDIF file contains all portable characters.
| 2. Be sure that all base64 encoded textual values in the file form valid UTF-8 strings. Keep in mind that
| X'00'-X'7F', or “7-bit ASCII”, is a subset of UTF-8.
| 3. Manually insert the `version: 1` tag as the first line in the LDIF file prior to running **ldif2db**.

db2ldif Program

Purpose

This program is used to dump entries from a directory stored in a relational database into a text file in LDAP Data Interchange Format (LDIF). Migration information pertaining to data stored by previous releases of the OS/390 LDAP Server is included in the output file.

Format

db2ldif [-o *output_file*] **[-s** *subtree*] **[-f** *config_file*] **[-d** *debug_level*] **[-t]**

Parameters

-o *output_file*
Specify the output file to contain the directory entries in LDIF. All entries from the specified subtree are written in LDIF to the output file. If the file is not in the current directory, a full path and file name must be specified. If the **-o** option is not specified, then the output from the program is written to **stdout**.

-s *subtree*
The subtree DN identifies the top entry of the subtree that is to be dumped to the LDIF output file. This entry, plus all below it in the directory hierarchy are written. If this option is not specified, all directory entries stored in the database will be written to the output file, based on the suffixes specified in the configuration file.

-f *config_file*
Specify the configuration file. Default is **/etc/ldap/slapd.conf**.

-d *debug_level*
Specify the debug level. See Table 5 on page 73 for a listing of the specific levels.

-t
Unload encrypted **userPassword** attributes in an encryption "tag visible" format for portability. This format of data may be acceptable for other LDAP providers to load into their LDAP directory. And, if it is not directly loadable, this format is easily modified for loading by another provider into its LDAP directory. This format cannot be loaded back into an OS/390 LDAP Server.

This parameter specifies that encrypted **userPassword** attribute values will be unloaded with their encryption tag in clear text, as follows:

```
userPassword: {tag}base64encoded_and_encryptedvalue
```

where *tag* is **none**, **crypt**, **MD5**, **SHA**, or **DES:***keylabel*.

Examples

```
userPassword: {none}321p90fa0fdvn;a

userPassword: {crypt}3sdfaf[a

userPassword: {SHA}24309gf[jgt

userPassword: {DES:kgup.data.key}3ajewomv..=
```

In this format, the tag is visible, and only the **userPassword** value itself is encrypted and base64 encoded.

Notes:

1. The tag is enclosed by a left brace and a right brace. One colon is used between the **userPassword** keyword and the value, as opposed to two colons in the standard LDIF format of **userPassword** dumped by **db2ldif**. This format cannot be read by the **ldif2db**. It is intended for other LDAP providers and tools that may require the encryption tag visible.
2. Clear text passwords without a tag could still exist in the RDBM backend if the password was not modified or **pwEncryption** was not configured on the server. The values would be unloaded as standard binary attributes in base64 encoding. Following is an example:

   ```
   userPassword:: kfa6903axs
   ```

3. The values returned by the crypt() algorithm are not portable to other X/Open-conformant systems. This means that user password values encoded by the crypt() algorithm and unloaded as tagged output using **db2ldif -t** are not portable when loaded by another platform's load utility.

All other command line inputs will result in a syntax error message, after which the proper syntax will be displayed.

For the LDAP Version 3 protocol, there is a related set of Internet Drafts which discuss the introduction of a version mechanism for use in creating LDIF files. Starting with Release 8, **db2ldif** will always create "tagged" LDIF files. The new LDIF tag consists of a single line at the top of the LDIF file:

```
version: 1
```

All characters contained in the `version: 1` LDIF file are portable characters represented in the local codepage. Strings containing nonportable characters (for instance, textual values containing multi-byte UTF-8 characters) must be base64 encoded.

Migration Information

Prior to Release 8, all releases of the OS/390 LDAP Server supported the LDAP Version 2 protocol exclusively and assumed all textual data exchanged with LDAP clients to be in the ISO8859-1 character set. The ISO8859-1 character set maps characters with values between X'00'-X'FF'. The LDAP Version 3 protocol requires textual data exchanged between LDAP clients and servers to be UTF-8. ISO8859-1 and UTF-8 identically map for the IA5 character set (X'00'-X'7F', or "7-bit ASCII"). However, for values greater than X'7F', the character mappings differ between ISO8859-1 and UTF-8. Since Release 8 of the OS/390 LDAP Server supports the LDAP Version 3 protocol, non-IA5 data stored by previous releases of the OS/390 LDAP Server needs to be migrated. The information required to migrate this data is obtained using the Release 8 version of **db2ldif**. The migration information can be reloaded into the relational database using the **ldif2db** program.

Migration information is only included in the resulting `version: 1` LDIF file if the directory entries include textual data that was stored by previous OS/390 LDAP Server releases **and** originated from data outside the IA5 character set.

Following are the various types of migration information that can be present in a `version: 1` LDIF file:

- "DF" - Distinguished Name, FAILED UTF-8 validation test.

  If portions of a directory entry's distinguished name (DN) contain UTF-8 character sequences that are not valid, the DN is converted to UTF-8 (using ISO8859-1 as the source of the conversion), base64 encoded, and then written to the LDIF file. The following commentary precedes the base64 encoded line written to the LDIF file:

```
#DF1 <the original DN, as represented in the local codepage>
#DF2 <the new DN, represented in the "escaped" local codepage (see page 33)>
```

Note: Prior releases of the OS/390 LDAP Server did not store DNs as base64 encoded strings; they were stored in the local codepage. This is also true for access control list (ACL) attribute values. Characters undefined by the local codepage were substituted with the EBCDIC substitution character (X'3F'). While the EBCDIC substitution character (X'3F') can be mapped directly to a valid UTF-8 character (X'1F'), the presence of substitution characters is considered suspicious and is flagged as a failure when encountered.

- "DP" - Distinguished Name, PASSED UTF-8 validation test.

  If portions of a directory entry's DN originated from values outside of the IA5 range, and also form valid UTF-8 character sequences, the resulting DN is simply base64 encoded before it is written to the LDIF file. No character set conversion is done for this data. The following commentary precedes the base64 encoded line written to the LDIF file:

```
#DP1 <the original DN, as represented in the local codepage>
#DP2 <the new DN, represented in the "escaped" local codepage (see page 33)>
```

- "AF" - Attribute value, FAILED UTF-8 validation test.

  If the directory entry contains a textual (nonbinary) base64 encoded attribute value which does not form valid UTF-8 character sequences, the value is converted to UTF-8 (using ISO8859-1 as the source for the conversion), base64 encoded, and then written to the LDIF file. The following commentary precedes the base64 encoded line written to the LDIF file:

```
#AF1 <the original value, as represented in the local codepage>
#AF2 <the new value, represented in the "escaped" local codepage (see page 33)>
```

- "AP" - Attribute value, PASSED UTF-8 validation.

  If the directory entry contains a textual (nonbinary) base64 encoded attribute value which forms valid UTF-8 character sequences, the value is written as it is to the resulting LDIF file. The following commentary precedes the base64 encoded line written to the LDIF file:

```
#AP1 <the original value, as represented in the local codepage>
#AP2 <the new value, represented in the "escaped" local codepage (see page 33)>
```

Notes:

1. The following ACL-related attribute types, if flagged, will not contain the local codepage representation flag (AF1/AP1):

   entryOwner
   ownerSource
   aclSource
   aclEntry

2. The informational commentary described above will not extend past column 77. Commentary values longer than this will be split and continued on the next line. Each line of continued commentary will include the identifier (DF, DP, and so on), plus an additional space between the identifier and the continued value.

3. In the event that an error occurs generating a commentary value, the value will be substituted with the following text:

   `Error generating value. LDAP return code=`*n*`.`

   where *n* is the LDAP error encountered.

In addition to character set migration issues, there may also be migration issues regarding attribute value and DN lengths. Multi-byte UTF-8 characters are represented internally by the OS/390 LDAP Server as 5-byte character sequences. So, for example, a string consisting of 5 double-byte UTF-8 characters is

actually 25 bytes in length as represented internally by the server. Attribute and DN size-limit restrictions are enforced by the server based on this internal representation.

- "ALX" - Attribute value, maximum allowable length exceeded.

  If an attribute value, as represented internally by the LDAP Server, exceeds the maximum size allowed by the schema, it is flagged with the additional commentary:

  ```
  #ALX attrType=<attribute type>, maxSize=<maximum allowable size as defined by the schema>, valueSize=<size>
  ```

  Note: ALX, if applicable, accompanies either an AF or AP sequence.

- "DLX" - Distinguished Name, maximum allowable length exceeded.

  If a directory entry's DN, as represented internally by the LDAP Server, exceeds the maximum allowable DN size (4000), it is flagged with the additional commentary:

  ```
  #DLX attrType=DN, maxSize=4000, valueSize=<size>
  ```

  Note: DLX, if applicable, accompanies either a DF or DP sequence.

At the end of the `version: 1` LDIF file, there is a report which summarizes the contents of the file. The contents of the report are as follows:

```
#ST numEntries=n, numSuspect=m
#SA attrType=x, maxBytesAllowed=y, maxBytesFound=z
#SA ...
#SA ...
 .
 .
 .
```

where:

ST is the summary of the total number of entries unloaded

n is the number of entries unloaded

m is the number of entries containing at least one value flagged as DF, DP, AF, or AP

SA is the summary of the largest attribute of this type which exceeds the allowable size

x is the attribute type

y is the maximum allowable size (in bytes) for values of this attribute type

z is the size (in bytes) that would be required to contain the largest attribute value found of this type

If **numSuspect**=0, the complete set of migration steps is **not** necessary for your installation. Whether or not this is the case, refer to "Coexistence and Migration with Previous Releases" on page 24 for instructions on how to proceed if you are migrating data from a previous OS/390 LDAP release.

Notes

The **db2ldif** program only dumps owner and ACL information for entries that have a specific owner or ACL. Any entry data with an inherited owner or ACL will not have owner or ACL information dumped.

Running the LDAP Operation Utilities

Several utility programs are provided with the LDAP Server. These utilities are versions of the examples that you can run. The utilities provide a way to add, modify, search and delete entries from the LDAP Server.

Each of the five programs:

- **ldapadd**
- **ldapmodify**
- **ldapmodrdn**
- **ldapsearch**
- **ldapdelete**

can be run from the OS/390 shell, or from TSO.

Running the LDAP Operation Utilities in the OS/390 Shell

In order to run any of these utilities in the shell, some environment variables need to be set properly. Ensure that **/bin** is included in the **PATH** environment variable. Also, make sure **STEPLIB** is set to *GLDHLQ*.SGLDLNK.

Each of these utilities accepts many possible parameters. See “Using the Command Line Utilities” on page 86 for a complete explanation of the parameters that can be supplied to each of the operation utility programs. In each of the examples that follow, the **-f** parameter is used to specify the full path name of a file containing the information to be added or modified. This should not be confused with the **-f** parameter for **ldif2db**, **db2ldif**, and **slapd** which is used for specifying the configuration file.

To run **ldapadd**, enter:

```
ldapadd -h hostname -D binddn -w password -f pathname
```

To run **ldapmodify**, enter:

```
ldapmodify -h hostname -D binddn -w password -f pathname
```

To run **ldapmodrdn**, enter:

```
ldapmodrdn -h hostname -D binddn -w password -f pathname
```

To run **ldapsearch**, enter:

```
ldapsearch -h hostname -D binddn -w password -b basedn filter [attrs]
```

To run **ldapdelete**, enter:

```
ldapdelete -h hostname -D binddn -w password dn
```

Note: When operating the LDAP server in multi-server mode with dynamic workload management enabled (see Chapter 5, “Configuring” on page 31 for additional information about LDAP server operating modes), the *hostname* value in the preceding commands should be in the form *group_name.sysplex_domain_name*, where *group_name* is the name of the **sysplexGroupName** identified in the server configuration files (see the search example on page 60) and *sysplex_domain_name* is the name or alias of the sysplex domain in which the servers operate.

Running the LDAP Operation Utilities in TSO

The LDAP operation utilities can be run from TSO. In order to do this, some elements of the environment need to be set up to locate the LDAP programs. First, the PDS (*GLDHLQ*.SGLDLNK) where the LDAP Server load modules were installed needs to be specified in one of **LINKLIB**, **LPALIB** or **TSOLIB**. Second, the PDS (*GLDHLQ*.SGLDEXEC) containing the CLISTs needed to run the utilities must be available in **SYSEXEC**.

Once this setup is complete, running these utilities follows the same syntax as would be used if running them in the OS/390 shell, except that the program names are eight characters or less. To run these utilities from TSO, use the following names:

OS/390 Shell Name	TSO Name
ldapadd	ldapadd
ldapmodify	ldapmdfy
ldapmodrdn	ldapmrdn
ldapsearch	ldapsrch
ldapdelete	ldapdlet

Using the Command Line Utilities

The **ldapdelete**, **ldapmodify**, **ldapadd**, **ldapmodrdn**, and **ldapsearch** utilities all use the **ldap_bind** API. When bind is invoked, several results can be returned. Following are bind results using various combinations of user IDs and passwords.

1. If specifying the administration DN, the password must be correctly specified or the bind will not be successful.
2. If a null DN is specified, or a 0 length DN is specified, you will receive unauthenticated access.
3. If a DN is specified, and it is non-null, a password must also be specified or an error will be returned.
4. If a DN and password are specified, but it does not fall under any suffix in the directory, a referral will be returned.
5. If a DN and password are specified, and are correct, the user is bound with that identity.
6. If a DN and password are specified, but the DN does not exist, unauthenticated access will be given.
7. If a DN and password are specified, and the DN exists, but the object does not have *userpassword*, an error message will be returned.
8. If a DN and password are specified, and the DN exists, but the password is of 0 length, then unauthenticated access will be given.

ldapdelete Utility

Purpose

The **ldapdelete** utility is a shell-accessible interface to the **ldap_delete** API.

The **ldapdelete** utility opens a connection to an LDAP Server, binds, and deletes one or more entries. If one or more *dn* arguments are provided, entries with those DNs are deleted. If no *dn* arguments are provided, a list of DNs is read from standard input (**<***entryfile*) or from *file* if the **-f** flag is used.

Format

```
ldapdelete [options] {-f file | < entryfile | dn... }
```

Parameters

options The following table shows the *options* you can use for the **ldapdelete** utility:

Table 6 (Page 1 of 2). ldapdelete Options

Option	Description
?	Print this text.
-V *version*	Specify the LDAP protocol level the client should use. The value for *version* can be **2** or **3**. The default is **2**.
-c	Continuous operation mode. Errors are reported, but **ldapdelete** will continue with deletions. The default is to exit after reporting an error.
-n	Show what would be done, but do not actually delete entries. Useful for debugging in conjunction with **-v**.
-v	Use verbose mode, with many diagnostics written to standard output.
-R	Do not automatically follow referrals.
-M	Manage referral objects as normal entries. This requires a protocol level of 3 (specify the **-V 3** parameter).
-d *debuglevel*	Set the LDAP debugging level to *debuglevel*.
-D *binddn*	Use *binddn* to bind to the LDAP directory. The *binddn* parameter should be a string-represented DN. The default is a NULL string.
-w *passwd*	Use *passwd* as the password for simple authentication. The default is a NULL string.
-h *ldaphost*	Specify the host on which the LDAP Server is running. The default is the local host. When the target host is an OS/390 LDAP Server operating in multi-server mode with dynamic workload management enabled (see Chapter 5, “Configuring” on page 31 for additional information about LDAP server operating modes), the *ldaphost* value should be in the form *group_name.sysplex_domain_name*, where *group_name* is the name of the **sysplexGroupName** identified in the server configuration file and *sysplex_domain_name* is the name or alias of the sysplex domain in which the target server operates.
-p *ldapport*	Specify the TCP port where the LDAP Server is listening. The default LDAP port is 389. If not specified and **-Z** is specified, the default LDAP SSL port 636 is used.
-Z	Use a secure SSL connection to communicate with the LDAP Server. The **-Z** option is not supported by non-SSL versions of this tool.

Table 6 (Page 2 of 2). ldapdelete Options	
Option	**Description**
-K *keyfile*	Specify the name of the SSL key database file. If the key database file is not in the current directory, specify the fully-qualified key database file name. If a key database file name is not specified, this utility looks for the presence of the **SSL_KEYRING** environment variable with an associated file name. Otherwise, no key database file will be used for server authentication and default trusted certification authority roots will be used. The key database file typically contains one or more certificates of certification authorities (CAs) that are trusted by the client. These types of X.509 certificates are also known as trusted roots. For more information on managing an SSL key database file, see the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for a description on the use of the **gskkyman** utility to manage the contents of a key database file. Also see "Securing Your LDAP Server with SSL" on page 65 for more information about SSL and certificates. This parameter is ignored if **-Z** is not specified.
-P *keyfilepw*	Specify the key database file password. This password is required to access the encrypted information in the key database file (including the private key). If the key database file does not contain a private key, which is possible on the LDAP client, then the key database file may have been created without a password. In this case, there is no need to specify a password here. This parameter is ignored if **-Z** is not specified.
-N *keyfiledn*	Specify the certificate name in the key database file.

-f *file* Read a series of lines from *file*, performing one LDAP delete for each line. In this case, the *filter* given on the command line is treated as a pattern where the first occurrence of %s is replaced with a line from *file*.

entryfile Specify a file containing DNs to delete on consecutive lines.

dn Specify distinguished name (DN) of an entry to delete. You can specify one or more *dn* arguments. Each *dn* should be a string-represented DN.

Example

The following command:

```
ldapdelete "cn=Delete Me, o=My Company, c=US"
```

attempts to delete the entry named with **commonName** `Delete Me` directly below `My Company` organizational entry. It may be necessary to supply a *binddn* and *passwd* for deletion to be allowed (see the **-D** and **-w** options).

Notes

If no *dn* arguments are provided, the **ldapdelete** command will wait to read a list of DNs from standard input. To break out of the wait, use <Ctrl-C> or <Ctrl-D>

SSL Notes

The contents of a client's key database file is managed with the **gskkyman** utility. See the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for information about the **gskkyman** utility. The **gskkyman** utility is used to define the set of trusted certification authorities (CAs) that are to be trusted by the client. By obtaining certificates from trusted CAs, storing them in the key database file, and marking them as trusted, you can establish a trust relationship with LDAP Servers that use certificates issued by one of the CAs that are marked as trusted.

If the LDAP Servers accessed by the client use server authentication, it is sufficient to define one or more trusted root certificates in the key database file. With server authentication, the client can be assured that the target LDAP Server has been issued a certificate by one of the trusted CAs. In addition, all LDAP transactions that flow over the SSL connection with the server are encrypted, including the LDAP credentials that are supplied on the **ldap_bind** API.

For example, if the LDAP Server is using a high-assurance VeriSign certificate, you should obtain a CA certificate from VeriSign, receive it into your key database file, and mark it as trusted. If the LDAP Server is using a self-signed **gskkyman** server certificate, the administrator of the LDAP Server can supply you with a copy of the server's certificate request file. Receive the certificate request file into your key database file and mark it as trusted.

Using this utility without the **-Z** parameter and calling the SSL-defined port on an LDAP Server (a non-SSL call to an SSL port) is not supported. Also, an SSL call to a non-SSL port is not supported.

Diagnostics

Exit status is 0 if no errors occur. Errors result in a nonzero exit status and a diagnostic message being written to standard error.

ldapmodify and ldapadd Utilities

The **ldapmodify** utility is a shell-accessible interface to the **ldap_modify** and **ldap_add** APIs. The **ldap_add** API is implemented as a renamed version of **ldapmodify**. When invoked as **ldapadd**, the **-a** (add new entry) flag is turned on automatically.

The **ldapmodify** utility opens a connection to an LDAP Server, binds, and modifies or adds entries. The entry information is read from standard input or from *file* through the use of the **-f** option.

Format

ldapmodify | **ldapadd** [*options*]

Parameters

options The following table shows the *options* you can use for the **ldapmodify** and **ldapadd** utilities:

Table 7 (Page 1 of 2). ldapmodify and ldapadd Options

Option	Description
?	Print this text.
-V *version*	Specify the LDAP protocol level the client should use. The value for *version* can be **2** or **3**. The default is **2**.
-c	Continuous operation mode. Errors are reported, but **ldapmodify** will continue with modifications. The default is to exit after reporting an error.
-n	Show what would be done, but do not actually modify entries. Useful for debugging in conjunction with **-v**.
-v	Use verbose mode, with many diagnostics written to standard output.
-R	Do not automatically follow referrals.
-M	Manage referral objects as normal entries. This requires a protocol level of 3 (specify the **-V 3** parameter).
-a	Add new entries. The default for **ldapmodify** is to modify existing entries. If invoked as **ldapadd**, this flag is always set.
-b	Assume that any values that start with a slash (/) are binary values and that the actual value is in a file whose path is specified in the place where values normally appear.
-r	Replace existing values by default.
-F	Force application of all changes regardless of the contents of input lines that begin with **replica:** (by default, **replica:** lines are compared against the LDAP Server host and port in use to decide if a replication log record should actually be applied).
-d *debuglevel*	Set the LDAP debugging level to *debuglevel*.
-f *file*	Read the entry modification information from *file* instead of from standard input.
-D *binddn*	Use *binddn* to bind to the LDAP directory. The *binddn* should be a string-represented DN. The default is a NULL string.
-w *passwd*	Use *passwd* as the password for simple authentication. The default is a NULL string.

Table 7 (Page 2 of 2). ldapmodify and ldapadd Options

Option	Description
-h *ldaphost*	Specify the host on which the LDAP Server is running. The default is the local host. When the target host is an OS/390 LDAP Server operating in multi-server mode with dynamic workload management enabled (see Chapter 5, "Configuring" on page 31 for additional information about LDAP server operating modes), the *ldaphost* value should be in the form *group_name.sysplex_domain_name*, where *group_name* is the name of the **sysplexGroupName** identified in the server configuration file and *sysplex_domain_name* is the name or alias of the sysplex domain in which the target server operates.
-p *ldapport*	Specify the TCP port where the LDAP Server is listening. The default LDAP port is 389. If not specified and **-Z** is specified, the default LDAP SSL port 636 is used.
-Z	Use a secure SSL connection to communicate with the LDAP Server. The **-Z** option is not supported by non-SSL versions of this tool.
-K *keyfile*	Specify the name of the SSL key database file. If the key database file is not in the current directory, specify the fully-qualified key database file name. If a key database file name is not specified, this utility will look for the presence of the **SSL_KEYRING** environment variable with an associated file name. Otherwise, no key database file will be used for server authentication and default trusted certification authority roots will be used. The key database file typically contains one or more certificates of certification authorities (CAs) that are trusted by the client. These types of X.509 certificates are also known as trusted roots. For more information on managing an SSL key database file, see the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for a description on the use of the **gskkyman** utility to manage the contents of a key database file. Also see "Securing Your LDAP Server with SSL" on page 65 for more information about SSL and certificates. This parameter is ignored if **-Z** is not specified.
-P *keyfilepw*	Specify the key database file password. This password is required to access the encrypted information in the key database file (including the private key). If the key database file does not contain a private key, which is possible on the LDAP client, then the key database file may have been created without a password. In this case, there is no need to specify a password here. This parameter is ignored if **-Z** is not specified.
-N *keyfiledn*	Specify the certificate name in the key database file.

Input Modes

The **ldapmodify** command as well as the **ldapadd** command accept two forms of input. The type of input is determined by the format of the first input line supplied to **ldapmodify** or **ldapadd**.

Note: The **ldapadd** command is equivalent to invoking the **ldapmodify -a** command.

The first line of input to the **ldapmodify** command (or **ldapadd** command) must denote the distinguished name of a directory entry to add or modify. This input line must be of the form:

dn:*distinguished_name*

or

distinguished_name

where **dn:** is a literal string and *distinguished_name* is the distinguished name of the directory entry to modify (or add). If **dn:** is found, the input mode is set to *LDIF mode*. If it is not found, the input mode is set to *modify mode*.

Note: The **ldapmodify** and **ldapadd** utilities do not support base64 encoded distinguished names.

LDIF Mode: When using LDIF mode style input, attribute types and values are delimited by colons (or double colons (::)). Furthermore, individual changes to attribute values are delimited with a **changetype:** input line. The general form of input lines for LDIF mode is:

change_record
<blank line>
change_record
<blank line>
.
.
.

An input file in LDIF mode consists of one or more *change_record* sets of lines which are separated by a single blank line. Each *change_record* has the following form:

dn:*distinguished_name*
[**changetype:**{**modify**|**add**|**modrdn**|**delete**}]
[*change_clause*
.
.
.]

Thus, a *change_record* consists of a line indicating the distinguished name of the directory entry to be modified, an optional line indicating the type of modification to be performed against the directory entry, along with one or more *change_clause* sets of lines. If the **changetype** line is omitted, then the change type is assumed to be **modify** unless the command invocation was **ldapmodify -a** or **ldapadd**, in which case the **changetype** is assumed to be **add**.

When the change type is **modify**, each *change_clause* is defined as a set of lines of the form:

add:x
{*attrtype*}{*sep*}{*value*}
.
.
.
–

or

replace:x
{*attrtype*}{*sep*}{*value*}
.
.
.
–

or

delete:{*attrtype*}
–

or

{*attrtype*}{*sep*}{*value*}
.
.
.

Specifying **replace** replaces all existing values for the attribute with the specified set of attribute values. Specifying **add** adds to the existing set of attribute values.

If an **add:x**, **replace:x**, or **delete:x** line (a change indicator) is specified, a line containing a hyphen (–) is expected as a closing delimiter for the changes. Attribute-value pairs are expected on the input lines that are found between the change indicator and hyphen line. If the change indicator line is omitted, the change is assumed to be **add** for the attribute values specified. However, if the **-r** option is specified on **ldapmodify**, then the *change_clause* is assumed to be **replace**. The separator, *sep*, can be either a single colon (:) or double colon (::). If a single colon is used as the separator, then all following text, including any white space after an optional space character after the separator is taken as the attribute value. Attribute values can be continued across multiple lines by using a single space character as the first character of the next line of input. If a double colon is used as the separator, then the input is expected to be in so-called **base64** format. This format is an encoding that represents every three binary bytes with four text characters. Refer to the **base64encode()** function in **/usr/lpp/ldap/examples/line64.c** for an implementation of this encoding.

Multiple attribute values are specified using multiple {*attrtype*}{*sep*}{*value*} specifications.

When the change type is **modify**, each *change_clause* is defined as a set of lines of the form:

{*attrtype*}{*sep*}{*value*}

As with change type of **modify**, the separator, *sep*, can be either a single colon (:) or double colon (::). If a single colon is used as the separator, then all following text, including any white space after an optional space character after the separator is taken as the attribute value. Attribute values can be continued across multiple lines by using a single space character as the first character of the next line of input. If a double colon is used as the separator, then the input is expected to be in so-called **base64** format.

When the change type is **modrdn**, each *change_clause* is defined as a set of lines of the form:

newrdn:*value*
deleteoldrdn:{**0**|**1**}

These are the parameters you can specify on a modify RDN LDAP operation. The value for the **newrdn** setting is the new RDN to be used when performing the modify RDN operation. Specify 0 for the value of the **deleteoldrdn** setting in order to save the old RDN and specify 1 to remove the old RDN.

When the change type is **delete**, no *change_clause* is specified.

The LDIF mode of input allows for almost any form of update to the LDAP directory to be accomplished. The one operation that cannot be performed by the **ldapmodify** command is the deletion of individual attribute values.

Modify Mode: The modify mode of input to the **ldapmodify** or **ldapadd** commands is not as flexible as the LDIF mode. However, it is sometimes easier to use than the LDIF mode.

When using modify mode style input, attribute types and values are delimited by an equal sign (=). The general form of input lines for modify mode is:

change_record
<blank line>
change_record
<blank line>
.
.
.

An input file in modify mode consists of one or more *change_record* sets of lines which are separated by a single blank line. Each *change_record* has the following form:

```
| distinguished_name
| [+|-]{attrtype} ={value_line1[\
| value_line2[\
| ...value_lineN]]}
| .
| .
| .
```

Thus, a *change_record* consists of a line indicating the distinguished name of the directory entry to be modified along with one or more attribute modification lines. Each attribute modification line consists of an optional add or delete indicator, an attribute type, and an attribute value. If a plus sign (+) is specified, then the modification type is set to **add**. If a hyphen (–) is specified then the modification type is set to **delete**. If the add or delete indicator is not specified, then the modification type is set to **add** unless the **-r** option is used, in which case the modification type is set to **replace**. Any leading or trailing white-space characters are removed from attribute values. If leading or trailing white-space characters are required for attribute values, then the LDIF mode of input must be used. Lines are continued using a backslash (\) as the last character of the line. If a line is continued, the backslash character is removed and the succeeding line is appended directly after the character preceding the backslash character. The new-line character at the end of the input line is not retained as part of the attribute value.

Multiple attribute values are specified using multiple *attrtype=value* specifications.

The modify mode is not as flexible as the LDIF mode of input. However, it does allow deletion of individual attribute values which the LDIF mode does not support.

Input Mode Examples:

Here are some examples of valid input for the **ldapmodify** command.

Adding a New Entry

```
dn:cn=Tim Doe, ou=Your Department, o=Your Company, c=US
changetype:add
cn: Tim Doe
sn: Doe
objectclass: organizationalperson
objectclass: person
objectclass: top
```

This example adds a new entry into the directory using name `cn=Tim Doe, ou=Your Department, o=Your Company, c=US`.

Adding Attribute Types

```
dn:cn=Tim Doe, ou=Your Department, o=Your Company, c=US
changetype:modify
add:x
telephonenumber: 888 555 1234
registeredaddress: td@yourcompany.com
registeredaddress: ttd@yourcompany.com
-
```

This example adds two new attribute types to the existing entry. Note that the **registeredaddress** attribute is assigned two values.

Changing the Entry Name

```
dn: cn=Tim Doe, ou=Your Department, o=Your Company, c=US
changetype:modrdn
newrdn: cn=Tim Tom Doe
deleteoldrdn: 0
```

This example changes the name of the existing entry to `cn=Tim Tom Doe, ou=Your Department, o=Your Company, c=US`. The old RDN, `cn=Tim Doe`, is retained as an additional attribute value of the **cn** attribute.

Replacing Attribute Values

```
dn: cn=Tim Tom Doe, ou=Your Department, o=Your Company, c=US
changetype:modify
replace:x
telephonenumber: 888 555 4321
registeredaddress: TTD@YOURCOMPANY.COM
-
```

This example replaces the attribute values for the **telephonenumber** and **registeredaddress** attributes with the specified attribute values.

Deleting and Adding Attributes

```
dn:cn=Tim Tom Doe, ou=Your Department, o=Your Company, c=US
changetype:modify
add:x
description: This is a very long attribute
  value that is continued on a second line.
   Note the spacing at the beginning of the
  continued lines in order to signify that
  the line is continued.
-
delete: phone
-
```

This example deletes the **telephonenumber** attribute and adds a **description** attribute. The description attribute value spans multiple lines.

Deleting an Entry

```
dn:cn=Tim Tom Doe, ou=Your Department, o=Your Company, c=US
changetype:delete
```

This example deletes the directory entry with name `cn=Tim Tom Doe, ou=Your Department, o=Your Company, c=US`.

Adding a New Entry

```
cn=Tim Doe, ou=Your Department, o=Your Company, c=US
cn=Tim Doe
sn=Doe
objectclass=organizationalperson
objectclass=person
objectclass=top
```

This example adds a new entry into the directory using name `cn=Tim Doe, ou=Your Department, o=Your Company, c=US`.

Adding New Attribute Types

```
cn=Tim Doe, ou=Your Department, o=Your Company, c=US
+telephonenumber=888 555 1234
+registeredaddress=td@yourcompany.com
+registeredaddress=ttd@yourcompany.com
```

This example adds two new attribute types to the existing entry. Note that the **registeredaddress** attribute is assigned two values.

Replacing Attribute Values

```
cn=Tim Doe, ou=Your Department, o=Your Company, c=US
telephonenumber=888 555 4321
registeredaddress=TTD@YOURCOMPANY.COM
```

Assuming that the command invocation was:

```
ldapmodify -r ...
```

this example replaces the attribute values for the **telephonenumber** and **registeredaddress** attributes with the specified attribute values. If the **-r** command line option was not specified, then the attribute values are added to the existing set of attribute values.

Deleting an Attribute Value

```
cn=Tim Doe, ou=Your Department, o=Your Company, c=US
-registeredaddress=ttd@yourcompany.com
```

This example deletes a single **registeredaddress** attribute value from the existing entry.

Adding an Attribute

```
cn=Tim Doe, ou=Your Department, o=Your Company, c=US
+description=This is a very long attribute \
value that is continued on a second line.  \
Note the backslash at the end of the line to \
be continued in order to signify that \
the line is continued.
```

This example adds a **description** attribute. The **description** attribute value spans multiple lines.

Input Format: An alternative input format is supported for compatibility with older versions of **ldapmodify**. This format consists of one or more entries separated by blank lines, where each entry looks like:

```
distinguished_name
attr=value
[attr=value ... ]
```

where *attr* is the name of the attribute and *value* is the value. By default, values are added. If the **-r** command-line flag is given, the default is to replace existing values with the new one. Note that it is permissible for a given attribute to appear more than once (for example, to add more than one value for an attribute). Also note that you can use a trailing backslash (\) to continue values across lines and preserve new lines in the value itself. The *attr* should be preceded by a dash (-) to remove a value. The equal sign (=) and value should be omitted to remove an entire attribute. The *attr* should be preceded by a plus sign (+) to add a value in the presence of the **-r** flag.

Examples

Following are some **ldapmodify** and **ldapadd** examples:

- Assuming that the file **/tmp/entrymods** exists and has the contents:

```
dn: cn=Modify Me, o=My Company, c=US
changetype: modify
replace: mail
mail: modme@MyCompany.com
-
add: title
title: Vice President
-
add: jpegPhoto
jpegPhoto: /tmp/modme.jpeg
-
delete: description
-
```

  the command:

```
ldapmodify -b -r -f /tmp/entrymods
```

  replaces the contents of the `Modify Me` entry's **mail** attribute with the value `modme@MyCompany.com`, adds a **title** of `Vice President`, and the contents of the file **/tmp/modme.jpeg** as a **jpegPhoto**, and completely removes the **description** attribute. The same modifications as above can be performed using the older **ldapmodify** input format:

```
cn=Modify Me, o=My Company, c=US
mail=modme@MyCompany.com
+title=Vice President
+jpegPhoto=/tmp/modme.jpeg
-description
```

- Assuming that the file **/tmp/newentry** exists and has the contents:

```
dn: cn=Joe Smith, o=My Company, c=US
objectClass: person
cn: Joseph Smith
cn: Joe Smith
sn: Smith
title: Manager
mail: jsmith@jsmith.MyCompany.com
uid: jsmith
```

  the command:

```
ldapadd -f /tmp/newentry
```

  adds a new entry for Joe Smith, using the values from the file **/tmp/newentry**.

- Assuming that the file **/tmp/newentry** exists and has the contents:

```
dn: cn=Joe Smith, o=My Company, c=US
changetype: delete
```

  the command:

```
ldapmodify -f /tmp/newentry
```

  removes Joe Smith's entry.

- Assuming `hostA` contains the referral object:

```
dn: o=ABC,c=US
ref: ldap://hostB:390/o=ABC,c=US
objectclass: referral
```

and `hostB` contains the organization object:

```
dn: o=ABC,c=US
o: ABC
objectclass: organization
telephoneNumber: 123-4567
```

and the file **/tmp/refmods** contains:

```
dn: o=ABC,c=US
changetype: modify
replace: ref
ref: ldap://hostB:391/o=ABC,c=US
-
```

and the file **/tmp/ABCmods** contains:

```
dn: o=ABC,c=US
changetype: modify
add: telephoneNumber
telephoneNumber: 123-1111
-
```

the command:

```
ldapmodify -h hostA -r -V 3 -M -f /tmp/refmods
```

replaces the **ref** attribute value of the referral object `o=ABC,c=US` in `hostA`, changing the TCP port address in the URL from 390 to 391.

The command:

```
ldapmodify -h hostB -p 391 -f /tmp/ABCmods
```

adds the **telephoneNumber** attribute value `123-1111` to `o=ABC,c=US` in `hostB`.

SSL Notes

The contents of a client's key database file is managed with the **gskkyman** utility. See the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for information about the **gskkyman** utility. The **gskkyman** utility is used to define the set of trusted certification authorities (CAs) that are to be trusted by the client. By obtaining certificates from trusted CAs, storing them in the key database file, and marking them as trusted, you can establish a trust relationship with LDAP Servers that use certificates issued by one of the CAs that are marked as trusted.

If the LDAP Servers accessed by the client use server authentication, it is sufficient to define one or more trusted root certificates in the key database file. With server authentication, the client can be assured that the target LDAP Server has been issued a certificate by one of the trusted CAs. In addition, all LDAP transactions that flow over the SSL connection with the server are encrypted, including the LDAP credentials that are supplied on the **ldap_bind** API.

For example, if the LDAP Server is using a high-assurance VeriSign certificate, you should obtain a CA certificate from VeriSign, receive it into your key database file, and mark it as trusted. If the LDAP Server is using a self-signed **gskkyman** server certificate, the administrator of the LDAP Server can supply you with a copy of the server's certificate request file. Receive the certificate request file into your key database file and mark it as trusted.

Using this utility without the **-Z** parameter and calling the SSL-defined port on an LDAP Server (a non-SSL call to an SSL port) is not supported. Also, an SSL call to a non-SSL port is not supported.

Diagnostics

Exit status is 0 if no errors occur. Errors result in a nonzero exit status and a diagnostic message being written to standard error.

ldapmodrdn Utility

Purpose

The **ldapmodrdn** utility is a shell-accessible interface to the **ldap_modrdn** API.

The **ldapmodrdn** utility opens a connection to an LDAP Server, binds, and modifies the RDN of entries. The entry information is read from standard input (**<***entryfile*), from *file* through the use of the **-f** option, or from the command-line pair *dn* and *newrdn*.

Format

```
ldapmodrdn [options] {-f file | < entryfile | dn newrdn }
```

Parameters

options The following table shows the *options* you can use for the **ldapmodrdn** utility:

Table 8 (Page 1 of 2). ldapmodrdn Options

Option	Description
?	Print this text.
-V *version*	Specify the LDAP protocol level the client should use. The value for *version* can be **2** or **3**. The default is **2**.
-c	Continuous operation mode. Errors are reported, but **ldapmodrdn** will continue with modifications. The default is to exit after reporting an error.
-n	Show what would be done, but do not actually change entries. Useful for debugging in conjunction with **-v**.
-r	Remove old RDN values from the entry. Default is to keep old values.
-v	Use verbose mode, with many diagnostics written to standard output.
-R	Do not automatically follow referrals.
-M	Manage referral objects as normal entries. This requires a protocol level of 3 (specify the **-V 3** parameter).
-d *debuglevel*	Set the LDAP debugging level to *debuglevel*.
-D *binddn*	Use *binddn* to bind to the LDAP directory. The *binddn* should be a string-represented DN. The default is a NULL string.
-w *passwd*	Use *passwd* as the password for simple authentication. The default is a NULL string.
-h *ldaphost*	Specify the host on which the LDAP Server is running. The default is the local host. When the target host is an OS/390 LDAP Server operating in multi-server mode with dynamic workload management enabled (see Chapter 5, “Configuring” on page 31 for additional information about LDAP server operating modes), the *ldaphost* value should be in the form *group_name.sysplex_domain_name*, where *group_name* is the name of the **sysplexGroupName** identified in the server configuration file and *sysplex_domain_name* is the name or alias of the sysplex domain in which the target server operates.
-p *ldapport*	Specify the TCP port where the LDAP Server is listening. The default LDAP port is 389. If not specified and **-Z** is specified, the default LDAP SSL port 636 is used.
-Z	Use a secure SSL connection to communicate with the LDAP Server. The **-Z** option is not supported by non-SSL versions of this tool.

Table 8 (Page 2 of 2). ldapmodrdn Options	
Option	**Description**
-K *keyfile*	Specify the name of the SSL key database file. If the key database file is not in the current directory, specify the fully-qualified key database file name. If a key database file name is not specified, this utility will look for the presence of the **SSL_KEYRING** environment variable with an associated file name. Otherwise, no key database file will be used for server authentication and default trusted certification authority roots will be used. The key database file typically contains one or more certificates of certification authorities (CAs) that are trusted by the client. These types of X.509 certificates are also known as trusted roots. For more information on managing an SSL key database file, see the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for a description on the use of the **gskkyman** utility to manage the contents of a key database file. Also see "Securing Your LDAP Server with SSL" on page 65 for more information about SSL and certificates. This parameter is ignored if **-Z** is not specified.
-P *keyfilepw*	Specify the key database file password. This password is required to access the encrypted information in the key database file (including the private key). If the key database file does not contain a private key, which is possible on the LDAP client, then the key database file may have been created without a password. In this case, there is no need to specify a password here. This parameter is ignored if **-Z** is not specified.
-N *keyfiledn*	Specify the certificate name in the key database file.

-f *file* — Read the entry modification information from *file* instead of from standard input or the command line (by specifying *dn* and *newrdn*). Standard input can also be supplied from a file (**<***entryfile*).

entryfile — Specify a file containing the old DN and new RDN on consecutive lines.

dn — Specify the DN of the entry to change.

newrdn — Specify the new RDN for the entry.

Input Format

If the command-line arguments *dn* and *newrdn* are given, *newrdn* replaces the RDN of the entry specified by the DN, *dn*. Otherwise, the contents of *file* (or standard input if no **-f** flag is given) should consist of one or more entries.

```
Distinguished Name (DN)
Relative Distinguished Name (RDN)
```

One or more blank lines may be used to separate each DN/RDN pair.

Example

Assuming that the file **/tmp/entrymods** exists and has the contents:

```
cn=Modify Me, o=My Company, c=US
cn=The New Me
```

the command:

```
ldapmodrdn -r -f /tmp/entrymods
```

changes the RDN of the `Modify Me` entry from `Modify Me` to `The New Me` and the old CN, `Modify Me` is removed.

SSL Notes

The contents of a client's key database file is managed with the **gskkyman** utility. See the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for information about the **gskkyman** utility. The **gskkyman** utility is used to define the set of trusted certification authorities (CAs) that are to be trusted by the client. By obtaining certificates from trusted CAs, storing them in the key database file, and marking them as trusted, you can establish a trust relationship with LDAP Servers that use certificates issued by one of the CAs that are marked as trusted.

If the LDAP Servers accessed by the client use server authentication, it is sufficient to define one or more trusted root certificates in the key database file. With server authentication, the client can be assured that the target LDAP Server has been issued a certificate by one of the trusted CAs. In addition, all LDAP transactions that flow over the SSL connection with the server are encrypted, including the LDAP credentials that are supplied on the **ldap_bind** API.

For example, if the LDAP Server is using a high-assurance VeriSign certificate, you should obtain a CA certificate from VeriSign, receive it into your key database file, and mark it as trusted. If the LDAP Server is using a self-signed **gskkyman** server certificate, the administrator of the LDAP Server can supply you with a copy of the server's certificate request file. Receive the certificate request file into your key database file and mark it as trusted.

Using this utility without the **-Z** parameter and calling the SSL-defined port on an LDAP Server (a non-SSL call to an SSL port) is not supported. Also, an SSL call to a non-SSL port is not supported.

Diagnostics

Exit status is 0 if no errors occur. Errors result in a nonzero exit status and a diagnostic message being written to standard error.

ldapsearch Utility

Purpose

The **ldapsearch** utility is a shell-accessible interface to the **ldap_search** routine.

The **ldapsearch** utility opens a connection to an LDAP Server, binds, and performs a search using the *filter*. If **ldapsearch** finds one or more entries, the *attributes* specified are retrieved and the entries and values are printed to standard output.

Note: Use of the approximate filter (~) is not supported on an OS/390 server.

Format

ldapsearch [*options***] *filter* [***attributes*...]**

Parameters

options The following table shows the *options* you can use for the **ldapsearch** utility:

Table 9 (Page 1 of 3). ldapsearch Options

Option	Description
?	Print this text.
-V *version*	Specify the LDAP protocol level the client should use. The value for *version* can be **2** or **3**. The default is **2**.
-S *method*	Specify the bind method to use. The default is **SIMPLE**. You can also specify **EXTERNAL** to indicate that a certificate (SASL external) bind is requested. The **EXTERNAL** method requires a protocol level of 3 (specify the **-V 3** parameter). You must also specify **-Z**, **-K**, and **-P** to use certificate bind. If there is more than one certificate in the key database file, use **-N** to specify the certificate or the default certificate will be used.
-n	Show what would be done, but do not actually perform the search. Useful for debugging in conjunction with **-v**.
-v	Run in verbose mode, with many diagnostics written to standard output.
-t	Write retrieved values to a set of temporary files. This option assumes values are nontextual (binary), such as **jpegPhoto** or **audio**. There is no character set translation performed on the values.
-A	Retrieve attributes only (no values). This is useful when you just want to see if an attribute is present in an entry and are not interested in the specific values.
-B	Do not suppress display of non-printable values. This is useful when dealing with values that appear in alternate character sets such as ISO-8859.1. This option is implied by the **-L** option.
-L	Display search results in LDIF format. This option also turns on the **-B** option, and causes the **-F** option to be ignored.
-R	Do not automatically follow referrals.
-M	Manage referral objects as normal entries. This requires a protocol level of 3 (specify the **-V 3** parameter).
-d *debuglevel*	Set the LDAP debugging level to *debuglevel*.
-F *sep*	Use *sep* as the field separator between attribute names and values. The default separator is an equal sign (=), unless the **-L** flag has been specified, in which case this option is ignored.

Table 9 (Page 2 of 3). ldapsearch Options

Option	Description
-f *file*	Read a series of lines from *file*, performing one LDAP search for each line. In this case, the *filter* given on the command line is treated as a pattern where the first occurrence of **%s** is replaced with a line from *file*. If *file* is a single hyphen (-) character, then the lines are read from standard input.
-b *searchbase*	Use *searchbase* as the starting point for the search instead of the default. If **-b** is not specified, this utility examines the **LDAP_BASEDN** environment variable for a *searchbase* definition. If you are running in TSO, set the **LDAP_BASEDN** environment variable using LE runtime environment variable **_CEE_ENVFILE**. See the *IBM OS/390 C/C++ Programming Guide* for more information. If you are running in the OS/390 shell, simply export the **LDAP_BASEDN** environment variable.
-s *scope*	Specify the scope of the search. The *scope* should be one of **base**, **one**, or **sub** to specify a base object, one-level, or subtree search. The default is **sub**.
-a *deref*	Specify how alias dereferencing is done. The *deref* should be one of **never**, **always**, **search**, or **find** to specify that aliases are never dereferenced, always dereferenced, dereferenced when searching, or dereferenced only when locating the base object for the search. The default is to never dereference aliases.
-l *timelimit*	Wait at most *timelimit* seconds for a search to complete. Also note the following: • If a client has passed a limit, then the smaller value of the client value, and the value read from **slapd.conf** will be used. • If the client has not passed a limit, and has bound as the **adminDN**, then the limit will be considered unlimited. • If the client has not passed a limit, and has not bound as the **adminDN**, then the limit will be that which was read from the **slapd.conf** file.
-z *sizelimit*	Limit the results of the search to at most *sizelimit* entries. This makes it possible to place an upper bound on the number of entries that are returned for a search operation. Also note the following: • If a client has passed a limit, then the smaller value of the client value, and the value read from **slapd.conf** will be used. • If the client has not passed a limit, and has bound as the **adminDN**, then the limit will be considered unlimited. • If the client has not passed a limit, and has not bound as the **adminDN**, then the limit will be that which was read from the **slapd.conf** file.
-D *binddn*	Use *binddn* to bind to the LDAP directory. The *binddn* should be a string-represented DN. The default is a NULL string.
-w *bindpasswd*	Use *bindpasswd* as the password for simple authentication. The default is a NULL string.
-h *ldaphost*	Specify the host on which the LDAP Server is running. The default is the local host. When the target host is an OS/390 LDAP Server operating in multi-server mode with dynamic workload management enabled (see Chapter 5, “Configuring” on page 31 for additional information about LDAP server operating modes), the *ldaphost* value should be in the form *group_name.sysplex_domain_name*, where *group_name* is the name of the **sysplexGroupName** identified in the server configuration file and *sysplex_domain_name* is the name or alias of the sysplex domain in which the target server operates.
-p *ldapport*	Specify the TCP port where the LDAP Server is listening. The default LDAP port is 389. If not specified and **-Z** is specified, the default LDAP SSL port 636 is used.

Table 9 (Page 3 of 3). ldapsearch Options	
Option	**Description**
-Z	Use a secure SSL connection to communicate with the LDAP Server. The **-Z** option is not supported by non-SSL versions of this tool.
-K *keyfile*	Specify the name of the SSL key database file. If the key database file is not in the current directory, specify the fully-qualified key database file name. If a key database file name is not specified, this utility will look for the presence of the **SSL_KEYRING** environment variable with an associated file name. Otherwise, no key database file will be used for server authentication and default trusted certification authority roots will be used. The key database file typically contains one or more certificates of certification authorities (CAs) that are trusted by the client. These types of X.509 certificates are also known as trusted roots. For more information on managing an SSL key database file, see the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for a description on the use of the **gskkyman** utility to manage the contents of a key database file. Also see "Securing Your LDAP Server with SSL" on page 65 for more information about SSL and certificates. This parameter is ignored if **-Z** is not specified.
-P *keyfilepw*	Specify the key database file password. This password is required to access the encrypted information in the key database file (including the private key). If the key database file does not contain a private key, which is possible on the LDAP client, then the key database file may have been created without a password. In this case, there is no need to specify a password here. This parameter is ignored if **-Z** is not specified.
-N *keyfiledn*	Specify the certificate name in the key database file.

filter Specify an IETF RFC 1558 compliant LDAP search filter. (See **ldap_search** in the *OS/390 LDAP Client Application Development Guide and Reference* for more information on filters.)

attributes Specify a space-separated list of attributes to retrieve. If no *attribute* list is given, all are retrieved.

Output Format

If one or more entries are found, each entry is written to standard output in the form:

```
Distinguished Name (DN)
attributename=value
attributename=value
attributename=value
...
```

Multiple entries are separated with a single blank line. If the **-F** option is used to specify a separator character, it will be used instead of the equal sign (=). If the **-t** option is used, the name of a temporary file is used in place of the actual value. If the **-A** option is given, only the `attributename` part is written.

Examples

Following are some **ldapsearch** examples:

- The command:

  ```
  ldapsearch "cn=karen smith" cn telephoneNumber
  ```

  performs a subtree search (using the default search base) for entries with a **commonName** of `karen smith`. The **commonName** and **telephoneNumber** values are retrieved and printed to standard output. The output might look something like this if two entries are found:

```
cn=Karen G Smith, ou=College of Engineering,
ou=Students, ou=People, o=IBM University, c=US
cn=Karen Smith
cn=Karen Grace Smith
cn=Karen G Smith
telephoneNumber=+1 313 555-9489
cn=Karen D Smith, ou=Information Technology Division,
ou=Faculty and Staff,
ou=People, o=IBM University, c=US
cn=Karen Smith
cn=Karen Diane Smith
cn=Karen D Smith
telephoneNumber=+1 313 555-2277
```

- The command:

  ```
  ldapsearch -t "uid=kds" jpegPhoto audio
  ```

  performs a subtree search using the default *searchbase* for entries with user ID of `kds`. The **jpegPhoto** and **audio** values are retrieved and written to temporary files. The output might look like this if one entry with one value for each of the requested attributes is found:

  ```
  cn=Karen D Smith, ou=Information Technology Division,
  ou=Faculty and Staff,
  ou=People, o=IBM University, c=US
  audio=/tmp/ldapsearch-audio-a19924
  jpegPhoto=/tmp/ldapsearch-jpegPhoto-a19924
  ```

- The command:

  ```
  ldapsearch -L -s one -b "c=US" "o=university*" o description
  ```

  performs a one-level search at the `c=US` level for all organizations whose **organizationName** begins with `university`. Search results are displayed in the LDIF format. The **organizationName** and **description** attribute values are retrieved and printed to standard output, resulting in output similar to this:

  ```
  dn: o=University of Alaska Fairbanks, c=US
  o: University of Alaska Fairbanks
  description: Preparing Alaska for a brave new tomorrow
  description: leaf node only
  dn: o=University of Colorado at Boulder, c=US
  o: University of Colorado at Boulder
  description: No personnel information
  description: Institution of education and research
  dn: o=University of Colorado at Denver, c=US
  o: University of Colorado at Denver
  o: UCD
  o: CU/Denver
  o: CU-Denver
  description: Institute for Higher Learning and Research
  dn: o=University of Florida, c=US
  o: University of Florida
  o: UFl
  description: Shaper of young minds
  ...
  ```

- The command:

  ```
  ldapsearch -h ushost -V 3 -M -b "c=US" "objectclass=referral"
  ```

  performs a subtree search for the `c=US` subtree within the server at host `ushost` (TCP port 389) and returns all referral objects. Note that the search is limited to the single server. No referrals are

followed to other servers to find additional referral objects. The output might look something like this if two referral objects are found:

```
o=IBM,c=US
objectclass=referral
ref=ldap://ibmhost:389/o=IBM,c=US

o=XYZ Company,c=US
objectclass=referral
ref=ldap://XYZhost:390/o=XYZ%20Company,c=US
```

- The command:

```
ldapsearch -h ushost -V 3 -s base -b ""  "objectclass=*"
```

provides the root DSE (DSA-specific entries, where a DSA is a directory server) information including the naming contexts of this server, URLs of referrals objects and alternate servers to contact if this server is unavailable, the LDAP V3 controls, SASL mechanisms and the versions of the LDAP protocol supported by this server. The result might look similar to this:

```
namingcontexts=cn=localhost
namingcontexts=o=IBM,c=US
altserver=ldap://host2.ibm.com:999
altserver=ldap://host3.ibm.com:999
ref=ldap://hostk.ibm.com:391
ref=ldap://host1.ibm.com:333
supportedsaslmechanisms=EXTERNAL
supportedldapversion=2
supportedldapversion=3
supportedcontrol=2.16.840.1.113730.3.4.2
supportedcontrol=1.3.18.0.2.10.2
```

SSL Notes

The contents of a client's key database file is managed with the **gskkyman** utility. See the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for information about the **gskkyman** utility. The **gskkyman** utility is used to define the set of trusted certification authorities (CAs) that are to be trusted by the client. By obtaining certificates from trusted CAs, storing them in the key database file, and marking them as trusted, you can establish a trust relationship with LDAP Servers that use certificates issued by one of the CAs that are marked as trusted.

If the LDAP Servers accessed by the client use server authentication, it is sufficient to define one or more trusted root certificates in the key database file. With server authentication, the client can be assured that the target LDAP Server has been issued a certificate by one of the trusted CAs. In addition, all LDAP transactions that flow over the SSL connection with the server are encrypted, including the LDAP credentials that are supplied on the **ldap_bind** API.

For example, if the LDAP Server is using a high-assurance VeriSign certificate, you should obtain a CA certificate from VeriSign, receive it into your key database file, and mark it as trusted. If the LDAP Server is using a self-signed **gskkyman** server certificate, the administrator of the LDAP Server can supply you with a copy of the server's certificate request file. Receive the certificate request file into your key database file and mark it as trusted. If the LDAP Server accessed by the client is configured to server and client authentication, then the client can transmit its certificate for authentication by the server by marking it as the default in its key database file.

Using this utility without the **-Z** parameter and calling the SSL-defined port on an LDAP Server (a non-SSL call to an SSL port) is not supported. Also, an SSL call to a non-SSL port is not supported.

Diagnostics

Exit status is 0 if no errors occur. Errors result in a nonzero exit status and a diagnostic message being written to standard error.

Running the LDAP Password Encryption Utility

The **db2pwden** utility is an administration utility used to migrate clear text user passwords in the RDBM backend to encrypted passwords. The **db2pwden** utility can be run from the OS/390 shell, or from TSO.

In order to run the **db2pwden** utility in the shell, some environment variables need to be set properly. Ensure that **/usr/sbin** is included in the **PATH** environment variable. Also, make sure **STEPLIB** is set to *GLDHLQ*.SGLDLNK.

db2pwden Utility

The **db2pwden** utility is provided to to encrypt all clear text user passwords in an already loaded RDBM backend. The utility runs as a client operation while the server is active, and causes the server to encrypt all the **userPassword** attribute values that are in clear text with the **pwEncryption** method configured on the LDAP Server. The utility must be run by the LDAP administrator.

Format

db2pwden [*options***]**

Parameters

options The following table shows the *options* you can use for the **db2pwden** utility:

Table 10 (Page 1 of 2). db2pwden Options

Option	Description
?	Print this text
-h *ldaphost*	Specify the host on which the LDAP Server is running. The default is the local host. When the target host is an OS/390 LDAP Server operating in multi-server mode with dynamic workload management enabled (see Chapter 5, “Configuring” on page 31 for additional information about LDAP server operating modes), the *ldaphost* value should be in the form *group_name.sysplex_domain_name*, where *group_name* is the name of the **sysplexGroupName** identified in the server configuration file and *sysplex_domain_name* is the name or alias of the sysplex domain in which the target server operates.
-p *ldapport*	Specify the TCP port where the LDAP Server is listening. The default LDAP port is 389. If not specified and **-Z** is specified, the default LDAP SSL port 636 is used.
-d *debuglevel*	Set the LDAP debugging level to *debuglevel*.
-D *binddn*	Use *binddn* to bind to the LDAP directory. The *binddn* must be the DN of the LDAP Server administrator as defined by **adminDN** in the LDAP Server configuration file. The *binddn* should be a string-represented DN. The default is a NULL string.
-w *bindpasswd*	Use *bindpasswd* as the password for simple authentication. The default is a NULL string.
-b *base*	Use *base* as the starting point for the update instead of the default. If **-b** is not specified, this utility examines the **LDAP_BASEDN** environment variable for a *base* definition. If you are running in TSO, set the **LDAP_BASEDN** environment variable using LE runtime environment variable **_CEE_ENVFILE**. See the *IBM OS/390 C/C++ Programming Guide* for more information. If you are running in the OS/390 shell, simply export the **LDAP_BASEDN** environment variable.
-Z	Use a secure SSL connection to communicate with the LDAP Server. The **-Z** option is not supported by non-SSL versions of this tool.

Table 10 (Page 2 of 2). db2pwden Options

Option	Description
-K *keyfile*	Specify the name of the SSL key database file. If the key database file is not in the current directory, specify the fully-qualified key database file name. If a key database file name is not specified, this utility will look for the presence of the **SSL_KEYRING** environment variable with an associated file name. Otherwise, no key database file will be used for server authentication and default trusted certification authority roots will be used. The key database file typically contains one or more certificates of certification authorities (CAs) that are trusted by the client. These types of X.509 certificates are also known as trusted roots. For more information on managing an SSL key database file, see the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for a description on the use of the **gskkyman** utility to manage the contents of a key database file. Also see "Securing Your LDAP Server with SSL" on page 65 for more information about SSL and certificates. This parameter is ignored if **-Z** is not specified.
-P *keyfilepw*	Specify the key database file password. This password is required to access the encrypted information in the key database file (including the private key). If the key database file does not contain a private key, which is possible on the LDAP client, then the key database file may have been created without a password. In this case, there is no need to specify a password here. This parameter is ignored if **-Z** is not specified.
-N *keyfiledn*	Specify the certificate name in the key database file.

Examples

Following are some **db2pwden** examples:

- The following command:

```
db2pwden -D "cn=admin" -w "secret"
```

  encrypts all the user passwords in the RDBM backend at the LDAP Server on the local host. The encryption method used is the **pwEncryption** method configured on the LDAP Server.

- The following command:

```
db2pwden -h ushost -p 391 -D "cn=admin" -w "secret" -b "o=university, c=US"
```

  encrypts all user passwords starting at the base `"o=university, c=US"` in the RDBM backend at the LDAP Server on host `ushost` at port `391`. The encryption method used is the **pwEncryption** method configured on the LDAP Server.

SSL Notes

The contents of a client's key database file is managed with the **gskkyman** utility. See the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for information about the **gskkyman** utility. The **gskkyman** utility is used to define the set of trusted certification authorities (CAs) that are to be trusted by the client. By obtaining certificates from trusted CAs, storing them in the key database file, and marking them as trusted, you can establish a trust relationship with LDAP Servers that use certificates issued by one of the CAs that are marked as trusted.

If the LDAP Servers accessed by the client use server authentication, it is sufficient to define one or more trusted root certificates in the key database file. With server authentication, the client can be assured that the target LDAP Server has been issued a certificate by one of the trusted CAs. In addition, all LDAP transactions that flow over the SSL connection with the server are encrypted, including the LDAP credentials that are supplied on the **ldap_bind** API.

For example, if the LDAP Server is using a high-assurance VeriSign certificate, you should obtain a CA certificate from VeriSign, receive it into your key database file, and mark it as trusted. If the LDAP Server is using a self-signed **gskkyman** server certificate, the administrator of the LDAP Server can supply you with a copy of the server's certificate request file. Receive the certificate request file into your key database file and mark it as trusted. If the LDAP Server accessed by the client is configured to server and client authentication, then the client can transmit its certificate for authentication by the server by marking it as the default in its key database file.

Using this utility without the **-Z** parameter and calling the SSL-defined port on an LDAP Server (a non-SSL call to an SSL port) is not supported. Also, an SSL call to a non-SSL port is not supported.

Diagnostics

Exit status is 0 if no errors occur. Errors result in a nonzero exit status and a diagnostic message being written to standard error.

Running ldapcp

The **ldapcp** command is a command-line program to assist in administering access to directory data maintained by SLAPD. It provides the ability to create, modify and delete access control lists. See Chapter 7, “Using the ldapcp Command” on page 115 for more information on **ldapcp**. The **ldapcp** command can be run from either the OS/390 shell or from TSO.

Running ldapcp in the OS/390 Shell

In order to run **ldapcp** in the shell, some environment variables need to be set properly. Ensure that **/bin** is included in the **PATH** environment variable. Also, make sure **STEPLIB** is set to *GLDHLQ*.SGLDLNK.

The **ldapcp** command can be run in either command-line mode or in interactive mode. In command-line mode, the full command to be sent to **ldapcp** is entered on the command line. In interactive mode, **ldapcp** is entered without a command on the command line, and, once started, **ldapcp** will display a prompt. Certain information, including the *binddn*, is required. If the user is using interactive mode, and the information is not supplied, **ldapcp** will prompt for the necessary information.

To run **ldapcp** in interactive mode, enter:

```
ldapcp
```

To run **ldapcp** in command-line mode, enter `ldapcp` followed by the necessary parameters. “Flags” on page 116 shows the flags that are valid for the **ldapcp** command.

Running ldapcp in TSO

The **ldapcp** command can be run from TSO. Following are the steps to do this:

1. Create your own data set.
2. Add the necessary environment variables to the data set.
3. Associate an **envvar** file with the utilities run from TSO using the data set name you just created:

   ```
   alloc da('yourdataset') fi(envvar) shr
   ```

   See “Running the LDAP Server Using Data Sets” on page 76 for information on how to create an **envvars** file.
4. Specify the PDS (*GLDHLQ*.SGLDLNK) where the LDAP Server load modules are installed:

   ```
   tsolib act dsn('GLDHLQ.SGLDLNK')
   ```
5. Make the PDS (*GLDHLQ*.SGLDEXEC) containing the CLISTs needed to run the utilities available in SYSEXEC:

   ```
   concatd f(SYSEXEC) da('LDAP.SGLDEXEC')
   ```

Once this setup is complete, running **ldapcp** follows the same syntax as would be used if running it in the OS/390 shell. See “Running ldapcp in the OS/390 Shell.”

Chapter 7. Using the ldapcp Command

The **ldapcp** command is an LDAP Directory Server Access Control List (ACL) and Group Administration Utility.

The **ldapcp** command can be used to do the following:

- Create ACL entries
- Delete ACL entries
- Modify ACL entries
- List ACL entries
- Get ACL entries for specific objects (directory entries)
- Get owners of specific objects (directory entries)
- Remove explicit owners of specific objects (directory entries)
- Create access control groups
- Delete access control groups
- List access control groups
- Add members to access control groups
- Delete members from access control groups
- List members of access control groups

Note that **ldapcp** can only be used to manage ACLs for RDBM entries. Also, it can only display groups and members that are themselves RDBM entities. It is not possible to display SDBM (RACF) group members using **ldapcp**. However, SDBM group or user IDs may appear in an ACL and will be displayed by **ldapcp** when listing ACLs for an object.

| You can find more information about access control in Chapter 13, “Using Access Control” on page 159.

Invoking ldapcp

The **ldapcp** command can be invoked in two modes of operation:

- Interactive-line mode is invoked by entering **ldapcp** without any subcommand arguments (flag arguments are permitted). In interactive-line mode the utility runs each command entered, displays the results, and is immediately ready to accept another command. Only when receiving a **quit** or **exit** command on a line by itself does the program end.
- Command-line mode is invoked by entering **ldapcp** with any flag arguments and a subcommand and verb immediately following the command processor name, indicating to start it in this mode. In command-line mode, the utility accepts the single command passed on invocation, displays the results, and ends immediately. All flags must precede any object names (directory entry names) and verbs.

For both modes above, all input will be read from **stdin**, all output will be written to **stdout**, and all errors and messages will be written to **stderr**.

The **ldapcp** command can be found in the **/bin** directory. In order to run **ldapcp**, be sure to include **/bin** in your **PATH** environment variable.

Syntax

The general format of the **ldapcp** command is:

ldapcp [*flags***] [***subcommand verb* **[***sub-verb***]** *argument***]**

Subcommands and verbs need only contain the minimum number of characters to uniquely identify them. All characters beyond those needed for uniqueness will be ignored. Case is also ignored.

Example

Following is an example of a full subcommand:

```
acl query object
```

Following is an example of the minimum abbreviation for the same command:

```
a q ob
```

Flags

The **ldapcp** command accepts flags at invocation to modify its default operation.

Valid flags are:

-c *reconnect_count*
: Specify the number of times **ldapcp** should attempt to reconnect to the server if communications are lost during processing. Defaults to zero if not specified.

-d *binddn*
: Any function which performs operations against the LDAP Server will authenticate against this DN prior to running. This flag is required when invoked in command-line mode.

  If invoked in interactive mode without providing this flag, the utility program will prompt for the *binddn* immediately upon startup.

-h *ldaphost*
: Specify the host on which the LDAP Server is running. If not specified, the default is the same host where **ldapcp** is running. When the server at the target host is operating in multi-server mode with dynamic workload management enabled (see Chapter 5, “Configuring” on page 31 for additional information about LDAP server operating modes), the *ldaphost* value should be in the form *group_name.sysplex_domain_name*, where *group_name* is the name of the **sysplexGroupName** identified in the server configuration file and *sysplex_domain_name* is the name or alias of the sysplex domain in which the target server operates.

-K *keydatabase*
: Specify the name of the SSL key database file. If the key database file is not in the current directory, specify the fully-qualified key database file name. If **-Z** is specified, **-K** is required. If **-Z** is not specified, **-K** is ignored.

  The key database file typically contains one or more certificates of certification authorities (CAs) that are trusted by the client. These types of X.509 certificates are also known as trusted roots. For more information on managing an SSL key database file, see “Securing Your LDAP Server with SSL” on page 65. Also, the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* describes how to create a key database file using the **gskkyman** utility.

-l *logfile_pathname*
: In addition to writing results to **stdout**, it writes the same output to the log file given as an argument to the option flag. By default, no log file is written. If the log file path name already exists, output is appended to the file.

-P *keydatabasepw* — Specify the key database password. This password is required to access the encrypted information in the key database file, if the key database file has been established with a password. See the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference* for more information on the **gskkyman** utility and key database file passwords. The encrypted information includes the set of trusted root certificates, and optionally a private key with an associated client X.509 certificate. This parameter is ignored if **-Z** is not specified.

-p *ldapport* — Specify a TCP port where the LDAP Server is listening. The default LDAP port is 389. If **-Z** is specified, port is assumed to be an SSL port. The default LDAP SSL port is 636.

-r — Provides parsable output. All fields in each line of output will be contained within double quotation marks. Any human-friendly header and trailer information is suppressed.

-v — Invokes **ldapcp** in verbose mode when a log file has been specified. It causes all commands to be written to the log file ahead of the command results. If errors occur, error messages are also written to the log file. By default, **-l** causes only normal results to be written to the log file. When this flag is used without the **-l** flag, there is no effect on operation (the **-v** flag is ignored).

-w *password* — This must be the password associated with the DN passed with the **-d** flag. This flag is required when invoked in command-line mode.

If invoked in interactive mode without providing this flag, the utility program will prompt for the password.

-Z — Use a secure SSL connection to communicate with the LDAP Server.

Using ldapcp to Administer Remote Server Data

Using the **-h** and **-p** parameters, it becomes possible to administer ACLs and groups on any LDAP Server implementing IBM's ACL implementation using **ldapcp**, including servers running on remote hosts. At this time, this includes LDAP Servers running on IBM UNIX (AIX®), IBM OS/400® and IBM OS/390 systems.

To ensure adequate protection of the ACL and group data, it is recommended that **ldapcp** be run using a secured, or SSL, connection to the server when the server is running on a remote host. If a secured connection is not used, the target server's privileged administrative Distinguished Name (DN) and associated password will be transmitted in clear text to the server. In addition, ACL and group information will also be transmitted in clear text to the server. Transmission in the clear exposes the user to the possibility that the privileged administrator identification may be acquired by unauthorized agents and used to perform server ACL and group administration, thus permitting access to potentially confidential information by unauthorized personnel.

In order to use secure communications for ACL and group administration with **ldapcp**, the target server must be configured to use SSL and to be listening on a secure port. The port specified on the **ldapcp -p** flag must be the secure port where the server is listening. To set up **ldapcp** to use SSL, follow the instructions in “Securing Your LDAP Server with SSL” on page 65, treating **ldapcp** as the client.

Subcommands

Arguments are to be specified within double quotation marks because they may contain imbedded blanks. In command-line mode, the quotation marks must be escaped so as not to be stripped off by the shell prior to **ldapcp** processing.

Following is an example:

```
\"cn=George, ou=Data Processing, o=ABC Company, c=US\"
```

If the input arguments are too long to specify on one line, the backslash character (\) can be used to continue on the next line. The backslash can be used within an argument or between arguments. Following is an example:

```
acl create "cn=Harry, ou=Personnel, o=ABC Company, c =US" \
     "access-id:cn=Phil, ou=Personnel,o=ABC Company,c=US:object:da:normal:rwcs:sensitive:rsc\
     :group:cn=Anybody:normal:rsc"
```

acl create

Creates an access control list.

Format

acl create *object_argument* **[***aclPropagate***]** *acl_argument*

The minimum abbreviation for this subcommand is:

a c *object_argument* **[***aclPropagate***]** *acl_argument*

Parameters

object_argument
The distinguished name of the object (entry) to which this ACL applies.

aclPropagate Is **TRUE** or **FALSE**. The default, if not specified, is **TRUE**.

acl_argument
Is specified as:

aclEntry [**:***aclEntry***...**]

Each *aclEntry* has the form:

subject_DN granted_rights

The *subject_DN* is:

priv_attr_type:*priv_attr_name*

where *priv_attr_type* is either **access-id** or **group**, and *priv_attr_name* is any valid DN which will represent the object (entry) to which privileges are being granted.

The *granted_rights* is specified as follows where *object_rights_list* is one or more elements of the set {**ad**}, and *attr_rights_list* is one or more elements of the set {**rwsc**}.

[**:object:***object_rights_list*] [**:normal:***attr_rights_list*] [**:sensitive:***attr_rights_list*] [**:critical:***attr_rights_list*]

Usage

Use this command to create an ACL for the specified directory entry. If **aclPropagate** is set to **TRUE**, then the ACL that is set will apply to this entry and every entry below this entry in the directory that does not have an ACL associated with it. Creating the ACL will create access permissions (or restrict access) to only the set of permissions granted in the ACL.

Example

```
acl create "cn=Donald,ou=Personnel,o=ABC Company,c=US" "true" \
"access-id:cn=Brian,ou=Personnel,o=ABC Company,c=US:object:a:\
normal:rwsc:sensitive:rsc:critical:s"

acl create "cn=Grant,ou=Personnel,o=ABC Company,c=US" \
"group:cn=Recruitment,ou=Personnel,o=ABC Company,c=US:normal:rsc:\
sensitive:sc"

acl create "cn=Cathy,ou=Personnel,o=ABC Company,c=US" "false" "access-id\
:cn=Diane,ou=Personnel,o=ABC Company,c=US:object:a:normal:rwsc:\
sensitive:rwsc:critical:rwsc"

acl create "cn=John,ou=Personnel,o=ABC Company,c=US" \
"access-id:cn=Jan,ou=Personnel,o=ABC Company,c=US:object:a"

acl create "cn=John,ou=Personnel,o=ABC Company,C=US" \
"access-id:racfid=user01,profiletype=user,sysplex=sysplexa \
:normal:r"
```

Output

None.

acl delete

Deletes an access control list.

Format

acl delete *object_argument*

The minimum abbreviation for this subcommand is:

a d *object_argument*

Parameters

object_argument The distinguished name of the object (entry) to which this ACL applies.

Usage

The **acl delete** subcommand removes access permissions that were granted from the ACL on this entry. If this was an ACL with **aclPropagate**=**TRUE**, then the ACLs on the descendent entries could also change.

Example

```
acl delete "cn=Harry, ou=Personnel, o=ABC Company, c =US"
```

Output

None.

acl modify

Modifies (replaces) an access control list.

Format

acl modify acl *object_argument* **[***aclPropagate***]** *acl_argument*

The minimum abbreviation for this subcommand is:

a m a *object_argument* **[***aclPropagate***]** *acl_argument*

Parameters

object_argument	The distinguished name of the object (entry) for which the ACL will be modified.
aclPropagate	Is **TRUE** or **FALSE**. The default, if not specified, is **TRUE**.
acl_argument	The new (replaced) ACL.

Usage

Use this command to replace an ACL that already exists for an entry in the directory.

In order to add a single ACL entry to an already-defined set of ACL entries within an ACL, you must:

1. Use **acl query object** to display the current ACL entry
2. Use **acl modify** to specify the complete ACL entry information consisting of the original ACL entry plus the ACL entry to be added.

The following syntax for the **acl modify** subcommand is also valid and produces the same results:

acl modify * *object_argument* **[***aclPropagate***]** *acl_argument*

Example

```
acl modify acl "cn=Angela,ou=Personnel,o=ABC Company,c=US" "true" \
"access-id:cn=Brian,ou=Personnel,o=ABC Company,c=US:object:a:\
normal:rwsc:sensitive:rsc:critical:s"

acl modify * "cn=Adam,ou=Personnel,o=ABC Company,c=US" \
"group:cn=Recruitment,ou=Personnel,o=ABC Company,c=US:normal:rsc:\
sensitive:sc"

acl modify acl "cn=Paul,ou=Personnel,o=ABC Company,c=US" "false" "access-id\
:cn=Brian,ou=Personnel,o=ABC Company,c=US:object:a:normal:rwsc:\
sensitive:rwsc:critical:rwsc"

acl modify * "cn=Diane,ou=Personnel,o=ABC Company,c=US" \
"access-id:cn=Donald,ou=Personnel,o=ABC Company,c=US:object:a"
```

Output

None.

acl modify owner

Modifies the owner information for an object (entry).

Format

acl modify owner *object_argument owner ownerPropagate_flag*

The minimum abbreviation for this subcommand is:

a m o *object_argument owner ownerPropagate_flag*

Parameters

object_argument	The distinguished name of the object (entry) for which the owner will be modified.
owner	The distinguished name of the owner.
ownerPropagate_flag	**TRUE** or **FALSE**. If set to **FALSE**, the owner for this object (entry) overrides the owner propagated from the parent entry. (See Chapter 13, “Using Access Control” on page 159 for more information about owner propagation.)

Usage

Use this command to change the designated owner distinguished name for an entry. The owner defaults to the owner specified in the nearest ancestor entry that is designated with **ownerPropagate**=**TRUE**. Modifying the owner to be an **accessGroup** distinguished name can allow multiple distinguished names to update the access permissions on the entry.

| The *inheritOnCreate_flag* parameter is not supported.

Example

```
acl modify owner "cn=Allan, ou=Personnel, o=ABC Company, c =US" \
          "access-id:cn=Jessica, ou=Personnel, o=ABC Company, c=US" \
          "TRUE" "FALSE"
```

Output

None.

acl query object

Gets the access control list for the specified object (entry).

Format

acl query object *object_argument*

The minimum abbreviation for this subcommand is:

a q ob *object_argument*

Parameters

object_argument	The distinguished name of the object (entry) for which the ACL will be returned.

Example

```
acl query object "cn=Anthony, ou=Personnel, o=ABC Company, c=US"
```

Output

```
object = cn=Anthony,ou=Personnel,o=ABC Company,c=US
aclSource = cn=Anthony,ou=Personnel,o=ABC Company,c=US
aclPropagate = FALSE
acl = access-id:cn=Steven, ou=Personnel, o=ABC Company, c=US:object:da:normal:rwcs:sensitive:rsc
acl = group:cn=Anybody:normal:rsc
```

Note: If `object` and `aclSource` are equal, it indicates that this object (entry) has an explicit ACL; otherwise, it inherits its ACL from a parent entry. The `aclSource` identifies the entry whose ACL applies to this entry. (See Chapter 13, “Using Access Control” on page 159 for more information about ACL propagation and inheritance.) This is controlled by the **aclPropagate** option and attribute.

Following is the output for the same query results when using the **-r** flag:

```
"cn=Anthony,ou=Personnel,o=ABC Company,c=US"
"cn=Anthony,ou=Personnel,o=ABC Company,c=US"
"FALSE" "access-id:cn=Steven,ou=Personnel,o=ABC Company,c=US:object:da:normal:rwcs:sensitive:rsc"
"group:cn=Anybody:normal:rsc"
```

acl query owner

Gets the object owner distinguished name for the specified object (entry).

Format

acl query owner *object_argument*

The minimum abbreviation for this subcommand is:

a q ow *object_argument*

Parameters

object_argument	The distinguished name of the object (entry) for which the owner will be returned.

Example

```
acl query owner "cn=Anthony, ou=Personnel, o=ABC Company, c=US"
```

Output

```
object       = cn=Anthony, ou=Personnel, o=ABC Company,c=US
owner        = access-id:cn=Anthony,ou=Personnel, o=ABC Company,c=US
               ownerPropagate:FALSE
explicit     = 1
owner_source = cn=Anthony, ou=Personnel,o=ABC Company, c=US
```

Note: If `explicit=1`, it indicates that the owner for this object (entry) has been explicitly specified; otherwise, the owner is inherited from a parent entry. The `owner_source` identifies the entry which owns this object. (See Chapter 13, “Using Access Control” on page 159 for more information about owner propagation and inheritance.) This is controlled by the **ownerPropagate** option and attribute.

Following is the output for same query results when using the **-r** flag:

```
"access-id:cn=Anthony,ou=Personnel,o=ABC Company,c=US" "FALSE" "FALSE" "1" "cn=Anthony,ou=Personnel,o=ABC Company,c=US"
```

acl remove owner

Removes the **entryOwner** attribute for an object (entry).

Format

acl remove owner *object_argument*

The minimum abbreviation for this subcommand is:

a r o *object_argument*

Parameters

object_argument The distinguished name of the object (entry) for which the **entryOwner** attribute will be removed.

Usage

Use this command to remove an owner that was explicitly specified for a directory entry. Once removed, the owner of the entry reverts back to the distinguished name in the nearest ancestor entry that is designated with **ownerPropagate**=**TRUE**.

Example

```
acl remove owner "cn=Harry, ou=Personnel, o=ABC Company, c =US"
```

Output

None.

exit

Ends the **ldapcp** ACL Administration Utility Program. It is only accepted in interactive mode.

Format

```
exit
```

The minimum abbreviation for this subcommand is:

```
e
```

Usage

See also “quit” on page 135.

group add

Adds members to a group.

Format

group add *group_argument member_argument* **[***member_argument* **...]**

The minimum abbreviation for this subcommand is:

g a *group_argument member_argument* **[***member_argument* **...]**

Parameters

group_argument	The distinguished name of the access group to which members will be added.
member_argument	The distinguished name of a member to be added to the group.

Usage

Use this command to add more distinguished names to an existing access group. The names will be added to the already existing set of names in the access group.

Example

```
group add "cn=Recruitment, ou=Personnel, o=ABC Company, c=US" \
          "cn=John, ou=Personnel, o=ABC Company, c=US" \
          "cn=Adam, ou=Personnel, o=ABC Company, c=US" \
          "cn=Cathy, ou=Personnel, o=ABC Company, c=US"
```

Output

None.

group create

Creates a new access control group.

Format

group create *group_argument member_argument* **[***member_argument* **...]** *common_name_argument*

The minimum abbreviation for this subcommand is:

g c *group_argument member_argument* **[***member_argument* **...]** *common_name_argument*

Parameters

group_argument	The distinguished name of the access control group which will be created.
member_argument	The distinguished name of a member to be added to the access control group (must be at least one member argument present).
common_name_argument	The common name of the access control group to be created.

Usage

Use this command to create an access group that did not exist before. The set of members for the group must be specified and contain at least one distinguished name.

Example

```
group create "cn=Recruitment, ou=Personnel, o=ABC Company, c=US" \
             "cn=Diane, ou=Personnel, o=ABC Company, c=US" \
             "cn=Donald, ou=Personnel, o=ABC Company, c=US" \
             "cn=Grant, ou=Personnel, o=ABC Company, c=US" \
             "cn=Recruitment"
```

Output

None.

group delete

Deletes an access control group.

Format

group delete group *group_argument*

The minimum abbreviation for this subcommand is:

g d g *group_argument*

Parameters

group_argument The distinguished name of the access control group to be deleted.

Usage

Use this command to remove an entire access group. Note that you must be careful when removing access groups. If entries are owned by the access group, their ACL can no longer be modified if the access group no longer exists.

The following syntax for the **group delete** subcommand is also valid and produces the same results:

group delete * *group_argument*

Example

```
group delete group "cn=Recruitment, ou=Personnel, o=ABC Company, c=US"
```

Output

None.

group delete member

Deletes a member or members from an access control group.

Format

group delete member *group_argument member_argument* **[***member_argument* **...]**

The minimum abbreviation for this subcommand is:

g d m *group_argument member_argument* **[***member_argument* **...]**

Parameters

group_argument	The distinguished name of the access control group from which a member will be deleted.
member_argument	The distinguished name of the member to be deleted from the access control group.

Usage

Use this command to delete individual member distinguished names from an access group. This is useful when a person changes departments or job assignments and no longer retains the responsibilities of the access group.

Example

```
group delete member "cn=Recruitment, ou=Personnel, o=ABC Company, c=US" \
                    "cn=Donald, ou=Personnel, o=ABC Company, c=US" \
                    "cn=Grant, ou=Personnel, o=ABC Company, c=US"
```

Output

None.

group list

Lists access control groups for a specific suffix.

Format

group list group *suffix_argument*

The minimum abbreviation for this subcommand is:

g l g *suffix_argument*

Parameters

suffix_argument	The suffix for the directory tree for which an access control group list is desired.

Usage

The following syntax for the **group list** subcommand is also valid and produces the same results:

group list * *suffix_argument*

Example

```
group list group  "ou=Personnel, o=ABC Company, c=US"
```

Output

```
suffix = ou=Personnel,o=ABC Company,c=US
count  = 2
groups = cn=Recruitment,ou=Personnel,o=ABC Company,c=US
         cn=Employee Services,ou=Personnel,o=ABC Company,c=US
```

Following is the output for the same query results when using the **-r** flag:

```
"2" "cn=Recruitment,ou=Personnel,o=ABC Company,c=US" "cn=Employee Services,ou=Personnel,o=ABC Company,c=US"
```

group list member

Lists members in an access control group.

Format

group list member *group_argument*

The minimum abbreviation for this subcommand is:

g l m *group_argument*

Parameters

group_argument The distinguished name of the access control group for which members will be listed.

Example

```
group list member  "cn=Employee Services, ou=Personnel, o=ABC Company, c=US"
```

Output

```
group   = cn=Employee Services,ou=Personnel,o=ABC Company,c=US
count   = 2
members = cn=Kristin,ou=Personnel,o=ABC Company,c=US
        = cn=Angela,ou=Personnel,o=ABC Company,c=US
```

Following is the output for the same query results when using the **-r** flag:

```
"2" "cn=Kristin,ou=Personnel,o=ABC Company,c=US" "cn=Angela,ou=Personnel,o=ABC Company,c=US"
```

help

Provides online help for **ldapcp**.

Format

help [*subcommand***] [***verb***]**

The minimum abbreviation for this subcommand is:

h [*subcommand***] [***verb***]**

Parameters

subcommand	Specify **acl**, **group**, **quit** or **exit**.
verb	See individual subcommand descriptions for list of appropriate verbs.

Example

```
help acl create
help acl
help
```

quit

Ends the **ldapcp** ACL Administration Utility Program. It is only accepted in interactive mode.

Format

quit

The minimum abbreviation for this subcommand is:

q

Chapter 8. Internationalization Support

This chapter discusses translated messages and UTF-8 support.

Translated Messages

The **LANG** and **NLSPATH** environment variables are set for the LDAP Server and the LDAP programs in the server's **envvars** file:

```
/etc/ldap/slapd.envvars
```

There are no default values for these variables. Figure 15 shows the default **slapd.envvars** file. Messages are also available in Japanese. The *lang* variable should be set to **LANG=Ja_JP**. These variables should also be set either in the environment variable file of the user or by exporting the variables in the shell for the user ID that will run the LDAP utilities.

Following is a sample **slapd.envvars** file.

```
#/*************************************************************
# *
# * Licensed Materials - Property of IBM
# * 5647-A01
# * (C) Copyright IBM Corp.  1997
# *
# *************************************************************/
#
LANG=En_US.IBM-1047
NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lib/nls/msg/En_US.IBM-1047/%N
```

Figure 15. slapd.envvars File

With Release 6, symbolic links to the English language message catalogs have also been established in **/usr/lib/nls/msg/C**. It is possible to run with either **LANG=C** or **LANG=En_US.IBM-1047** and access the English language LDAP message catalogs.

UTF-8 Support

UTF stands for “UCS (Unicode) Transformation Format”. The UTF-8 encoding can be used to represent any Unicode character. Depending on a Unicode character's numeric value, the corresponding UTF-8 character is a 1, 2, or 3 byte sequence. Table 11 shows the mapping between Unicode and UTF-8. Refer to IETF RFC 2279 *UTF-8, a transformation format of ISO 10646* for more information on UTF-8.

Table 11. Mapping Between Unicode and UTF-8

Unicode Range (hexadecimal)	UTF-8 Octet Sequence (binary)
0000-007F	0xxxxxxx
0080-07FF	110xxxxx 10xxxxxx
0800-FFFF	1110xxxx 10xxxxxx 10xxxxxx

The LDAP Version 3 protocol specifies that all data exchanged between LDAP clients and servers be UTF-8. Beginning with Release 8, the OS/390 LDAP Server supports UTF-8 data exchange as part of its Version 3 protocol support.

Note: For UTF-8 data stored in an OS/390 LDAP Server's DB2 backend (RDBM), collation for single-byte UTF-8 characters is relative to the server's locale. For multi-byte UTF-8 characters, collation is relative to the numeric value of the equivalent Unicode character.

Part 2. Usage

Chapter 9. Data Model

The LDAP data model is closely aligned with the X.500 data model. In this model, a directory service provides a hierarchically organized set of *entries*. Each of these entries is represented by an *object class*. The object class of the entry determines the set of *attributes* which are required to be present in the entry as well as the set of attributes that can optionally appear in the entry. An attribute is represented by an *attribute type* and one or more *attribute values*. In addition to the attribute type and values, each attribute has an associated *syntax* which describes the format of the attribute values. Examples of attribute syntaxes for LDAP include **caseignorestring** and **binary**.

To summarize, the directory is made up of entries. Each entry contains a set of attributes. These attributes can be single or multivalued (have one or more values associated with them). The object class of an entry determines the set of attributes that must and the set of attributes that may exist in the entry. Refer to the *OS/390 DCE Application Development Guide: Directory Services* for more information on the X.500 Directory Information Model.

Another directory access protocol is XDS/XOM. In XDS/XOM, a complex set of arrays of structures is used to represent a directory entry. In LDAP, this is somewhat simplified. With the LDAP API, a set of C language utility routines is used to extract attribute type and value information from directory entry information returned from an LDAP search operation. Unlike XDS/XOM, attribute values are provided to the calling program in either null-terminated character string form or in a simple structure that specifies a pointer and a length value. Further, attribute types are provided to the program as null-terminated character strings instead of object identifiers.

Chapter 10. Distinguished Names

Every entry in the directory has a *distinguished name (DN)*. The DN is the name that uniquely identifies an entry in the directory. A DN is made up of attribute:value pairs, separated by commas. For example:

```
cn=Ben Gray,ou=editing,o=New York Times,c=US
cn=Lucille White,ou=editing,o=New York Times,c=US
cn=Tom Brown,ou=reporting,o=New York Times,c=US
```

Any of the attributes defined in the directory schema may be used to make up a DN. The attributes and values used, however, must exist as attributes of the entry.

The order of the component attribute:value pairs is important. The DN contains one component for each level of the directory hierarchy. LDAP DNs begin with the most specific attribute (usually some sort of name), and continue with progressively broader attributes, often ending with a country attribute. The maximum length of a DN is 4,000 characters in OS/390.

Relative Distinguished Names

Each component of a DN is referred to as a *relative distinguished name (RDN)*. It identifies an entry distinctly from any other entries which have the same parent. In the examples above, the RDN `cn=Ben Gray` separates the first entry from the second entry, (with RDN `cn=Lucille White`). The attribute:value pair or pairs making up the RDN for an entry must also be present as an attribute:value pair or pairs in the entry. (This is not true of the other components of the DN.)

RDNs can contain multiple attribute:value pairs. So-called multivalued RDNs use two or more attribute:value pairs from the directory entry to define the name of the entry relative to its parent. An example where this would be useful would be where a directory hierarchy of users was being defined for a large university. This hierarchy would be segmented by campus. A problem is encountered, however, when it is discovered that there is more than one John Smith at the downtown campus. The RDN cannot simply be the name of the user. What can be done, however, is add a unique value to the RDN, thus ensuring its uniqueness across the campus. Typically universities hand out serial numbers to their students. Coupling the student number with the person's name is one method of solving the problem of having a unique RDN under a parent in the directory hierarchy. The entry's RDN might look something like: `cn=John Smith+studentNumber=123456`. The plus sign (+) is used to delimit separate attribute:value pairs within an RDN. The entry's DN might look like:

```
cn=John Smith+studentNumber=123456, ou=downtown, o=Big University, c=US
```

Distinguished Name Syntax

The Distinguished Name (DN) syntax supported by this server is based on IETF RFC 1779 *A String Representation of Distinguished Names*. A semicolon (;) character may be used to separate RDNs in a distinguished name, although the comma (,) character is the typical notation.

White space (blank) characters may be present on either side of the comma or semicolon. The white space characters are ignored, and the semicolon replaced with a comma.

In addition, space characters may be present between an attribute:value pair and a plus sign (+), between an attribute type and an equal sign (=), and between an equal sign (=) and an attribute value. These space characters are ignored when parsing.

A value may be surrounded by quotation marks, which are not part of the value. Inside the quoted value, the following characters can occur without any escaping:

- A space or pound sign (#) character occurring at the beginning of the string
- A space character occurring at the end of the string
- One of the characters
  - apostrophe (')
  - equal sign (=)
  - plus sign (+)
  - backslash (\)
  - less than sign (<)
  - greater than sign (>)
  - semicolon (;)

Alternatively, a single character to be escaped may be prefixed by a backslash (\). This method may be used to escape any of the characters listed above, plus the quotation mark.

This notation is designed to be convenient for common forms of name. This section gives a few examples of distinguished names written using this notation. First is a name containing three components:

```
OU=Sales+CN=J. Smith,O=Widget Inc.,C=US
```

This example shows a method of escaping a comma in an organization name:

```
CN=R. Smith,O=Big Company\, Inc.,C=US
```

RACF-style Distinguished Names

If you are using SDBM (described in Chapter 12, "Accessing RACF Information" on page 149), the format of the DNs is restricted in order to match the schema of the underlying RACF data. A RACF-style DN contains three required attributes:

racfid Specifies the user ID or group ID.
profiletype Specifies **user** or **group**.
sysplex Specifies the required attribute for the SDBM suffix.

A RACF-style DN may contain additional attributes that make up the naming context for SDBM. For example, if the naming context for SDBM has been specified as:

```
suffix sysplex=mySysplex,c=US
```

in the SLAPD configuration file, any RACF-style DN would end with:

```
sysplex=mySysplex,c=US
```

Following is an example of the DN format and a sample DN:

racfid=*userid*,profiletype=*user*,sysplex=*mysysplex*

```
racfid=ID1,profiletype=user,sysplex=mySysplex,c=US
```

Chapter 11. Directory Schema

Entries in the directory are made up of attribute:value pairs. Attributes may have one or multiple values. Every entry contains an **objectClass** attribute that identifies what type of information the entry contains. In fact, the object class dictates which other attributes may be present in an entry. The directory schema defines the valid attribute types and object classes which may appear in the directory. Attribute type definitions define the maximum length and syntax of its values. Object class definitions specify which attributes must be present in an object of that class, as well as attributes that may be present. The standard schema definitions shipped with this product are listed in the schema files which are shown in Appendix A, "Configuration Files" on page 241.

The LDAP Server comes with a set of defined attribute types and object classes that can be used to get started using the directory. These are installed into the **/usr/lpp/ldap/etc** directory.

Changing the Configuration Files

The files in **/usr/lpp/ldap/etc** must not contain any customizations. This ensures that any service applied to the LDAP Server does not overlay any local customization of these files. Instead, the easiest approach for customizing the LDAP Server is to make a copy of the original file from the **/usr/lpp/ldap/etc** directory and modify this copy in the **/etc/ldap** directory. The following set of commands is useful in doing this (we use **slapd.conf** as an example):

```
cd /etc/ldap
cp /usr/lpp/ldap/etc/slapd.conf slapd.conf
```

The **cp** command creates a copy of the **/usr/lpp/ldap/etc/slapd.conf** file contents in a new file called **/etc/ldap/slapd.conf**. The **/etc/ldap/slapd.conf** file can now be modified using **oedit** or **vi**. The same technique should be used for the other configuration files created by the LDAP Server install upon the first need to make changes to these files. For your reference, the configuration files shipped in the **/usr/lpp/ldap/etc** directory by the LDAP Server install are:

- **slapd.conf**
- **slapd.envvars**
- **slapd.at.conf** (Do not modify)
- **slapd.at.system** (Do not modify)
- **slapd.oc.conf** (Do not modify)
- **slapd.oc.system** (Do not modify)
- **slapd.at.racf** (Do not modify)
- **slapd.oc.racf** (Do not modify)
- **slapd.cb.at.conf** (Do not modify)
- **slapd.cb.oc.conf** (Do not modify)
- **schema.system.at** (Do not modify)
- **schema.system.oc** (Do not modify)
- **schema.IBM.at** (Do not modify)
- **schema.IBM.oc** (Do not modify)
- **schema.user.at** (Do not modify)
- **schema.user.oc** (Do not modify)

As indicated in the list above, it is recommended that you only modify **slapd.conf** or **slapd.envvars**, if necessary. If you need to add additional schema entries, refer to "Customizing the Schema" on page 146.

Customizing the Schema

It is recommended that existing schema files shipped by IBM (shown in the previous section) should not be modified. Rather, it is recommended that a separate file or set of files be created to hold attribute types and object classes that you need to add. These files can then be included by the server configuration file in **/etc/ldap/lpp/slapd.conf** using the **include** keyword.

The intent of the LDAP directory is that it can be adapted to meet the specific needs of each organization. In directory service terms, this means that it is possible that a company or organization will either modify the existing object classes to suit its own specific needs, or, more commonly, add additional object classes and attribute types that closely model the people, places, and things in the organization. The way in which the LDAP directory is customized in terms of object classes and attribute types is to extend the shipped set of attribute types and object classes.

Adding a New Object Class

To add a new object class to the set of object classes that the LDAP directory understands and accepts, simply add a new object class definition to a configuration file, include the file in the **/etc/ldap/lpp/slapd.conf** file, and restart the LDAP Server. If the new object class contains attributes that are also new, these attribute types must also be defined before the object class will be accepted by the LDAP Server. The same configuration file can be used to define both attribute types and object classes.

As an example, consider a company who wants to keep employee contact information in their directory service. All employees have a full name (common name), surname, telephone number, job title, and e-mail address. However, some employees also have home phone numbers, FAX numbers, and pager numbers. Each of the data elements here corresponds to an attribute type to be stored in an entry. Specifying an object class definition for this type of user would look like:

```
objectclass localPerson
    requires
        objectClass,
        cn,
        sn,
        tn,
        longtitle,
        mailaddress
    optional
        homePhone,
        faxNumber,
        pagerNumber
```

Since `longtitle`, `homePhone`, `faxNumber`, and `pagerNumber` do not exist as attribute types in the default **slapd.at.conf** or **schema.user.at** files, attribute definitions are required for each of these. For example:

```
attribute  longtitle    ces  title       256  normal
attribute  homePhone    cis  homephone   32   normal
attribute  faxNumber    cis  fax         32   normal
attribute  pagerNumber  cis  pager       32   normal
```

A separate file should be created to contain these definitions. For example, a file named `schema.local` could be created in the **/etc/ldap** directory to hold these schema definitions. The **/etc/ldap/slapd.conf** file would then be updated in the global section with the line:

```
include /etc/ldap/schema.local
```

Updating Attribute Types and Object Classes

There are a number of attribute types and object classes that are used by the LDAP Server internally. The directory service depends upon these attribute types and object classes in order to operate. Most of these attribute types and object classes are contained in the **slapd.at.system**, **slapd.oc.system**, **schema.system.at**, and **schema.system.oc** files. In addition, the **slapd.at.conf**, **slapd.oc.conf**, **schema.user.at** and **schema.user.oc** contain a few attribute definitions used by the LDAP Server. Comments in these files indicate which attributes are required.

Keep in mind that the include directive (see "Configuration File Global Options" on page 33 for details on the **include** option) for configuration files can be used within the attribute type, object class definitions, or both. This can be useful if there is a need to allow different portions of a company to add their own object classes and attribute types without interfering with each other. It is recommended that the attribute type and object class definitions be in separate files from the rest of the configuration commands. This is how the default **slapd.at.conf**, **slapd.oc.conf**, **schema.*.at**, and **schema.*.oc** files are shipped.

For compatibility with previous releases of the LDAP Server, the default schema used by the LDAP Server remains as defined in the files:

- **/usr/lpp/ldap/etc/slapd.at.system**
- **/usr/lpp/ldap/etc/slapd.oc.system**
- **/usr/lpp/ldap/etc/slapd.at.conf**
- **/usr/lpp/ldap/etc/slapd.oc.conf**
- **/usr/lpp/ldap/etc/slapd.at.racf**
- **/usr/lpp/ldap/etc/slapd.oc.racf**
- **/usr/lpp/ldap/etc/slapd.cb.at.conf**
- **/usr/lpp/ldap/etc/slapd.cb.oc.conf**

A more extensive set of schema is defined by the files in:

- **/usr/lpp/ldap/etc/schema.system.at**
- **/usr/lpp/ldap/etc/schema.system.oc**
- **/usr/lpp/ldap/etc/schema.IBM.at**
- **/usr/lpp/ldap/etc/schema.IBM.oc**
- **/usr/lpp/ldap/etc/schema.user.at**
- **/usr/lpp/ldap/etc/schema.user.oc**

It is recommended that if you are just starting to use the LDAP Server, you should modify the default configuration to include the **schema.*** files instead of the **slapd.*.conf** files.

The schema definitions in **schema.system.*** define attribute types and object classes required by the LDAP Server. Do not modify these files. The files must be included in your configuration if you use the new schema definition files.

The schema definitions in **schema.IBM.*** define the attribute types and object classes required by IBM directory-enabled products. It is recommended that you include these files in your configuration so that the LDAP Server is ready to support these products.

The schema definitions in **schema.user.*** define attribute types and object classes found in industry-standard schemas. These attribute types and object classes have been extracted from various sources, including the X.520 CCITT standard, IETF RFC 1274 *The COSINE and Internet X.500 Schema*, IETF RFC 2252 *Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions*, and IETF RFC 2256 *A Summary of the X.500(96) User Schema for use with LDAP v3*.

There are a number of attribute types and object classes in these files which are required by the LDAP Server. Comments in the files indicate which attribute types and object classes are required. These files must be included in your configuration if you use the new schema definition files.

To extend this schema, define additional attribute types and object classes in a separate file or set of files, include these files in the LDAP Server configuration file, and restart the LDAP Server. If you are running in a sysplex, the schema must be updated on all systems in the sysplex. All of the servers do not need to be restarted at the same time but they must all be restarted before the new attribute types and object classes are available for use across the sysplex.

Chapter 12. Accessing RACF Information

RACF provides definitions of users and groups, as well as access control for resources. The LDAP Server can provide LDAP access to the user and group information stored in RACF.

Using the RACF database function of the LDAP Server, you can:

- Add new users and groups to RACF
- Modify RACF information for users and groups
- Retrieve RACF information for users and groups
- Delete users and groups from RACF

The SDBM database of the LDAP Server implements portions of the **adduser**, **addgrp**, **altuser**, **altgrp**, **deluser**, **delgrp**, **listuser**, and **listgrp** RACF commands. An individual user has the same authority through SDBM as with normal RACF commands.

The SDBM database allows for directory authentication (or bind) using the RACF user ID and password. The RACF user and group information that make up an identity can be used to establish access control on other LDAP directory entities. This expands use of the RACF identity to the rest of the LDAP-managed namespace.

If the SDBM database is to be used for authentication purposes only, consider having your clients use the **authenticateOnly** server control, to streamline bind processing. This supported control overrides any extended group membership searching and default group membership gathering and is supported for Version 3 clients. See Appendix F, “Supported Server Controls” on page 397 for more information.

Note that the SDBM backend only updates the default RACF on a given system. That is, the **AT** and **ONLYAT** clauses of the RACF commands, used to redirect RACF commands, are not exploited by SDBM.

See the *OS/390 Security Server (RACF) Command Language Reference* for more information about the supported RACF commands.

See “Setting Up Your Server to Run with SDBM” on page 64 for information on getting your LDAP Server configured with SDBM.

Mapping LDAP-Style Names to RACF Attributes

Following are two tables that show the RACF attribute name and the corresponding LDAP-style attribute name for both user (Table 12) and group (Table 13 on page 152).

Table 12 (Page 1 of 4). Mapping of LDAP-Style Names to RACF Attributes (User)

RACF Segment Name	RACF Attribute Name in altuser/adduser String	LDAP-Style Attribute Name
User base or Group base	Not modifiable; listuser displays as CREATED	racfAuthorizationDate
User base or Group base	OWNER	racfSegmentOwner
User base or Group base	DATA	racfInstallationData
User base or Group base	MODEL	racfDatasetModel
DFP segment - common to group or user	DATAAPPL	SAFDfpDataApplication

Table 12 (Page 2 of 4). Mapping of LDAP-Style Names to RACF Attributes (User)

RACF Segment Name	RACF Attribute Name in altuser/adduser String	LDAP-Style Attribute Name
DFP segment - common to group or user	DATACLAS	SAFDfpDataClass
DFP segment - common to group or user	MGMTCLAS	SAFDfpManagementClass
DFP segment - common to group or user	STORCLAS	SAFDfpStorageClass
User base	Multi-value: ADSP, SPECIAL, OPERATIONS, GRPACC, AUDITOR, OIDCARD, UAUDIT	racfAttributes
User base	PASSWORD	racfPassword
User base	Not modifiable - displayed as PASS-INT	racfPasswordInterval
User base	Not modifiable - displayed as PASSDATE	racfPasswordChangeDate
User base	NAME	racfProgrammerName
User base	DFLTGRP	racfDefaultGroup
User base	Not modifiable - displayed as LAST-ACCESS	racfLastAccess
User base	SECLEVEL	racfSecurityLevel
User base	ADDCATEGORY	racfSecurityCategoryList
User base	REVOKE	racfRevokeDate
User base	RESUME	racfResumeDate
User base	WHEN(DAYS())	racfLogonDays
User base	WHEN(TIME())	racfLogonTime
User base	CLAUTH	racfClassName
User base	Not modifiable - displayed as GROUP	racfConnectGroupName
User base	Not modifiable - displayed as AUTH	racfConnectGroupAuthority
User base	Not modifiable - displayed as UACC	racfConnectGroupUACC
User base	SECLABEL	racfSecurityLabel
TSO segment	ACCTNUM	SAFAccountNumber
TSO segment	COMMAND	SAFDefaultCommand
TSO segment	DEST	SAFDestination
TSO segment	HOLDCLASS	SAFHoldClass
TSO segment	JOBCLASS	SAFJobClass
TSO segment	MSGCLASS	SAFMessageClass
TSO segment	PROC	SAFDefaultLoginProc
TSO segment	SIZE	SAFLogonSize
TSO segment	MAXSIZE	SAFMaximumRegionSize

Table 12 (Page 3 of 4). Mapping of LDAP-Style Names to RACF Attributes (User)

RACF Segment Name	RACF Attribute Name in altuser/adduser String	LDAP-Style Attribute Name
TSO segment	SYSOUTCLASS	SAFDefaultSysoutClass
TSO segment	USERDATA	SAFUserdata
TSO segment	UNIT	SAFDefaultUnit
TSO segment	SECLABEL	SAFTsoSecurityLabel
LANGUAGE segment	PRIMARY	racfPrimaryLanguage
LANGUAGE segment	SECONDARY	racfSecondaryLanguage
CICS® segment	OPIDENT	racfOperatorIdentification
CICS segment	OPCLASS	racfOperatorClass
CICS segment	OPPRTY	racfOperatorPriority
CICS segment	XRFSOFF	racfOperatorReSignon
CICS segment	TIMEOUT	racfTerminalTimeout
OPERPARM segment	STOR	racfStorageKeyword
OPERPARM segment	AUTH	racfAuthKeyword
OPERPARM segment	MFORM	racfMformKeyword
OPERPARM segment	LEVEL	racfLevelKeyword
OPERPARM segment	MONITOR	racfMonitorKeyword
OPERPARM segment	ROUTCODE	racfRoutcodeKeyword
OPERPARM segment	LOGCMDRESP	racfLogCommandResponseKeyword
OPERPARM segment	MIGID	racfMGIDKeyword
OPERPARM segment	DOM	racfDOMKeyword
OPERPARM segment	KEY	racfKEYKeyword
OPERPARM segment	CMDSYS	racfCMDSYSKeyword
OPERPARM segment	UD	racfUDKeyword
OPERPARM segment	MSCOPE	racfMscopeSystems
OPERPARM segment	ALTGROUP	racfAltGroupKeyword
OPERPARM segment	AUTO	racfAutoKeyword
WORKATTR segment	WANAME	racfWorkAttrUserName
WORKATTR segment	WABLDG	racfBuilding
WORKATTR segment	WADEPT	racfDepartment
WORKATTR segment	WAROOM	racfRoom
WORKATTR segment	WAADDR1	racfAddressLine1
WORKATTR segment	WAADDR2	racfAddressLine2
WORKATTR segment	WAADDR3	racfAddressLine3
WORKATTR segment	WAADDR4	racfAddressLine4
WORKATTR segment	WAACCNT	racfWorkAttrAccountNumber
User OMVS segment	UID	racfOmvsUid
User OMVS segment	HOME	racfOmvsHome
User OMVS segment	PROGRAM	racfOmvsInitialProgram

Table 12 (Page 4 of 4). Mapping of LDAP-Style Names to RACF Attributes (User)

RACF Segment Name	RACF Attribute Name in altuser/adduser String	LDAP-Style Attribute Name
Netview segment	IC	racfNetviewInitialCommand
Netview segment	CONS	racfDefaultConsoleName
Netview segment	CTL	racfCTLKeyword
Netview segment	MSGRECVR	racfMessageReceiverKeyword
Netview segment	OPCLASS	racfNetviewOperatorClass
Netview segment	DOMAINS	racfDomains
Netview segment	NGMFADM	racfNGMFADMKeyword
DCE segment	UUID	racfDCEUUID
DCE segment	DCENAME	racfDCEPrincipal
DCE segment	HOMECELL	racfDCEHomeCell
DCE segment	HOMEUUID	racfDCEHomeCellUUID
DCE segment	AUTOLOGIN	racfDCEAutoLogin
User OVM segment	UID	racfOvmUid
User OVM segment	HOME	racfOvmHome
User OVM segment	PROGRAM	racfOvmInitialProgram
User OVM segment	FSROOT	racfOvmFileSystemRoot
User OVM segment	HOMEUUID	racfOvmHomeUUID

Table 13 (Page 1 of 2). Mapping of LDAP-Style Names to RACF Attributes (Group)

RACF Segment Name	RACF Attribute Name in altuser/adduser string	LDAP-Style Attribute Name
User base or Group base	Not modifiable; listuser displays as CREATED	racfAuthorizationDate
User base or Group base	OWNER	racfSegmentOwner
User base or Group base	DATA	racfInstallationData
User base or Group base	MODEL	racfDatasetModel
Group base	SUPGROUP	racfSuperiorGroup
Group base	TERMUACC	racfGroupNoTermUAC
Group base	Not modifiable - listgrp displays as SUBGROUP(s)	racfSubGroupName
Group base	Not modifiable - listgrp displays as USERID(s)	racfGroupUserids
Group base	Not modifiable - displayed as ACCESS	racfGroupUserAccess
Group OMVS	GID	racfOmvsGroupId
Group OVM	GID	racfOvmGroupId
DFP segment - common to group or user	DATAAPPL	SAFDfpDataApplication
DFP segment - common to group or user	DATACLAS	SAFDfpDataClass

Table 13 (Page 2 of 2). Mapping of LDAP-Style Names to RACF Attributes (Group)

RACF Segment Name	RACF Attribute Name in altuser/adduser string	LDAP-Style Attribute Name
DFP segment - common to group or user	MGMTCLAS	SAFDfpManagementClass
DFP segment - common to group or user	STORCLAS	SAFDfpStorageClass

There are several RACF fields that can be added using an **ldapadd** operation that cannot be viewed with an **ldapsearch** operation. These fields are related to groups and authorities within groups. They are **racfConnectGroupAuthority** and **racfConnectGroupUACC** and they correspond to the **AUTH** and **UACC** lines displayed by **listuser** for each group a user belongs to. These fields can be viewed using the RACF **listuser** command.

RACF Namespace Entries

When the SDBM database is used to make RACF information accessible over the LDAP protocol, the top three entries in the hierarchy are reserved, read-only, and generated by the server. The purpose of these reserved entries is to enable a hierarchical representation of RACF names and groups in a sysplex. For example, the top three entries in Figure 16 are:

- `sysplex=Sysplex1,o=IBM,c=US` (suffixDN)
- `profileType=User,sysplex=Sysplex1,o=IBM,c=US`
- `profileType=Group,sysplex=Sysplex1,o=IBM,c=US`

The value of **sysplex** in the top DN is generated from the suffix line in the **slapd.conf** file for the SDBM database entry (see “Setting Up Your Server to Run with SDBM” on page 64). The keyword **sysplex** is required to be in the suffix.

Following is a high-level diagram of the RACF backend.

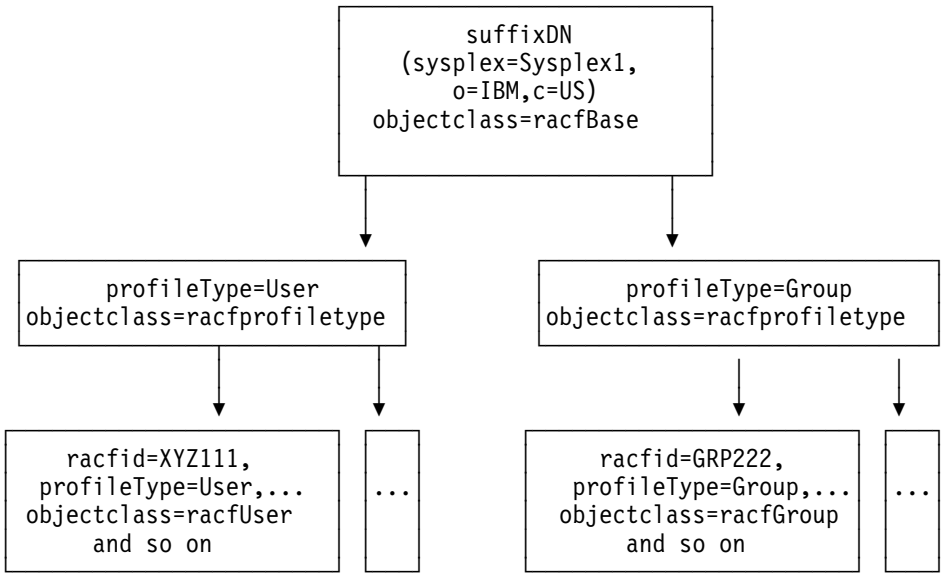


Figure 16. RACF Namespace Hierarchy

SDBM Operational Behavior

Table 14 on page 154 shows how the SDBM database behaves during different LDAP operations.

Table 14. RACF Backend Behavior

Target DN	Add	Modify	Delete	Modrdn	Compare	Search base	Search one level	Search subtree	Bind
suffixDN	Error: Unwilling to perform	Error: Unwilling to perform	Error: Unwilling to perform	Error: Unwilling to perform	Compare attribute	Return requested attributes	Perform a base search against each subordinate of this entry	See "Searching Entire RACF Database" on page 155	Error: No credentials
profiletype =User, suffixDN	Error: Unwilling to perform	Error: Unwilling to perform	Error: Unwilling to perform	Error: Unwilling to perform	Compare attribute	Return requested attributes	See "Searching Entire RACF Database" on page 155	See "Searching Entire RACF Database" on page 155	Error: No credentials
profiletype =Group, suffixDN	Error: Unwilling to perform	Error: Unwilling to perform	Error: Unwilling to perform	Error: Unwilling to perform	Compare attribute	Return requested attributes	See "Searching Entire RACF Database" on page 155	See "Searching Entire RACF Database" on page 155	Error: No credentials
racfid =XYZ111, profiletype =User, suffixDN	Perform an **adduser** RACF command using USER= XYZ111	Perform an **altuser** RACF command using USER= XYZ111	Perform a **deluser** RACF command using USER= XYZ111	Error: Unwilling to perform.	Compare requested attribute with data returned from **listuser**	Perform a **listuser** RACF command using USER= XYZ111	Empty search results (this is a leaf node in the hierarchy)	Perform a **listuser** RACF command using USER= XYZ111	If bind type is not simple, error: Unwilling to perform, else use **_passwd()** to verify the user ID and password combination
racfid =GRP222, profiletype =Group, suffixDN	Perform an **addgrp** RACF command using GROUP= GRP222	Perform an **altgrp** RACF command using GROUP= GRP222	Perform a **delgrp** RACF command using GROUP= GRP222	Error: Unwilling to perform.	Compare requested attribute with data returned from **listgrp**	Perform a **listgrp** RACF command using GROUP= GRP222	Empty search results (this is a leaf node in the hierarchy)	Perform a **listgrp** RACF command using GROUP= GRP222	Error: No credentials

Note about modify: If a request is made to delete a specific attribute value for an attribute where specific values cannot be selectively deleted, **LDAP_UNWILLING_TO_PERFORM** is returned. There are two attributes where specific attribute values are accepted: **racfAttributes** and **racfSecurityCategoryList**. If an attempt is made to delete any attribute that has no corresponding delete command in RACF, **LDAP_UNWILLING_TO_PERFORM** is returned.

If LDAP is running with an SDBM backend, the **ldap_modify** and **ldap_add** APIs can return **LDAP_OTHER** and have completed a partial update to an entry in RACF. The results will match what would occur if the update were done using the RACF **altuser** command. If several RACF attributes are

being updated and one of them is in error, RACF reports on the error, but still updates the other attributes. The RACF message text is also returned in the result.

SDBM Search Capabilities

Table 15 shows the search filters that are supported.

Table 15. RACF Backend Search Filters

Search Base	Filters Supported
`sysplex=YourSysplex` (root of the sysplex directory)	racfid=<*any_value*> objectclass=*
`profileType=user,sysplex=YourSysplex`	racfid=<*any_value*> objectclass=*
`profileType=group,sysplex=YourSysplex`	racfid=<*any_value*> objectclass=*
`racfid=abcdefg,profileType=user,sysplex=YourSysplex`	objectclass=*
`racfid=abcdefg,profileType=group,sysplex=YourSysplex`	objectclass=*

Complex search filters that include NOT, AND, OR, LE, or GE constructs are not supported.

Searching Entire RACF Database: For searches requiring the entire RACF database to be queried, for example, a subtree search from one of the top three directories, you must specify DN (distinguished name) as the only attribute to be returned. You may then obtain more specific data about a particular user/group on a follow-up search using a specific DN as the search base.

Using LDAP Operation Utilities with SDBM

The LDAP operation utilities described in “Using the Command Line Utilities” on page 86 can be used to update data in RACF. Following are some examples.

If the LDIF file **add.mods** contains:

```
dn: racfid=newuser,profiletype=user,sysplex=sysplexa
objectclass: racfUser
racfid: newuser
```

The following command will add user ID "`newuser`" to RACF, assuming *bind_dn* has the authority to add a user:

```
ldapadd -h ldaphost -p ldapport -D bind_dn -w passwd -f add.mods
```

Note that the only required attribute to add a user is the user ID specified as **racfid**. This mimics the RACF **adduser** command.

Now, to add a TSO segment for `newuser`, the LDIF file **mod.mods** could contain:

```
dn: racfid=newuser,profiletype=user,sysplex=sysplexa
changetype: modify
SAFAccountNumber: 123
SAFHoldClass: H
SAFLogonSize: 1024
```

The command:

```
ldapmodify -h ldaphost -p ldapport -D binddn -w passwd -f mod.mods
```

modifies the RACF user profile for user ID **newuser**, adding a TSO segment with the specified values.

Now, to see the information in RACF for **newuser**, the following search command can be performed:

```
ldapsearch -h ldaphost -p ldapport -D binddn -w passwd
  -b "racfid=newuser,profiletype=user,sysplex=sysplexa" "objectclass=*"
```

The results that are returned is everything that RACF displays on a **listuser** command, but using LDAP-style attribute names. Following is an example for **newuser**:

```
racfid=newuser,profiletype=USER,sysplex=sysplexa
objectclass=racfUser
objectclass=racfBaseCommon
objectclass=racfBaseUserSegment
objectclass=SAFTsoSegment
racfid=NEWUSER
racfprogrammername=UNKNOWN
racfowner=racfid=bindid,profiletype=USER,sysplex=sysplexa
racfauthorizationdate=98.001
racfdefaultgroup=racfid=groupid,profiletype=GROUP,sysplex=sysplexa
racfpasswordchangedata=00.000
racfpasswordinterval=90

racfattributes=NONE
racfrevokedata=NONE
racfresumedate=NONE
racflastaccess=UNKNOWN
racfclassname=NONE
racfinstallationdata=NO-INSTALLATION-DATA
racfdatasetmodel=NO-MODEL-NAME
racflogondays=ANYTIME
racflogontime=ANYTIME
racfsecuritylevel=NONE SPECIFIED
racfsecuritycategorylist=NONE SPECIFIED
racfsecuritylabel=NONE SPECIFIED
safaccountnumber=123
safholdclass=H
saflogonsize=00001024
safmaximumregionsize=00000000
safuserdata=0000
1 matches
```

The following command removes the `newuser` user profile for *dn* from RACF (the equivalent of a RACF **deluser** command):

```
ldapdelete -h ldaphost -p ldapport -D binddn -w passwd "racfid=newuser,profiletype=user,sysplex=sysplexa"
```

Deleting Attributes

The **racfAttributes** attribute is treated as a multi-valued attribute in LDAP which represents the flags in the **USER BASE** RACF segment. The available values are:

- ADSP
- AUDITOR
- GRPACC
- OIDCARD
- OPERATIONS
- SPECIAL
- UAUDIT

If a request is made to delete the **racfAttributes** attribute and no values are provided, SDBM generates commands to delete all **racfAttributes** attribute values. Deleting a specific value for **racfAttributes** requires that the value itself be specified on the delete operation. That is, to remove the **SPECIAL** value of **racfAttributes**, specify **-racfAttributes=SPECIAL** in the modification (see "ldapmodify and ldapadd Utilities" on page 90 for more information on command syntax). If a user ID has **SPECIAL**, **ADSP**, **OPERATIONS** as attributes in RACF, deletion of **SPECIAL** and **OPERATIONS** can be accomplished by specifying each of these in a delete clause:

```
racfid=YourID,Profiletype=USER,sysplex=YourSysplex
-racfAttributes=SPECIAL
-racfAttributes=OPERATIONS
```

Following are some additional examples of deleting attributes:

```
dn: racfid=YourID,Profiletype=USER,sysplex=YourSysplex
changetype: modify
delete: racfProgrammerName

returns: LDAP_UNWILLING_TO_PERFORM
```

The **racfProgrammerName** attribute is one that cannot be deleted.

```
racfid=YourID,Profiletype=USER,sysplex=YourSysplex
-racfBuilding=001

returns: LDAP_UNWILLING_TO_PERFORM
```

You cannot specify a value to be removed for **racfBuilding**.

```
racfid=YourID,Profiletype=USER,sysplex=YourSysplex
-racfAttributes=SPECIAL

Expected result: successful removal of only the SPECIAL attribute
and LDAP_SUCCESS returned

dn: racfid=YourID,Profiletype=USER,sysplex=YourSysplex
changetype: modify
delete: racfBuilding

Expected result: successful removal of the attribute racfBuilding
and LDAP_SUCCESS returned

dn: racfid=YourID,Profiletype=USER,sysplex=YourSysplex
changetype: modify
delete: racfAttributes

expected result: successful removal of all attribute values
(including ADSP, AUDITOR, GRPACC, OIDCARD, OPERATIONS, SPECIAL, and UAUDIT)
and LDAP_SUCCESS returned
```

After this operation, racfAttributes will be listed as **none**.

Chapter 13. Using Access Control

Access control of information in the LDAP Server is specified by setting up Access Control Lists (ACLs). ACLs provide a means to protect information stored in an LDAP directory. Administrators use ACLs to restrict access to different portions of the directory, or specific directory entries. LDAP directory entries are related to each other by a hierarchical tree structure. Each directory entry (or object), contains the entry's distinguished name, a set of attributes and their corresponding values. ACLs and groups may be created and managed through the LDAP Directory Server Access Control List and Group Administration Utility (**ldapcp**) described in Chapter 7, “Using the ldapcp Command” on page 115; they may also be created using the **ldif2db** program (see “ldif2db Program” on page 79 for more information).

ACLs are represented by a set of attributes which appear to be a part of the entry. The attributes associated with access control, such as **owner**, **ownerPropagate**, **acl** and **aclPropagate** are unusual in that they are logically associated with each entry, but can have values which depend upon other entries higher in the hierarchy. Depending upon how they are established, these attribute values can be explicit to an entry, or inherited from an ancestor entry.

Use of LDAP's SDBM database allows a user to be authenticated to the directory namespace using the RACF ID and password. The RACF identity becomes associated with the user's RACF-style distinguished name on the bind operation. It is then possible to set up ACLs for namespace entries managed by the RDBM backend using RACF-style user and group DNs. This controls access to RDBM database directory entities using the RACF user ID or group.

| **Note:** The **inheritOnCreate** attribute is not supported. See “Withdrawal of Support for inheritOnCreate
| Attribute” on page 30 for more information.

Access Control Attributes

Access to LDAP directory entries and attributes is defined by Access Control Lists (ACLs). Each entry in the directory contains a special set of attribute/value pairs which describe who is allowed to access information within that entry. Table 16 shows the set of attributes which are related to access control. More in-depth information about each attribute is given following the table.

Table 16. Access Control Attributes

Attribute	Definition
aclEntry	A multivalued attribute which describes access to attributes of the associated LDAP entry, as well as permissions on the entry itself.
aclPropagate	A **TRUE** or **FALSE** flag which indicates whether the ACL should be propagated down the entire directory hierarchy.
aclSource	An attribute that is not user-modifiable which identifies the directory entry with which the ACL information is associated.
entryOwner	The owner of this particular directory entry. The **entryOwner** receives complete access to all attributes of the entry.
ownerPropagate	A **TRUE** or **FALSE** flag which indicates if the owner should be propagated down the entire directory hierarchy.
ownerSource	An attribute that is not user-modifiable which identifies the directory entry with which the owner information is associated.

aclEntry Attribute

An **aclEntry** is a multivalued attribute which contains information pertaining to the access allowed to the entry and each of its attributes. An **aclEntry** lists the following types of information:

- Who has rights to the entry (scope of the protection). Also called the subject.
- What classes of attributes that subject has access to (attribute access classes).
- What rights does the subject have (access permissions).

Scope of Protection: The scope of the protection is based on the following two types of privilege attributes:

access-id The distinguished name of an entity or entry being granted access.

group The distinguished name of the group entity or entry being granted access.

Access control groups have an object class of **accessGroup**. The **accessGroup** object class is a subclass of the **groupOfNames** object class. Groups identified in an **aclEntry** attribute value must be of object class **accessGroup**.

Privilege attributes take the form of *type*:*name* where *type* refers to either **access-id** or **group** and *name* is the distinguished name.

Examples

```
access-id:cn=personA, ou=deptXYZ, o=IBM, c=US
```

In this example, the DN type is `access-id` and the DN itself is `cn=personA, ou=deptXYZ, o=IBM, c=US`.

```
group:cn=deptXYZRegs, o=IBM, c=US
```

In this example, the DN type is `group` and the DN itself is `cn=deptXYZRegs, o=IBM, c=US`.

```
access-id:racfid=YourID,profileType=user,sysplex=YourSysplex
group:racfid=YourGroup,profileType=group,sysplex=YourSysplex
```

This is an example of how to use the RACF identity established with SDBM in an RDBM ACL.

Attribute Access Classes: Attributes requiring similar permission for access are grouped together in classes. Attributes are assigned to an attribute access class within the **slapd.at.*** or **schema.*.at** schema files. The three user-modifiable attribute access classes are:

- **normal**
- **sensitive**
- **critical**

Each of these attribute access classes is discrete. If a user has write permission to **sensitive** attributes, then the user does not automatically have write permission to **normal** attributes. This permission must be explicitly defined.

The default attribute access class for an attribute is **normal** and all users have read access to **normal** attributes. There are two additional attribute access classes used internally by LDAP for system attributes, but users are not permitted to define attributes in these classes.

For example, a person's name would typically be defined in the **normal** class. Perhaps a social security number would be considered **sensitive**, and any password information for the user would be considered **critical**. Following is an example attribute definition:

```
attribute       userPassword    bin               userPassword    128     critical
attribute       dn              distinguishedName dn              1000    normal
```

In the above example, **userPassword** has been assigned membership in the **critical** attribute access class. Similarly, **dn** has been assigned membership in the **normal** attribute access class.

Access Permissions: Following is the set of access permissions.

Table 17. Permissions Which Apply to an Entire Entry

Add	Add an entry
Delete	Delete an entry

Table 18. Permissions Which Apply to Attribute Access Classes

Read	Read attribute value
Write	Write attribute value
Search	Search entries with specified attributes
Compare	Compare attributes

Syntax: Following is the **aclEntry** attribute value syntax:

subject_DN granted_rights

The *subject_DN* is:

priv_attr_type:*priv_attr_name*

where *priv_attr_type* is either **access-id** or **group**, and *priv_attr_name* is any valid DN which will represent the object (entry) to which privileges are being granted.

The *granted_rights* is specified as follows where *object_rights_list* is one or more elements of the set {**ad**}, and *attr_rights_list* is one or more elements of the set {**rwsc**}.

[**:object:***object_rights_list*] [**:normal:***attr_rights_list*] [**:sensitive:***attr_rights_list*] [**:critical:***attr_rights_list*]

Put all together, each attribute value has the form:

<*access-id*|*group*>:<*distinguishedName*> [:*object*:<**a**|**d**|**ad**>][:<**normal**|**sensitive**|**critical**>:<[**r**][**w**][**s**][**c**]>]

aclPropagate Attribute

Each entry with an explicit ACL has associated with it an **aclPropagate** attribute. This attribute indicates whether the entry's explicit ACL is propagated down the directory hierarchy to its descendents, or whether the explicit ACL for this entry is an override for an ACL inherited from the nearest propagating ancestor ACL. See “Propagating ACLs” on page 163 for more information.

aclSource Attribute

Each entry has an associated **aclSource**. This reflects the DN with which the ACL is associated. This attribute is kept by the server, but may be retrieved for administrative purposes.

The derivation of **aclSource** is further explained in “Propagating ACLs” on page 163.

entryOwner Attribute

Each entry has an associated **entryOwner**. The **entryOwner** might be a user or a group, similar to what is allowed within the **aclEntry**. However, the **entryOwner** subject has certain privileges over the entry.

Entry owners are, in essence, the administrators for a particular entry. They have full access on that particular entry, similar to the administrator DN. Note that the administrator DN has full permission on any entry in the database.

Entry owners are not constrained by permissions given in the **aclEntry**; they have complete access to any entry attribute, and can add and delete as desired.

In addition, entry owners (and the administrator DN) are the only people who are allowed to change the attributes related to access control.

ownerSource Attribute

Each entry also has an associated **ownerSource**. This reflects the DN with which the owner values are associated. This attribute is kept by the server, but can be retrieved for administrative purposes.

ownerPropagate Attribute

Owner propagation works exactly the same as ACL propagation. By default, owners are inherited down the hierarchy tree, and their owner propagate attribute is set to **TRUE**. If set to **FALSE**, the owner becomes an override, pertaining only to the particular entry.

Access Determination

Each of the LDAP access permissions is discrete. One permission does not imply another permission. If a user (**access-id**) is given an **aclEntry** for a particular entry, this is the permission set the user receives. If no permission is given to the user, the user receives permission based on the entry's effective ACL, and the user receives the combined permissions of all listed access groups of which they are a member.

If the user is not listed on the ACL, and is not a member of any listed access group, then the user receives the permissions listed under the `group:cn=Anybody` ACL entry. If this ACL entry does not exist, access to the entry is completely denied.

Following are examples for permissions:

```
group:cn=Anybody:normal:rsc
```

In this example, members of the group `cn=Anybody` have permission to read, search and compare all attributes within the **normal** attribute access class.

```
access-id:cn=personA, ou=deptXYZ, o=IBM, c=US:object:ad:normal:rwsc:sensitive:rwsc:critical:rsc
```

In this example, the user corresponding to **access-id** `cn=personA, ou=deptXYZ, o=IBM, c=US` has permission to add and delete entries below the entry, to delete the entry, to read, write, search and compare both **normal** and **sensitive** attributes, and to read, search and compare **critical** attributes.

Attribute Classes and Searching

There are two different ways of performing a search in LDAP: one retrieves both attributes and values and the other retrieves just attribute names. In order to retrieve both attributes and values, the user must have both **r** (read) and **s** (search) permission to the corresponding attributes class. If only the attribute name is desired, only **s** (search) permission is necessary. Returning just attribute names indicates presence of the attribute in the entry.

There are two parts to a search operation related to access control: the attributes creating the search filter and the requested attributes. Access-checking is performed against both groups of attributes.

Filter

If the user does not have access permission to all parts of the search filter, no attributes are returned. Attributes which are intended to be used as search filters should be placed in the **normal** attribute access class. In addition, any entry with world-readable attributes should grant **r** and **s** permissions to attributes in the **normal** attribute access class. World-readable attributes consist of:

- all ACL attributes
- **createdBy**
- **modifiedBy**
- **createTimeStamp**
- **lastModifiedTimeStamp**

Requested Attributes

If the user has the necessary permission on all parts of the filter, the server returns as much information as possible. All requested world-readable attributes are returned, as well as any other attributes to which the user has access.

For example, let the **aclEntry** be

```
group:cn=Anybody:normal:rsc:sensitive:c:critical:c
```

and let `personA` perform an anonymous search

```
ldapsearch -L -b "c=US" cn=LastName title userpassword telephoneNumber
```

where `title` is a **normal** attribute, `telephoneNumber` is a **sensitive** attribute, and `userpassword` is a **critical** attribute.

Users performing anonymous searches are given the permission granted to the **cn=Anybody** group. In this example, permission exists to the filter since **cn** is in the **normal** attribute access class, and **cn=Anybody** has **r** and **s** permission to the **normal** attribute access class. What is returned however, is only the title for any matching entry. The **telephoneNumber** and **userPassword** are not returned since **cn=Anybody** does not have permissions on the **sensitive** and **critical** attribute access classes.

Propagating ACLs

ACLs can be set on any entry in the hierarchy. LDAP ACLs can propagate down through the directory hierarchy. These ACLs, called propagating ACLs, have the **aclPropagate** attribute set to **TRUE**. All descendents of this entry will inherit the ACL set at that point, unless overridden. In order to specify an ACL different from that of its parent, this new ACL must be explicitly set.

When setting the new ACL, there is again a choice of whether to propagate the ACL. If set to **TRUE**, the ACL will propagate down to all descendants. If set to **FALSE**, the ACL is no longer a propagated ACL; it instead becomes an override ACL. The ACL is not propagated down through the hierarchy, but instead applies only to the one particular entry within the hierarchy. If unspecified, **aclPropagate** is set to **TRUE**.

An entry without an explicit ACL receives its ACL from the nearest propagating ancestor ACL. Propagated ACLs do not accumulate as the depth in the tree increases. The scope of a propagated ACL is from the explicitly-set propagating ACL down through the tree until another explicitly-set propagating ACL is found.

Example of Propagation

Following is the explicit ACL for entry `ou=deptXYZ, o=IBM, c=US` :

```
aclPropagate:  TRUE
aclEntry:   group:cn=deptXYZRegs, o=IBM, c=US:normal:rcs:sensitive:rsc
aclEntry:   access-id:cn=personA, ou=deptXYZ, o=IBM, c=US:object:ad:normal:rwsc:sensitive:rwsc:critical:rsc
aclEntry:   group:cn=Anybody:normal:rsc
aclSource:  ou=deptXYZ, o=IBM, c=US
```

In the absence of an explicit ACL for entry `cn=personA, ou=deptXYZ, o=IBM, c=US`, the following is the implicit ACL for the entry:

```
aclPropagate:  TRUE
aclEntry:   group:cn=deptXYZRegs, o=IBM, c=US:normal:rcs:sensitive:rsc
aclEntry:   access-id:cn=personA, ou=deptXYZ, o=IBM, c=US:object:ad:normal:rwsc:sensitive:rwsc:critical:rsc
aclEntry:   group:cn=Anybody:normal:rsc
aclSource:  ou=deptXYZ, o=IBM, c=US
```

In this example, a propagating ACL has been set on `ou=deptXYZ, o=IBM, c=US`. No ACL has been set on the descendant `cn=personA, ou=deptXYZ, o=IBM, c=US`. Therefore, the descendant inherits its ACL value from the nearest ancestor with a propagating ACL. This happens to be `ou=deptXYZ, o=IBM, c=US`, which is reflected in the **aclSource** field. The **aclEntry** and **aclPropagate** values are identical to the explicit propagating ACL set at `ou=deptXYZ, o=IBM, c=US`.

Example of Overrides

Following is an explicit ACL for entry `o=IBM, c=US`:

```
aclPropagate:  TRUE
aclEntry:   group:cn=IBMRegs, o=IBM, c=US:normal:rcs:sensitive:rsc
aclEntry:   group:cn=Anybody:normal:rsc
aclSource:  o=IBM, c=US
```

Following is an explicit ACL for entry `ou=deptXYZ, o=IBM, c=US`:

```
aclPropagate:  FALSE
aclEntry:   group:cn=deptXYZRegs, o=IBM, c=US:normal:rcs:sensitive:rsc
aclEntry:   access-id:cn=personA, ou=deptXYZ, o=IBM, c=US:object:ad:normal:rwsc:sensitive:rwsc:critical:rsc
aclEntry:   group:cn=Anybody:normal:rsc
aclSource:  ou=deptXYZ, o=IBM, c=US
```

| Note that in the explicit ACLs above, **aclSource** is the same as the entry DN. This is required to add an
| explicit ACL.

Following is an implicit ACL for entry `cn=personA, ou=deptXYZ, o=IBM, c=US`:

```
aclPropagate:  TRUE
aclEntry:  group:cn=IBMRegs, o=IBM, c=US:normal:rcs:sensitive:rsc
aclEntry:  group:cn=Anybody:normal:rsc
aclSource:  o=IBM, c=US
```

In this example, a propagating ACL has been set on o=IBM, c=US. An override ACL has been set (**aclPropagate** is **FALSE**) on the descendant ou=deptXYZ, o=IBM, c=US. Therefore, the ACL set at ou=deptXYZ, o=IBM, c=US pertains only to that particular entry.

The descendant cn=personA, ou=deptXYZ, o=IBM, c=US inherits its ACL value from the nearest ancestor with a propagating ACL (which is o=IBM, c=US as reflected in the **aclSource**). The ACL on ou=deptXYZ is skipped because **aclPropagate** is **FALSE**.

Other Examples

In these examples, the administrator DN will be cn=admin, c=US.

The following example shows the default ACL:

```
entryOwner:  access-id:cn=admin,c=US
ownerPropagate:  TRUE
aclPropagate:  TRUE
aclEntry:  group:cn=Anybody:normal:rsc
aclSource:  default
ownerSource:  default
```

The following example shows a typical ACL for entry cn=personA, ou=deptXYZ, o=IBM, c=US:

```
entryOwner:  access-id:deptXYZMgr, ou=deptXYZ, o=IBM, c=US
ownerPropagate:  TRUE
aclPropagate:  TRUE
aclEntry:  group:cn=deptXYZRegs, o=IBM, c=US:normal:rcs:sensitive:rsc
aclEntry:  access-id:cn=personA, ou=deptXYZ, o=IBM, c=US:object:ad:normal:rwsc:sensitive:rwsc:critical:rsc
aclEntry:  group:cn=Anybody:normal:rsc
aclSource:  ou=deptXYZ, o=IBM, c=US
ownerSource:  ou=deptXYZ, o=IBM, c=US
```

This is an inherited ACL and an inherited owner. Both owner properties and ACL properties are inherited from entry ou=deptXYZ, o=IBM, c=US. In this example, members of group cn=deptXYZRegs, o=IBM, c=US have permission to read, search and compare entries in both the **normal** and **sensitive** attribute access classes. They do not have permission to add or delete entries under this entry. Nor do they have permission to access any information or change any information on attributes in the **critical** attribute access class. Unauthenticated, as well as all other bound users, have permission to read, search, and compare attributes in the **normal** attribute access class only. The personA has add and delete permission on the entry; read, write, search, and compare permissions on **normal** and **sensitive** attributes; and read, search, and compare permission on **critical** attributes.

Access Control Groups

Access control groups provide a mechanism for applying the same **aclEntry** attribute values to an entry for multiple users without having to create an explicit **aclEntry** for each user, by including users as members of an access control group.

Access control groups have an object class of **accessGroup**. The **accessGroup** is a subclass of **groupOfNames**.

Each group entry contains a multivalued attribute consisting of member DNs. Groups cannot contain group DNs.

Upon deletion of an access control group, the access control group is also deleted from all ACLs to which it has been applied.

Creating ACLs and Owners Using LDIF-Format Input to ldif2db

As mentioned on page 159, ACLs and groups may be created using the **ldif2db** program. By creating a file containing entries to be added which are in LDAP Data Interchange Format (LDIF), ACLs and groups may be added to the LDAP directory using this file as input to the **ldif2db** program. Following are examples of entries to be added with ACL attributes, owner attributes or both, represented in LDIF format.

The following example creates ACL entries for the Distinguished Name (DN) `cn=Donald,ou=Personnel,o=ABC Company,c=US`:

```
dn: cn=Donald,ou=Personnel,o=ABC Company,c=US
objectclass: person
cn: Donald
sn: DONALD
userPassword: DonaldSecret
aclSource: cn=Donald,ou=Personnel,o=ABC Company,c=US
aclEntry: access-id:cn=Brian,ou=Personnel,o=ABC Company,c=US:object:a
aclEntry: access-id:cn=Brian,ou=Personnel,o=ABC
Company,c=US:normal:rwsc:sensitive:rsc:critical:s
aclEntry: group:cn=Grant,ou=Personnel,o=ABC
Company,c=US:normal:rsc:sensitive:sc
```

This example creates an explicit ACL for the DN `cn=Donald,ou=Personnel,o=ABC Company,c=US`. The ACL that is created contains the **aclEntry** values (specifying access permissions for this DN) for `access-id:cn=Brian,ou=Personnel,o=ABC Company,c=US` and `group:cn=Grant,ou=Personnel,o=ABC Company,c=US`.

Two points in this example should be noted:

- Although the **aclEntry** for `access-id:cn=cn=Brian,ou=Personnel,o=ABC Company,c=US` was broken into separate **aclEntry** attribute values for object and user (**normal**, **sensitive**) DN types, they could be combined into one, if desired.
- Whenever an **aclEntry** is present for a given DN, it *must* be accompanied by a corresponding **aclSource** whose attribute value equals that of the DN for which the ACLs are being created; if the DN for the entry does not equal the DN for the **aclSource**, the **aclEntry** values will be ignored.

The following example will create an explicit owner for the DN `cn=Allan,ou=Personnel,o=ABC Company,c=US`:

```
dn: cn=Allan,ou=Personnel,o=ABC Company,c=US
objectclass: person
cn: Allan
sn: ALLAN
userPassword: AllanSecret
ownerSource: cn=Allan,ou=Personnel,o=ABC Company,c=US
entryOwner: access-id:cn=Jessica,ou=Personnel,o=ABC Company,c=US
ownerPropagate: TRUE
```

This example creates an explicit owner for the DN `cn=Allan,ou=Personnel,o=ABC Company,c=US`. The owner created for this DN is `access-id:cn=Jessica,ou=Personnel,o=ABC Company,c=US`, and will be propagated to children entries in the directory hierarchy, as directed by the **ownerPropagate** attribute being set to **TRUE**.

It should be noted that whenever an **entryOwner** is present for a given DN, it *must* be accompanied by a corresponding **ownerSource** whose attribute value equals that of the DN for which the **entryOwner** is

being created. If the DN for the entry does not equal the DN for the **ownerSource**, the **entryOwner** will be ignored.

When adding ACLs and owners for the same DN in an LDIF file, the ACL and owner attributes must be combined in the same entry stanza. Following is an example of such a stanza:

```
dn: cn=Donald,ou=Personnel,o=ABC Company,c=US
objectclass: person
cn: Donald
sn: DONALD
userPassword: DonaldSecret
aclSource: cn=Donald,ou=Personnel,o=ABC Company,c=US
aclEntry: access-id:cn=Brian,ou=Personnel,o=ABC Company,c=US:object:a
aclEntry: access-id:cn=Brian,ou=Personnel,o=ABC
Company,c=US:normal:rwsc:sensitive:rsc:critical:s
aclEntry: group:cn=Grant,ou=Personnel,o=ABC
Company,c=US:normal:rsc:sensitive:sc
ownerSource: cn=Donald,ou=Personnel,o=ABC Company,c=US
entryOwner: access-id:cn=Jessica,ou=Personnel,o=ABC Company,c=US
ownerPropagate: TRUE
```

Note that when adding ACLs and owners through an LDIF file, the **aclPropagate** and **ownerPropagate** attributes may be omitted from the stanza, in which case these attributes for the entry in question are set to their default values (**TRUE** for both).

Chapter 14. Replication

Once the OS/390 LDAP Server is installed and configured, users can access the directory, add objects, delete objects, or perform search operations to retrieve particular sets of information.

Replication is a process which keeps multiple databases in sync. Through replication, a change made to one database is propagated to one or more additional databases. In effect, a change to one database shows up on multiple different databases. This means there are two types of databases: masters and replicas.

Master — All changes to the database are made to the master server. The master server is then responsible for propagating the changes to all other databases. It is important to note that while there can be multiple databases representing the same information, only one of those databases can be the master.

Replica — Each of the additional servers which contain a database replica. These replica databases are identical to the master database.

Replication from a Master server to a Replica is only supported when the Master server is running in single-server mode. Refer to Chapter 5, “Configuring” on page 31 for more information about server operating modes.

Password Encryption and Replication

To ensure data integrity and the proper working of the LDAP Servers in the replication environment, the **pwEncryption** option in the configuration files for the master and slave servers must be the same. If one of the servers involved in replication is a non-OS/390 server, then the administrator must choose a **pwEncryption** method that is supported by both servers for correct operation of replication. If no encryption methods are common between the servers, then password encryption should not be used.

For crypt encryption, note that the values returned by the crypt algorithm are not portable to other X/Open-conformant systems. This means the crypt() algorithm cannot be used for replication between OS/390 and a non-OS/390 server.

For DES encryption, where both the master and slave servers are OS/390 LDAP Servers, the same DES key label and data key must be defined on both OS/390 systems through the ICSF KGUP and CKDS facilities. (See the information on managing cryptographic keys in the *OS/390 ICSF Administrator's Guide* for more details.) This key label must be used in the configuration files of both of the LDAP Servers involved in replication.

Benefits of Replication

There are several benefits realized through replication. The single greatest benefit is providing a means of faster searches. Instead of having all search requests directed at a single server, the search requests can be spread among several different servers. This improves the response time for the request completion.

Additionally, the replica provides a backup to the master server. Even if the master server crashes, or is unreadable, the replica can still fulfill search requests, and provide access to the data.

Although replication is not supported when operating multiple concurrent server instances against the same RDBM database (multi-server operating mode), similar benefits are afforded when operating in this mode.

Master Server

In order for the replication process to occur, the following must happen:

- The master must be aware of each replica that is to receive the change information.
- Each replica must be aware of the master server for the database that it serves.

The master server becomes aware of the existence of the replica databases when objects of type **replicaObject** are added to the directory. Each of these objects represents a particular replica server. The attribute/value pair within the replica object provide the information the server needs in order to find the replica and send any updates to that server.

Replica Objects

The **replicaObject** object class is provided in the system schema files **slapd.oc.system** and **schema.system.oc**. Like all other LDAP object class definitions, the **replicaObject** has mandatory and optional attributes. Each of the **replicaObject** attributes are single-valued. The following is an example of a **replicaObject** definition.

Table 19. Replica Object Schema Definition (Mandatory Attributes)

Attribute	Description and Example
replicaHost	Represents the Internet name of the machine. This could be an IP address, such as, 127.0.0.1, or a DNS name such as `myMachine.endicott.ibm.com`. Example: `replicaHost: myMachine.endicott.ibm.com`
replicaBindDN	Specifies the LDAP distinguished name that the master uses to bind to the replica when sending directory updates. The **replicaBindDN** and the **masterServerDN** in the replica configuration file must be the same. Example: `replicaBindDN: cn=Master`
replicaCredentials	Contains the authentication information needed for the master server to authenticate to the replica using the **replicaBindDn**. Example: `replicaCredentials: secret`
cn	Forms the RDN of the LDAP distinguished name. Example: `cn: myReplica`

In Table 19 when the master server receives and successfully finishes an update request, the update is also sent to `myMachine.endicott.ibm.com` on port 389. The master performs a bind operation using the **replicaBindDN** of `cn=Master` and password of `secret`.

In addition, there are several attributes available that provide additional flexibility in configuring a replica server. For instance, an added description could better describe the replica server, and it could listen on a different port then the default port of 389. Adding a description and changing the port to 400 is shown in Table 20 on page 171

Table 20. Replica Object Schema Definition (Optional Attributes)

Attribute	Description and Example
replicaPort	Describes the port number on which the replica is listening for incoming requests. By default, the server listens on port 389. Example: `replicaPort: 400`
replicaUpdateTimeInterval	Delays the propagation of additional updates for specified number of seconds. The default is for the server to send updates immediately. Example: `replicaUpdateTimeInterval: 3600`
replicaUseSSL	Determines whether the master should replicate over SSL. The default is for the the master not to replicate using SSL. Example: `replicaUseSSL: TRUE`
description	Provides an additional text field for extra information pertaining to the replica object. Example: `description: Replica Machine in the fourth floor lab.`
seeAlso	This optional attribute identifies another directory server entry that may contain information related to this entry. Example: `seeAlso: cn=Alternate Code, ou=Software, o=IBM, c=US`
replicaBindMethod	This optional attribute identifies the bind method to be used. If it is specified, it must be set to **simple**. Example: `replicaBindMethod: simple`

Localhost Suffix

During installation, the object corresponding to the **cn=localhost** is automatically created. There are several important characteristics of this object:

- It has an object class of **replicaObject**.
- It is the only suffix which is automatically created.
- It is the only object which is automatically created.
- It cannot be removed.
- Objects under **cn=localhost** are pertinent only to that particular machine.

Adding Replica Objects

The placement of the replica object within the LDAP directory tree is critical. All replicas that have **objectclass=replicaObject** must be added under the **cn=localhost** suffix. This is the only type of object that can be added under the **cn=localhost** suffix. An object with **objectclass=replicaObject** cannot be added under any other suffix, such as **c=US**. Similarly, an object with **objectclass=person** cannot be added under **cn=localhost**. Following is an example of a replica object definition using LDIF format.

```
dn: cn=myReplica,cn=localhost
cn: myReplica
objectclass: replicaObject
replicaHost: myMachine.endicott.ibm.com
replicaBindDn: cn=Master
replicaCredentials: secret
replicaPort: 400
replicaUseSSL: FALSE
description: "Replica machine in the fourth floor lab."
```

Replica Server

Initialization, or population, of a replica database requires several steps.

Populating a Replica

1. Stop the LDAP master server.
2. Perform a **db2ldif** of the master server's directory contents if there are any entries.
3. Perform an **ldif2db** with a single added directory entry which defines a **replicaObject** entry into the master server's directory contents.
4. If the master database does not contain any entries, no further action must be taken to ensure that the replica and master server are in sync and the master server can now be restarted; otherwise, continue to the next step.
5. Transport the LDIF file created in step 2 to the replica server's location.
6. Stop the replica server if it is running.
7. Perform an **ldif2db** on the replica server.
8. Start the replica server.
9. Start the master server.

Configuring the Replica

The key to a successful replica configuration rests in ensuring that the values in the **replicaObject** accurately represent the target (or relevant) values on the replica server. The following values between the **replicaObject** and the master server, and the replica configuration must be identical and the replica configuration on a replica server must be identical.

Replica Object	Replica Server Configuration
replicaPort	**port**
replicaHost	**masterServer**
replicaBindDn	**masterServerDN**
replicaCredentials	**masterServerPW**

Notes:

1. It is recommended that the **masterServerDN** be a DN that is dedicated specifically to replication. It should not be used for any other operations.
2. The **masterServer**, **masterServerDN**, and **masterServerPW** entries must follow the database definition entry in the SLAPD configuration file.
3. The **replicaHost** and **masterServer** represent the same server in different formats. The **masterServer** is the LDAP URL equivalent to the value for the **replicaHost**.

LDAP Operations on Replicas

The basic type of LDAP operations:

- Searches
- Updates

Update operations, such as add, delete, modify, and modify RDN, should not be performed against a replica server. These operations fail if performed using the utilities (**ldapadd**, **ldapdelete**, **ldapmodify**, **ldapmodrdn**). Changes must be made to the master database, which then propagates the change to the replica.

If update operations are sent to a replica server, the master server set in the replica configuration is returned and the operation is referred to the master server and is then propagated to the replica server. For compatibility purposes, if **masterServer** is not specified, the first default referral found in the replica configuration file is used as the master server and all update operations are returned to that server.

In order to maintain database integrity, the **ldif2db** program should be used on a replica only when initially populating the replica database. If **ldif2db** is used to add entries to a replica server after initial population, these changes are not reflected in the master database. The replica database is corrupted, and could give erroneous information. See “SSL and Replication” on page 174 for information about securing a database.

Changing a Replica to a Master

At some point, it may become desirable to change one of the replicas to be the master. Perhaps the machine where the replica server is installed is being upgraded, and the customer wishes this replica to now be the master LDAP Server.

The following procedure should be followed to change a replica to a master:

1. Ensure all of the data from the master resides in the replica. This can be done by dumping both databases using **db2ldif** and comparing the output. If the replica is out of sync with the master, follow the procedure for correcting out-of-sync conditions.
2. Remove the **replicaObjects** from the master server.
3. Stop the master and the replica servers.
4. Remove the **masterServer**, **masterServerDN**, and **masterServerPW** directives from the replica's configuration file.
5. If the original master is being eliminated, simply drop the databases on the original master. (See “Creating the LDAP Server DB2 Database and Table Spaces” on page 14 for examples of the SPUFI commands needed to drop the databases.) The new master can now be started.
6. If the original master is going to become a replica:

a. Add the **replicaObject** to the new master database using **ldapadd** or **ldif2db**.

b. Add the **masterServer** directive to the new replica's configuration file. Make sure it points to the new master.

c. Add the **masterServerDN** and **masterServerPW** directives to the new replica's configuration file.

d. Start the servers.

SSL and Replication

SSL can be used to communicate between a Master and Replica LDAP Server.

Replica Server with SSL Enablement

Set the replica server up for SSL just like a normal SSL server. It needs its own public-private key pair and certificate, and the configuration file needs the standard SSL keywords set (**sslKeyRingFile** and **sslKeyRingFilePW** configuration file options). See "Securing Your LDAP Server with SSL" on page 65 for more information.

Master Server with SSL Enablement

The master server acts like an SSL client to the replica server.

To set up the master server, you must:

1. Run the **gskkyman** utility (see the *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference*), this time as if you were a client. You should use the same key database file that contains the master server's key pair and certificate. Receive the replica's self-signed certificate and mark it as trusted.
2. In the master server's configuration file:
   - Set the **sslKeyRingFile** to the replica key database file name created above. (The **replKeyRingFile** option is no longer evaluated by the LDAP Server.)
   - Set the **sslKeyRingFilePW** to the password for the replica key database file. (The **replKeyRingFile** option is no longer evaluated by the LDAP Server.)
3. In the replica object:
   - Set the **replicaPort** keyword to the replica's secure port number.
   - Set the **replicaUseSSL** keyword to **TRUE**.

See "Securing Your LDAP Server with SSL" on page 65 for more information.

Since the master server acts like an SSL client to the replica server, the master server binds with the replica server. The bind method used is **simple** bind. The SASL external bind method is not supported for replication.

Troubleshooting

If the replica server does not seem to be receiving updates from the master server, there are several possible reasons. Check the following conditions for a possible quick fix:

- Check for messages from the LDAP Server.
- Use **ldapsearch** with a base of **cn=localhost** and a filter of **objectClass=*** to verify that the replica object exists in the master database, and was specified correctly to match with the replica server.

- Check that the values listed in the replica object for that replica match those of the replica server configuration. Specifically, the **replicaPort**, **replicaBindDN**, and **replicaCredentials** should be verified.
- Check that the **replicaUpdateTimeInterval** specified in the server replica object has been set correctly.
- Verify that the replica server is running by performing an **ldapsearch** against the replica.
- Check that the default referral specified in the replica's configuration file points to the master server.
- If the **replicaObject** attribute **replicaUseSSL** is set to **TRUE**, verify the **replicaObject** attribute **replicaPort** is set to the SSL port configured on the replica server, and verify the **sslKeyRingFile** and **sslKeyRingFilePW** are correct.

Recovering from Out-of-Sync Conditions

If a replica becomes out-of-sync with its master for any reason, and normal replication processing is not correcting the situation, it may be necessary to reload the replica.

The following procedure should be followed to reload a replica:

1. Stop both the master and replica servers.
2. Drop and recreate the table spaces on the replica server. (See “Creating the LDAP Server DB2 Database and Table Spaces” on page 14 for an example of the SPUFI commands needed to drop and recreate the table spaces.)
3. Run **db2ldif** on the master, getting the LDIF-formatted version of the data in the master.
4. Run **ldif2db** on the replica, using the LDIF-formatted data retrieved from the master, above.
5. Start both servers.

Chapter 15. Referrals

Referrals provide a way for servers to refer clients to additional directory servers. With referrals you can:

- Distribute namespace information among multiple servers
- Provide knowledge of where data resides within a set of interrelated servers
- Route client requests to the appropriate server

Following are some of the advantages of using referrals:

- Distribute processing overhead, providing primitive load balancing
- Distribute administration of data along organizational boundaries
- Provide potential for widespread interconnection, beyond an organization's own boundaries.

This chapter describes how to use the referral object class and the **ref** attribute to construct entries in an LDAP directory server containing references to other LDAP directory servers. Also described in this chapter is how to associate multiple servers using referrals and an example of associating a set of servers through referrals and replication (see Chapter 14, “Replication” on page 169).

Using the Referral Object Class and the ref Attribute

The referral object class and the **ref** attribute are used to facilitate distributed name resolution or to search across multiple servers. The **ref** attribute appears in an entry named in the referencing server. The value of the **ref** attribute points to the corresponding entry maintained in the referenced server. While the distinguished name (DN) in a value of the **ref** attribute is typically that of an entry in a naming context below the naming context held by the referencing server, it is permitted to be the distinguished name of any entry. A multi-valued **ref** attribute may be used to indicate different locations for the same resource. If the **ref** attribute is multi-valued, all the DNs in the values of the **ref** attribute should have the same value.

Creating Entries

Following is an example configuration that illustrates the use of the **ref** attribute.

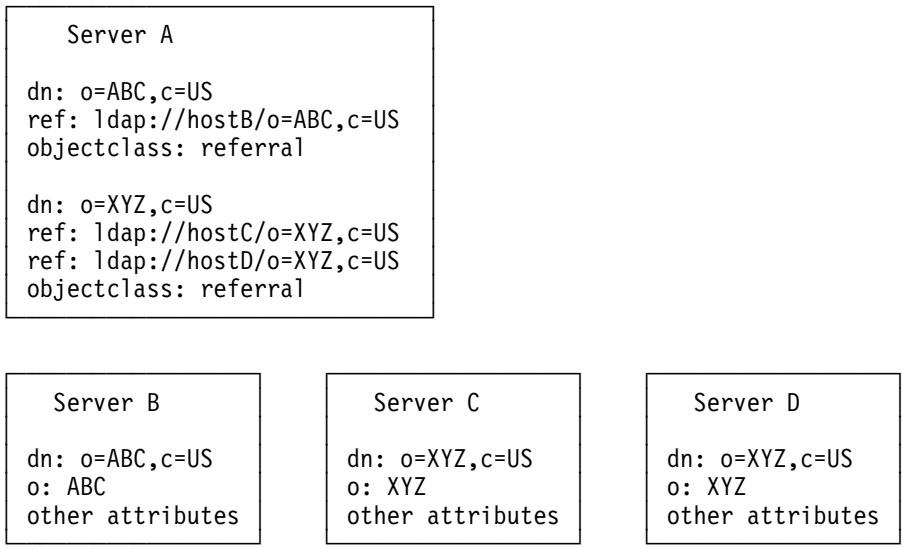


Figure 17. Example Using ref Attribute

In the example, Server A holds references to two entries: `o=ABC,c=US` and `o=XYZ,c=US`. For the `o=ABC,c=US` entry, Server A holds a reference to Server B and for the `o=XYZ,c=US` entry, Server A holds references to two equivalent servers, Server C and Server D.

The recommended setup of referrals is to structure the servers into a hierarchy based on the subtrees they manage. Then, provide "forward" referrals from servers that hold higher information and set the default referral to point back to its parent server.

Associating Servers with Referrals

In order to associate servers through referrals:

- Use referral objects to point to other servers for subordinate references
- Define the default referral to point somewhere else, typically to the parent server

These steps are defined below.

Pointing to Other Servers

Use referral objects to point to the other servers for subordinate references. That is, portions of the namespace below this server which it does not service directly.

Referral objects, like other objects, go in the backend (DB2). Referral objects consist of:

dn: Specifies the distinguished name. It is the portion of the namespace served by the referenced server.

objectclass: Specifies **referral**.

ref: Specifies the LDAP URL of the server. This URL should consist of the **ldap://** identifier, the *hostname*:*port*, and a DN. The DN requires a slash (/) before it to delimit it from the *hostname*:*port*, and should match the DN of the referral object. The **ref:** attribute may be multi-valued, with each value specifying the LDAP URL of a different server. When multiple values are used, each LDAP URL should contain the same DN, and each server should hold equivalent information for the portion of the namespace represented by the DN.

Following is an example:

```
dn:           o=IBM,c=US
objectclass:  referral
ref:          ldap://Host1:389/o=IBM,c=US
ref:          ldap://Host2:389/o=IBM,c=US
ref:          ldap://Host3:1389/o=IBM,c=US
```

The server can have any number of referral objects within its database. However, the objects must essentially be descendents of its suffix.

Defining the Default Referral

Define the default referral to point to another server which services other portions of the namespace unknown to the referencing server. The default referral can be used to point to:

- The immediate parent of this server (in a hierarchy)
- A "more knowledgeable" server, such as the uppermost server in the hierarchy
- A "more knowledgeable" server which possibly serves a disjoint portion of the namespace.

The default referral goes in the configuration file and not the backend. The default referral is described in the configuration file with the **referral** keyword and an LDAP URL. Multiple default referrals may be specified. However, each one specified is considered equivalent; that is, each server referenced by a default referral should present the same view of the namespace to its clients.

Note: The default referral LDAP URL does not include the DN portion. It should just have the **ldap://** identifier and the *hostname*:*port*. For example:

```
referral  ldap://host3.ibm.com:999
```

See "Operating in Multi-server Mode Without Dynamic Workload Management Enabled" on page 48 and "Operating in Multi-server Mode With Dynamic Workload Management Enabled" on page 49 for information about setting up multiple default referrals.

SSL note: A non-SSL client referral to an SSL port is not supported. Also, an SSL client referral to a non-SSL port is not supported.

Processing Referrals

When clients request information from servers which do not hold the needed data, servers can pass back referral URLs which indicate one or more other servers to contact. The clients can then request the information from the referenced server. The OS/390 client API, by default, chases referrals returned from servers. However, client applications can suppress referral chasing through the **ldap_set_option()** API. This option's scope is the LDAP handle, so a client could open multiple connections to one or more servers, some of which would chase referrals automatically, and some of which would not.

Servers present the referral URLs differently depending on the LDAP protocol version being used by the client. Referrals are presented to LDAP Version 2 clients in the error string, as the protocol does not provide a specific mechanism for indicating referrals. In LDAP Version 3, protocol elements are specifically defined to allow servers to present referral information to clients.

Using LDAP Version 2 Referrals

LDAP Version 2 referrals are presented as part of the error string passed back to the client. Since clients do not generally examine the error string on results indicating **LDAP_SUCCESS**, some other error code is needed. Thus, on wholly successful operations, the server passes back a result of **LDAP_PARTIAL_RESULTS** instead of **LDAP_SUCCESS** to indicate the presence of referral information. On any result other than **LDAP_SUCCESS**, referral information might also be present in the error string. Note that some servers might return a result of **LDAP_PARTIAL_RESULTS** with no referral information if the server does not have a default referral defined and the client makes a request for a portion of the namespace outside the server (and not below it in the hierarchy).

The referral information in the error string looks like this:

```
Referral:\n
ldap://hostname:port/DN\n
...
```

where `Referral:` is followed by a new line code (`\n`) and `ldap://hostname:port/DN\n` is an LDAP URL followed by a new line code. The ellipses (`...`) indicate a list of multiple referrals; that is, more LDAP URLs followed by new line codes. Multiple referrals are only presented for partial search results when it is necessary to contact more than one additional server to complete the entire request. This would indicate that multiple referral objects were found in the referencing server that matched the search criteria. The client contacts every server presented in the list to continue the search request. For referral objects that have multi-valued **ref** attributes, the server sends only one of the LDAP URLs to a client using LDAP Version 2 protocol. This is because there is no provision for distinguishing between equivalent servers to

contact (as indicated by multi-valued **ref** attributes) and multiple servers which must be contacted to complete a search request.

Limitations with LDAP Version 2 Referrals: One limitation of LDAP Version 2 referrals is the lack of support for alternate referrals. As noted earlier, a referral object can have a multi-valued **ref** attribute which indicates different servers which hold equivalent information. As will be seen below, it is only possible for the LDAP Version 2 referral mechanism to contain a single list of referrals for a given request. Thus, one cannot tell if it is a list of servers, all of which must be contacted to complete a request, a list of equivalent servers of which only one should be contacted, or a combination of both.

A second limitation of referrals in LDAP Version 2 is that operations can sometimes be ambiguous in their intent regarding whether the operation was targeted for "real" objects in the namespace, as opposed to the referral objects themselves. For searches, referral objects are only presented as referrals, since the usual intent of a search is to look at the real objects in the namespace. Server administrators must therefore use other means to examine existing referral objects, such as examining the database, or reviewing **ldif2db** output. For update operations, default referrals for upward references are presented as referrals, so that read-only replica servers can forward update operations to the master replica. However, subordinate references indicated by a referral object are not followed for update operations, rather they operate on the referral object itself. This is necessary to allow an administrator the ability to delete or modify existing referral objects. Erroneous changes caused by misdirected update operations are generally avoided through access protection and schema rules.

Using LDAP Version 3 Referrals

In LDAP Version 3, referrals are defined as part of the protocol. The LDAP Version 2 limitations mentioned above are overcome by elements of the protocol and extensions to the protocol. There are two methods of passing back referral information in the LDAP Version 3 protocol: referrals and search continuation references.

A result code of **LDAP_REFERRAL** is presented by the server to indicate that the contacted server does not hold the target entry of the request. The referral field is present in the result message and indicates another server (or set of servers) to contact. Referrals can be returned in response to any operation request except unbind and abandon which do not have responses. When multiple URLs are present in a given referral response, each one must be equally capable of being used to progress the operation.

A referral is not returned for a one-level or subtree search in which the search scope spans multiple naming contexts, and several different servers would need to be contacted to complete the operation. Instead, one or more search continuation references are returned. Search continuation references are intermixed with returned search entries. Each one contains a URL to another server (or set of servers) to contact, and represents an unexplored subtree of the namespace which potentially satisfies the search criteria. When multiple URLs are present in a given search continuation reference, each one must be equally capable of being used to progress the operation. By using a separate response for each unexplored subtree, LDAP Version 3 overcomes the limitation of LDAP Version 2, allowing alternate, equivalent servers to be included in each response.

As mentioned earlier, the other limitation in LDAP Version 2 referral processing is related to the inability of a client to specify whether a request was targeted for a normal object or a referral object. For LDAP Version 3, this difficulty is overcome with a protocol extension in the form of the **manageDsaIT** control. (Appendix F, "Supported Server Controls" on page 397 describes **manageDsaIT** in detail.) For typical client requests where the control is absent, whenever the server encounters an applicable referral object while processing the request, either a referral or search continuation reference is presented. When the client request includes this control, the server does not present any referrals or search continuation references, but instead treats the referral objects as normal objects. In this case, even superior references through the use of default referrals are suppressed. The shipped command line utilities

support the **-M** option to indicate that the requestor is managing the namespace, and therefore wishes to examine and manipulate referral objects as if they were normal objects.

Example: Associating Servers Through Referrals and Replication

Following are the steps involved in distributing the namespace using referrals.

1. Plan your namespace hierarchy.

```
country - US
company - IBM, Lotus
organizationalUnit - IBM Austin, IBM Endicott, IBM HQ
```

2. Set up multiple servers, each containing portions of the namespace.

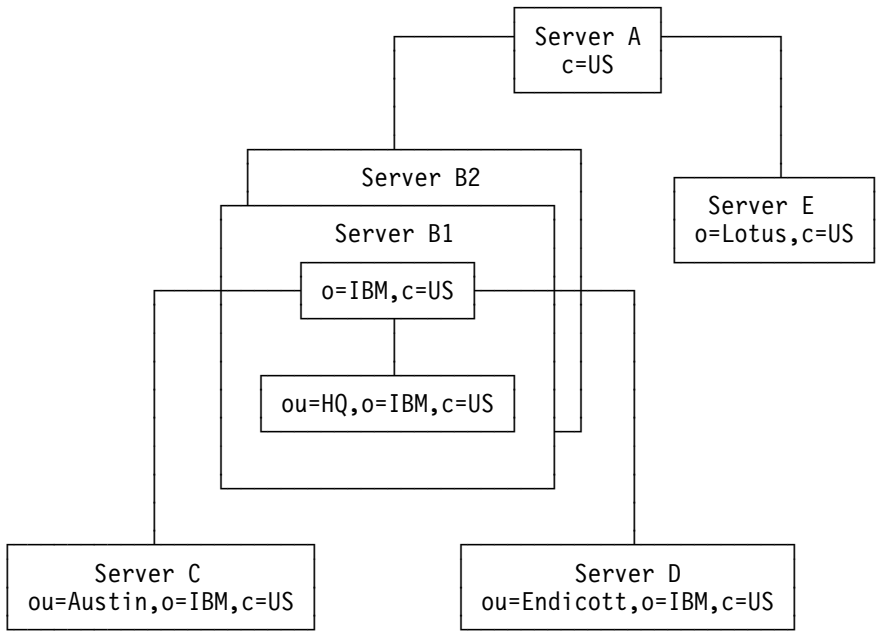


Figure 18. Setting up the Servers

Following is a description of each server:

Server A	Perhaps just a server used to locate other servers in the US. With no other knowledge, clients can come here first to locate information for anyone in the US.
Server B1	A hub for all data pertaining to IBM in the US. Holds all HQ information directly. Holds all knowledge (referrals) of where other IBM data resides.
Server B2	A replica of Server B1.
Server C	Holds all IBM Austin information.
Server D	Holds all IBM Endicott information.
Server E	Holds all Lotus™ information.

3. Set up referral objects to point to the descendents in other servers.

```
dn: ou=HQ,o=IBM,c=US                        <----Entry Data
objectClass: organizationalUnit
description: Headquarters
...                                         <----Entries

dn: o=IBM,c=US                              <----Pointer to Servers B1 and B2
objectClass: referral
ref: ldap://ibm.com:389/o=IBM,c=US
ref: ldap://ibm.com:390/o=IBM,c=US

dn: o=Lotus,c=US                            <----Pointer to Server E
objectClass: referral
ref: ldap://lotus.com:389/o=Lotus,c=US
```

Figure 19. Server A Database (LDIF Input)

4. Servers can also define one or more default referrals which point to “more knowledgeable” servers for anything that is not underneath them in the namespace.

   The default referrals go in the configuration file, not the backend.

   Note: The default referral LDAP URLs do not include the DN portion.

```
# General section

referral        ldap://ibm.com:389
referral        ldap://ibm.com:390

port            789

maxthreads      200
maxconnections  200
.
.
.
# rdbm database definitions

database        rdbm
suffix          "cn=localhost"
suffix          "ou=Endicott,o=IBM,c=US"
```

Figure 20. Server D Configuration File

5. Putting it all together.

   Figure 21 on page 183, Figure 22 on page 184, and Figure 23 on page 185 show these same six servers, showing the referral objects in the database as well as the default referrals which are used for superior references. Also included in Servers B1 and B2 are sample definitions for replication.

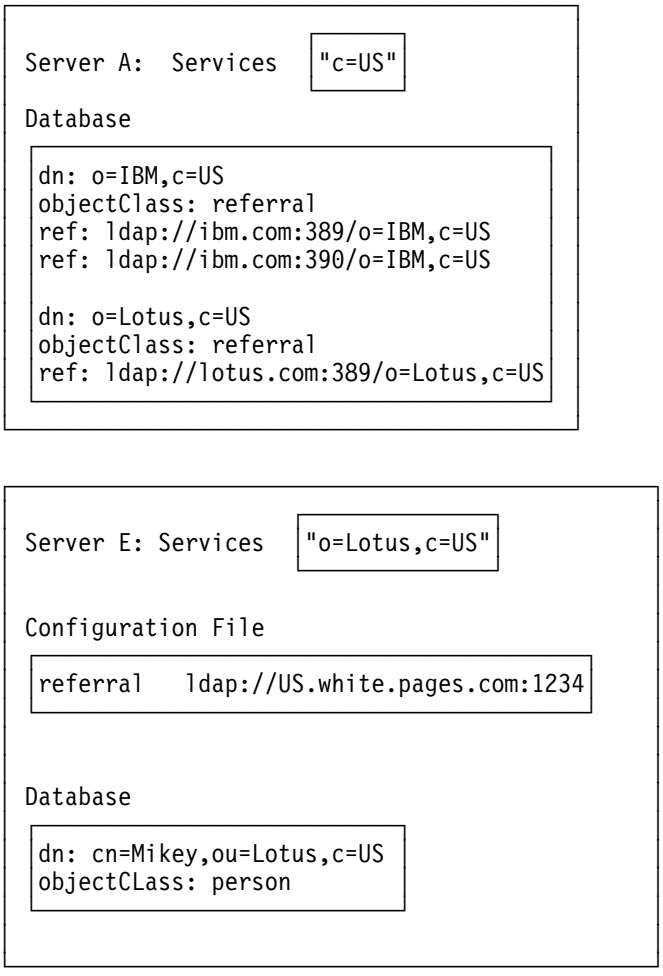


Figure 21. Referral Example Summary (Servers A and E)

```
Server B1:  Services    "o=IBM,c=US"

Configuration File

 referral  ldap://US.white.pages.com:1234

Database

 dn:                  cn=ReplicaB2,cn=localhost
 objectClass:         replicaObject
 replicaHost:         ibm.com
 replicaPort:         390
 replicaBindDN:       cn=Master
 replicaCredentials: secret
 ...

 dn: ou=Austin,o=IBM,c=US
 objectClass: referral
 ref: ldap://austin.ibm.com:389/ou=Austin,o=IBM,c=US

 dn: ou=Endicott,o=IBM,c=US
 objectClass: referral
 ref: ldap://endicott.ibm.com:789/ou=Endicott,o=IBM,c=US
```

```
Server B2:  Services    "o=IBM,c=US"

Configuration File

 referral  ldap://US.white.pages.com:1234

Database

 masterServer:    ldap://ibm.com:389
 masterServerDN:  cn=Master
 masterServerPW:  secret

 dn: ou=Austin,o=IBM,c=US
 objectClass: referral
 ref: ldap://austin.ibm.com:389/ou=Austin,o=IBM,c=US

 dn: ou=Endicott,o=IBM,c=US
 objectClass: referral
 ref: ldap://endicott.ibm.com:789/ou=Endicott,o=IBM,c=US
```

Figure 22. Referral Example Summary (Servers B1 and B2)

Figure 23. Referral Example Summary (Servers C and D)

Chapter 16. Organizing the Directory Namespace

Directory services are meant to help organize the computing environment of the enterprise. To do this, directory services are meant to be used to help find all the resources at one's disposal. Information that is typically found in a directory consists of configuration information for services offered in the enterprise, locating information for people, places, and things in the enterprise, as well as descriptive information about services and resources available in the enterprise. The directory service should be thought of as the spot that can be queried to find whatever is desired in the enterprise.

When designing the format and organization of the directory service for an enterprise, the intended usage scenarios should be considered. These usage characteristics can have an impact on how the directory namespace should be organized so as to offer reasonable performance.

There are two general areas of directory namespace design to be considered. First, the types of information and the layout of where that information will be placed in the directory namespace must be determined. Additional information types can be added at a later date, but there should be some overall design of where in the directory namespace these types of information should be placed. Second, based on the usage characteristics of the users in the enterprise, the number of distinct directory servers and the namespace subtree or subtrees that they support must be considered.

As an example, consider an enterprise that consisted of two physical locations, one in Los Angeles, CA and one in New York City, NY. People in New York City access information about people, places, and things in Los Angeles often, while the people in Los Angeles rarely access information items in New York City. To offer good performance for both locations, a separate directory server could be installed and run in each location. These LDAP Servers would manage information about the people, places, and things that reside in their respective locations. In addition, because the New York City personnel access information about things in the Los Angeles location, the information from the Los Angeles LDAP Server could be replicated to an additional LDAP Server at the New York City LDAP Server. This would allow the New York City personnel to access information about the Los Angeles location by contacting a local server. In Los Angeles, however, directory requests about items in the New York City portion of the enterprise namespace are redirected (that is, referred) to the New York City LDAP Server for the information. This would save managing a replicated set of information at the expense of slightly longer access times on the less-requested information.

The next two sections discuss information layout in the namespace and partitioning an enterprise namespace across multiple LDAP Servers. These sections are followed by a small example.

Information Layout

A directory is meant to provide information about people, places, and things in the enterprise. The most direct use of a directory is to hold information on how to contact other people in the enterprise. This has commonly been known as the *internal phone book*. With the widespread enhancements in technology, people are now more accessible than ever. We have pagers, answering machines, cellular phones, and e-mail. In trying to communicate with someone we might need to know about all of this information. Modeling a person object class based on the attributes about a person that are important to others in the enterprise is an easy way to support an online *internal phone book* using an LDAP directory service. In addition to people, different organizations within an enterprise can also be modeled by creating new object classes and attribute types. This would allow storage in the LDAP directory of locating information for useful services in the organization like benefits, travel reservations, and human resources.

Another application of directory services is the ability to model or store information about places. A place could be a conference room, which might have attributes of **numberOfSeats**, **projectorType**,

phoneNumber, **calendarLocation**, **dataPortType**, **officeNumber**, and **buildingNumber**. Using this method, different conference rooms within a company could be located and compared. Another example of a place would be the whole site in which employees work. An object class for a site LDAP directory entry might be made up of **streetAddress**, **generalManagerDN**, **siteMap**, and **cafeteriaLocation**.

Things abound within the enterprise. Under this category falls computers, copiers, FAX machines, printers, and computer software, as well as configuration information for daemons that use an LDAP directory service. Each of these can be modeled with attribute types used inside object classes specific to the device or program.

In laying out where entries should appear in the directory hierarchy, by far the most common method of naming things is to start with the country in which the company is organized, followed by the name of the company, treated as an organization attribute type. Thus, the top level suffix for LDAP directory service names for entries within the company usually follows the form: `o=CompanyName, c=US` (for US-based countries). Below this suffix it is common for organizational unit object classes to be used to represent departments or sites within an organization. Below these organizational entries the actual entry representing a person, place, or thing would be defined. When organizing the information layout for the namespace, the intended usage should be considered to ensure the best performance.

Example of Building an Enterprise Directory Namespace

Let us look at an example configuration that exhibits the features capable with the OS/390 LDAP Server. To set the stage, we will consider a moderately sized company that has personnel working in three locations across the United States. Big Company, Inc. has corporate headquarters in Chicago, IL, and two satellite facilities, one in Los Angeles, CA and the other in New York City, NY. The information technology staff would like to make available information about all of the company's computing and office services using an LDAP directory. In order to facilitate local modifications as necessary of the information in the directory, as well as provide improved response time for accessing local information, each site will have an LDAP Server running. The server running at each site will be responsible for managing the directory information that pertains to that site.

The first thing to do is determine the name of the root of the directory namespace for Big Company, Inc. Typically, the name for the company will consist of the country of origin along with the company's given name. In LDAP directory terminology, the company is an organization. In this example, we chose:

```
o=Big Company, c=US
```

as the company's name is Big Company and is located in the United States. Choosing a name of this format helps ensure that when a global namespace coordinator is established, the company's chosen *root* will fit nicely into the overall directory namespace.

Next to choose are the names of the three locations under which the directory information is stored. At this point, the namespace could be organized in a number of ways. One way would be to organize by functional unit (regardless of location). This model is useful if individuals (or computers, or other equipment or services) typically remain with the functional unit as opposed to being tied to the individual or physical location. Another way would be to organize based on the physical locations of the parts of the organization. This is useful if the people, places, and things to be stored in the directory typically do not move between locations. This latter approach will be used in the example. So, with three locations, three names are defined below the overall company distinguished name:

```
ou=Los Angeles, o=Big Company, c=US
ou=Chicago, o=Big Company, c=US
ou=New York City, o=Big Company, c=US
```

Since separate LDAP Servers will be established at each location, these names represent the root of the subtree stored and managed by the directory server at each location.

For administration, each site will have a different directory administrator. To define this administrator, an administrator distinguished name and password need to be defined for each location. To start, the following names will be used:

```
AdminDN "cn=Administrator, ou=Los Angeles, o=Big Company, c=US"

AdminDN "cn=Administrator, ou=Chicago, o=Big Company, c=US"

AdminDN "cn=Administrator, ou=New York City, o=Big Company, c=US"
```

Since the Chicago location is also the corporate headquarters, the LDAP directory at this location will be used to store information about the entire company as well as information about the Chicago site.

We now have enough information to set up the base configuration files for each of the three LDAP Servers that will be used to supply this information. Following are the files needed to set up the LDAP Servers on each site. Note that what is shown is the minimal setup required. Other options could be specified in addition to these, see Chapter 5, “Configuring” on page 31 for configuration file options.

```
# Configuration file for the Chicago LDAP Server
include /etc/ldap/schema.system.at
include /etc/ldap/schema.system.oc
include /etc/ldap/schema.IBM.at
include /etc/ldap/schema.IBM.oc
include /etc/ldap/schema.user.at
include /etc/ldap/schema.user.oc

security none

adminDN "cn=Administrator, ou=Chicago, o=Big Company, c=US"

database rdbm GLDBRDBM
suffix "cn=localhost"
suffix "o=Big Company, c=US"

dbuserid user1
tbspaceentry  dbtbspace
tbspace4k     dbtb4k
tbspace32k    dbtb32k
# end of configuration file
```

Figure 24. Chicago Base Configuration

```
# Configuration file for the Los Angeles LDAP Server
referral ldap://ldap.chicago.bigcompany.com

include /etc/ldap/schema.system.at
include /etc/ldap/schema.system.oc
include /etc/ldap/schema.IBM.at
include /etc/ldap/schema.IBM.oc
include /etc/ldap/schema.user.at
include /etc/ldap/schema.user.oc

security none

adminDN "cn=Administrator, ou=Los Angeles, o=Big Company, c=US"

database rdbm GLDBRDBM
suffix "cn=localhost"
suffix "ou=Los Angeles, o=Big Company, c=US"

dbuserid user2
tbspaceentry  dbtbspace
tbspace4k     dbtb4k
tbspace32k    dbtb32k
# end of configuration file
```

Figure 25. Los Angeles Base Configuration

```
# Configuration file for the New York City LDAP Server
referral ldap://ldap.chicago.bigcompany.com

include /etc/ldap/schema.system.at
include /etc/ldap/schema.system.oc
include /etc/ldap/schema.IBM.at
include /etc/ldap/schema.IBM.oc
include /etc/ldap/schema.user.at
include /etc/ldap/schema.user.oc

security none

adminDN "cn=Administrator, ou=New York City, o=Big Company, c=US"

database rdbm GLDBRDBM
suffix "cn=localhost"
suffix "ou=New York City, o=Big Company, c=US"

dbuserid user2
tbspaceentry  dbtbspace
tbspace4k     dbtb4k
tbspace32k    dbtb32k
# end of configuration file
```

Figure 26. New York City Base Configuration

The referral line indicates the default place to refer connecting clients when the LDAP Server does not contain the information requested by the client. It is called the *default referral*. It is in the form of an LDAP URL. After the scheme name (`ldap`), the LDAP URL contains a TCP/IP DNS host name for another

LDAP Server. In this example, it is assumed that the TCP/IP host on which the Chicago LDAP Server is running is `ldap.chicago.bigcompany.com`. The Chicago LDAP Server does not have a default referral defined. This keeps directory searches from inadvertently going over the Internet from within the company.

The `security none` line indicates that SSL connection support should not be enabled at this time. The `adminDN` line indicates the distinguished name that should be used to connect to the LDAP Server in order to have complete control over the data content held by the LDAP Server.

The `database` line indicates that all following lines pertain to the RDBM storage method.

The next lines are used to connect to DB2 using the correct user ID as well as identify the DB2 table spaces that were defined by the SPUFI script that was run to create these table spaces for the LDAP Server (see “Creating the LDAP Server DB2 Database and Table Spaces” on page 14).

After these files have been created, one or more of the LDAP Servers can be started. However, there will be no initial data in the DB2 tables. The next section introduces the **ldif2db** tool that can be used to bulk-load entries into the LDAP Server.

Priming the Directory Servers with Information

In order to start using the LDAP Servers that we have defined, it is recommended that at least the top levels of directory information be bulk-loaded into the respective databases. This provides a basis from which to add more entries into the directory namespace either with additional invocations of the bulk-loading facility called **ldif2db**, using the **ldapadd** and **ldapmodify** tools, or using the LDAP C language API and the LDAP protocol.

The **ldif2db** utility program used to bulk-load entries into the directory server database can be run either while SLAPD is running or not. It makes direct updates to the directory database tables without using the LDAP protocol. This method results in faster loads of large amounts of directory information. The tool takes as input a sequential file that contains the data describing directory entries to be added into the directory namespace. The format of the information is noted as LDAP Interchange Format (LDIF).

Using LDIF Format to Represent LDAP Entries

The LDAP Data Interchange Format (LDIF) is used to represent LDAP entries in a simple text format. An LDIF file contains groups of attribute information which will be treated as an entry to be added to the directory. The general format of an LDIF entry is:

```
[id]
dn: distinguished name
attrtype1: attrvalue1
attrtype2: attrvalue2
...
```

where *id* is the optional entry ID (a positive decimal number). Normally, you would not supply the *id*, allowing the database creation tools to do that for you.

A line may be continued by starting the next line with a single space or tab character. For example:

```
dn: ou=departments, ou=New York City, o=Big Co
 mpany, c=US
```

Multiple attribute values are specified on separate lines. For example:

```
objectclass: organizationalunit
ou: departments
```

If an *attrvalue* contains a nonprinting character, or begins with a space or a colon (:), the *attrtype* is followed by a double colon (::) and the value is encoded in base64 notation. For example, the value `" begins with a space"` would be encoded like this:

```
cn:: IGJlZ2lucyB3aXRoIGEgc3BhY2U=
```

Multiple entries within the same LDIF file are separated by blank lines. Here is an example of an LDIF file containing three entries.

```
dn: ou=New York City, o=Big Company, c=US
objectclass: organizationalunit
ou: New York City

dn: ou=fax machines, ou=New York City, o=Big Company, c=US
objectclass: organizationalunit
ou: fax machines

dn: ou=computers, ou=New York City, o=Big Company, c=US
objectclass: organizationalunit
ou: computers
```

Note: Trailing spaces are not trimmed from values in an LDIF file. Also, multiple internal spaces are not compressed. If you do not want them in your data, do not put them there.

Multiple attribute values for the same attribute type are specified on multiple lines within the specification of a directory entry. For example:

```
dn: cn=John Doe, ou=New York City, o=Big Company, c=US
objectclass: person
cn: John Doe
phonenumber: 555-1111
phonenumber: 555-2222
sn: Doe
```

Generating the File

A file is typically generated using an existing source of information and some tools to format the data into the LDIF format. Note that the order of entries in the LDIF file is important. In order for an entry specified in the LDIF file to be successfully added to the directory, its parent entry must first exist in the directory namespace. For this reason, the top level entries in the directory namespace subtree that the particular LDAP Server will support should be first in the LDIF file.

For our example, we will define just the a minimal set of entries to get the directory server useful at each location. This will include two referral entries for the Chicago location. The meaning of these entries will be discussed in more detail in the following sections.

Here is the base set of LDIF files to set up the directory namespace at each location. For the Los Angeles location:

```
dn: ou=Los Angeles, o=Big Company, c=US
objectclass: organizationalunit
ou: Los Angeles

dn: ou=fax machines, ou=Los Angeles, o=Big Company, c=US
objectclass: organizationalunit
ou: fax machines
```

```
dn: ou=computers, ou=Los Angeles, o=Big Company, c=US
objectclass: organizationalunit
ou: computers

dn: ou=departments, ou=Los Angeles, o=Big Company, c=US
objectclass: organizationalunit
ou: departments
```

For the New York City location:

```
dn: ou=New York City, o=Big Company, c=US
objectclass: organizationalunit
ou: New York City

dn: ou=fax machines, ou=New York City, o=Big Company, c=US
objectclass: organizationalunit
ou: fax machines

dn: ou=computers, ou=New York City, o=Big Company, c=US
objectclass: organizationalunit
ou: computers

dn: ou=departments, ou=New York City, o=Big Company, c=US
objectclass: organizationalunit
ou: departments
```

For the Chicago location:

```
dn: o=Big Company, c=US
objectclass: organization
o: Big Company

dn: ou=Los Angeles, o=Big Company, c=US
objectclass: referral
ref: ldap://ldap.losangeles.bigcompany.com/ou=Los Angeles,o=Big Company,c=US

dn: ou=New York City, o=Big Company, c=US
objectclass: referral
ref: ldap://ldap.newyorkcity.bigcompany.com/ou=New York City,o=Big Company,c=US

dn: ou=Chicago, o=Big Company, c=US
objectclass: organizationalunit
ou: Chicago

dn: ou=fax machines, ou=Chicago, o=Big Company, c=US
objectclass: organizationalunit
ou: fax machines

dn: ou=computers, ou=Chicago, o=Big Company, c=US
objectclass: organizationalunit
ou: computers

dn: ou=departments, ou=Chicago, o=Big Company, c=US
objectclass: organizationalunit
ou: departments
```

These files will now be used with the **ldif2db** bulk-loading facility as follows. From the OS/390 shell, TSO, or batch environment, run:

```
ldif2db -i ldif.filename -f config.filename
```

From the OS/390 shell environment, **ldif2db** will read from the standard input so that:

```
cat ldif.filename | ldif2db -f config.filename
```

is equivalent.

After running this command on each respective directory server system, the directory namespace will be formed and the servers can now be used to hold and supply information.

Two entries added to the Chicago location directory server database deserve some special attention. These are the referral objects that were noted in the LDIF file for the Chicago location. Notice that the referral object has the identical distinguished name as the root of the LDAP directory namespace that is served by the Los Angeles and New York City servers. These entries, coupled with the default referral specification in all of the configuration files for the LDAP Servers enable searches of the Big Company namespace to originate at any of the three directory servers and resolve to the correct server to obtain the information.

A referral redirects a client request to a different LDAP Server that can presumably handle the request (or refer the client to another server that can handle the request). In our example, if a client connects to the New York City server requesting a name that is under the Los Angeles portion of the namespace, the New York City server will send back a referral to the client based on the default referral. This will point the client at the Chicago directory server. The Chicago server will resolve the request down to the referral object for distinguished name `ou=Los Angeles, o=Big Company, c=US` and refer the client to the Los Angeles server. Finally, the client will contact the Los Angeles server and obtain the information requested.

Setting Up for Replication

As people start using the directory service in their daily routines at Big Company, Inc., the information technology staff notices that the people in New York City are doing a lot of work with the people in Los Angeles. So much, in fact, that an analysis of the TCP/IP traffic between New York City and Los Angeles shows that much of the traffic is directory access requests, presumably to look up phone numbers or FAX numbers for people in Los Angeles. The information technology staff decides to improve directory lookup response time, as well as lessen the directory lookup traffic between New York City and Los Angeles, by creating a replica of the Los Angeles directory server's information in New York City. This will allow local access to this information by the New York City users and cut down on the amount of requests from New York that must travel to Los Angeles to be completed.

Defining Another LDAP Server

To set up a replica of the LDAP Server information in Los Angeles, a second LDAP Server must be defined and started in New York City. This server can reside on the same system as the first LDAP Server, though if this is chosen, the TCP/IP port that this replica server listens on must be different from the other LDAP Server running on the system. As an alternative, the replica server could run on a different system, allowing it to listen on the default LDAP port. The configuration file for the replica server in New York City will be very similar to the configuration files for the New York City server and the Los Angeles server. This configuration file must contain some additional items that pertain to replication. Here is what the contents of the New York City Los Angeles replica server should contain:

```
# Configuration file for the New York City Los Angeles replica LDAP Server
referral ldap://ldap.chicago.bigcompany.com

include /etc/ldap/schema.system.at
include /etc/ldap/schema.system.oc
include /etc/ldap/schema.IBM.at
include /etc/ldap/schema.IBM.oc
include /etc/ldap/schema.user.at
include /etc/ldap/schema.user.oc

port 2001

security none

adminDN "cn=Administrator, ou=Los Angeles, o=Big Company, c=US"

database rdbm GLDBRDBM
suffix "cn=localhost"
suffix "ou=Los Angeles, o=Big Company, c=US"

dbuserid user2
tbspaceentry  dbtbspace
tbspace4k     dbtb4k
tbspace32k    dbtb32k

masterServer ldap://ldap.losangeles.bigcompany.com
masterServerDN "cn=Replicator, ou=Los Angeles, o=Big Company, c=US"
# end of configuration file
```

Figure 27. New York City Los Angeles Replica Configuration

The additional lines at the end of the configuration file specify the only “user” that can update entries in the replica LDAP Server. The values here must match the values entered at the “source” location when the replica is defined.

Preparing the Replica

The next step is to get the LDAP replica primed with the existing information in the Los Angeles server and set up the Los Angeles server to replicate to the New York City replica. The set of steps to perform (described in “Populating a Replica” on page 172) ensures that the replicas are in sync and that no update is lost during this synchronization. Once the replica is defined at the source location, updates to the directory information will be logged to be sent to the replica server when possible.

To initially synchronize the data between the LDAP master server and the LDAP replica server, perform the steps in “Populating a Replica” on page 172.

While there are a number of manual steps to perform, there is a small consolation that the steps at different locations are not interleaved. All work can be done at the source location and then all work can be performed at the target (replica) location.

Resynching the Replica and Master Servers: If it is noticed that a replica's contents are out of sync with the information at the master server, the information can be resynched by following the steps shown in “Recovering from Out-of-Sync Conditions” on page 175.

Notifying Users of the Replica

At this point, the New York City users can be notified that a second LDAP Server is now available for their use. The notification should contain either the LDAP URL of the new LDAP replica server or the host name and port number of the LDAP replica server, as well as the base of the LDAP subtree that is published by the replica. As updates are made to the Los Angeles LDAP Server, these updates will be propagated to the replica server in New York City. See Chapter 14, “Replication” on page 169 for more details on replication.

What Big Company, Inc. now has in place is an Enterprise Directory service that can be used by whatever enterprise distributed processing tasks require lookup or configuration information. These enterprise distributed processing tasks and applications may require some changes to make use of the directory service, but the result will be the ability to view, find, and modify the configuration of the enterprise by looking at and modifying the contents of the LDAP directory.

Part 3. Messages

Chapter 17. LDAP Server Messages

This part contains the messages returned by the LDAP Server. The messages are ordered alphanumerically.

GLD0001E *option_flag* **flag is not allowed. Usage:** *command_name* **[-d debuglevel] [-f configfile] [-p portnumber] [-s sysloglevel].**

Severity: Error

Explanation: The LDAP Server cannot start because the parameters specified on start-up were not correct.

System Action: The program ends.

Operator Response: Correct the parameters and start the LDAP Server again.

GLD0002I Configuration file successfully read.

Severity: Informational

Explanation: The LDAP Server successfully read the configuration file.

System Action: The program continues.

Operator Response: None.

GLD0003I Starting necessary replication threads.

Severity: Informational

Explanation: The LDAP Server is starting the threads needed to support the requested replication.

System Action: The program continues.

Operator Response: None.

GLD0004I Terminating slapd.

Severity: Informational

Explanation: The LDAP Server is ending, probably due to a SIGTERM signal.

System Action: The program ends.

Operator Response: None.

GLD0006I No values for type *type***.**

Severity: Attention

Explanation: The LDAP Server is unable to process the requested operation because no values were supplied for the specified type.

System Action: The program continues. The request fails.

Operator Response: None.

GLD0008I Unrecognized database type (*type***).**

Severity: Attention

Explanation: The LDAP Server encountered an error processing the configuration file. The specified database type is not supported.

System Action: The program continues. If all required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: Correct the configuration file and try again.

GLD0010I Reading configuration file *file_name***.**

Severity: Informational

Explanation: The LDAP Server is processing the specified configuration file.

System Action: The program continues.

Operator Response: None.

GLD0011I *command_name***: line** *line_number***: must appear inside a database definition. Ignored.**

Severity: Attention

Explanation: The LDAP Server encountered an error processing the configuration file. The specified line is associated with a database definition and must appear in the database section of the configuration file. The specified line is ignored.

System Action: The program continues. If all required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: None. If there are no complete database definitions, the configuration will fail. If the configuration fails, correct the configuration file and try again.

GLD0012I *command_name***: line** *line_number***: incorrect configuration line. Ignored.**

Severity: Attention

Explanation: The LDAP Server encountered an error processing the configuration file because the specified line is not correct. The line is ignored.

System Action: The program continues. If all

required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: None. If the configuration information is incomplete, the configuration will fail. If the configuration fails, correct the configuration file and try again.

GLD0013I *command_name*: **line** *line_number*: **incorrect number of parameters specified.**

Severity: Attention

Explanation: The LDAP Server encountered an error processing the configuration file because the specified line does not supply all of the required parameters.

System Action: The program continues. If all required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: None. If the configuration information is incomplete, the configuration will fail. If the configuration fails, correct the configuration file and try again.

GLD0014E Unable to open file *file_name*. **Try specifying the full path name.**

Severity: Error

Explanation: The LDAP Server is unable to open the specified configuration file.

System Action: The program ends.

Operator Response: Correct the file name and restart the server, or contact the administrator.

Administrator Response: Correct the file name or permissions and restart the server.

GLD0015I *command_name*: **line** *line_number*: **unknown directive** *directive* **outside database definition. Ignored.**

Severity: Attention

Explanation: The LDAP Server encountered an error processing the configuration file because the specified line contains an unrecognized directive or keyword. The specified line is ignored.

System Action: The program continues. If all required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: None. If there are no complete database definitions, the configuration will fail. If the configuration fails, correct the configuration file and try again.

GLD0016E A ber_alloc failed.

Severity: Error

Explanation: The LDAP Server is unable to allocate the necessary storage to continue processing the request.

System Action: The program ends.

Operator Response: Increase the storage for the LDAP Server and restart the server.

GLD0017I Closing socket *socket_number*.

Severity: Informational

Explanation: The LDAP Server is closing the specified socket.

System Action: The program continues.

Operator Response: None.

GLD0019E Error while trying to allocate memory.

Severity: Error

Explanation: The LDAP Server is unable to allocate the necessary storage to continue processing.

System Action: The program ends.

Operator Response: Increase the storage for the LDAP Server and restart the server.

GLD0020I Exceeded maximum number of connections, currently set at *max_connections*.

Severity: Attention

Explanation: The LDAP Server is unable to process the request because all available connections are in use.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

Programmer Response: Submit the request again.

GLD0021E Unable to create necessary thread.

Severity: Error

Explanation: The LDAP Server is unable to obtain the necessary resources to create a required thread. If the failure occurs during startup of the LDAP Server, the program will end. If the failure occurs while processing a new client request or initializing secure communications for a client request, the program will continue, but the request will not complete.

System Action: The program ends.

Operator Response: Verify the OMVS Kernel is operating correctly. Save the dump and contact the system programmer.

Administrator Response: If the problem occurs during client processing, modify the settings of the `maxthreads` and `waitingthreads` parameters in the configuration file to support additional load. If the problem persists, contact the service representative.

GLD0022I *LDAP_server_version* **Starting slapd.**

Severity: Informational

Explanation: The LDAP Server is starting.

System Action: The program continues.

Operator Response: None.

GLD0023I Server listening for incoming client requests on *port_number***.**

Severity: Informational

Explanation: The LDAP Server is listening on the specified port for client requests.

System Action: The program continues.

Operator Response: None.

GLD0027E slapd unable to start because all backends failed to configure.

Severity: Error

Explanation: The LDAP Server is unable to start because the defined backends have not configured successfully.

System Action: The program ends.

Operator Response: None.

Administrator Response: Check for other messages regarding errors during configuration. Correct the configuration file and restart the server.

GLD0028E Configuration error: server using port *port_number* **for both SSL and non-SSL.**

Severity: Error

Explanation: The LDAP Server is listening on the specified port for both secure and nonsecure requests.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the SSL port in the configuration file and restart the server.

GLD0038E A command that is not supported or is not valid was entered from the console.

Severity: Error

Explanation: The LDAP Server received a command from the operator console that is not supported or is not correct.

System Action: The program continues.

Operator Response: Correct the command and try again.

GLD0039E The __console() function failed with errno *errno***.**

Severity: Error

Explanation: The LDAP Server received an unexpected return code from system function **__console()**. The *OS/390 C/C++ Run-Time Library Reference*, SC28-1663 contains more information about the **errno**. Error messages normally written to the console will only appear in the log.

System Action: The program continues.

Operator Response: Save the dump, if any, and contact the system programmer. If the problem persists, contact the service representative.

GLD0040I The value for the 'maxthreads' option is out of range (*min_threads***,** *max_threads***). The default (***max_threads***) will be used.**

Severity: Attention

Explanation: The LDAP Server encountered an error when processing the `maxthreads` parameter of the configuration file. The server will continue with the default value.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0041E An administrator DN must be specified in the adminDn line.

Severity: Error

Explanation: The LDAP Server encountered a blank `adminDN` parameter in the configuration file.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the `adminDN` parameter in the configuration file and restart the server.

GLD0042E Attempt to initialize administrative connection failed, *errno* **(***errno_string***).**

Severity: Error

Explanation: The LDAP Server was unable to establish a connection with the administrative utilities because system function **unlink()** returned the specified **errno**. The *OS/390 C/C++ Run-Time Library Reference*, SC28-1663 contains more information about the error.

System Action: The program continues.

Operator Response: None.

Administrator Response: If the problem persists, contact the service representative.

GLD0043E Unable to connect to replica *replica_name* **on port** *port_number***. Please verify that the replica is started.**

Severity: Error

Explanation: The LDAP Server was unable to connect to the specified server to perform replication.

System Action: The program continues. Replication to the specified server cannot continue.

Operator Response: Verify that the replica server is started.

Administrator Response: Verify that the replica server is started, or contact the operator to start the replica server. Verify that the replica server information is correct.

GLD0044I Error *error_value* **occurred during replication to** *replica_name***: add failed on entry** *distinguished_name***.**

Severity: Attention

Explanation: The LDAP Server was unable to replicate the specified add operation to the replica server. The replication attempt will be tried again.

System Action: The program continues.

Operator Response: None.

Administrator Response: None. If the failure continues, contact the service representative.

GLD0045I Error *error_value* **occurred during replication to** *replica_name***: delete failed on entry** *distinguished_name***.**

Severity: Attention

Explanation: The LDAP Server was unable to replicate the specified delete operation to the replica server. The replication attempt will be tried again.

System Action: The program continues.

Operator Response: None.

Administrator Response: None. If the failure continues, contact the service representative.

GLD0046I Error *error_value* **occurred during replication to** *replica_name***: modrdn failed on entry** *distinguished_name***.**

Severity: Attention

Explanation: The LDAP Server was unable to replicate the specified modify RDN operation to the replica server. The replication attempt will be tried again.

System Action: The program continues.

Operator Response: None.

Administrator Response: None. If the failure continues, contact the service representative.

GLD0047I Error *error_value* **occurred during replication to** *replica_name***: modify failed on entry** *distinguished_name***.**

Severity: Attention

Explanation: The LDAP Server was unable to replicate the specified modify operation to the replica server. The replication attempt will be tried again.

System Action: The program continues.

Operator Response: None.

Administrator Response: None. If the failure continues, contact the service representative.

GLD0048E Creation of socket failed; errno *errno* **(***errno_string***).**

Severity: Error

Explanation: The LDAP Server received the specified error from system function **socket()**. Refer to the *OS/390 C/C++ Run-Time Library Reference*, SC28-1663 for an explanation of the **errno** returned.

System Action: The program ends.

Operator Response: Ensure TCP/IP is operating correctly. Save the diagnostic information and contact the system programmer.

Administrator Response: If the problem persists, contact the service representative.

GLD0049E Attempt to setsockopt() failed; errno *errno* **(***errno_string***).**

Severity: Error

Explanation: The LDAP Server received the specified error from system function **setsockopt()**. Refer to the *OS/390 C/C++ Run-Time Library Reference*, SC28-1663 for an explanation of the **errno** returned.

System Action: The program ends.

Operator Response: Ensure TCP/IP is operating correctly. Save the diagnostic information and contact the system programmer.

Administrator Response: If the problem persists, contact the service representative.

GLD0050E Attempt to bind failed; errno *errno* **(***errno_string***).**

Severity: Error

Explanation: The LDAP Server received an error from system function **bind()**. Refer to the *OS/390 C/C++ Run-Time Library Reference*, SC28-1663 for an explanation of the **errno** returned.

System Action: The program ends.

Operator Response: Ensure TCP/IP is operating correctly. Save the diagnostic information and contact the system programmer.

Administrator Response: If the problem persists, contact the service representative.

GLD0051E The listen() failed; errno *errno* **(***errno_string***).**

Severity: Error

Explanation: The LDAP Server received an error from system function **listen()**. Refer to the *OS/390 C/C++ Run-Time Library Reference*, SC28-1663 for an explanation of the **errno** returned.

System Action: The program ends.

Operator Response: Ensure TCP/IP is operating correctly. Save the diagnostic information and contact the system programmer.

Administrator Response: If the problem persists, contact the service representative.

GLD0052I Configuration read securePort *port_number***.**

Severity: Informational

Explanation: The LDAP Server has assigned the `securePort` to the specified value based on the value read from the configuration file.

System Action: The program continues.

Operator Response: None.

GLD0053I Configuration read security of *security_value***.**

Severity: Informational

Explanation: The LDAP Server has assigned the `security` to the specified value based on the value read from the configuration file.

System Action: The program continues.

Operator Response: None.

GLD0054I Value connectionsAllowed is set to *connectionsAllowed***.**

Severity: Informational

Explanation: The LDAP Server has assigned the specified value for `connectionsAllowed`.

System Action: The program continues.

Operator Response: None.

GLD0055I Configuration read cipher specifications mask to be *cipher_mask***.**

Severity: Informational

Explanation: The LDAP Server has assigned the cipher specification mask to the specified value based on the value read from the configuration file.

System Action: The program continues.

Operator Response: None.

GLD0056I Non-SSL port initialized to *port_number***.**

Severity: Informational

Explanation: The LDAP Server has initialized the port for nonsecure communications to the specified value.

System Action: The program continues.

Operator Response: None.

GLD0057I SSL port initialized to *port_number***.**

Severity: Informational

Explanation: The LDAP Server has initialized the port for secure communications to the specified value.

System Action: The program continues.

Operator Response: None.

GLD0058I Local UNIX socket name initialized to *local_socket_name***.**

Severity: Informational

Explanation: The LDAP Server has initialized the name for the local UNIX socket to the specified value.

System Action: The program continues.

Operator Response: None.

GLD0059E SocketInit failed for port *port_number***.**

Severity: Error

Explanation: The LDAP Server was unable to initialize the specified port for communications. Refer to additional messages to identify the specific error.

System Action: The program ends.

Operator Response: Capture additional messages. See actions for additional messages.

Administrator Response: If the problem persists, contact the service representative.

GLD0060I Non-SSL port override from command line, value = *value***.**

Severity: Informational

Explanation: The LDAP Server has assigned the nonsecure port to the specified value based on the value supplied on the command line.

System Action: The program continues.

Operator Response: None.

GLD0061I SSL port override from command line, value = *value***.**

Severity: Informational

Explanation: The LDAP Server has assigned the secure port to the specified value based on the value supplied on the command line.

System Action: The program continues.

Operator Response: None.

GLD0062I The setsockopt() was successful for socket *socket_number***.**

Severity: Informational

Explanation: The LDAP Server successfully modified the socket options for the specified socket.

System Action: The program continues.

Operator Response: None.

GLD0063E Unknown error *error_code* **reported by SSL initialization.**

Severity: Error

Explanation: The LDAP Server was unable to complete initialization required for secure communications.

System Action: The program ends.

Operator Response: Contact the service representative.

Administrator Response: None.

GLD0064E File I/O error on opening SSL keyring file *file_name***.**

Severity: Error

Explanation: The LDAP Server was unable to read the `keyring` file required for secure communications. Secure communications cannot continue.

System Action: The program ends.

Operator Response: Verify that the file system is operating correctly.

Administrator Response: Ensure that the file permissions on the SSL keyring file are correct.

GLD0065E Open of SSL keyring file *file_name* **failed.**

Severity: Error

Explanation: The LDAP Server was unable to open the `keyring` file required for secure communications. Secure communications cannot continue.

System Action: The program ends.

Operator Response: Verify that the file system is operating correctly.

Administrator Response: Ensure that the file permissions on the SSL keyring file are correct.

GLD0066E SSL keyring file *file_name* **is in an unknown format.**

Severity: Error

Explanation: The LDAP Server found that the `keyring` file format is incorrect. Secure communications cannot continue.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure that the `keyring` file is correct.

GLD0067E The password supplied for keyring file *file_name* **is not correct.**

Severity: Error

Explanation: The LDAP Server found that the password supplied for the `keyring` file is incorrect. Secure communications cannot continue.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the password for the `keyring` file in the configuration file and restart the server.

GLD0068E A memory allocation error occurred processing SSL keyring file *file_name*.

Severity: Error

Explanation: The LDAP Server is unable to allocate the necessary storage to continue processing the `keyring` file.

System Action: The program ends.

Operator Response: Increase the storage for the LDAP Server and restart the server.

GLD0069E Unknown SSL error *error_code* reported attempting SSL handshake.

Severity: Error

Explanation: The LDAP Server was unable to complete initialization of socket required for secure communications.

System Action: The program ends.

Operator Response: Contact the service representative.

Administrator Response: None.

GLD0070E An incorrect SSL keyring identifier *identifier* was specified.

Severity: Error

Explanation: The LDAP Server found that the identifier supplied for the `keyring` file is incorrect. Secure communications cannot continue.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the identifier for the `keyring` file in the configuration file and restart the server.

GLD0071E An incorrect value of *cipher_spec* was given for the SSL cipher specification.

Severity: Error

Explanation: The LDAP Server found that the cipher supplied for the `keyring` file is incorrect. Secure communications cannot continue.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the cipher for the `keyring` file in the configuration file and restart the server.

GLD0072E No SSL ciphers were specified.

Severity: Error

Explanation: The LDAP Server found that no ciphers were supplied for secure communications. Secure communications cannot continue.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the ciphers for the `keyring` file in the configuration file and restart the server.

GLD0073E The default SSL keyring certificate has expired in file *file_name*.

Severity: Error

Explanation: The LDAP Server found that the default certificate in the `keyring` file is no longer valid.

System Action: The program ends.

Operator Response: None.

Administrator Response: Refresh the certificate and restart the server.

GLD0074E The default SSL keyring certificate is incorrect in file *file_name*.

Severity: Error

Explanation: The LDAP Server found that the default certificate in the `keyring` file is not correct.

System Action: The program ends.

Operator Response: None.

Administrator Response: Refresh the certificate and restart the server.

GLD0075E The default SSL keyring certificate is of an unsupported type in file *file_name*.

Severity: Error

Explanation: The LDAP Server found that the default certificate in the `keyring` file is not the correct type.

System Action: The program ends.

Operator Response: None.

Administrator Response: Obtain a supported certificate and restart the server.

GLD0076E No certificate exists in SSL keyring file *file_name***.**

Severity: Error

Explanation: The LDAP Server found that no certificate exists in the `keyring` file.

System Action: The program ends.

Operator Response: None.

Administrator Response: Refresh the certificate and restart the server.

GLD0077E The underlying socket was closed.

Severity: Error

Explanation: The LDAP Server was unable to complete secure communications because the socket is closed.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact the service representative.

GLD0078E An SSL time-out has occurred.

Severity: Error

Explanation: The LDAP Server was unable to complete secure communications because the time allowed for the communication has expired.

System Action: The program ends.

Operator Response: None.

Administrator Response: None.

GLD0079E An SSL error of *error_code* **was encountered contacting replica server** *replica_name***, port** *port_number***.**

Severity: Error

Explanation: The LDAP Server was unable to complete secure communications to the specified replication server because of the specified SSL error.

System Action: The program continues. Replication fails.

Operator Response: None.

Administrator Response: Contact the service representative.

GLD0080E Attempted to use an SSL connection with a non-SSL version of the product.

Severity: Error

Explanation: The LDAP Server determined that a client attempted secure communications when the server version was not configured to support secure communications.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: If the problem persists, consider configuring the server to support secure communications.

Programmer Response: Correct the request to use nonsecure communications.

GLD0081I Connection to host *host_name***, port** *port_number***, established using SSL.**

Severity: Attention

Explanation: The LDAP Server was able to restart replication using secure communication to the specified replica server.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0083I Successfully reconnected to replica host *replica_name* **on port** *port_number***.**

Severity: Attention

Explanation: The LDAP Server was able to restart replication to the specified replica server.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0084E Connection to replica *replica_name* **on port** *port_number* **has failed. Verify that the replica is started.**

Severity: Error

Explanation: The LDAP Server detected that the connection to the specified replication server ended.

System Action: The program continues. Replication to the specified server cannot continue.

Operator Response: Verify that the replica server is started.

Administrator Response: Verify that the replica server is started, or contact the operator to start the replica server. Verify that the replica server information is correct.

GLD0089I Attention: configuration file *file_name* is empty.

Severity: Attention

Explanation: The LDAP Server encountered an empty configuration file.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0090E A modify command that is not valid was entered from the console: *command*.

Severity: Error

Explanation: The LDAP Server received a modify command from the operator console that is not correct.

System Action: The program continues.

Operator Response: Correct the command and try again.

GLD0091I Successfully set debug level to *debug_level* from console command.

Severity: Informational

Explanation: The LDAP Server set the debug level to the specified value.

System Action: The program continues.

Operator Response: None.

GLD0092E Unable to open any configuration file. No configuration file specified at startup, tried DDname *DD_name* and default name *file_name*.

Severity: Error

Explanation: The LDAP program is unable to open any configuration file. No configuration file name was specified at startup using the **-f** option. The program tried the specified DDname and then the default configuration file, but was unable to find any configuration file. The program cannot start without a configuration file.

System Action: The program ends.

Operator Response: Correct the configuration file name and restart the server, or contact the administrator.

Administrator Response: Correct the file name or permissions and restart the server.

GLD0094I Bind attempt by *bind_dn* was unsuccessful.

Severity: Attention

Explanation: The LDAP Server was unable to successfully complete the bind request.

System Action: The program continues.

Operator Response: None.

GLD0107E The configuration file produces a loop within the include statements.

Severity: Error

Explanation: A loop was detected within the configuration file processing

System Action: The program ends.

Operator Response: Contact the administrator.

Administrator Response: Check the configuration file include statements

GLD0108I No object class was specified for entry *entry_dn*.

Severity: Attention

Explanation: All entries must specify an object class.

System Action: The entry is not added and program continues.

Operator Response: Verify the `ldif` file syntax.

Administrator Response: None.

GLD0109I The required attribute *attribute_name* is missing for entry *entry_dn*.

Severity: Attention

Explanation: A required attribute for the specified object class was not provided.

System Action: The entry is not added and program continues.

Operator Response: Verify the `ldif` file syntax.

Administrator Response: Refer to the schema files for a list of the attributes required for each object class.

GLD0110I The attribute *attribute_name* is not allowed for entry *entry_dn*.

Severity: Attention

Explanation: A specified attribute is not allowed for the given object class.

System Action: The entry is not added and program continues.

Operator Response: Verify the `ldif` file syntax.

Administrator Response: Refer to the schema files for a list of the attributes required for each object class.

GLD0111E An error occurred while processing schema file *file_name*: **line** *line_number* **syntax :<oc clause> ::= objectclass <ocname> [requires <attrlist>] [allows <attrlist>]**

Severity: Error

Explanation: The LDAP Server object class schema file contains a schema definition that is not valid.

System Action: The program ends.

Operator Response: None.

Administrator Response: None.

GLD0114E A client sent nonsecured communications to the SSL port.

Severity: Error

Explanation: The LDAP Server determined that a client sent unencrypted data to the secure port. The request from the client is ended.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: If a command utility such as **ldapsearch** is called by the client and secure communications is intended, make sure the **-Z** (use secure communications) parameter is specified. If the client does not intend to use secure communications then specify the nonsecure port.

Programmer Response: If secure communications is intended, make sure the client calls **ldap_ssl_start**. If secure communications is not intended then specify the nonsecure port.

GLD0115I Workload Manager enablement initialization successful for group=*sysplex_groupname***, server=***sysplex_servername* **on host** *host_name*.

Severity: Informational

Explanation: The LDAP Server successfully registered with Sysplex Workload Manager. The LDAP Server will operate in multi-server mode. Multiple concurrent servers may be running on a given OS/390 image, and multiple concurrent servers may be running on multiple OS/390 images in a parallel sysplex, all of which use the same LDAP DB2 database. Note that **no** replication may be performed by any of these servers. If replication is desired, only one server instance may be running which uses a given LDAP DB2 database, and the server must be configured to run in single-server mode.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0116I Workload Manager enablement initialization failed for group=*sysplex_groupname***, server=***sysplex_servername* **on host** *host_name*. **No Workload Manager support will be activated for this server. RC =** *return_code*.

Severity: Attention

Explanation: The LDAP Server registration with Sysplex Workload Manager failed.

System Action: The program continues.

Operator Response: None.

Administrator Response: Contact the service representative and provide the return code in this message.

GLD0117I Workload Manager enablement termination successful for group=*sysplex_groupname***, server=***sysplex_servername* **on host** *host_name*.

Severity: Informational

Explanation: The LDAP Server successfully unregistered from Sysplex Workload Manager.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0118I Workload Manager enablement termination failed for group=*sysplex_groupname***, server=***sysplex_servername* **on host** *host_name*. **RC =** *return_code*.

Severity: Attention

Explanation: The LDAP Server was unable to unregister from Sysplex Workload Manager.

System Action: The program continues.

Operator Response: None.

Administrator Response: If the problem persists, contact the service representative.

GLD0119I Workload Manager enablement initialization failed for group=*sysplex_groupname*, server=*sysplex_servername*. No Workload Manager support will be activated for this server. RC = *return_code*.

Severity: Attention

Explanation: The LDAP Server was unable to register with Sysplex Workload Manager. This is an internal program error. Contact the service representative with the RC value displayed.

System Action: The program continues.

Operator Response: None.

Administrator Response: Contact the service representative.

GLD0120I Workload Manager enablement termination failed for group=*sysplex_groupname*, server=*sysplex_servername*. RC = *return_code*.

Severity: Attention

Explanation: The LDAP Server was unable to deregister from Sysplex Workload Manager. This is an internal program error. Contact the service representative with the RC value displayed.

System Action: The program continues.

Operator Response: None.

Administrator Response: Contact the service representative.

GLD0121I The object class *object_class* specified for entry *distinguished_name* is not defined in the schema.

Severity: Attention

Explanation: An object class specified in a request for the given DN is not defined in the current schema.

System Action: The entry is not added and program continues.

Operator Response: None.

Administrator Response: Verify the object class in the request and the object classes defined in the schema configuration files and try the request again.

GLD0122I Slapd is ready for requests.

Severity: Informational

Explanation: The LDAP Server is listening and is ready for requests.

System Action: The program continues.

Operator Response: None.

GLD0123I Workload Manager enablement initialization failed for group=*sysplex_groupname*, server=*sysplex_servername* on host *host_name* because this group/server combination is already registered on this host. No Workload Manager support will be activated for this server. RC = *return_code*.

Severity: Attention

Explanation: The LDAP Server registration with Sysplex Workload Manager failed because another server has already been registered by the same name in the same sysplex group.

System Action: The program continues.

Operator Response: Ensure that the `sysplexServerName` in the server configuration file (slapd.conf) is unique among all servers started with the same `sysplexGroupName`. If the problem persists, contact the service representative.

Administrator Response: Ensure that the `sysplexServerName` in the server configuration file (slapd.conf) is unique among all servers started with the same `sysplexGroupName`. If the problem persists, contact the service representative.

GLD0124I Dynamic workload management enabled. Server will operate in multi-server mode. sysplexServerName = *sysplex_server_name*, sysplexGroupName = *sysplex_group_name*.

Severity: Informational

Explanation: The LDAP Server will operate in multi-server mode. Multiple concurrent servers may be running on a given OS/390 image, and multiple concurrent servers may be running on multiple OS/390 images in a parallel sysplex, all of which use the same LDAP DB2 database. Note that **no** replication may be performed by any of these servers. If replication is desired, only one server instance may be running which uses a given LDAP DB2 database, and the server must be configured to run in single-server mode.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0125I Only one of the sysplexGroupName or sysplexServerName keywords is present in the configuration file.

Severity: Attention

Explanation: One of the sysplex keywords `sysplexGroupName` or `sysplexServerName` was found in the server configuration file. If either of these keywords is present, the other must also be present, and both keywords must be accompanied by non-null arguments.

System Action: The program continues. However, configuration may fail.

Operator Response: None.

Administrator Response: None.

GLD0126E The length of *sysplex_keyword* must be <= *maximum_argument_length*.

Severity: Error

Explanation: The length of the `sysplexGroupName` argument must not be greater than 18 characters. The length of the `sysplexServerName` argument must not be greater than 8 characters.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the keyword argument and restart the server.

GLD0127I Workload Manager enablement initialization failed due to memory allocation error. No Workload Manager support will be activated for this server.

Severity: Informational

Explanation: The LDAP Server could not allocate memory needed to register with Workload Manager. The LDAP Server will continue to operate, but no Workload Manager functions will be available.

System Action: The program continues.

Operator Response: None.

Administrator Response: Start the server again when system memory demands have been reduced.

GLD0128E The certificate sent by the client or the server certificate in key file *file_name* is not valid.

Severity: Error

Explanation: The LDAP Server found that the certificate sent by the client or the default certificate in the `keyring` file is not valid.

System Action: The program continues. The client request fails.

Operator Response: None.

Administrator Response: Verify that the server certificate in the keyring file certificate information are the issuer Certificate Authority and the expiration time of the certificate.

GLD0129I The value for the 'maxConnections' option is out of range (*min_Connection*, *max_Connection*). The default (*max_Connection*) will be used.

Severity: Attention

Explanation: The LDAP Server is unable to process the request because the value of `maxConnections` is out of range. The server will continue with the default value

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0130I The value for the 'waitingthreads' option is out of range (*min_threads*, *max_threads*). The default (*max_threads*) will be used.

Severity: Attention

Explanation: The LDAP Server encountered an error when processing the `waitingthreads` parameter of the configuration file. The server will continue with the default value.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0131I The value for the sizelimit option is not numeric. The default (*default_sizelimit*) will be used.

Severity: Attention

Explanation: The LDAP Server determined that the value specified for the `sizelimit` option of the configuration file is not numeric. The server will continue with the default value.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0132I The value for the timelimit option is not numeric. The default (*default_timelimit*) will be used.

Severity: Attention

Explanation: The LDAP Server determined that the value specified for the `timelimit` option of the configuration file is not numeric. The server will continue with the default value.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0133E Unable to process the objectclass definition in file *configuration_file* **at line** *line_number***.**

Severity: Error

Explanation: The LDAP Server encountered an error during processing of the `objectclass` option at the specified line in the specified configuration file.

System Action: The program continues. However, configuration may fail.

Operator Response: None.

Administrator Response: Correct the objectclass definition and restart the server.

GLD0134E Unable to process the attribute definition in file *configuration_file* **at line** *line_number***.**

Severity: Error

Explanation: The LDAP Server encountered an error during processing of the `attribute` option at the specified line in the specified configuration file.

System Action: The program continues. However, configuration may fail.

Operator Response: None.

Administrator Response: Correct the attribute definition and restart the server.

GLD0135I The value specified for the readonly option is not valid. The default (*default_readonly***) will be used.**

Severity: Attention

Explanation: The LDAP Server encountered an error during processing of the `readonly` option of the configuration file. The default value will be used.

System Action: The program continues.

Operator Response: None.

Administrator Response: If readonly is desired, correct the configuration file and restart the server.

GLD0136E The value specified for the masterserverDN option is not valid.

Severity: Error

Explanation: The LDAP Server encountered an error during processing of the `masterserverDN` option of the configuration file. The `masterserverDN` cannot have a NULL value.

System Action: The program continues. However, configuration may fail.

Operator Response: None.

Administrator Response: Correct the configuration file and restart the server.

GLD0137E The value specified for the masterserverDN option is 'cn=Anybody'. This is not permitted.

Severity: Error

Explanation: The LDAP Server encountered an error during processing of the `masterserverDN` option of the configuration file. The `masterserverDN` cannot be 'cn=Anybody'.

System Action: The program continues. However, configuration may fail.

Operator Response: None.

Administrator Response: Correct the configuration file and restart the server.

GLD0138I The value for the maxConnections option is not numeric. The default (*default_max_connections***) will be used.**

Severity: Attention

Explanation: The LDAP Server determined that the value specified for the `maxConnections` option of the configuration file is not numeric. The server will continue with the default value.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0139I The value for the waitingThreads option is not numeric. The default (*default_waiting_threads***) will be used.**

Severity: Attention

Explanation: The LDAP Server determined that the value specified for the `waitingThreads` option of the configuration file is not numeric. The server will continue with the default value.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0140I The value for the maxThreads option is not numeric. The default (*default_maximum_threads*) will be used.

Severity: Attention

Explanation: The LDAP Server determined that the value specified for the `maxThreads` option of the configuration file is not numeric. The server will continue with the default value.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD0141I A backend (*backend_address*) of type *backend_type* failed to configure.

Severity: Attention

Explanation: The LDAP Server encountered an error during configuration of the specified backend. This backend will not be available when the server starts.

System Action: The program continues. Other backends that configure successfully will be available. If no backends configure successfully, an additional message will appear, and the program will end.

Operator Response: None.

Administrator Response: Check for other messages regarding errors during configuration. Correct the configuration file and restart the server.

GLD0142I The value *URL_value* for the *configuration_option* option is not a valid URL.

Severity: Attention

Explanation: The LDAP Server determined that the value provided for the specified option of the configuration file is not valid. An LDAP URL is expected.

System Action: The program continues. However, configuration may fail.

Operator Response: None.

Administrator Response: Correct the configuration file parameter and restart the server.

GLD0143E The LDAP program cannot create required configuration structures.

Severity: Error

Explanation: The LDAP program encountered an error when establishing a required configuration structure. The program cannot start.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure adequate memory for the program. Correct any other reported errors and restart the program.

GLD0144E The LDAP Server encountered an error during configuration.

Severity: Error

Explanation: The LDAP Server encountered an error during configuration. The program cannot start. See other messages regarding errors.

System Action: The program ends.

Operator Response: None.

Administrator Response: Examine other messages and correct any other reported errors. Restart the LDAP Server.

GLD0145I The value for the -d command line option is not numeric. Value is ignored.

Severity: Attention

Explanation: The LDAP Server determined that the value specified for the **-d** option on the command line invocation of the program is not numeric. The value is ignored. The **-d** option specifies the level of debug statements to be produced.

System Action: The program continues.

Operator Response: If debug is needed, restart the server with corrected command line options.

Administrator Response: None.

GLD0146I The value for the -s command line option is not numeric. Value is ignored.

Severity: Attention

Explanation: The LDAP Server determined that the value specified for the **-s** option on the command line invocation of the program is not numeric. The value is ignored. The **-s** option identifies the secure port.

System Action: The program continues.

Operator Response: Restart the server with corrected command line options.

Administrator Response: None.

GLD0147I The value for the -p command line option is not numeric. Value is ignored.

Severity: Attention

Explanation: The LDAP Server determined that the value specified for the **-p** option on the command line invocation of the program is not numeric. The value is ignored. The **-p** option identifies the port.

System Action: The program continues.

Operator Response: Restart the server with corrected command line options.

Administrator Response: None.

GLD0148E The dllload function failed loading path *loadpath* with errno=*errno*, errno2=*errno2*.

Severity: Error

Explanation: The LDAP Server was unable to load the requested library. See explanations of the **errno** and **errno2** values for more information.

System Action: The program continues. However, configuration may fail.

Operator Response: None.

Administrator Response: Correct the error and restart the server.

GLD0149I The value specified for the extendedGroupSearching option is not valid. The default value of (*default_extendedGroupSearching*) will be used.

Severity: Attention

Explanation: The LDAP Server encountered an error during processing of the `extendedGroupSearching` option of the configuration file. The default value will be used.

System Action: The program continues.

Operator Response: None.

Administrator Response: If extended group searching is desired, correct the configuration file and restart the server.

GLD0150E The LDAP program requires an RDBM backend but none is configured.

Severity: Error

Explanation: An RDBM backend must be configured for this LDAP program to run. Either no RDBM backend was configured or an error was encountered during RDBM configuration. The LDAP program cannot start.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure RDBM configuration is present. Correct any other reported errors and restart the program.

GLD0151E Object class *object_class* requires attribute type *attribute_type* which is not defined.

Severity: Error

Explanation: The LDAP Server encountered an error during processing of the object class definitions.

System Action: The program ends.

Operator Response: Comment the specification of the object class in error and restart the server. Contact the administrator to have the specification of the object class corrected.

Administrator Response: Correct the specification of the object class in error and restart the server.

GLD0152I The value specified (*value_specified*) for the verifySchema option is not correct. The default value of (*default_verifySchema*) will be used.

Severity: Attention

Explanation: The LDAP Server encountered an error during processing of the `verifySchema` option in the configuration file. The default value will be used.

System Action: The program continues.

Operator Response: None.

Administrator Response: If the default value for `verifySchema` is not desired, correct the configuration file and restart the server.

GLD2001I ACL: User is not allowed to specify ACL or owner values at object creation time. Inherited values will be used for these attributes.

Severity: Attention

Explanation: An error occurred when trying to create an object because the ACL or owner value was specified, but the parent object has `inheritOnCreate` set to true. The object is created with the ACL and owner attribute values inherited from the parent object. The ACL and owner attribute values can be modified to the desired values.

System Action: The program continues.

Operator Response: None.

Administrator Response: Modify the ACL and owner attributes in a separate request.

Programmer Response: An additional request is needed to modify the ACL and owner attributes.

GLD2003I Error code *error_code* **from odbc string: "** *odbc_string* **"** *substring* **.**

Severity: Informational

Explanation: An error occurred when performing DB2 operations. The request fails. There may be an additional message with information about SQL return codes.

System Action: The program continues.

Operator Response: None.

GLD2004I ldif2db: *number* **entries have been successfully added out of** *number* **attempted.**

Severity: Informational

Explanation: The utility **ldif2db** successfully added the specified number of entries to the database.

System Action: The program continues.

Operator Response: None.

GLD2005E ldif2db: Database is set to read-only in the configuration file.

Severity: Error

Explanation: The utility **ldif2db** cannot add any entries to the database because the database is configured to allow only reads, and not updates.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure the database is configured correctly and try again.

GLD2007E *program_name***: line** *line_number***: incorrect configuration line:** *option* **takes parameters** *values***.**

Severity: Error

Explanation: The relational data store for the LDAP Server cannot be configured because of the specified error in the configuration file.

System Action: The program continues. If all required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: Correct the configuration file and try again.

GLD2008E *command_name***: line** *line_number***: incorrect index line:** *option* **takes parameters** *values***.**

Severity: Error

Explanation: The relational data store for the LDAP Server cannot be configured because of the specified error in the index portion of the configuration file.

System Action: The program continues. If all required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: Correct the configuration file and try again.

GLD2009E *program_name***: line** *line_number***: incorrect configuration line: too many parameters.**

Severity: Error

Explanation: The relational data store for the LDAP Server cannot be configured because the specified line in the configuration file contains too many parameters.

System Action: The program continues. If all required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: Correct the configuration file and try again.

GLD2010E *program_name***: line** *line_number***: incorrect configuration line: unrecognized keyword.**

Severity: Error

Explanation: The relational data store for the LDAP Server cannot be configured because the specified line in the configuration file contains an unrecognized keyword.

System Action: The program continues. If all required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: Correct the configuration file and try again.

GLD2011E *keyword* **parameter is missing from configuration file.**

Severity: Error

Explanation: The relational data store for the LDAP Server cannot be configured because the configuration file is missing a required parameter.

System Action: The program continues. If all

required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: Correct the configuration file and try again.

GLD2013E *program_name*: **line** *line_number*: **incorrect configuration line:** *attribute_name* **attribute is not defined in the schema file.**

Severity: Error

Explanation: The relational data store for the LDAP Server cannot be configured because the specified line in the configuration file contains an attribute not defined in the schema.

System Action: The program continues. If all required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: Correct the configuration file or the schema file and try again.

GLD2014E *program_name*: **line** *line_number*: **incorrect configuration line:** *attribute_name* **attribute is binary and cannot be indexed.**

Severity: Error

Explanation: The requested index cannot be created because the type of the attribute does not allow indexing. The index is not created.

System Action: The program continues. If all required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: Correct the configuration file or the schema file and try again.

GLD2015E *program_name*: **line** *line_number*: **incorrect configuration line: 'none' cannot be combined with other index types.**

Severity: Error

Explanation: The requested index cannot be created because the index types requested are conflicting. The index is not created.

System Action: The program continues. If all required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: Correct the configuration file or the schema file and try again.

GLD2016E *program_name*: **line** *line_number*: **incorrect configuration line:** *index_type* **is not a valid index type. Valid index types are: 'eq', 'approx', or 'sub'.**

Severity: Error

Explanation: The requested index cannot be created because the index types requested are not correct.

System Action: The program continues. If all required parameters are not set correctly, the configuration will fail.

Operator Response: None.

Administrator Response: Correct the configuration file and try again.

GLD2017I '*distinguished_name*' **does not have permission to all parts of the filter. The search operation cannot return any results.**

Severity: Attention

Explanation: The requested data cannot be returned because the requestor is not permitted to read all of the data requested.

System Action: The program continues. The operation returns no data.

Operator Response: None.

Administrator Response: Correct the ACL permissions for the data and try again.

Programmer Response: Modify the request and try again. If access is needed to the data, contact the Administrator.

GLD2018I Only one attribute-value pair may be compared at a time.

Severity: Attention

Explanation: The request cannot be completed because more than one attribute-value pair is being compared in the request.

System Action: The program continues. The operation returns no data.

Operator Response: None.

Administrator Response: None.

Programmer Response: Modify the request and try again.

GLD2019I Attribute presence checking may only be applied to one attribute at a time.

Severity: Attention

Explanation: The request cannot be completed because the request is attempting to test the presence of more than one attribute.

System Action: The program continues. The operation returns no data.

Operator Response: None.

Administrator Response: None.

Programmer Response: Modify the request and try again.

GLD2020I When adding (or deleting) the ownerPropagate field, the entryOwner field must also be added (or deleted).

Severity: Attention

Explanation: The request cannot be completed because the request is attempting to modify the `ownerPropagate` flag without also modifying the `entryOwner` field.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Modify the request and try again.

GLD2021I When adding (or deleting) the aclPropagate field, the aclEntry field must also be added (or deleted).

Severity: Attention

Explanation: The request cannot be completed because the request is attempting to modify the `aclPropagate` field without also modifying the `aclEntry` field.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Modify the request and try again.

GLD2022I An explicit ACL must be set in order to change or delete this attribute value.

Severity: Attention

Explanation: The request cannot be completed because an explicit ACL does not exist.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Modify the request and try again.

GLD2023I An explicit owner must be set in order to change or delete this attribute value.

Severity: Attention

Explanation: The request cannot be completed because an explicit owner does not exist.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Modify the request and try again.

GLD2024I This object already has an explicit ACL set. Use the ldapmodify -r command to change the ACL.

Severity: Attention

Explanation: The request cannot be completed because an explicit ACL already exists for the requested object.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Use **ldapmodify** for this request and try again.

GLD2025I Cannot create an explicit owner; an explicit owner already exists on this entry.

Severity: Attention

Explanation: The request cannot be completed because an explicit owner already exists for the requested entry.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Use **ldapmodify** for this request and try again.

GLD2026I Cannot delete and change the same attribute.

Severity: Attention

Explanation: The request cannot be completed because the request is attempting to both delete and modify the same attribute.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Modify the request and try again.

GLD2027I The object operation permission specified is not valid. Valid permissions are 'a' and 'd'.

Severity: Attention

Explanation: The request cannot be completed because the permissions being specified for the object are not correct.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the permissions in the request and try again.

GLD2028I The access class permission specified is not valid. Valid permissions are 'r', 'w', 'c' and 's'.

Severity: Attention

Explanation: The request cannot be completed because the access class permissions specified are not correct.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the permissions in the request and try again.

GLD2029I The DnType *type* specified in the aclentry is not valid. Valid values are: 'access-id' and 'group'.

Severity: Attention

Explanation: The request cannot be completed because the `DnType` specified is not correct.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2030I The syntax specified for entryOwner is not valid. Syntax of this field is dnType:dn.

Severity: Attention

Explanation: The request cannot be completed because the syntax of the `entryOwner` field is not correct.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2031I The prefix *prefix* specified for a permission set is not valid. Valid prefixes are: 'normal', 'critical', 'sensitive', and 'object'.

Severity: Attention

Explanation: The request cannot be completed because the prefix specified for the permissions is not correct.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2032I The specified permission set *permission* is not valid.

Severity: Attention

Explanation: The request cannot be completed because the permissions specified are not correct.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2033I Objects of type ReplicaObject must be added under the cn=localhost entry. Cannot add new entry '*distinguished_name*'.

Severity: Attention

Explanation: The request cannot be completed because an attempt is being made to add a `ReplicaObject` under a `cn` other than `cn=localhost`.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2034I Only objects of type ReplicaObject can be added under the cn=localhost entry. Cannot add new entry '*distinguished_name*'.

Severity: Attention

Explanation: The request cannot be completed because an attempt is being made to add an object under `cn=localhost` that is not a `ReplicaObject`.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2035E The dsnaoini CLI initialization file was not specified.

Severity: Error

Explanation: The LDAP Server (or LDAP utility) cannot complete initialization of the DB2 backend because dsnaoini was not specified in the slapd.conf file or there is no DD specified for DSNAOINI in the JCL.

System Action: The program ends.

Operator Response: None.

Administrator Response: Specify the dsnaoini in the slapd.conf file or the DD DSNAOINI in the JCL.

GLD2036I Attribute *attribute_name* was not found in the schema definition.

Severity: Attention

Explanation: An error occurred because the specified attribute was not found in the schema definition.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: Verify that the schema definitions are correct.

Programmer Response: Correct the attribute in the request and try again.

GLD2037I Entry '*distinguished_name*' already exists.

Severity: Attention

Explanation: The request cannot be completed because an attempt is being made to add an entry that already exists in the database.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2038I Entry '*distinguished_name*' violates the schema definition.

Severity: Attention

Explanation: The request cannot be completed because an attempt is being made to add an entry that does not match the schema definition.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2039I Parent entry does not exist for entry '*distinguished_name*'.

Severity: Attention

Explanation: The request cannot be completed because no parent entry exists for the entry being added.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2040I Entry size *hex_value* **is greater than the maximum size supported (***hex_value***).**

Severity: Attention

Explanation: The request cannot be completed because the specified data exceeds the maximum supported size for an entry.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2041I SQL call returned unexpected return code *return_code* **on call:** *SQL_call***.**

Severity: Attention

Explanation: The request cannot be completed because an unexpected return code was received from an SQL call.

System Action: The program continues. The request fails.

Operator Response: Ensure DB2 is operating correctly.

Administrator Response: None.

GLD2042I Object '*distinguished_name***' does not exist.**

Severity: Attention

Explanation: The request cannot be completed because the specified object does not exist.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2043I The attribute *attribute_name* **with value** *value* **already exists.**

Severity: Attention

Explanation: The request cannot be completed because the attribute-value pair already exists in the database.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2044I Entry '*distinguished_name***' does not contain attribute** *attribute_name***.**

Severity: Attention

Explanation: The request cannot be completed because the entry does not contain the specified attribute.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2045I Entry '*distinguished_name***' does not contain attribute** *attribute_name* **with value** *value***.**

Severity: Attention

Explanation: The request cannot be completed because the entry does not contain the specified attribute with the specified value.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2046I RDN '*relative_distinguished_name***' is not valid.**

Severity: Attention

Explanation: The request cannot be completed because the specified Relative Distinguished Name (RDN) is not correct.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2047I Error: Attribute *attribute_name* has a maximum value length of *length*. Current attribute value is of length *length*.

Severity: Attention

Explanation: The request cannot be completed because the specified attribute value exceeds the maximum length supported for this attribute.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Correct the request and try again.

GLD2048E ldif2db: Error, cannot find input file *file_name*.

Severity: Error

Explanation: The **ldif2db** utility could not find the input `ldif` file.

System Action: The program stops.

Operator Response: Verify the name of the input file and try again.

Administrator Response: None.

Programmer Response: None.

GLD2049I ldif2db: Error, attribute *attribute_name* on line *line_number* is not preceded by a dn.

Severity: Attention

Explanation: The `ldif` file has an entry that does not begin with a distinguished name (DN).

System Action: The program continues.

Operator Response: Verify the name and format of the `ldif` file.

Administrator Response: None.

Programmer Response: None.

GLD2050E The dynalloc function call failed. Errcode = *error_code*, Infocode = *info_code*, dsnaoini = *dataset_name*.

Severity: Error

Explanation: The LDAP Server (or LDAP utility) could not complete initialization because the **dynalloc()** function call failed.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the error code and make sure the correct dsnaoini file is specified in the LDAP Server configuration file.

GLD2051I ODBC error, SQL data is: native return code=*SQL_code*, SQL state=*SQL_state*, SQL message=*SQL_message*.

Severity: Informational

Explanation: An error occurred when performing DB2 operations. The information in the message is the data available from SQL at the time of the error.

System Action: The program continues.

Operator Response: None.

Administrator Response: Evaluate and resolve the DB2 problem with the information provided. If the problem cannot be resolved, contact the service representative.

GLD2052E An error occurred during *backend_type* backend initialization, rc = *return_code*.

Severity: Error

Explanation: An error occurred during backend initialization.

System Action: The program ends.

Operator Response: None.

Administrator Response: Make sure that DB2 is started and running properly. Check for any additional error messages that may be given. Fix these errors and try again. If the problem cannot be resolved, contact the service representative.

GLD2057I Only one entryOwner per object is allowed. Error setting entry owner dn: '*distinguished_name*'

Severity: Attention

Explanation: Only one `entryOwner` is allowed per object. Additional values will not be accepted.

System Action: The modification is not made and the program continues.

Operator Response: None.

Administrator Response: Correct the entry and try the request again.

GLD2058I Distinguished name '*distinguished_name*' may have partial access.

Severity: Attention

Explanation: An error occurred when obtaining the group permissions for the specified distinguished name. The DN may have been granted partial access.

System Action: The program continues.

Operator Response: None.

Administrator Response: Examine the ACL and group entries for the specified DN, and correct any malformed entries.

GLD2060I Referral object modifications detected. Referral cache being updated. *date_and_time*

Severity: Informational

Explanation: The program detected that referral entries have changed, or have been added or deleted. The memory-resident referral cache will be updated to reflect the changes.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD2061I A referral cache update failed. Using old cache.

Severity: Informational

Explanation: The program attempted to refresh the referral cache, but the refresh operation failed. The existing cache contents will be used.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD2062I The LDAP program will operate in single-server mode.

Severity: Informational

Explanation: The LDAP program will operate in single-server mode. Only one instance of the LDAP Server or LDAP utility may be running against a given LDAP DB2 database. Replication will be performed if this is the master server and replication objects are present. If the intent is to establish multiple concurrent LDAP Server or LDAP utility instances using the same LDAP DB2 database, all of those instances must be configured to operate in multi-server mode.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD2063I The LDAP Server will operate in multi-server mode.

Severity: Informational

Explanation: The LDAP Server (or LDAP utility) will operate in multi-server mode. Multiple concurrent servers may be running on a given OS/390 image, and multiple concurrent servers may be running on multiple OS/390 images in a parallel sysplex, all of which use the same LDAP DB2 database. Note that **no** replication may be performed by any of these servers. If replication is desired, only one server instance may be running which uses a given LDAP DB2 database, and the server must be configured to run in single-server mode.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD2064E Current DB schema version = *current_db_version***. DB schema must be migrated to version** *minimum_db_version* **or higher to operate in multi-server mode.**

Severity: Error

Explanation: If the database version is not 5.1 or higher and the server is running in a multi-server environment, the application will be terminated. To avoid this termination the database must be migrated to version 5.1 or higher.

System Action: The program ends.

Operator Response: None.

Administrator Response: Migrate the database schema to version 5.1 or higher and restart the server.

GLD2065I Current DB schema version is *current_db_version***; maximum supported DB schema version is** *database_implementation_version***.**

Severity: Informational

Explanation: Current DB schema version is not the highest version supported by the level of LDAP code being run. The highest DB schema version supported by this level of LDAP is provided.

System Action: The program continues.

Operator Response: None.

Administrator Response: The LDAP program will continue to operate in this mode. However, it may be necessary to upgrade the DB schema version to the highest level supported if new functions are needed. See the Migration section of the *OS/390 Security Server LDAP Server Administration and Usage Guide* for more information about upgrading DB schema versions.

GLD2066I Multiserver keyword argument is 'n' or 'N', but the LDAP Server will operate in multi-server mode.

Severity: Informational

Explanation: The `multiserver` keyword argument found in the server configuration file was either 'n' or 'N'. However, because both `sysplexServerName` and `sysplexGroupName` keywords have valid arguments, the

LDAP Server (or LDAP utility) must operate in multi-server mode.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD2067E The LDAP Server will operate in multi-server mode. The tbspacemutex keyword must also be present.

Severity: Error

Explanation: The `multiserver` keyword was present in the LDAP Server configuration file; the `tbspacemutex` keyword must also be present.

System Action: The program ends.

Operator Response: None.

Administrator Response: Add the `tbspacemutex` keyword to the server configuration file and restart the server.

GLD2068I Unable to acquire database lock on attempt *count*.

Severity: Informational

Explanation: Repeated attempts to acquire the database lock have failed. If this condition persists, it may indicate unrecoverable errors. Note that occasional failures are normal during periods of high database update activity.

System Action: The program continues.

Operator Response: None.

Administrator Response: If this condition persists, contact the service representative.

GLD2069I Locking or unlocking related to referral cache refresh operation has failed. Future referral cache refreshes may be unreliable.

Severity: Attention

Explanation: A locking operation associated with a referral cache refresh has failed. Depending upon whether this was a transient failure or a persistent failure, and depending upon the state of the cache refresh mechanism at the time of failure, future attempts to refresh the cache may fail to update the cache properly.

System Action: The program continues.

Operator Response: None.

Administrator Response: If this message is repeated frequently, stop the LDAP Server and restart it.

GLD2070I Locking or unlocking related to referral cache operation has failed. Future referral cache operations may be unreliable.

Severity: Attention

Explanation: A locking operation associated with a referral cache reference has failed. Depending upon whether this was a transient failure or a persistent failure, and depending upon the state of the caching mechanism at the time of failure, future attempts to reference the cache may yield unexpected results.

System Action: The program continues.

Operator Response: None.

Administrator Response: If this message is repeated frequently, stop the LDAP Server and restart it.

GLD2071I A database operation related to referral cache operation has failed. Future referral cache operations may be unreliable.

Severity: Attention

Explanation: A database operation associated with a referral cache reference has failed. Depending upon whether this was a transient failure or a persistent failure, and depending upon the state of the caching mechanism at the time of failure, future attempts to reference the cache may yield unexpected results.

System Action: The program continues.

Operator Response: None.

Administrator Response: If this message is repeated frequently, stop the LDAP Server and restart it.

GLD2072I Initialization related to referral cache operation has failed. rc = *return_code*.

Severity: Attention

Explanation: Initialization related to referral cache operation has failed. Depending upon whether this was a transient failure or a more permanent failure, the problem may be resolved by restarting the server.

System Action: The program ends.

Operator Response: None.

Administrator Response: If this message repeats after restarting the server several times, contact the service representative.

GLD2073E The LDAP program found no RDBM database.

Severity: Error

Explanation: The LDAP program cannot continue because the RDBM database was not configured correctly.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure the database is configured correctly and try again.

GLD2074E Overlapping RDBM backend suffixes found in configuration file: suffixes '*first_overlapping_suffix*' and '*second_overlapping_suffix*' overlap.

Severity: Error

Explanation: The presence of overlapping suffixes was detected in the server configuration file for an RDBM backend. Overlapping suffixes (suffixes for which the hierarchy is identical to the extent of the shorter of the two) may not be specified. Eliminate the overlap in suffixes and restart the server.

System Action: The program ends.

Operator Response: None.

Administrator Response: Eliminate the overlap in RDBM suffixes and restart the server.

GLD2075I Parent of new entry is a referral object. Cannot add new entry '*distinguished_name*'.

Severity: Attention

Explanation: The request cannot be completed because an entry cannot be added directly below a referral object. The entry must be added to the namespace below the actual entry which is referenced by the referral object.

System Action: The program continues. The request fails.

Operator Response: None.

Administrator Response: None.

Programmer Response: Direct the request to the correct location in the namespace and try again.

GLD2076I ldif2db found aclentry attributes for an entry where aclsource differs from entry dn '*distinguished_name*'; aclentry attributes are ignored.

Severity: Attention

Explanation: When adding the specified entry, **ldif2db** determined there is an incompatability between the value of the `aclsource` attribute and the presence of the `aclentry` attribute. To specify an explicit ACL with `aclentry`, the value of the attribute `aclsource` must be the entry DN. The entry is added with default ACL information.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the entry using **ldapmodify**, or delete the entry, correct the `ldif` file and run **ldif2db** again.

GLD2077I ldif2db found entryowner attributes for an entry where ownersource differs from entry dn '*distinguished_name*'; entryowner attribute is ignored.

Severity: Attention

Explanation: When adding the specified entry, **ldif2db** determined there is an incompatability between the value of the `ownersource` attribute and the presence of the `entryowner` attribute. To specify an explicit owner with `entryowner`, the value of the `ownersource` attribute must be the entry DN. The entry is added with default owner information.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the entry using **ldapmodify**, or delete the entry, correct the `ldif` file and run **ldif2db** again.

GLD2078E The LDAP program encountered an error during configuration.

Severity: Error

Explanation: An LDAP program encountered an error while processing the configuration file. The program cannot continue. See additional messages for more information about the error encountered.

System Action: The program ends.

Operator Response: None.

Administrator Response: Examine the additional messages. Correct the configuration file and run the program again.

GLD2079E The LDAP program found no backends.

Severity: Error

Explanation: An LDAP program encountered an empty list of configured backends. The program cannot continue.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure there is at least one database line specified in the configuration file and run the program again.

GLD2080E The LDAP program found incomplete database information.

Severity: Error

Explanation: An LDAP program encountered an empty list of RDBM-specific information. The program cannot continue.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure the database is configured correctly and run the program again.

GLD2081I ldif2db failed to load entry with dn '*distinguished_name***' around line** *line_number***, LDAPRC=***return_code***.**

Severity: Informational

Explanation: **ldif2db** encountered the specified error when loading the entry with the specified distinguished name. The entry can be located near the specified line number in the input `ldif` file.

System Action: The program continues.

Operator Response: None.

Administrator Response: Depending upon the LDAP return code, it may be necessary to correct the entry before trying the load again. Refer to the *OS/390 Security Server LDAP Client Application Development Guide and Reference* manual for more information regarding LDAP return codes.

GLD2082I ldif2db failed to create UTF8 data from source at line *line_number***, LDAPRC=***return_code***.**

Severity: Informational

Explanation: **ldif2db** encountered the specified error when trying to establish a UTF-8 encoding for part of an entry. The failing attribute can be located at the specified line number in the input `ldif` file.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the entry and try the load again.

GLD2083I ldif2db failed during load of entry at line *line_number***, LDAPRC=***return_code***.**

Severity: Informational

Explanation: **ldif2db** encountered the specified error when trying to load the entry around the specified line number in the input `ldif` file.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the entry and try the load again.

GLD2084E The LDAP program does not support this encryption method.

Severity: Error

Explanation: The LDAP Server configuration file contains the keyword `pwEncryption` with an incorrect value. Correct values are **none**, **crypt**, **MD5**, **SHA**, or **DES**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the configuration file and try again.

GLD2085E OCSF setup failed, encryption method *method* **is not available.**

Severity: Error

Explanation: The `pwEncryption` value specified in the configuration file requires the OCSF product to be installed and available on the system.

System Action: The program ends.

Operator Response: None.

Administrator Response: Verify OCSF setup and try again or change the pwEncyption value in the configuration file to a method that does not need OCSF (**none** or **crypt**).

GLD2087I OCSF setup failed, but pwEncryption method *method* **is available.**

Severity: Attention

Explanation: The `pwEncryption` value specified in the configuration file does not require the OCSF product to be installed and available on your system, but any value already encrypted in MD5, SHA, or DES will not compare correctly on a bind.

System Action: The program continues.

Operator Response: None.

Administrator Response: If OCSF is needed, verify OCSF setup and then stop and start the LDAP program again.

GLD2088E DES key label not available, encyption method *method* is not available.

Severity: Error

Explanation: The `pwEncryption` value specified in the configuration file requires the OCSF product and the ICSF product to be installed and available on your system. It also requires a valid CKDS and the key corresponding to the DES key label in the configuration file to be available on your system.

System Action: The program ends.

Operator Response: None.

Administrator Response: Verify OCSF, ICSF, and CKDS setup and try again or change the pwEncyption value in the configuration file to a method that does not need OCSF, ICSF, or CKDS.

GLD2089E The DES key label specified with pwEncryption in the configuration file is too long.

Severity: Error

Explanation: The DES key label specified with `pwEncryption` in the configuration file can be a maximum of 64 characters long.

System Action: The program ends.

Operator Response: None.

Administrator Response: Set up a key with a key label with a valid length and try again.

GLD2090E The format of pwEncryption with DES is incorrect in the configuration file.

Severity: Error

Explanation: The format of `pwEncryption` with DES should be **DES**:*keylabel*.

System Action: The program ends.

Operator Response: None.

Administrator Response: Format the `pwEncryption` value correctly and try again.

GLD4001E Requested message number *max_number* beyond bounds of internal table.

Severity: Error

Explanation: The message specified by the message number cannot be displayed because it is not in the internal message table and was not found in any catalog.

System Action: The program continues.

Operator Response: Contact the service representative. An internal error has occurred.

GLD4002I Cannot open message catalog file *file_name*.

Severity: Attention

Explanation: The program was unable to locate the specified message catalog. Possible reasons include: message catalogs installed incorrectly or, **LANG** or **NLSPATH** environment variables may not be set or may be set incorrectly.

System Action: The program continues.

Operator Response: Verify the full path name to the message catalogs. Verify that the **LANG** and **NLSPATH** variables have the correct values to reach this path. Ensure the message catalogs are installed correctly and have the correct permissions. If the problem persists, contact your service representative.

GLD4003E Memory allocation failed; cannot continue. Program terminated.

Severity: Error

Explanation: An LDAP facility was unable to allocate the necessary memory to continue processing. The program is ending.

System Action: The program ends.

Operator Response: Ensure that the program has sufficient memory and try again. If the problem persists, contact the service representative.

GLD4004E Unable to write error message to console. Return code=*return_code*.

Severity: Error

Explanation: An LDAP program received an error from system routine **__console()** when attempting to write a message to the operator's console.

System Action: The program continues.

Operator Response: Contact the service representative.

GLD4005I Environment variable file not found. Environment variables not set. Continuing.

Severity: Attention

Explanation: The LDAP program was unable to find its environment variable file. Program continues with environment variables unset.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD4006I Environment variable file cannot be opened. Environment variables not set. Continuing.

Severity: Attention

Explanation: The LDAP program was unable to open its environment variable file. Program continues with environment variables unset.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD4007I Environment variable file contents in error. Environment variables not set. Continuing.

Severity: Attention

Explanation: The LDAP program encountered an incorrect line in its environment variable file. Program continues with environment variables unset.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD4008I Environment variables not set because error encountered. Continuing.

Severity: Attention

Explanation: The LDAP program encountered an unexpected error when processing its environment variable file. Program continues with environment variables unset.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD4009E The initialization of ServerGlobal structure failed.

Severity: Error

Explanation: The LDAP program encountered an error when establishing the ServerGlobal data structure. The program cannot continue without this structure.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure adequate storage for the program and try again.

GLD4010E The initialization of ConfigInfo structure failed.

Severity: Error

Explanation: The LDAP program encountered an error when establishing the ConfigInfo data structure. The program cannot continue without this structure.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure adequate storage for the program and try again.

GLD4011E The initialization of ThreadControl structure failed.

Severity: Error

Explanation: The LDAP program encountered an error when establishing the ThreadControl data structure. The program cannot continue without this structure.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure adequate storage for the program and try again.

GLD4012E The initialization of ReplInfo structure failed.

Severity: Error

Explanation: The LDAP program encountered an error when establishing the ReplInfo data structure. The program cannot continue without this structure.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure adequate storage for the program and try again.

GLD5001E Memory allocation failed; cannot continue. Program terminated.

Severity: Error

Explanation: The SDBM backend was unable to allocate the necessary memory to continue processing. The program is ending.

System Action: The program ends.

Operator Response: Ensure that the LDAP Server has sufficient memory and try again. If the problem persists, contact the service representative.

GLD5002E The __passwd function failed; not loaded from a program controlled library.

Severity: Error

Explanation: The LDAP Server with an SDBM backend needs to be loaded from a program controlled dataset for the **__passwd** function to work.

System Action: The client request fails with an operations error. The program continues.

Operator Response: Ensure that the LDAP Server and the SDBM backend are loaded from a program controlled dataset. Stop and start the server after checking the dataset.

GLD5003E SDBM backend internal error. Program continues.

Severity: Error

Explanation: The SDBM backend received an unexpected error from the **__passwd()** function while processing a bind request.

System Action: The client request fails with an operations error. The program continues.

Operator Response: None.

Administrator Response: Correct the request and try again. If the problem persists, contact the service representative.

GLD6001E Port number must be numeric, in the range 1 - 65535.

Severity: Error

Explanation: **ldapcp** is unable to communicate with any LDAP Server because the specified port number is not valid.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the port number parameter supplied to **ldapcp** and try again.

GLD6002I No debug level match for argument: *debug_level***. This debug level ignored.**

Severity: Attention

Explanation: **ldapcp** is unable to interpret the specified debug level.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the debug level parameter supplied to **ldapcp** and try again.

GLD6003I No syslog level match for argument: *syslog_level***. This syslog level ignored.**

Severity: Attention

Explanation: **ldapcp** is unable to interpret the specified syslog level.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the syslog level parameter supplied to **ldapcp** and try again.

GLD6004E Flag on invocation line not valid, or missing required argument: *bad_flag***.**

Severity: Error

Explanation: A flag or argument supplied to **ldapcp** was incorrect.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6005E More than one occurrence of -d detected. Only one permitted.

Severity: Error

Explanation: More than one distinguished name was supplied to **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6006I -v flag requires -I flag. -v ignored.

Severity: Attention

Explanation: The **-v** flag was supplied to **ldapcp** without the **-I** flag. If used, both are required.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6007I More than one occurrence of -I detected. First one will be used.

Severity: Attention

Explanation: More than one file path name was supplied to **ldapcp** to use to log activity. The messages will be placed into the file path name specified first.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6008E More than one occurrence of -w detected. Only one permitted.

Severity: Error

Explanation: More than one password was supplied to **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6009I No DN entered. Enter DN now.

Severity: Attention

Explanation: No distinguished name was supplied to **ldapcp**, and one is required.

System Action: The program continues.

Operator Response: None.

Administrator Response: Enter the DN parameter at the prompt.

GLD6010I No password entered. Enter password now.

Severity: Attention

Explanation: No password was supplied to **ldapcp**, and one is required.

System Action: The program continues.

Operator Response: None.

Administrator Response: Enter the password parameter at the prompt.

GLD6011E Unable to allocate storage.

Severity: Error

Explanation: **ldapcp** was unable to obtain the storage needed for processing.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure that the userID where **ldapcp** is being used has adequate storage to run **ldapcp** and try again. If the problem persists, contact your service representative.

GLD6012E Distinguished name required in command line mode.

Severity: Error

Explanation: No distinguished name was supplied to **ldapcp**, and one is required.

System Action: The program ends.

Operator Response: None.

Administrator Response: Supply the DN parameter on the command line or use **ldapcp** in interactive mode and you will receive a prompt for the information.

GLD6013E Password required in command line mode.

Severity: Error

Explanation: No password was supplied to **ldapcp**, and one is required.

System Action: The program ends.

Operator Response: None.

Administrator Response: Supply the password parameter on the command line or use **ldapcp** in interactive mode and you will receive a prompt for the information.

GLD6014E Object must be one of acl, group, help, quit, or exit.

Severity: Error

Explanation: None of the objects supplied to **ldapcp** were correct.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the object and try again.

GLD6015I Object must be one of acl, group, help, quit, or exit. Ignored.

Severity: Attention

Explanation: One of the objects supplied to **ldapcp** was incorrect. **ldapcp** will continue with the remaining objects.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the object and try again.

GLD6016I Unable to restore terminal echo. Typed characters will not display.

Severity: Attention

Explanation: **ldapcp** has turned terminal echo off and is unable to turn it back on. **ldapcp** will continue with terminal echo off.

System Action: The program continues.

Operator Response: None.

Administrator Response: If terminal echo is desired, exit from **ldapcp** and the OMVS shell, re-enter the shell and try **ldapcp** again.

GLD6017E No buffer space left for command line.

Severity: Error

Explanation: The command supplied to **ldapcp** is too long.

System Action: The program ends.

Operator Response: None.

Administrator Response: Modify the command to shorten it, or use **ldapcp** in interactive mode.

GLD6018I Communicating with server on default port.

Severity: Informational

Explanation: **ldapcp** is using the default port number, 389, to communicate with the LDAP Server.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD6019I Communicating with server on port *port_number***.**

Severity: Informational

Explanation: **ldapcp** is using the specified port number to communicate with the LDAP Server.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD6020I Unable to open logfile *file_name***; return code on open =***errno***. No messages will be logged.**

Severity: Attention

Explanation: **ldapcp** received an error from system function **fopen()** when attempting to open the specified file to log messages. This error is the result of file system or file name/file path-related problems. Some possible causes: The file path name is too long; the maximum number of open files has been exceeded; the target file system is full; a path name prefix component is not a directory; the path prefix is on a read-only file system.

System Action: The program continues.

Operator Response: None.

Administrator Response: Contact your service representative and provide the return code and file name displayed in the message.

GLD6021I Unable to open logfile *file_name***; create or append not permitted. No messages will be logged.**

Severity: Attention

Explanation: **ldapcp** received an access error from system function **fopen()** when attempting to open the specified file to log messages. **ldapcp** will continue without logging messages.

System Action: The program continues.

Operator Response: None.

Administrator Response: If a message log is desired, stop the program. If a log file already exists with the requested logfile name, ensure that it is writable by the userID running this program. If a log file does not already exist with the requested path, ensure that the userID running the program has access to all components of the path prefix, and can write files in the target directory. Then run **ldapcp** again.

GLD6022I Program terminated. RC=*return_code***.**

Severity: Informational

Explanation: **ldapcp** is ending.

System Action: The program ends.

Operator Response: None.

Administrator Response: None.

GLD6023I Selected debug levels: 0x*hex_value***.**

Severity: Informational

Explanation: **ldapcp** is using the specified debug levels during processing.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD6024I Selected syslog levels: 0x*hex_value*.

Severity: Informational

Explanation: **ldapcp** is using the specified syslog levels during processing.

System Action: The program continues.

Operator Response: None.

Administrator Response: None.

GLD6025E Verb must be one of create, delete, modify, query or remove.

Severity: Error

Explanation: None of the verbs supplied to **ldapcp** were correct.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the verb and try again.

GLD6027E No object argument was input for this command.

Severity: Error

Explanation: The command specified requires an object argument, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command arguments and try again.

GLD6029E No ACL entry argument was input for this command.

Severity: Error

Explanation: The command specified requires an ACL entry argument, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command arguments and try again.

GLD6031E No owner argument was input for this command.

Severity: Error

Explanation: The command specified requires an owner argument, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command arguments and try again.

GLD6033E No subverb was input for this command.

Severity: Error

Explanation: The command specified requires a subverb, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command and try again.

GLD6035E Extra input for this command has been entered and is ignored.

Severity: Error

Explanation: The command specified supplied extraneous data. This data will be ignored.

System Action: The program continues.

Operator Response: None.

Administrator Response: None. Correct the data in additional commands.

GLD6036I Call to sigaction failed with rc=*return_code*.

Severity: Attention

Explanation: **ldapcp** received the specified return code from system function sigaction() when attempting to establish the signal handler.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6037I SIGINT received; program terminated.

Severity: Informational

Explanation: **ldapcp** received a SIGINT signal. **ldapcp** is ending.

System Action: The program ends.

Operator Response: None.

Administrator Response: None.

GLD6038I SIGQUIT received; program terminated.

Severity: Informational

Explanation: **ldapcp** received a SIGQUIT signal. **ldapcp** is ending.

System Action: The program ends.

Operator Response: None.

Administrator Response: None.

GLD6039E Program unable to establish signal handler. Program terminated.

Severity: Error

Explanation: **ldapcp** is unable to complete signal handling setup. **ldapcp** is ending.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6040I Program caught signal for which handler was not established.

Severity: Informational

Explanation: **ldapcp** received a signal. **ldapcp** is ending.

System Action: The program ends.

Operator Response: None.

Administrator Response: None.

GLD6041I More than one occurrence of -p detected. First one will be used.

Severity: Attention

Explanation: More than one port number was supplied to **ldapcp** to use to communicate with the server. **ldapcp** will use the port number specified first.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6042E Memory allocation failed; cannot continue. Program terminated.

Severity: Error

Explanation: **ldapcp** was unable to obtain the storage needed for processing.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure that the userID where **ldapcp** is being used has adequate storage to run **ldapcp** and try again. If the problem persists, contact your service representative.

GLD6043I Codepage conversion error occurred with rc=*return_code***.**

Severity: Attention

Explanation: The program was unable to successfully complete conversion of messages from the stored codepage to the local codepage.

System Action: The program continues.

Operator Response: None.

Administrator Response: Contact your service representative, and provide the return code displayed in the message.

GLD6044E No group argument was input for this command.

Severity: Error

Explanation: The command specified requires a group argument, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command arguments and try again.

GLD6046E No count argument was input for this command.

Severity: Error

Explanation: The command specified requires a count argument, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command arguments and try again.

GLD6048E No member argument was input for this command.

Severity: Error

Explanation: The command specified requires a member argument, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command arguments and try again.

GLD6050E No cn argument was input for this command.

Severity: Error

Explanation: The command specified requires a common name argument, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command arguments and try again.

GLD6052E No suffix argument was input for this command.

Severity: Error

Explanation: The command specified requires a suffix argument, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command arguments and try again.

GLD6054I Unable to perform codepage conversions from codepage *code_page* **to codepage** *code_page***.**

Severity: Attention

Explanation: The conversion between the stored catalog codeset and the local codeset is not supported.

System Action: The program ends.

Operator Response: None

Administrator Response: Ensure that support for both codepages in the message is installed on the system.

GLD6055I No debug level match for environment variable LDAP_DEBUG: *value***. This debug level ignored.**

Severity: Attention

Explanation: **ldapcp** is unable to interpret the debug level specified in the environment variable LDAP_DEBUG. The debug level is ignored.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the debug level in the environment variable and try again.

GLD6056I Selected debug level includes value in environment variable LDAP_DEBUG.

Severity: Informational

Explanation: In addition to any debug level specified on the command line using the -D flag, the program is also using any debug directives requested via the LDAP_DEBUG environment variable.

System Action: The program continues.

Operator Response: None

Administrator Response: None

GLD6057E Verb must be one of add, create, delete, or list.

Severity: Error

Explanation: None of the verbs supplied to **ldapcp** for the group were correct.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the verb and try again.

GLD6059E Subverb must be one of *, group, or member.

Severity: Error

Explanation: The group delete command requires a subverb, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command and try again.

GLD6060E Subverb must be one of *, group, or member.

Severity: Error

Explanation: The group list command requires a subverb, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command and try again.

GLD6061I Member maximum of 256 exceeded. Extra input for this command is ignored.

Severity: Attention

Explanation: A maximum of 256 members can be added to a group per invocation of this command.

System Action: The program continues.

Operator Response: None.

Administrator Response: Invoke the 'group add' command to add the additional members.

GLD6062E Subverb must be one of *, ACL, or owner.

Severity: Error

Explanation: The acl modify command requires a subverb, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command and try again.

GLD6063E Subverb must be one of object, or owner.

Severity: Error

Explanation: The acl query command requires a subverb, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command and try again.

GLD6064E Subverb must be one of owner.

Severity: Error

Explanation: The acl remove command requires the owner subverb, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command and try again.

GLD6065I LDAP client error.

Severity: Attention

Explanation: The client request to the server failed. This is often due to a connection failure, or a bind failure, or an unbind failure.

System Action: The program continues.

Operator Response: Ensure that the DN provided to the program is authorized to bind to the target server.

Administrator Response: Ensure that a server is running on the local host.

GLD6066E A memory allocation error occurred.

Severity: Error

Explanation: **ldapcp** was unable to obtain the storage needed for processing.

System Action: The program ends.

Operator Response: None.

Administrator Response: Ensure that the userID where **ldapcp** is being used has adequate storage to run **ldapcp** and try again. If the problem persists, contact your service representative.

GLD6067E Administration API error. Unknown error.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6068E The LDAP Server is not running, cannot perform function.

Severity: Error

Explanation: **ldapcp** was unable to communicate with the LDAP Server.

System Action: The program ends.

Operator Response: Start the LDAP Server.

Administrator Response: Contact the operator to have the LDAP Server restarted.

GLD6069E No ownerPropagate flag argument was input for this command.

Severity: Error

Explanation: The command specified requires an `ownerPropagate` flag argument, but none was specified.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command arguments and try again.

GLD6071E ownerPropagate flag must be one of TRUE or FALSE.

Severity: Error

Explanation: The `ownerPropagate` flag supplied on the command was not correct.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the command arguments and try again.

GLD6073E All arguments for this command must be within double quotation marks.

Severity: Error

Explanation: Arguments were input to this command that were not within double quotation marks.

System Action: The program ends.

Operator Response: None.

Administrator Response: Invoke the command again with the arguments within double quotation marks.

GLD6074E Administration API error. Parameter is not valid.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6075E Administration API error. DN parameter is not valid.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6076E Administration API error. PW parameter is not valid.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6077E LDAP client error. Bind failed.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6078E Administration API error. Handle to free not valid.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6079E LDAP client error. Object not valid.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6080E LDAP client error. Permission denied.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6081E LDAP client error. DN syntax error.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6082E LDAP client error. The LDAP Server is down.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6083E LDAP client error. Already exists.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6084E LDAP client error. Object already exists.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6085E LDAP client error. Time limit exceeded.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6086E LDAP client error. Size limit exceeded.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6087E Administration API error. System error.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6088E Administration API error. The configuration file was not found.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6089E Administration API error. Permission denied to access configuration file.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6090E Administration API error. Busy.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6091E Administration API error. A value is missing.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6092E Administration API error. A file name is missing.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6093E Administration API error. Not found.

Severity: Error

Explanation: An internal error is detected by **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: Contact your service representative.

GLD6094E No explicit owner found for this distinguished name.

Severity: Error

Explanation: An operation was requested for an owner attribute, and no explicit owner attribute exists for this distinguished name.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the distinguished name and try again.

GLD6134E Too many attempts at entering password. Program terminated.

Severity: Error

Explanation: No valid password was provided to **ldapcp** on 3 consecutive attempts. This error will occur if a non-white-space password has been used. This error could occur if there is no controlling terminal, or if **stdin** refers to a TTY device.

System Action: The program terminates.

Operator Response: Restart the program and provide a valid password.

Administrator Response: None.

GLD6138I Subject DN must be preceded by 'access-id:' or 'group:'.

Severity: Informational

Explanation: The DN input as owner must be preceded by the string 'access-id:' or by the string 'group:'.

System Action: The program continues.

Operator Response: Correct the owner DN and resubmit the request.

Administrator Response: None.

GLD6139I AclEntry string not formed correctly.

Severity: Informational

Explanation: The aclEntry input as part of the ACL to create is not formed correctly. Either required components are missing or values are not in valid range, or components are not properly delimited by colons.

System Action: The program continues.

Operator Response: Correct the aclEntry string and resubmit the request.

Administrator Response: None.

GLD6141I More than one occurrence of -h detected. First one will be used.

Severity: Attention

Explanation: More than one host address was supplied to **ldapcp** for the server. **ldapcp** will use the host address specified first.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6142I Reconnect count not numeric, defaulting to zero.

Severity: Attention

Explanation: The count of attempts to reconnect to the host was not numeric. The reconnect count has defaulted to zero. No attempts will be made to reconnect to the server if it stops responding during **ldapcp** operations.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6143E SSL was requested, but no key file name was supplied.

Severity: Error

Explanation: SSL connection was requested with the -Z parameter. Use of SSL requires that the -K parameter naming the SSL key file be supplied.

System Action: The program ends.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6144E LDAPCP was unable to connect to the server at host: *host_name***, port:** *port_number***.**

Severity: Error

Explanation: **ldapcp** was unable to connect to the requested server. Either the **ldap_open()** or **ldap_bind()** call failed.

System Action: The program ends.

Operator Response: Ensure the requested LDAP Server is running.

Administrator Response: Ensure the parameters supplied to **ldapcp** are correct and try again.

GLD6145I More than one occurrence of -K detected. First one will be used.

Severity: Attention

Explanation: More than one SSL key file name was supplied to **ldapcp**. **ldapcp** will use the SSL key file name specified first.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6146I More than one occurrence of -P detected. First one will be used.

Severity: Attention

Explanation: More than one SSL key file password was supplied to **ldapcp**. **ldapcp** will use the SSL key file password specified first.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6147I More than one occurrence of -N detected. First one will be used.

Severity: Attention

Explanation: More than one SSL key file DN was supplied to **ldapcp**. **ldapcp** will use the SSL key file DN specified first.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6148I More than one occurrence of -c detected. First one will be used.

Severity: Attention

Explanation: More than one reconnect count was supplied to **ldapcp**. **ldapcp** will use the reconnect count specified first.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the parameters supplied to **ldapcp** and try again.

GLD6149I No information available for this entry.

Severity: Informational

Explanation: Information is not available for the specified entry. This is either because the information is not available in the database, or because **ldapcp** does not support retrieving the information from this database type. For example, ACL and group information for SDBM entries is only available through RACF and not through **ldapcp**.

System Action: The program ends.

Operator Response: None.

Administrator Response: None.

GLD6151E No explicit ACL found for this distinguished name.

Severity: Error

Explanation: An operation was requested for an ACL attribute, and no explicit ACL attribute exists for this distinguished name.

System Action: The program continues.

Operator Response: None.

Administrator Response: Correct the distinguished name and try again.

Part 4. Appendixes

Appendix A. Configuration Files

This appendix shows the following LDAP Server configuration files:

The slapd.conf File

```
#
# Licensed Materials - Property of IBM
# 5647-A01
# (C) Copyright IBM Corp. 1997, 1999
#
#
#/*******************************************************************
#* This file is shipped in code page IBM-1047 and must remain in     *
#* code page IBM-1047.                                               *
#*******************************************************************/
#
#/*******************************************************************
# * Filename slapd.conf
# *
# * This file is the sample LDAP Server configuration file for OS/390.
# *******************************************************************/

include  /usr/lpp/ldap/examples/sample_server/slapd.at.system
```

Figure 28 (Part 1 of 4). The slapd.conf File

```
include  /usr/lpp/ldap/examples/sample_server/slapd.oc.system
include  /usr/lpp/ldap/examples/sample_server/slapd.at.conf
include  /usr/lpp/ldap/examples/sample_server/slapd.oc.conf
port            1800
security        none

# Update these values to reflect requirements
# Remove the '#' to uncomment these options.
#
# maxthreads     nnn
# maxconnections nnn
# waitingthreads nnn

timelimit       3600
sizelimit       500

# The following adminDN option should be updated to contain a
# distinguished name within one of the suffixes defined below.
# NOTE: An entry must exist in the Directory for this distinguished
# name and it will be used when evaluating an LDAP bind operation
# for the AdminDN.
#
adminDN         "cn=LDAP Administrator, o=Your Company, c=US"

######################################################################
# rdbm database definitions
######################################################################

database        rdbm GLDBRDBM

# The following options must be filled in with appropriate values
# for your DB2 setup, prior to attempting to run with the DB2 backend.

#
# The following dataset should be built by issuing the following command:
#     oget '/usr/lpp/ldap/examples/sample_server/dsnaoini.db2ini' \
#                 'XXXXXX.DSNAOINI'
# where the backslash (\) signifies a line continuation and should not
# be typed.  The OGET command works from TSO, ISPF option 6, and by using
# PF6 when running the OE shell.  The XXXXXX in the command above
# should be replaced with the high level qualifier of your choice.
# Be sure to set the dsnaoini option below to the name of the
# created dataset.
#
dsnaoini        XXXXXX.DSNAOINI
```

Figure 28 (Part 2 of 4). The slapd.conf File

```
#
# The servername option value must match the name of the DATA SOURCE
# stanza in the dataset named by the dsnaoini option value.  This value
# is DB2 installation dependent.  It is assumed to be LOC1 for this example.
#
servername      LOC1

#
# The dbuserid option must specify the userid under which the DB2 tables
# that the LDAP server uses were created.  If ldif2db is used to prime the
# LDAP Directory, then the userid specified here should be the userid that
# ran the ldif2db command.  It is assumed to be ldapuser for this example.
#
dbuserid        ldapuser

#
# The databasename option value must match the database name
# that was created using SPUFI.  In this example, the ldapspfi.spufi
# file uses the name ldapdb.
#
databasename    ldapdb

#
# The tbspaceentry option value must match the large tablespace name
# that was created using SPUFI.  In this example, the ldapspfi.spufi
# file uses the name ldaplg.
#
tbspaceentry    ldaplg

#
# The tbspace4k option value must match the database name
# that was created using SPUFI.  In this example, the ldapspfi.spufi
# file uses the name ldap4k.
#
tbspace4k       ldap4k

#
# The tbspace32k option value must match the database name
# that was created using SPUFI.  In this example, the ldapspfi.spufi
# file uses the name ldap32k.
#
tbspace32k      ldap32k
```

Figure 28 (Part 3 of 4). The slapd.conf File

```
| #
| # The tbspacemutex option value must match the database name
| # that was created using SPUFI.  In this example, the ldapspfi.spufi
| # file uses the name ldapls.
| #
| tbspacemutex    ldapls


| #
| # This portion of the LDAP Directory hierarchy is reserved for the use
| # of the LDAP server.  It should appear in the configuration file and
| # not be modified.
| #
| suffix          "cn=localhost"

| #
| # Note that all LDAP entries stored by the LDAP server must appear below
| # this distinguished name.  The sample.ldif file for this example has
| # all names ending with "o=Your Company, c=US".
| #
| suffix          "o=Your Company, c=US"

| index           cn              eq
| index           ou              eq
| index           sn              eq
| index           telephoneNumber eq
| index           title           eq
| readOnly        off
```

Figure 28 (Part 4 of 4). The slapd.conf File

The slapd.at.system File

```
| #/*****************************************************************
| #* This file is shipped in code page IBM-1047 and must remain in     *
| #* code page IBM-1047.                                               *
| #*****************************************************************/
| #/*************************************************************
| # *
| # * Licensed Materials - Property of IBM
| # * 5647-A01
| # * (C) Copyright IBM Corp. 1998, 1999
| # *
| # *************************************************************/
| # Filename: slapd.at.system
| #
| # This is the LDAP Server Private System Attribute Type Definition file
| #   for OS/390
| #
| # WARNING:  Do not alter the attribute type definitions in this file.
| #

| # For referral objects:
| attribute ref        ces  ref      100  normal


| # New attributes defined for replication

| attribute replicaHost         cis     replicaHost        100  normal
| attribute replicaBindDN       dn      replicaBindDN     1000  critical
| attribute replicaCredentials  bin     replicaCred        128  critical
| attribute replicaPort         cis     replicaPort         10  normal
| attribute replicaBindMethod   cis     replicaBindMethod  100  normal
| attribute replicaUseSSL       cis     replicaUseSSL       10  normal
| attribute replicaUpdateTimeInterval  cis replicaUpdateInt 20  normal


| # System (operational) attributes

| attribute modifiersName       dn      modifiersName     1000  system
| attribute modifyTimestamp     cis     modifyTimestamp     20  system
| attribute creatorsName        dn      creatorsName      1000  system
| attribute createTimestamp     cis     createTimestamp     20  system
| # The following two attribute types are marked OBSOLETE by the LDAP v3 drafts
| attribute lastModifiedBy      dn      lastModifiedBy    1000  system
| attribute lastModifiedTime    cis     lastModifiedTime    20  system
```

| *Figure 29 (Part 1 of 2). The slapd.at.system File*

```
| # Restricted attributes

| attribute aclEntry           cis      aclEntry         32700  restricted
| attribute aclPropagate       cis      aclPropagate         5  restricted
| attribute aclSource          dn       aclSource         1000  restricted
| attribute entryOwner         dn       entryOwner        1000  restricted
| attribute ownerPropagate     cis      ownerPropagate       5  restricted
| attribute ownerSource        dn       ownerSource       1000  restricted


| #
| # WARNING:  Do not alter the attribute type definitions in this file.
```

Figure 29 (Part 2 of 2). The slapd.at.system File

The slapd.oc.system File

```
#/*******************************************************************
#* This file is shipped in code page IBM-1047 and must remain in     *
#* code page IBM-1047.                                               *
#*******************************************************************/
#
#/*************************************************************
# *
# * Licensed Materials - Property of IBM
# * 5647-A01
# * (C) Copyright IBM Corp. 1998, 1999
# *
# *************************************************************/
# Filename: slapd.oc.system
#
# This is the LDAP Server Private System Object Class Definition file
#   for OS/390
#
# WARNING:  Do not alter the object class definitions in this file.
#
# NOTE: The LDAP Server depends upon the definitions of object classes
#       contained in this file for correct operation.  Do not remove these
#       object classes from the configuration of the LDAP Server.
#
objectclass replicaObject
   requires
      objectClass,
      cn,
      replicaBindDN,
      replicaCredentials,
      replicaHost
   allows
      description,
      seeAlso,
      replicaPort,
      replicaBindMethod,
      replicaUseSSL,
      replicaUpdateTimeInterval

objectclass referral
   requires
      objectClass,
      ref

objectclass accessGroup
   requires
```

Figure 30 (Part 1 of 2). The slapd.oc.system File

```
      objectClass,
      member,
      cn
   allows
      businessCategory,
      description,
      o,
      ou,
      owner,
      seeAlso

objectclass container
   requires
      cn

#
# WARNING:  Do not alter the object class definitions in this file.
```

Figure 30 (Part 2 of 2). The slapd.oc.system File

The slapd.at.conf File

```
| #/*****************************************************************
| #* This file is shipped in code page IBM-1047 and must remain in     *
| #* code page IBM-1047.                                               *
| #*****************************************************************/
| #/*************************************************************
| # *
| # * Licensed Materials - Property of IBM
| # * 5647-A01
| # * (C) Copyright IBM Corp.  1997, 1999
| # *
| # *************************************************************/
| #/*************************************************************
| #
| # Filename slapd.at.conf
| # This file must contain definitions for any attribute types that are
| # referenced by subsequent object class definitions.
| #

| # NOTE: The LDAP Server depends upon the definitions of the objectClass
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute objectClass         cis   objectclass     128   normal

| # NOTE: The LDAP Server depends upon the definitions of the commonName
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute cn   commonName     cis   cn              128   normal

| # NOTE: The LDAP Server depends upon the definitions of the surName
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute sn   surName        cis   sn              128   normal

| attribute serialNumber        cis   serialNumber     64   normal
| attribute c   countryName     cis   c               128   normal
| attribute l   localityName    cis   l               128   normal
| attribute st   stateOrProvince cis  st              128   normal
| attribute street streetAddress cis  street          128   normal

| # NOTE: The LDAP Server depends upon the definitions of the organizationName
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute o   organizationName cis  o               128   normal

| # NOTE: The LDAP Server depends upon the definitions of the organizationalUnit
```

| *Figure 31 (Part 1 of 4). The slapd.at.conf File*

```
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute ou   organizationalUnit cis   ou          128   normal

| attribute title               cis  title           128   normal

| # NOTE: The LDAP Server depends upon the definitions of the description
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute description          cis  description    1024   normal

| attribute searchGuide          ces  searchGuide    5000   normal
| attribute enhancedSearchGuide  ces  enhancedGuide  5000   normal

| # NOTE: The LDAP Server depends upon the definitions of the businessCategory
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute businessCategory     cis  businessCategory 128  normal

| attribute postalAddress        cis  postalAddress   500   normal
| attribute postalCode           cis  postalCode       40   normal
| attribute postOfficeBox        cis  postOfficeBox    40   normal
| attribute physicalDeliveryOfficeName  cis  physicalDelivery  128   normal

| # NOTE: The LDAP Server depends upon the definitions of the telephoneNumber
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute telephoneNumber      tel  telephoneNumber  32   normal

| attribute telexNumber          cis  telexNumber      28   normal
| attribute teletexTerminalIdentifier cis  teletexTerminalId 1000   normal
| attribute facsimileTelephoneNumber fax tel fax       32   normal
| attribute x121Address         ces   x121Address      15   normal
| attribute internationaliSDNNumber ces iSDNNumber     16   normal
| attribute registeredAddress   cis   registeredAddress 500 normal
| attribute destinationIndicator cis  destIndicator   128   normal
| attribute preferredDeliveryMethod cis  prefDeliveryMeth  1000   normal
| attribute presentationAddress     ces  presentationAddr  1000   normal
| attribute supportedApplicationContext cis supportAppContext 1000 normal

| # NOTE: The LDAP Server depends upon the definitions of the member
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute member              dn   member        1000   normal
```

| *Figure 31 (Part 2 of 4). The slapd.at.conf File*

```
| # NOTE: The LDAP Server depends upon the definitions of the owner
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute owner               dn   owner         1000   normal

| attribute roleOccupant        dn   roleOccupant  1000   normal

| # NOTE: The LDAP Server depends upon the definitions of the seeAlso
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute seeAlso             dn   seeAlso       1000   normal

| # NOTE: The LDAP Server depends upon the definitions of the userPassword
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute userPassword        bin  userPassword         128   critical

| attribute userCertificate     bin  userCertificate   250000   critical
| attribute cACertificate       bin  cACertificate     250000   critical
| attribute authorityRevocationList bin  authRevocationLst 250000 critical
| attribute certificateRevocationList bin  certRevocationLst 250000 critical
| attribute deltaRevocationList bin   deltRevocationLst 250000 critical
| attribute crossCertificatePair bin  crossCertPair     250000 critical
| attribute name                cis   name           32768   normal
| attribute givenName           cis   givenName        128   normal
| attribute initials            cis   initials          20   normal
| attribute generationQualifier  cis   generationQualif  10   normal
| attribute x500UniqueIdentifier bin   x500UniqueId     128   normal
| attribute dnQualifier         cis   dnQualifier      128   normal
| attribute protocolInformation  bin   protocolInfo    5000   normal
| attribute dn   distinguishedName dn  dn              1000   normal
| attribute uniqueMember        dn    uniqueMember    1000   normal
| attribute houseIdentifier     cis   houseIdentifier 32768  normal
| attribute uid                 cis   uid              256   normal
| attribute textEncodedOrAddress cis   textEncodedOrAddr 256  normal
| attribute mail                cis   mail             256   normal
| attribute info                cis   info            2048   normal
| attribute drink   favouritedrink cis drink            256   normal
| attribute roomNumber          cis   roomNumber       256   normal
| attribute photo               bin   photo         250000   normal
| attribute userClass           cis   userClass        256   normal
| attribute host                cis   host             256   normal
| attribute manager             dn    manager         1000   normal
| attribute documentIdentifier  cis   documentIdent    256   normal
| attribute documentTitle       cis   documentTitle    256   normal
```

| *Figure 31 (Part 3 of 4). The slapd.at.conf File*

```
attribute documentVersion      cis   documentVersion  256   normal
attribute documentLocation     cis   documentLocation 256   normal
attribute documentAuthor       dn    documentAuthor  1000   normal
attribute homePhone            tel   homePhone         32   sensitive
attribute secretary            dn    secretary       1000   normal
attribute otherMailbox         cis   otherMailbox      40   normal
attribute dc   domainComponent cis   dc                64   normal
attribute dnsRecord            cis   dnsRecord        128   normal
attribute associatedDomain     cis   associatedDomain 128   normal
attribute associatedName       dn    associatedName  1000   normal
attribute homePostalAddress    cis   homePostalAddress 500  sensitive
attribute personalTitle        cis   personalTitle     50   normal
attribute mobile mobileTelephoneNumber tel mobile      32   normal
attribute pager pagerTelephoneNumber   tel pager       32   normal
attribute co    friendlyCountryName    cis   co       128   normal
attribute uniqueIdentifier     cis   uniqueIdentifier 128   normal
attribute organizationalStatus cis   orgStatus        256   normal
attribute janetMailbox         cis   janetMailbox     256   normal
attribute mailPreferenceOption cis   mailPrefOption    40   normal
attribute buildingName         cis   buildingName     256   normal
attribute personalSignature    bin   personalSignature 50000 normal
attribute dITRedirect          dn    dITRedirect     1000   normal
attribute audio                bin   audio         250000   normal
attribute documentPublisher    cis   documentPublisher 256  normal
attribute jpegPhoto            bin   jpegPhoto     250000   normal
attribute url                  ces   url              100   normal
# knowledgeInformation is flagged as "no longer used" in IETF-ASID v3 drafts
attribute knowledgeInformation cis   knowledgeInfo  32768   normal
attribute abstract             cis   abstract         500   normal
attribute documentAuthorCommonName cis  docAuthorCN   128   normal
attribute documentAuthorSurName    cis  docAuthorSN   128   normal
attribute documentStore        cis   documentStore    128   normal
attribute keywords             cis   keywords         256   normal
attribute obsoletedByDocument  cis   obsoletedByDoc   256   normal
attribute obsoletesDocument    cis   obsoletesDoc     256   normal
attribute subject              cis   subject          100   normal
attribute updatedByDocument    cis   updatedByDocument 256   normal
attribute updatesDocument      cis   updatesDocument   256   normal
attribute labeledURI           ces   labeledURI        100   normal
attribute dSAQuality           ces   dSAQuality       5000   normal
attribute singleLevelQuality   ces   singleLevelQual  5000   normal
attribute subtreeMinimumQuality ces  subtreeMinQuality 5000  normal
attribute subtreeMaximumQuality ces  subtreeMaxQuality 5000  normal
```

Figure 31 (Part 4 of 4). The slapd.at.conf File

The slapd.oc.conf File

```
#/*******************************************************************
#* This file is shipped in code page IBM-1047 and must remain in     *
#* code page IBM-1047.                                               *
#*******************************************************************/
#/*************************************************************
# *
# * Licensed Materials - Property of IBM
# * 5647-A01
# * (C) Copyright IBM Corp.  1997, 1999
# *
# *************************************************************/
#
# Filename: slapd.oc.conf
# This file contains object class definitions.
#

# NOTE: The LDAP Server depends upon the definitions of the top
#       object class.  Do not remove this object class from the
#       configuration of the LDAP Server.
objectclass top
   requires
      objectClass

objectclass country
   requires
      objectClass,
      c
   allows
      searchGuide,
      description

objectclass locality
   requires
      objectClass
   allows
      description,
      l,
      searchGuide,
      seeAlso,
      st,
      street

objectclass organization
   requires
      objectClass,
```

Figure 32 (Part 1 of 12). The slapd.oc.conf File

```
      o
   allows
      businessCategory,
      description,
      destinationIndicator,
      facsimileTelephoneNumber,
      internationaliSDNNumber,
      l,
      physicalDeliveryOfficeName,
      postOfficeBox,
      postalAddress,
      postalCode,
      preferredDeliveryMethod,
      registeredAddress,
      searchGuide,
      seeAlso,
      st,
      street,
      telephoneNumber,
      teletexTerminalIdentifier,
      telexNumber,
      userPassword,
      x121Address

objectclass organizationalUnit
   requires
      objectClass,
      ou
   allows
      businessCategory,
      description,
      destinationIndicator,
      facsimileTelephoneNumber,
      internationaliSDNNumber,
      l,
      physicalDeliveryOfficeName,
      postOfficeBox,
      postalAddress,
      postalCode,
      preferredDeliveryMethod,
      registeredAddress,
      searchGuide,
      seeAlso,
      st,
      street,
```

Figure 32 (Part 2 of 12). The slapd.oc.conf File

```
      telephoneNumber,
      teletexTerminalIdentifier,
      telexNumber,
      userPassword,
      x121Address

# NOTE: The LDAP Server depends upon the definitions of the person
#       object class.  Do not remove this object class from the
#       configuration of the LDAP Server.
objectclass person
   requires
      objectClass,
      sn,
      cn
   allows
      description,
      seeAlso,
      telephoneNumber,
      userPassword

objectclass organizationalPerson
   requires
      objectClass,
      sn,
      cn
   allows
      description,
      destinationIndicator,
      facsimileTelephoneNumber,
      internationaliSDNNumber,
      l,
      ou,
      physicalDeliveryOfficeName,
      postOfficeBox,
      postalAddress,
      postalCode,
      preferredDeliveryMethod,
      registeredAddress,
      seeAlso,
      st,
      street,
      telephoneNumber,
      teletexTerminalIdentifier,
      telexNumber,
      title,
```

Figure 32 (Part 3 of 12). The slapd.oc.conf File

```
      userPassword,
      x121Address

objectclass organizationalRole
   requires
      objectClass,
      cn
   allows
      description,
      destinationIndicator,
      facsimileTelephoneNumber,
      internationaliSDNNumber,
      l,
      ou,
      physicalDeliveryOfficeName,
      postOfficeBox,
      postalAddress,
      postalCode,
      preferredDeliveryMethod,
      registeredAddress,
      roleOccupant,
      seeAlso,
      st,
      street,
      telephoneNumber,
      teletexTerminalIdentifier,
      telexNumber,
      x121Address

objectclass groupOfNames
   requires
      objectClass,
      member,
      cn
   allows
      businessCategory,
      description,
      o,
      ou,
      owner,
      seeAlso

objectclass groupOfUniqueNames
   requires
      objectClass,
```

Figure 32 (Part 4 of 12). The slapd.oc.conf File

```
      uniqueMember,
      cn
   allows
      businessCategory,
      description,
      o,
      ou,
      owner,
      seeAlso

objectclass residentialPerson
   requires
      objectClass,
      sn,
      cn
   allows
      businessCategory,
      description,
      destinationIndicator,
      facsimileTelephoneNumber,
      internationaliSDNNumber,
      l,
      physicalDeliveryOfficeName,
      postOfficeBox,
      postalAddress,
      postalCode,
      preferredDeliveryMethod,
      registeredAddress,
      seeAlso,
      st,
      street,
      telephoneNumber,
      teletexTerminalIdentifier,
      telexNumber,
      userPassword,
      x121Address

objectclass applicationProcess
   requires
      objectClass,
      cn
   allows
      description,
      l,
      ou,
```

Figure 32 (Part 5 of 12). The slapd.oc.conf File

```
      seeAlso

objectclass applicationEntity
   requires
      objectClass,
      presentationAddress,
      cn
   allows
      description,
      l,
      o,
      ou,
      seeAlso,
      supportedApplicationContext

objectclass dSA
   requires
      objectClass,
      presentationAddress,
      cn
   allows
      knowledgeInformation

objectclass device
   requires
      objectClass,
      cn
   allows
      description,
      l,
      o,
      ou,
      owner,
      seeAlso,
      serialNumber

objectclass cRLDistributionPoint
   requires
      objectClass,
      cn
   allows
      certificateRevocationList,
      authorityRevocationList,
      deltaRevocationList
```

Figure 32 (Part 6 of 12). The slapd.oc.conf File

```
objectclass pilotObject
   requires
      objectClass
   allows
      audio,
      dITRedirect,
      info,
      jpegPhoto,
      lastModifiedBy,
      lastModifiedTime,
      manager,
      photo,
      uniqueIdentifier

objectclass newPilotPerson
   requires
      objectClass,
      sn,
      cn
   allows
      businessCategory,
      description,
      drink,
      homePhone,
      homePostalAddress,
      janetMailbox,
      mail,
      mailPreferenceOption,
      mobile,
      organizationalStatus,
      otherMailbox,
      pager,
      personalSignature,
      personalTitle,
      preferredDeliveryMethod,
      roomNumber,
      secretary,
      seeAlso,
      telephoneNumber,
      textEncodedOrAddress,
      uid,
      userClass,
      userPassword

objectclass account
```

Figure 32 (Part 7 of 12). The slapd.oc.conf File

```
   requires
      objectClass,
      uid
   allows
      description,
      host,
      l,
      o,
      ou,
      seeAlso

objectclass document
   requires
      objectClass,
      documentIdentifier
   allows
      abstract,
      audio,
      documentAuthorCommonName,
      documentAuthorSurName,
      cn,
      dITRedirect,
      description,
      documentAuthor,
      documentLocation,
      documentPublisher,
      documentStore,
      documentTitle,
      documentVersion,
      info,
      jpegPhoto,
      keywords,
      l,
      lastModifiedBy,
      lastModifiedTime,
      manager,
      o,
      obsoletedByDocument,
      obsoletesDocument,
      ou,
      photo,
      seeAlso,
      subject,
      uniqueIdentifier,
      updatedByDocument,
```

Figure 32 (Part 8 of 12). The slapd.oc.conf File

```
      updatesDocument

objectclass room
   requires
      objectClass,
      cn
   allows
      description,
      roomNumber,
      seeAlso,
      telephoneNumber

objectclass documentSeries
   requires
      objectClass,
      cn
   allows
      description,
      l,
      o,
      ou,
      seeAlso,
      telephoneNumber

objectclass domain
   requires
      objectClass,
      dc
   allows
      associatedName,
      businessCategory,
      description,
      destinationIndicator,
      facsimileTelephoneNumber,
      internationaliSDNNumber,
      l,
      o,
      physicalDeliveryOfficeName,
      postOfficeBox,
      postalAddress,
      postalCode,
      preferredDeliveryMethod,
      registeredAddress,
      searchGuide,
      seeAlso,
```

Figure 32 (Part 9 of 12). The slapd.oc.conf File

```
      st,
      street,
      telephoneNumber,
      teletexTerminalIdentifier,
      telexNumber,
      userPassword,
      x121Address

objectclass rFC822localPart
   requires
      objectClass,
      dc
   allows
      associatedName,
      businessCategory,
      cn,
      description,
      destinationIndicator,
      facsimileTelephoneNumber,
      internationaliSDNNumber,
      l,
      o,
      physicalDeliveryOfficeName,
      postOfficeBox,
      postalAddress,
      postalCode,
      preferredDeliveryMethod,
      registeredAddress,
      searchGuide,
      seeAlso,
      sn,
      st,
      street,
      telephoneNumber,
      teletexTerminalIdentifier,
      telexNumber,
      userPassword,
      x121Address

objectclass dNSDomain
   requires
      objectClass,
      dc
   allows
      associatedName,
```

Figure 32 (Part 10 of 12). The slapd.oc.conf File

```
      businessCategory,
      dnsRecord,
      description,
      destinationIndicator,
      facsimileTelephoneNumber,
      internationaliSDNNumber,
      l,
      o,
      physicalDeliveryOfficeName,
      postOfficeBox,
      postalAddress,
      postalCode,
      preferredDeliveryMethod,
      registeredAddress,
      searchGuide,
      seeAlso,
      st,
      street,
      telephoneNumber,
      teletexTerminalIdentifier,
      telexNumber,
      userPassword,
      x121Address

objectclass domainRelatedObject
   requires
      objectClass,
      associatedDomain

objectclass friendlyCountry
   requires
      objectClass,
      c,
      co
   allows
      description,
      searchGuide

objectclass pilotOrganization
   requires
      objectClass,
      ou,
      o
   allows
      buildingName,
```

Figure 32 (Part 11 of 12). The slapd.oc.conf File

```
      businessCategory,
      description,
      destinationIndicator,
      facsimileTelephoneNumber,
      internationaliSDNNumber,
      l,
      physicalDeliveryOfficeName,
      postOfficeBox,
      postalAddress,
      postalCode,
      preferredDeliveryMethod,
      registeredAddress,
      searchGuide,
      seeAlso,
      st,
      street,
      telephoneNumber,
      teletexTerminalIdentifier,
      telexNumber,
      userPassword,
      x121Address

objectclass labeledURIObject
   requires
      objectClass
   allows
      labeledURI

objectclass oldQualityLabelledData
   requires
      objectClass,
      singleLevelQuality
   allows
      subtreeMaximumQuality,
      subtreeMinimumQuality

objectclass qualityLabelledData
   requires
      objectClass,
      singleLevelQuality
   allows
      subtreeMaximumQuality,
      subtreeMinimumQuality

objectclass pilotDSA
   requires
      objectClass,
      dSAQuality
```

Figure 32 (Part 12 of 12). The slapd.oc.conf File

The slapd.at.racf File

```
#/*****************************************************************
#* This file is shipped in code page IBM-1047 and must remain in     *
#* code page IBM-1047.                                               *
#*****************************************************************/
#/*************************************************************
# *
# * Licensed Materials - Property of IBM
# * 5647-A01
# * (C) Copyright IBM Corp.  1999
# *
# *************************************************************/
# Filename: slapd.at.racf
#
# This is the LDAP Server RACF Attribute Type Definition file
#   for OS/390
#
# WARNING:  Do not alter the attribute type definitions in this file.
#

# Note that attributes 'o' and 'c',defined in the slapd.at.conf file,
# are required by the sdbm(RACF) backend.

# The tablename of '_nocreate' indicates that if rdbm and sdbm backends are
# supported by the same server, no rdbm attribute tables will be created
# for the sdbm attributes.

# The LDAP Server Administration and Usage Guide contains a mapping between
# these attributes and the RACF command clauses for modifying and adding
# RACF information.
# Refer to the RACF Command Reference for information regarding value
# specification including valid values and formats. Some information
# regarding display and entry formats for date fields is
# provided in this file as comments for ease of use.

# For racf backend top objects:
attribute sysplex        cis   _nocreate    8   sensitive

# Valid values for profileType are USER and GROUP
attribute profileType    cis   _nocreate    5   sensitive


# racfid can be either a RACF userid or a RACF Group id
attribute racfid         cis   _nocreate    8   sensitive
```

Figure 33 (Part 1 of 5). The slapd.at.racf File

```
# These definitions are for the racfBaseCommon objectclass

attribute racfAuthorizationDate  cis   _nocreate    6   sensitive
# racfAuthorizationDate is displayed in yy.ddd format.
attribute racfOwner              cis   _nocreate    8   sensitive
attribute racfInstallationData   cis   _nocreate  255   sensitive
attribute racfDatasetModel       cis   _nocreate   44   sensitive

# These definitions are for the racfGroup objectclass

attribute racfSuperiorGroup      cis   _nocreate    8   sensitive
attribute racfGroupNoTermUAC     cis   _nocreate    9   sensitive
# Valid values for racfGroupNoTermUAC are TERMUAC and NOTERMUAC
attribute racfSubGroupName       cis   _nocreate    8   sensitive
attribute racfGroupUserids       cis   _nocreate    8   sensitive
attribute racfGroupUserAccess    cis   _nocreate    7   sensitive

# These definitions are for the racfUser objectclass

attribute racfAttributes         cis   _nocreate   12   sensitive
attribute racfPassword           cis   _nocreate    8   critical
attribute racfPasswordInterval   cis   _nocreate    3   sensitive
# racfPasswordInterval is displayed as ddd.
attribute racfPasswordChangeDate cis   _nocreate    6   critical
# racfPasswordChangeDate is displayed as yy.ddd.
attribute racfProgrammerName     cis   _nocreate   20   sensitive
attribute racfDefaultGroup       cis   _nocreate    8   sensitive
attribute racfLastAccess         cis   _nocreate   15   sensitive
# racfLastAccess is displayed as yy.ddd/hh:mm:ss
attribute racfSecurityLevel      cis   _nocreate   15   critical
attribute racfSecurityCategoryList cis _nocreate   15   sensitive
attribute racfRevokeDate         cis   _nocreate    8   sensitive
# racfRevokeDate is specified in yy/mm/dd format.
attribute racfResumeDate         cis   _nocreate    8   sensitive
# racfResumeDate is specified in yy/mm/dd format.
attribute racfLogonDays          cis   _nocreate   29   sensitive
attribute racfLogonTime          cis   _nocreate   13   sensitive
attribute racfClassName          cis   _nocreate    8   sensitive
attribute racfConnectGroupName   cis   _nocreate    8   sensitive
attribute racfConnectGroupAuthority cis _nocreate   8   sensitive
attribute racfConnectGroupUACC   cis  _nocreate     8   sensitive
attribute racfSecurityLabel      cis  _nocreate     8   sensitive

# Attribute definitions for SAF DFP objectclass
attribute SAFDfpDataApplication  cis  _nocreate     8   sensitive
```

Figure 33 (Part 2 of 5). The slapd.at.racf File

```
attribute SAFDfpDataClass        cis  _nocreate     8   sensitive
attribute SAFDfpManagementClass  cis  _nocreate     8   sensitive
attribute SAFDfpStorageClass     cis  _nocreate     8   sensitive

# Attribute definitions for GROUP OMVS objectclass
attribute racfOmvsGroupId        cis  _nocreate     4   sensitive

# Attribute definitions for GROUP OVM objectclass
attribute racfOvmGroupId         cis  _nocreate     4   sensitive


# attribute definitions for SAF TSO Segment
attribute SAFAccountNumber       cis  _nocreate    40   sensitive
attribute SAFDefaultCommand      cis  _nocreate    80   sensitive
attribute SAFDestination         cis  _nocreate     8   sensitive
attribute SAFHoldClass           cis  _nocreate     1   sensitive
attribute SAFJobClass            cis  _nocreate     1   sensitive
attribute SAFMessageClass        cis  _nocreate     1   sensitive
attribute SAFDefaultLoginProc    cis  _nocreate     8   sensitive
attribute SAFLogonSize           cis  _nocreate     4   sensitive
attribute SAFMaximumRegionSize   cis  _nocreate     4   sensitive
attribute SAFDefaultSysoutClass  cis  _nocreate     1   sensitive
attribute SAFUserdata            cis  _nocreate     2   sensitive
attribute SAFDefaultUnit         cis  _nocreate     8   sensitive
attribute SAFTsoSecurityLabel    cis  _nocreate     8   sensitive

# Attribute definitions for the LANGUAGE segment
attribute racfPrimaryLanguage    cis  _nocreate    15   sensitive
attribute racfSecondaryLanguage  cis  _nocreate    15   sensitive

# attribute definitions for RACF CICS Segment
attribute racfOperatorIdentification cis _nocreate  3   sensitive
attribute racfOperatorClass      cis _nocreate     62   sensitive
attribute racfOperatorPriority   cis _nocreate      3   sensitive
attribute racfOperatorReSignon   cis _nocreate      7   sensitive
attribute racfTerminalTimeout    cis _nocreate      4   sensitive

# Attribute definitions for RACF OperParm Segment
attribute racfStorageKeyword     cis _nocreate      4   sensitive
attribute racfAuthKeyword        cis _nocreate      6   sensitive
attribute racfMformKeyword       cis _nocreate      1   sensitive
attribute racfLevelKeyword       cis _nocreate      3   sensitive
attribute racfMonitorKeyword     cis _nocreate      9   sensitive
attribute racfRoutcodeKeyword    cis _nocreate      7   sensitive
attribute racfLogCommandResponseKeyword cis _nocreate 6 sensitive
```

Figure 33 (Part 3 of 5). The slapd.at.racf File

```
attribute racfMGIDKeyword        cis _nocreate      3   sensitive
attribute racfDOMKeyword         cis _nocreate      6   sensitive
attribute racfKEYKeyword         cis _nocreate      8   sensitive
attribute racfCMDSYSKeyword      cis _nocreate      8   sensitive
attribute racfUDKeyword          cis _nocreate      3   sensitive
attribute racfMscopeSystems      cis _nocreate      8   sensitive
attribute racfAltGroupKeyword    cis _nocreate      8   sensitive
attribute racfAutoKeyword        cis _nocreate      3   sensitive

# Attribute definitions for the RACF Work Attributes Segment
attribute racfWorkAttrUserName   cis _nocreate     60   sensitive
attribute racfBuilding           cis _nocreate     60   sensitive
attribute racfDepartment         cis _nocreate     60   sensitive
attribute racfRoom               cis _nocreate     60   sensitive
attribute racfAddressLine1       cis _nocreate     60   sensitive
attribute racfAddressLine2       cis _nocreate     60   sensitive
attribute racfAddressLine3       cis _nocreate     60   sensitive
attribute racfAddressLine4       cis _nocreate     60   sensitive
attribute racfWorkAttrAccountNumber cis _nocreate 255   sensitive

# attribute definitions for RACF User OMVS Segment
attribute racfOmvsUid            cis _nocreate     10   sensitive
attribute racfOmvsHome           cis _nocreate   1023   sensitive
attribute racfOmvsInitialProgram cis _nocreate   1023   sensitive

# attribute definitions for RACF NetView Segment
attribute racfNetviewInitialCommand cis _nocreate 255   sensitive
attribute racfDefaultConsoleName cis    _nocreate   8   sensitive
attribute racfCTLKeyword         cis    _nocreate   8   sensitive
attribute racfMSGRCVRKeyword     cis    _nocreate   3   sensitive
attribute racfNetviewOperatorClass cis  _nocreate   4   sensitive
attribute racfDomains            cis    _nocreate   5   sensitive
attribute racfNGMFADMKeyword     cis    _nocreate   3   sensitive

# Attribute definitions for RACF DCE Segment
attribute racfDCEUUID            cis   _nocreate   36   sensitive
attribute racfDCEPrincipal       cis   _nocreate 1023   sensitive
attribute racfDCEHomeCell        cis   _nocreate 1023   sensitive
attribute racfDCEHomeCellUUID    cis   _nocreate   36   sensitive
attribute racfDCEAutoLogin       cis   _nocreate    3   sensitive
# Valid values for racfDCEAutoLogin are YES and NO.

# Attribute definitions for RACF User Ovm Segment
attribute racfOvmUid             cis   _nocreate   10   sensitive
attribute racfOvmHome            cis   _nocreate 1023   sensitive
```

Figure 33 (Part 4 of 5). The slapd.at.racf File

```
attribute racfOvmInitialProgram  cis   _nocreate 1023   sensitive
attribute racfOvmFileSystemRoot  cis   _nocreate 1023   sensitive
attribute racfOvmHomeUUID        cis   _nocreate   36   sensitive

#
# WARNING:  Do not alter the attribute type definitions in this file.
```

Figure 33 (Part 5 of 5). The slapd.at.racf File

The slapd.oc.racf File

```
#/*****************************************************************
#* This file is shipped in code page IBM-1047 and must remain in     *
#* code page IBM-1047.                                               *
#*****************************************************************/
#
#/*************************************************************
# *
# * Licensed Materials - Property of IBM
# * 5647-A01
# * (C) Copyright IBM Corp.  1999
# *
# *************************************************************/
# Filename: slapd.oc.racf
#
# This is the LDAP Server Private System Object Class Definition file
#   for OS/390 for access to RACF data via LDAP
#
# WARNING:  Do not alter the object class definitions in this file.
#
# Definitions for RACF Namespace Tree Head

# Structural objectclass representing 1st level of RACF namespace; extends top
# An racfbase entry denotes the top of the directory tree that holds
# information accessible from RACF.
objectclass racfbase
   requires
      objectClass,
      sysplex

# Structural objectclass representing 2nd level of RACF namespace; extends top
# Entries with this object class appear below racfbase entries in the
# directory hierarchy.
objectclass racfProfileType
       requires
      objectClass,
      profileType

# Abstract objectclass representing common base segment attributes; extends top
# Entries with this object class appear below racfProfileType entries
# in the directory hierarchy.
objectclass racfBaseCommon
   requires
      objectClass
   allows
      racfAuthorizationDate,
```

Figure 34 (Part 1 of 5). The slapd.oc.racf File

```
      racfOwner,
      racfInstallationData,
      racfDatasetModel

# Structural objectclass representing a RACF User; extends racfBaseCommon
objectclass racfUser
   requires
      objectClass,
      racfid
   allows
      racfAttributes,
      racfPassword,
      racfPasswordInterval,
      racfPasswordChangeDate,
      racfProgrammerName,
      racfDefaultGroup,
      racfLastAccess,
      racfSecurityLevel,
      racfSecurityCategoryList,
      racfRevokeDate,
      racfResumeDate,
      racfLogonDays,
      racfLogonTime,
      racfClassName,
      racfConnectGroupName,
      racfConnectGroupAuthority,
      racfConnectGroupUACC,
      racfSecurityLabel

# Structural objectclass representing a RACF Group; extends racfBaseCommon
objectclass racfGroup
   requires
      objectClass,
      racfid
   allows
      racfSuperiorGroup,
      racfGroupNoTermUAC,
      racfSubGroupName,
      racfGroupUserids,
      racfGroupUserAccess

# Auxiliary objectclass for racfUser and racfGroup; extends top
objectclass SAFDfpSegment
   requires
      objectClass
```

Figure 34 (Part 2 of 5). The slapd.oc.racf File

```
   allows
      SAFDfpDataApplication,
      SAFDfpDataClass,
      SAFDfpManagementClass,
      SAFDfpStorageClass

# Auxiliary objectclass for racfGroup; extends top
objectclass racfGroupOmvsSegment
   requires
      objectClass
   allows
      racfOmvsGroupId

# Auxiliary objectclass for racfGroup; extends top
objectclass racfGroupOvmSegment
   requires
      objectClass
   allows
      racfOvmGroupId

# Auxiliary objectclass for racfUser; extends top
objectclass SAFTsoSegment
   requires
      objectClass
   allows
      SAFAccountNumber,
      SAFDefaultCommand,
      SAFDestination,
      SAFHoldClass,
      SAFJobClass,
      SAFMessageClass,
      SAFDefaultLoginProc,
      SAFLogonSize,
      SAFMaximumRegionSize,
      SAFDefaultSysoutClass,
      SAFUserdata,
      SAFDefaultUnit,
      SAFTsoSecurityLabel

# Auxiliary objectclass for racfUser; extends top
objectclass racfCicsSegment
   requires
      objectClass
   allows
      racfOperatorIdentification,
```

Figure 34 (Part 3 of 5). The slapd.oc.racf File

```
      racfOperatorClass,
      racfOperatorPriority,
      racfOperatorReSignon,
      racfTerminalTimeout

# Auxiliary objectclass for racfUser; extends top
objectclass racfLanguageSegment
   requires
      objectClass
   allows
      racfPrimaryLanguage,
      racfSecondaryLanguage

# Auxiliary objectclass for racfUser; extends top
objectclass racfOperparmSegment
   requires
      objectClass
   allows
      racfStorageKeyword,
      racfAuthKeyword,
      racfMformKeyword,
      racfLevelKeyword,
      racfMonitorKeyword,
      racfRoutcodeKeyword,
      racfLogCommandResponseKeyword,
      racfMGIDKeyword,
      racfDOMKeyword,
      racfKEYKeyword,
      racfCMDSYSKeyword,
      racfUDKeyword,
      racfMscopeSystems,
      racfAltGroupKeyword,
      racfAutoKeyword

# Auxiliary objectclass for racfUser; extends top
objectclass racfWorkAttrSegment
   requires
      objectClass
   allows
      racfWorkAttrUserName,
      racfBuilding,
      racfDepartment,
      racfRoom,
      racfAddressLine1,
      racfAddressLine2,
```

Figure 34 (Part 4 of 5). The slapd.oc.racf File

```
      racfAddressLine3,
      racfAddressLine4,
      racfWorkAttrAccountNumber

# Auxiliary objectclass for racfUser; extends top
objectclass racfUserOmvsSegment
   requires
      objectClass
   allows
      racfOmvsUid,
      racfOmvsHome,
      racfOmvsInitialProgram

# Auxiliary objectclass for racfUser; extends top
objectclass racfNetviewSegment
   requires
      objectClass
   allows
      racfNetviewInitialCommand,
      racfDefaultConsoleName,
      racfCTLKeyword,
      racfMSGRCVRKeyword,
      racfNetviewOperatorClass,
      racfDomains,
      racfNGMFADMKeyword

# Auxiliary objectclass for racfUser; extends top
objectclass racfDCESegment
   requires
      objectClass
   allows
      racfDCEUUID,
      racfDCEPrincipal,
      racfDCEHomeCell,
      racfDCEHomeCellUUID,
      racfDCEAutoLogin

# Auxiliary objectclass for racfUser; extends top
objectclass racfUserOvmSegment
   requires
      objectClass
   allows
      racfOvmUid,
      racfOvmHome,
      racfOvmInitialProgram,
      racfOvmFileSystemRoot,
      racfOvmHomeUUID

#
# WARNING:  Do not alter the object class definitions in this file.
```

Figure 34 (Part 5 of 5). The slapd.oc.racf File

The slapd.cb.at.conf File

```
# ------------------------------------------------------------------
# This file is shipped in code page IBM-1047 and must remain in
# code page IBM-1047.
# ------------------------------------------------------------------
#
# Licensed Materials - Property of IBM
# 5647-A01
# (C) Copyright IBM Corp. 1998, 1999
#
# ------------------------------------------------------------------
# Filename slapd.cbseries.at.conf
# This file contains definitions for attribute types that are used
# by the CBSeries component of OS/390.
# ------------------------------------------------------------------

# ------------------------------------------------------------------
# The following attribute types are used in managing the CBSeries
# Naming Service.
#
attribute TypelessRDN ces TypelessRDN 1024  normal
attribute seq#  sequenceNumber cis sequenceNumber   250   normal
attribute bt    bindingType cis  bindingType      15    normal
attribute oref  objectReference cis  objectReference  32700 normal
# ------------------------------------------------------------------

# ------------------------------------------------------------------
# The following attribute types are used in managing the CBSeries
# Interface Repository.
#
attribute bbop_def_kind  ces bbop_def_kind  10 normal
attribute bbop_id        ces bbop_id        1024 normal
attribute bbop_name      ces bbop_name      300 normal
attribute bbop_version   ces bbop_version   100 normal
attribute bbop_def_in    ces bbop_def_in    9000 normal
attribute bbop_abs_name  ces bbop_abs_name  3000 normal
attribute bbop_type_def  ces bbop_type_def  9000 normal
attribute bbop_value     bin bbop_value     3000 normal
attribute bbop_members   bin bbop_members   9000 normal
attribute bbop_disc_td   ces bbop_disc_td   9000 normal
attribute bbop_orig_td   ces bbop_orig_td   9000 normal
attribute bbop_mode      ces bbop_mode      10 normal
attribute bbop_res_def   ces bbop_res_def   9000 normal
attribute bbop_params    bin bbop_params    9000 normal
attribute bbop_contexts  bin bbop_contexts  3000 normal
attribute bbop_except    bin bbop_except    9000 normal
```

Figure 35 (Part 1 of 2). The slapd.cb.at.conf File

```
attribute bbop_base_int  bin bbop_base_int  9000 normal
attribute bbop_pkstring  bin bbop_pkstring  5000 normal
attribute bbop_refer     bin bbop_refer     9000 normal
# ------------------------------------------------------------------
```

Figure 35 (Part 2 of 2). The slapd.cb.at.conf File

The slapd.cb.oc.conf File

```
# ------------------------------------------------------------------
# This file is shipped in code page IBM-1047 and must remain in
# code page IBM-1047.
# ------------------------------------------------------------------

#
# Licensed Materials - Property of IBM
# 5647-A01
# (C) Copyright IBM Corp. 1998, 1999
#

# ------------------------------------------------------------------
# Filename slapd.cbseries.oc.conf
# This file contains definitions for object classes that are used
# by the CBSeries component of OS/390.
# ------------------------------------------------------------------

# ------------------------------------------------------------------
# The following object classes are used in managing the CBSeries
# Naming Service.
#
objectclass INamingService
 requires
  objectClass,
  TypelessRDN,
  bt,
  oref
 allows
  seq#
# ------------------------------------------------------------------

# ------------------------------------------------------------------
# The following object classes are used in managing the CBSeries
# Interface Repository.
#
objectclass BBOPIRRepository
 requires
  objectClass,
  TypelessRDN,
  bbop_def_kind

objectclass BBOPIRRepositoryId
 requires
  objectClass,
  TypelessRDN,
```

Figure 36 (Part 1 of 5). The slapd.cb.oc.conf File

```
|    bbop_def_kind
|   allows
|    bbop_abs_name,
|    bbop_pkstring

|  objectclass BBOPIRModuleDef
|   requires
|    objectClass,
|    TypelessRDN,
|    bbop_def_kind
|   allows
|    bbop_id,
|    bbop_name,
|    bbop_version,
|    bbop_def_in,
|    bbop_abs_name

|  objectclass BBOPIRConstantDef
|   requires
|    objectClass,
|    TypelessRDN,
|    bbop_def_kind
|   allows
|    bbop_id,
|    bbop_name,
|    bbop_version,
|    bbop_def_in,
|    bbop_abs_name,
|    bbop_type_def,
|    bbop_value

|  objectclass BBOPIRStructDef
|   requires
|    objectClass,
|    TypelessRDN,
|    bbop_def_kind
|   allows
|    bbop_id,
|    bbop_name,
|    bbop_version,
|    bbop_def_in,
|    bbop_abs_name,
|    bbop_members,
|    bbop_refer
```

| *Figure 36 (Part 2 of 5). The slapd.cb.oc.conf File*

```
| objectclass BBOPIRUnionDef
|  requires
|   objectClass,
|   TypelessRDN,
|   bbop_def_kind
|  allows
|   bbop_id,
|   bbop_name,
|   bbop_version,
|   bbop_def_in,
|   bbop_abs_name,
|   bbop_disc_td,
|   bbop_members,
|   bbop_refer

| objectclass BBOPIREnumDef
|  requires
|   objectClass,
|   TypelessRDN,
|   bbop_def_kind
|  allows
|   bbop_id,
|   bbop_name,
|   bbop_version,
|   bbop_def_in,
|   bbop_abs_name,
|   bbop_members,
|   bbop_refer

| objectclass BBOPIRAliasDef
|  requires
|   objectClass,
|   TypelessRDN,
|   bbop_def_kind
|  allows
|   bbop_id,
|   bbop_name,
|   bbop_version,
|   bbop_def_in,
|   bbop_abs_name,
|   bbop_orig_td,
|   bbop_refer

| objectclass BBOPIRExceptionDef
|  requires
```

| *Figure 36 (Part 3 of 5). The slapd.cb.oc.conf File*

```
|    objectClass,
|    TypelessRDN,
|    bbop_def_kind
|   allows
|    bbop_id,
|    bbop_name,
|    bbop_version,
|    bbop_def_in,
|    bbop_abs_name,
|    bbop_members,
|    bbop_refer

|  objectclass BBOPIRAttributeDef
|   requires
|    objectClass,
|    TypelessRDN,
|    bbop_def_kind
|   allows
|    bbop_id,
|    bbop_name,
|    bbop_version,
|    bbop_def_in,
|    bbop_abs_name,
|    bbop_type_def,
|    bbop_mode

|  objectclass BBOPIROperationDef
|   requires
|    objectClass,
|    TypelessRDN,
|    bbop_def_kind
|   allows
|    bbop_id,
|    bbop_name,
|    bbop_version,
|    bbop_def_in,
|    bbop_abs_name,
|    bbop_res_def,
|    bbop_params,
|    bbop_mode,
|    bbop_contexts,
|    bbop_except

|  objectclass BBOPIRInterfaceDef
|   requires
```

| *Figure 36 (Part 4 of 5). The slapd.cb.oc.conf File*

```
|    objectClass,
|    TypelessRDN,
|    bbop_def_kind
|   allows
|    bbop_id,
|    bbop_name,
|    bbop_version,
|    bbop_def_in,
|    bbop_abs_name,
|    bbop_base_int,
|    bbop_refer
|  # ------------------------------------------------------------------
```

Figure 36 (Part 5 of 5). The slapd.cb.oc.conf File

The schema.system.at File

```
# ------------------------------------------------------------------
# This file is shipped in code page IBM-1047 and must remain in
# code page IBM-1047.
# ------------------------------------------------------------------

#
# Licensed Materials - Property of IBM
# 5647-A01
# (C) Copyright IBM Corp. 1999
#

# ------------------------------------------------------------------
# This is the LDAP Server Private System Attribute Type Definition file
#   for OS/390.
#
# WARNING:  Do not alter the attribute type definitions in this file.
#
# NOTE: The LDAP Server depends upon the definitions of attribute types
#       contained in this file for correct operation.  Do not remove these
#       attribute types from the configuration of the LDAP Server.
# ------------------------------------------------------------------

attribute   aliasedObjectName
            aliasedentryname          dn     aliasedObject    1000 normal
attribute   aclEntry                  cis    aclEntry        32700 restricted
attribute   aclPropagate              cis    aclPropagate        5 restricted
attribute   aclSource                 dn     aclSource        1000 restricted
attribute   altServer                 ces    altServer        2048 normal
attribute   createTimestamp           cis    createTimestamp    20 system
attribute   creatorsName              dn     creatorsName     1000 system
attribute   displayName               cis    displayName       128 normal
attribute   entryOwner                dn     entryOwner       1000 restricted

attribute   lastModifiedBy            dn     lastModifiedBy   1000 system

attribute   lastModifiedTime          cis    lastModifiedTime   20 system
# NOTE: Before using this attribute, See the Migration
# section of the documentation.
#attribute   lastModifiedTime         cis    lastModifiedTime   30 system

attribute   modifiersName             dn     modifiersName    1000 system
attribute   modifyTimestamp           cis    modifyTimestamp    20 system
attribute   namingContexts            dn     namingContexts   1000 normal
attribute   ownerPropagate            cis    ownerPropagate      5 restricted
attribute   ownerSource               dn     ownerSource      1000 restricted
```

Figure 37 (Part 1 of 2). The schema.system.at File

```
| attribute   ref                       ces    ref              100 normal

| attribute   replicaBindDN             dn     replicaBindDN    1000 critical
| attribute   replicaBindMethod         cis    replicaBindMethod 100 normal
| attribute   replicaCredentials
|             replicaBindCredentials    bin    replicaCred       128 critical
| attribute   replicaHost               cis    replicaHost       100 normal
| attribute   replicaPort               cis    replicaPort        10 normal
| attribute   replicaUpdateTimeInterval cis    replicaUpdateInt   20 normal
| attribute   replicaUseSSL             cis    replicaUseSSL      10 normal
| attribute   subschemaSubentry         dn     subschemaSubent  1000 system
| attribute   supportedControl          cis    supportedControl 2048 normal
| attribute   supportedExtension        cis    supportedExtensio 2048 normal
| attribute   supportedLDAPVersion      cis    supportedLDAPVers  11 normal
| attribute   supportedSASLMechanisms   cis    supportedSASLMech 2048 normal
| # ------------------------------------------------------------------
| # WARNING:  Do not alter the attribute type definitions in this file.
| # ------------------------------------------------------------------
```

| *Figure 37 (Part 2 of 2). The schema.system.at File*

The schema.system.oc File

```
# ------------------------------------------------------------------
# This file is shipped in code page IBM-1047 and must remain in
# code page IBM-1047.
# ------------------------------------------------------------------

#
# Licensed Materials - Property of IBM
# 5647-A01
# (C) Copyright IBM Corp. 1999
#

# ------------------------------------------------------------------
# This is the LDAP Server Private System Object Class Definition file
#   for OS/390.
#
# WARNING:  Do not alter the object class definitions in this file.
#
# NOTE: The LDAP Server depends upon the definitions of object classes
#       contained in this file for correct operation.  Do not remove these
#       object classes from the configuration of the LDAP Server.
# ------------------------------------------------------------------

objectclass   accessGroup
        requires
                objectClass,
                cn,
                member
        allows
                businessCategory,
                seeAlso,
                owner,
                ou,
                o,
                description

objectclass   accessRole
        requires
                objectClass,
                cn,
                member
        allows
                businessCategory,
                seeAlso,
                owner,
                ou,
```

Figure 38 (Part 1 of 2). The schema.system.oc File

```
                o,
                description

objectclass   alias
        requires
                objectClass,
                aliasedObjectName

objectclass   aliasObject
        requires
                objectClass,
                aliasedObjectName

objectclass   referral
        requires
                objectClass,
                ref

objectclass   replicaObject
        requires
                objectClass,
                cn,
                replicaBindDN,
                replicaHost,
                replicaCredentials
        allows
                description,
                seeAlso,
                replicaPort,
                replicaBindMethod,
                replicaUseSSL,
                replicaUpdateTimeInterval

# ------------------------------------------------------------------
# WARNING:  Do not alter the object class definitions in this file.
# ------------------------------------------------------------------
```

Figure 38 (Part 2 of 2). The schema.system.oc File

The schema.IBM.at File

```
# ------------------------------------------------------------------
# This file is shipped in code page IBM-1047 and must remain in
# code page IBM-1047.
# ------------------------------------------------------------------
#
# Licensed Materials - Property of IBM
# 5647-A01
# (C) Copyright IBM Corp. 1999
#
# ------------------------------------------------------------------
# This is the LDAP Server IBM-defined Attribute Type Definition file
#   for OS/390.
#
# WARNING:  Do not alter the attribute type definitions in this file.
# ------------------------------------------------------------------

attribute   acceleratorCapabilities    cis    accelCap            11 normal
attribute   accessHint                 dn     accessHint        1000 normal
attribute   accountHint                dn     accountHint       1000 normal
attribute   accountService             dn     accountService    1000 normal
attribute   AccountSuffix              dn     AccountSuffix     1000 normal
attribute   actionPending              cis    actionPending        5 normal
attribute   addMajor                   cis    addMajor            11 normal
attribute   addMinor                   cis    addMinor            11 normal
attribute   addressWidth               cis    addressWidth        11 normal
attribute   ansiLevel                  cis    ansiLevel           32 normal
attribute   appl
            applicationName            cis    appl               256 normal
attribute   applSoftwareHint           dn     applSoftwareHint  1000 normal
attribute   applSystemHint             dn     applSystemHint    1000 normal
attribute   audibleAlarm               cis    audibleAlarm         5 normal
attribute   authenticationType         cis    authType           128 normal
attribute   autoInitiate               cis    autoInitiate         5 normal
attribute   autoTerminate              cis    autoTerminate        5 normal
attribute   availablefordirsync        cis    AVAILABLEFORDSYN    10 normal
attribute   backupName                 cis    backupName          64 normal
attribute   backupTelephoneNumber      tel    backupTelNumber     32 normal
attribute   bandWidth                  cis    bandWidth           11 normal
attribute   bankLabel                  cis    bankLabel          100 normal
attribute   bbop_abs_name
            bbop-abs-name              ces    bbop_abs_name     3000 normal
attribute   bbop_base_int
            bbop-base-int              bin    bbop_base_int     9000 normal
attribute   bbop_contexts
            bbop-contexts              bin    bbop_contexts     3000 normal
```

Figure 39 (Part 1 of 14). The schema.IBM.at File

```
| attribute   bbop_def_in
|             bbop-def-in               ces    bbop_def_in       9000 normal
| attribute   bbop_def_kind
|             bbop-def-kind             ces    bbop_def_kind       10 normal
| attribute   bbop_disc_td
|             bbop-disc-td              ces    bbop_disc_td      9000 normal
| attribute   bbop_except
|             bbop-except               bin    bbop_except       9000 normal
| attribute   bbop_id
|             bbop-id                   ces    bbop_id           1024 normal
| attribute   bbop_members
|             bbop-members              bin    bbop_members      9000 normal
| attribute   bbop_mode
|             bbop-mode                 ces    bbop_mode           10 normal
| attribute   bbop_name
|             bbop-name                 ces    bbop_name          300 normal
| attribute   bbop_orig_td
|             bbop-orig-td              ces    bbop_orig_td      9000 normal
| attribute   bbop_params
|             bbop-params               bin    bbop_params       9000 normal
| attribute   bbop_pkstring
|             bbop-pkstring             bin    bbop_pkstring     5000 normal
| attribute   bbop_refer
|             bbop-refer                bin    bbop_refer        9000 normal
| attribute   bbop_res_def
|             bbop-res-def              ces    bbop_res_def      9000 normal
| attribute   bbop_type_def
|             bbop-type-def             ces    bbop_type_def     9000 normal
| attribute   bbop_value
|             bbop-value                bin    bbop_value        3000 normal
| attribute   bbop_version
|             bbop-version              ces    bbop_version       100 normal
| attribute   binProperty               bin    binProperty     250000 normal
| attribute   binPropertyType           cis    binPropertyType    128 normal
| attribute   BIOSDate                  cis    BIOSDate            30 normal
| attribute   blockSize                 cis    blockSize           11 normal
| attribute   breachDescription         cis    breachDescription  256 normal
| attribute   bt
|             bindingType               cis    bindingType         15 normal
| attribute   buildNumber               ces    buildNumber         64 normal
| attribute   busAttributes             ces    busAttributes       32 normal
| attribute   bytesPerSector            cis    bytesPerSector      11 normal
| attribute   cableManagementStrategy   cis    cableMgmtStrat     250 normal
| attribute   capability                cis    capability         128 normal
| attribute   capacity                  cis    capacity            11 normal
```

| *Figure 39 (Part 2 of 14). The schema.IBM.at File*

```
| attribute   caption                    cis    caption            128 normal
| attribute   card
|             cardName                   cis    card               256 normal
| attribute   CDS-CELL
|             CDS_CELL                   bin    cdscell           1024 normal
| attribute   CDS-REPLICA
|             CDS_REPLICA                bin    cdsreplicas       1024 normal
| attribute   cellularTelephoneNumber    tel    cellTelNumber       32 normal
| attribute   certificateExpirationDate  cis    certificateExpira 1024 normal
| attribute   certifierId                cis    certifierId       1024 normal
| attribute   certifierPassword          ces    certifierPassword   63 critical
| attribute   cesProperty                ces    cesProperty      32700 normal
| attribute   cesPropertyType            cis    cesPropertyType    128 normal
| attribute   chassisTypes               cis    chassisTypes        11 normal
| attribute   cid
|             configID                   cis    cid                256 normal
| attribute   cisProperty                cis    cisProperty      32700 normal
| attribute   cisPropertyType            cis    cisPropertyType    128 normal
| attribute   city                       cis    city               256 normal
| attribute   clienttypereg              cis    CLIENTTYPEREG       20 normal
| attribute   codeSet                    cis    codeSet             64 normal
| attribute   companyName                cis    companyName         64 normal
| attribute   compressed                 cis    compressed           5 normal
| attribute   compressionMethod          cis    compressionMethod  100 normal
| attribute   configPtr                  dn     configPtr         1000 normal
| attribute   connectorType              cis    connectorType       11 normal
| attribute   controllerTimeouts         cis    controlTimeouts     11 normal
| attribute   countryCode                cis    countryCode         16 normal
| attribute   countryreg                 cis    countryreg        1024 normal
| attribute   coverLetterStatus          cis    CoverLetterStatus    5 normal
| attribute   createAddressBookEntry     cis    createAddressBook    5 normal
| attribute   createFullTextIndex        cis    createFullTextInd    5 normal
| attribute   createIdFile               cis    createIdFile         5 normal
| attribute   createMailDatabase         cis    createMailDatabas    5 normal
| attribute   createNorthAmericanId      cis    createNorthAmeric    5 normal
| attribute   createNotesUser            cis    createNotesUser      5 normal
| attribute   currentBitsPerPixel        cis    curBitsPerPix       11 normal
| attribute   currentClockSpeed          cis    currentClockSpeed   11 normal
| attribute   currentHorizontalResolution cis   currentHorzResol    11 normal
| attribute   currentNumberOfColumns     cis    currentNbrCols      11 normal
| attribute   currentNumberOfRows        cis    curNbrOfRows        11 normal
| attribute   currentRefreshRate         cis    curRefreshRate      11 normal
| attribute   currentRequiredOrProduced  cis    currentReqOrProd    11 normal
| attribute   currentScanMode            cis    currentScanMode     11 normal
| attribute   currentTimeZone            cis    currentTimeZone     11 normal
```

| *Figure 39 (Part 3 of 14). The schema.IBM.at File*

```
| attribute   currentVerticalResolution  cis    curVertRes          11 normal
| attribute   dataWidth                  cis    dataWidth           11 normal
| attribute   db2additionalParameters    cis    db2additionalPara 1024 normal
| attribute   db2ARLibrary               cis    db2ARLibrary       256 normal
| attribute   db2authenticationLocation  cis    db2authentication   64 normal
| attribute   db2databaseAlias           cis    db2databaseAlias  1024 normal
| attribute   db2databaseName            cis    db2databaseName   1024 normal
| attribute   db2databaseRelease
|             db2Release                 cis    db2databaseReleas   64 normal
| attribute   db2gwPtr                   dn     db2gwPtr          1000 normal
| attribute   db2instanceName
|             instance                   cis    db2instanceName    256 normal
| attribute   db2nodeAlias               cis    db2nodeAlias      1024 normal
| attribute   db2nodeName                cis    db2nodeName       1024 normal
| attribute   db2nodePtr                 dn     db2nodePtr        1000 normal
| attribute   db2Type                    cis    db2Type             64 normal
| attribute   DCEPrincipalName           cis    DCEPrincipalName  2048 normal
| attribute   defaultBlockSize           cis    defaultBlockSize    11 normal
| attribute   defaultpassword            ces    defaultpassword     63 critical
| attribute   depth                      cis    depth               11 normal
| attribute   detectedErrorState         cis    detErrorState       11 normal
| attribute   deviceID                   ces    deviceID            64 normal
| attribute   deviceMap                  ces    deviceMap           64 normal
| attribute   diskDriveIndex             cis    diskDriveIndex      11 normal
| attribute   displayType                cis    displayType         11 normal
| attribute   distributedOS              cis    distributedOS        5 normal
| attribute   domainuserid               cis    domainuserid      1024 normal
| attribute   dominogroupmembers         cis    dominogroupmember 1024 normal
| attribute   doubleClickRate            cis    doubleClickRate     11 normal
| attribute   driverName                 cis    driverName         256 normal
| attribute   driveType                  cis    driveType           11 normal
| attribute   editable                   cis    editable             5 normal
| attribute   eNetworkHostName           cis    eNetworkHostName  1024 normal
| attribute   eNetworkPort               cis    eNetworkPort      1024 normal
| attribute   eNTDomainGroupID           cis    eNTDomainGroupID  1024 normal
| attribute   eNTGroupAttributes         cis    eNTGroupAttribute   11 normal
| attribute   eNTPasswordSync            ces    eNTPasswordSync   1024 normal
| attribute   eNTUserBadPwCount          cis    eNTUserBadPwCount   11 normal
| attribute   eNTUserFlags               cis    eNTUserFlags        11 normal
| attribute   eNTUserLogonHours          cis    eNTUserLogonHours   11 normal
| attribute   eNTUserMaxStorage          cis    eNTUserMaxStorage   11 normal
| attribute   eNTUserPasswordExpired     cis    eNTUserPasswordEx   11 normal
| attribute   eNTUserPrimaryGroupId      cis    eNTUserPrimaryGro   11 normal
| attribute   eNTUserPriv                cis    eNTUserPriv         11 normal
| attribute   eNTUserUnitsPerWeek        cis    eNTUserUnitsPerWe   64 normal
```

| *Figure 39 (Part 4 of 14). The schema.IBM.at File*

```
| attribute   errorMethodology           cis    errorMethodology   100 normal
| attribute   externalCacheEnabled       cis    extCacheEnabled      5 normal
| attribute   externalTelephoneNumber    tel    extTelNumber        32 normal
| attribute   family                     cis    family              11 normal
| attribute   fileSystem                 cis    fileSystem          64 normal
| attribute   floor                      cis    floor               64 normal
| attribute   formFactor                 cis    formFactor          11 normal
| attribute   fullName                   cis    fullName          1024 normal
| attribute   groupid                    cis    groupid           1024 normal
| attribute   GroupSuffix                dn     GroupSuffix       1000 normal
| attribute   GroupType                  cis    GroupType            1 normal
| attribute   hardwareVersion            ces    hardwareVersion     64 normal
| attribute   headedness                 cis    headedness          11 normal
| attribute   heatGeneration             cis    heatGeneration      11 normal
| attribute   height                     cis    height              11 normal
| attribute   horizontalResolution       cis    horizRes            11 normal
| attribute   horizontalSize             cis    horizontalSize      11 normal
| attribute   hostBusType                ces    hostBusType         32 normal
| attribute   hostedSoftwarePtr          dn     hostedSoftwarePtr 1000 normal
| attribute   hostingBoard               cis    hostingBoard         5 normal
| attribute   hotSwappable               cis    hotSwappable         5 normal
| attribute   httppassword               ces    HTTPPASSWORD        63 critical
| attribute   httppasswordsync           cis    HTTPPASSWORDSYN      5 normal
| attribute   identificationCode         ces    idCode              64 normal
| attribute   identifyingNumber          ces    identifyingNumber   64 normal
| attribute   idFilePath                 cis    idFilePath        1024 normal
| attribute   idtype                     cis    idtype              17 normal
| attribute   initialPassword            ces    initialPassword     63 critical
| attribute   initialPopulation          cis    initialPopulation    5 normal
| attribute   installDate                cis    installDate         30 normal
| attribute   installSoftwarePtr         dn     installSoftwarePt 1000 normal
| attribute   interfaceType              ces    interfaceType       64 normal
| attribute   interleavePosition         cis    interleavePos       11 normal
| attribute   internalCacheEnabled       cis    intCacheEnabled      5 normal
| attribute   internalTelephoneNumber    tel    intTelNumber        32 normal
| attribute   internetAddress            cis    internetAddress   1024 normal
| attribute   ioAccessSupported          ces    ioAccessSupported   32 normal
| attribute   IPLAction                  cis    IPLAction           11 normal
| attribute   IRQNumber                  cis    IRQNumber           11 normal
| attribute   isCompatible               cis    isCompatible        11 normal
| attribute   isLocked                   cis    isLocked             5 normal
| attribute   jarFileName                ces    jarFileName        256 normal
| attribute   jobCountSinceLastReset     cis    jobCtSinceLastRes   11 normal
| attribute   key                        ces    key                256 critical
| attribute   keyLocation                ces    keyLocation        256 critical
```

| *Figure 39 (Part 5 of 14). The schema.IBM.at File*

```
| attribute   languageEdition            ces    languageEdition     32 normal
| attribute   languagesSupported         cis    langSupported       11 normal
| attribute   launchable                 cis    launchable           5 normal
| attribute   layout                     cis    layout             100 normal
| attribute   LdifFileName               dn     LdifFileName      1000 normal
| attribute   listname                   cis    listname          1024 normal
| attribute   loadPercentage             cis    loadPercentage      11 normal
| attribute   localadmin                 cis    localadmin        1024 normal
| attribute   localPath                  ces    localPath          256 normal
| attribute   location                   cis    location           256 normal
| attribute   locationName               cis    locationName       256 normal
| attribute   lockPresent                cis    lockPresent          5 normal
| attribute   logicalUnit                cis    logicalUnit         11 normal
| attribute   mailDomain                 cis    mailDomain        1024 normal
| attribute   mailFile                   cis    mailFile          1024 normal
| attribute   mailFileOwnerAccess        cis    mailFileOwnerAcce    8 normal
| attribute   mailFileTemplate           cis    mailFileTemplate   256 normal
| attribute   mailProgram                cis    mailProgram       1024 normal
| attribute   mailServer                 cis    mailServer        1024 normal
| attribute   mailSystem                 cis    MAILSYSTEM           1 normal
| attribute   maintenanceUnitForSoftware dn     mainUnitForSfw    1000 normal
| attribute   managerName                cis    managerName        256 normal
| attribute   managerTelephoneNumber     tel    mgrTelNumber        32 normal
| attribute   manufacturer               ces    manufacturer       256 normal
| attribute   maxBlockSize               cis    maxBlockSize        11 normal
| attribute   maxCDB                     cis    maxCDB              11 normal
| attribute   maxClockSpeed              cis    maxClockSpeed       11 normal
| attribute   maxDataWidth               cis    maxDataWidth        11 normal
| attribute   maxFailedLogins            cis    maxFailedLogins     11 normal
| attribute   maximumComponentLength     cis    maxComponentLen     11 normal
| attribute   maxMediaSize               cis    maxMediaSize        11 normal
| attribute   maxMemorySupported         cis    maxMemSupported     11 normal
| attribute   maxNumberControlled        cis    maxNbrControlled    11 normal
| attribute   maxNumberOfProcesses       cis    maxNumProcesses     11 normal
| attribute   maxRefreshRate             cis    maxRefreshRate      11 normal
| attribute   maxScatter                 cis    maxScatter          11 normal
| attribute   maxTransferRate            cis    maxTransferRate     11 normal
| attribute   mciHDWCollectDateTime      cis    mciHDWColDateTime   30 normal
| attribute   mciHDWCollectVersion       cis    mciHDWCollVersion   64 normal
| attribute   mciPTFCollectDateTime      cis    mciPTFColDateTime   30 normal
| attribute   mciPTFCollectVersion       cis    mciPTFCollVersion   64 normal
| attribute   mciSFWCollectDateTime      cis    mciSFWColDateTime   30 normal
| attribute   mciSFWCollectVersion       cis    mciSFWCollVersion   64 normal
| attribute   mediaAccessDeviceCapabilities cis mediaAccDevCap      11 normal
| attribute   mediaLoaded                cis    mediaLoaded          5 normal
```

| *Figure 39 (Part 6 of 14). The schema.IBM.at File*

```
| attribute   mediaType                cis    mediaType           11 normal
| attribute   memoryType               cis    memoryType          11 normal
| attribute   minBlockSize             cis    minBlockSize        11 normal
| attribute   minPasswordLength        cis    minPasswordLength   11 normal
| attribute   minRefreshRate           cis    minRefreshRate      11 normal
| attribute   model                    cis    model               64 normal
| attribute   modelNumber              cis    modelNumber         10 normal
| attribute   modelSubNumber           cis    modelSubNumber      10 normal
| attribute   msgFileName              ces    msgfilename        256 normal
| attribute   nameFormat               ces    nameFormat          64 normal
| attribute   networkAddress           ces    networkAddress      40 normal
| attribute   nextContainerDN          dn     nextContainerDN   1000 normal
| attribute   nfiAdapterMemory         cis    nfiAdapterMemory    11 normal
| attribute   nfiBaseMemory            cis    nfiBaseMemory       11 normal
| attribute   nfiBoardMemory           cis    nfiBoardMemory      11 normal
| attribute   nfiCacheableMemory       cis    nfiCacheMem         11 normal
| attribute   nfiCmWindowEnd           cis    nfiCmWindowEnd      20 normal
| attribute   nfiCmWindowEndGMT        cis    nfiCmWinEndGMT       2 normal
| attribute   nfiCmWindowStart         cis    nfiCmWindowStart    20 normal
| attribute   nfiCmWindowStartGmt      cis    nfiCmWinStartGMT     2 normal
| attribute   nfiDedicatedIRQ          cis    nfiDedicatedIRQ     64 normal
| attribute   nfiDistWindowEnd         cis    nfiDistWindowEnd    20 normal
| attribute   nfiDistWindowEndGMT      cis    nfiDistWindEndGMT    2 normal
| attribute   nfiDistWindowStart       cis    nfiDistWindStart    20 normal
| attribute   nfiDistWindowStartGMT    cis    nfiDistWinStGMT      2 normal
| attribute   nfiHDWCollectDateTime    cis    nfiHDWColDT         30 normal
| attribute   nfiInventoryServerID     ces    nfiInvServerID      64 normal
| attribute   nfiIPXAddress            cis    nfiIPXAddress       64 normal
| attribute   nfiKeywords              ces    nfiKeywords        256 normal
| attribute   nfiLocalDmMgr            cis    nfiLocalDmMgr        5 normal
| attribute   nfiLocalServerInv        cis    nfiLocalServerInv    5 normal
| attribute   nfiLogicalGroupID        cis    nfiLogicalGroupID   64 normal
| attribute   nfiManagingID            cis    nfiManagingID       64 normal
| attribute   nfiMemoryDetected        cis    nfiMemoryDetected   11 normal
| attribute   nfiMode                  ces    nfiMode             16 normal
| attribute   nfiNbrKeywords           cis    nfiNbrKeywords      11 normal
| attribute   nfiNetIdCPName           ces    nfiNetIdCPName      32 normal
| attribute   nfiNVRAM                 cis    nfiNVRAM            11 normal
| attribute   nfiParallelPorts         cis    nfiParallelPorts    11 normal
| attribute   nfiPreviousServerID      ces    nfiPreServiceID     64 normal
| attribute   nfiProtocolType          cis    nfiProtocolType     16 normal
| attribute   nfiReferenceDisk         cis    nfiReferenceDisk    11 normal
| attribute   nfiRemoteDmMgr           cis    nfiRemoteDmMgr       5 normal
| attribute   nfiRemoteServerID        ces    nfiRemServerID      64 normal
| attribute   nfiRen                   cis    nfiRen              16 normal
```

| *Figure 39 (Part 7 of 14). The schema.IBM.at File*

```
| attribute   nfiRgn                      ces    nfiRgn              16 normal
| attribute   nfiSerialPorts              cis    nfiSerialPorts      11 normal
| attribute   nfiServerName               cis    nfiServerName      128 normal
| attribute   nfiSFWCollectDateTime       cis    nfiSFWColDT         30 normal
| attribute   nfiSharedIRQ                cis    nfiSharedIRQ        64 normal
| attribute   nfiSnaAddress               ces    nfiSnaAddress       32 normal
| attribute   nfiTargetType               cis    nfiTargetType        2 normal
| attribute   ntDefaultPassword           ces    ntDefaultPassword   64 normal
| attribute   ntDomain                    cis    ntDomain          2048 normal
| attribute   ntDomainUserID              cis    ntDomainUserID    2048 normal
| attribute   ntuserauthflags             cis    ntuserauthflags     11 normal
| attribute   ntUserNumLogons             cis    ntUserNumLogons     11 normal
| attribute   ntUserWorkstations          cis    ntUserWorkstation 1024 normal
| attribute   numberOfBlocks              cis    numberOfBlocks      11 normal
| attribute   numberOfButtons             cis    numberOfButtons     11 normal
| attribute   numberOfColorPlanes         cis    nbrOfColorPlanes    11 normal
| attribute   numberOfFunctionKeys        cis    nbrOfFunctionKeys   11 normal
| attribute   numberOfLicensedUsers       cis    numLicensedUsers    11 normal
| attribute   numberOfMediaSupported      cis    nbrOfMediaSup       11 normal
| attribute   numberOfPowerCords          cis    nbrOfPowerCords     11 normal
| attribute   numberOfVideoPages          cis    nbrOfVideoPages     11 normal
| attribute   numberWarnDays              cis    numberWarnDays      11 normal
| attribute   objectClassCaption          cis    objectClassCap     128 normal
| attribute   optIdentifier               cis    optIdentifier      128 normal
| attribute   optIdentifierName           cis    optIdentifierName  128 normal
| attribute   oref
|             objectReference             cis    objectReference  32700 normal
| attribute   OS400CardCategory           cis    OS400CardCategory   11 normal
| attribute   OS400CardFamilyLevel        cis    OS400CardFamLevel   11 normal
| attribute   OS400Level                  ces    OS400Level          12 normal
| attribute   OS400ProductID              cis    OS400ProductID     256 normal
| attribute   OS400ProductOption          cis    OS400ProductOptio   11 normal
| attribute   OS400PTFMax                 ces    OS400PTFMax         12 normal
| attribute   OS400PTFMin                 ces    OS400PTFMin         12 normal
| attribute   OS400PTFSaveFileStatus      cis    OS400PTFSFStatus     5 normal
| attribute   OS400PTFSupersedingPTFId    cis    OS400PTFSsPTFId    256 normal
| attribute   OS400SupportedState         cis    OS400SupportedSta    5 normal
| attribute   osPtr                       dn     osPtr             1000 normal
| attribute   osType                      cis    osType              11 normal
| attribute   otherConnectorTypeDescription cis otherConnTypeDesc  256 normal
| attribute   otherFamilyDescription      cis    otherFamilyDesc     64 normal
| attribute   otherOSTypeDescription      cis    otherOSTypeDesc     64 normal
| attribute   otherPrincipalPtr           dn     otherPrincipalPtr 1000 normal
| attribute   otherProtocolSupportedDescription cis othProcSupDesc 256 normal
| attribute   otherVideoArchitectureDescription cis othVideoArchDesc 256 normal
```

| *Figure 39 (Part 8 of 14). The schema.IBM.at File*

```
| attribute   overwriteaddressbook       cis    OverwriteAddres      5 normal
| attribute   overwriteidfile            cis    OVERWRITEIDFILE      5 normal
| attribute   paperSizesSupported        cis    paperSizesSup       11 normal
| attribute   partitions                 cis    partitions          11 normal
| attribute   passwordCheckMethods       ces    pwCheckMethods    1024 normal
| attribute   passwordDictFiles          ces    passwordDictFiles 1024 normal
| attribute   passwordExpireTime         cis    passwordExpireTim   30 normal
| attribute   passwordGenerator          dn     passwordGenerator 1000 critical
| attribute   passwordMaxRepeatedChars   cis    pwMaxRepChars       11 normal
| attribute   passwordMinAlphaChars      cis    pwMinAlphaChars     11 normal
| attribute   passwordMinDiffChars       cis    pwMinDiffChars      11 normal
| attribute   passwordMinOtherChars      cis    pwMinOtherChars     11 normal
| attribute   passwordRegistry           cis    passwordRegistry   128 normal
| attribute   passwordReuseNum           cis    passwordReuseNum    11 normal
| attribute   passwordTimeReuse          cis    passwordTimeReuse   11 normal
| attribute   physicalElementLocation    dn     phyElementLoc     1000 normal
| attribute   physicalPosition           cis    physicalPosition   256 normal
| attribute   planarSpeed                cis    planarSpeed         11 normal
| attribute   pointingDeviceResolution   cis    pointingDevRes      11 normal
| attribute   pointingType               cis    pointingType        11 normal
| attribute   pointingTypeResolution     cis    pointingTypeRes     11 normal
| attribute   portName                   cis    portName           256 normal
| attribute   positionInRow              cis    positionInRow       11 normal
| attribute   powerManagementCapabilities cis   powerMgmtCap        11 normal
| attribute   powerManagementSupported   cis    powerMgmtSupport     5 normal
| attribute   prereqTarget               dn     prereqTarget      1000 normal
| attribute   primaryBIOS                cis    primaryBIOS          5 normal
| attribute   primaryBusType             cis    primaryBusType      32 normal
| attribute   primaryOwnerContact        cis    primaryOwnerCon    256 normal
| attribute   primaryOwnerName           cis    primaryOwnerName   128 normal
| attribute   principalName
|             principal                  cis    principalName      256 normal
| attribute   principalPtr               dn     principalPtr      1000 normal
| attribute   printerCapabilities        cis    printerCaps         11 normal
| attribute   printerStatus              cis    printerStatus       11 normal
| attribute   processorNumber            cis    processorNumber     11 normal
| attribute   profiles                   cis    profiles          1024 normal
| attribute   profileType                cis    _nocreate            8 sensitive
| attribute   propertyType               cis    propertyType       128 normal
| attribute   proposedAltFullNameLanguage cis   PROPOSEDALTFNL     256 normal
| attribute   proposedAltOrgUnit         cis    proposedAltOrgUni 1024 normal
| attribute   protectionManagement       cis    protManagement      11 normal
| attribute   protocolSupported          cis    protocolSupported   11 normal
| attribute   providerName               ces    providerName        64 normal
| attribute   publisherType              dn     publisherType     1000 normal
```

| *Figure 39 (Part 9 of 14). The schema.IBM.at File*

```
| attribute   queueName                  cis    queueName          256 normal
| attribute   racfAddressLine1           cis    _nocreate           60 sensitive
| attribute   racfAddressLine2           cis    _nocreate           60 sensitive
| attribute   racfAddressLine3           cis    _nocreate           60 sensitive
| attribute   racfAddressLine4           cis    _nocreate           60 sensitive
| attribute   racfAltGroupKeyword        cis    _nocreate            8 sensitive
| attribute   racfAttributes             cis    _nocreate           12 sensitive
| attribute   racfAuthKeyword            cis    _nocreate            6 sensitive
| attribute   racfAuthorizationDate      cis    _nocreate            6 sensitive
| attribute   racfAutoKeyword            cis    _nocreate            3 sensitive
| attribute   racfBuilding               cis    _nocreate           60 sensitive
| attribute   racfClassName              cis    _nocreate            8 sensitive
| attribute   racfCMDSYSKeyword          cis    _nocreate            8 sensitive
| attribute   racfConnectGroupAuthority  cis    _nocreate            8 sensitive
| attribute   racfConnectGroupName       cis    _nocreate            8 sensitive
| attribute   racfConnectGroupUACC       cis    _nocreate            8 sensitive
| attribute   racfCTLKeyword             cis    _nocreate            8 sensitive
| attribute   racfDatasetModel           cis    _nocreate           44 sensitive
| attribute   racfDCEAutoLogin           cis    _nocreate            3 sensitive
| attribute   racfDCEHomeCell            cis    _nocreate         1023 sensitive
| attribute   racfDCEHomeCellUUID        cis    _nocreate           36 sensitive
| attribute   racfDCEPrincipal           cis    _nocreate         1023 sensitive
| attribute   racfDCEUUID                cis    _nocreate           36 sensitive
| attribute   racfDefaultConsoleName     cis    _nocreate            8 sensitive
| attribute   racfDefaultGroup           cis    _nocreate            8 sensitive
| attribute   racfDepartment             cis    _nocreate           60 sensitive
| attribute   racfDomains                cis    _nocreate            5 sensitive
| attribute   racfDOMKeyword             cis    _nocreate            6 sensitive
| attribute   racfGroupNoTermUAC         cis    _nocreate            9 sensitive
| attribute   racfGroupUserAccess        cis    _nocreate            7 sensitive
| attribute   racfGroupUserids           cis    _nocreate            8 sensitive
| attribute   racfid                     cis    _nocreate            8 sensitive
| attribute   racfInstallationData       cis    _nocreate          255 sensitive
| attribute   racfKEYKeyword             cis    _nocreate            8 sensitive
| attribute   racfLastAccess             cis    _nocreate           15 sensitive
| attribute   racfLevelKeyword           cis    _nocreate            3 sensitive
| attribute   racfLogCommandResponseKeyword
|                                        cis    _nocreate            6 sensitive
| attribute   racfLogonDays              cis    _nocreate           29 sensitive
| attribute   racfLogonTime              cis    _nocreate           13 sensitive
| attribute   racfMformKeyword           cis    _nocreate            1 sensitive
| attribute   racfMGIDKeyword            cis    _nocreate            3 sensitive
| attribute   racfMonitorKeyword         cis    _nocreate            9 sensitive
| attribute   racfMscopeSystems          cis    _nocreate            8 sensitive
| attribute   racfMSGRCVRKeyword         cis    _nocreate            3 sensitive
```

| *Figure 39 (Part 10 of 14). The schema.IBM.at File*

```
| attribute   racfNetviewInitialCommand  cis    _nocreate          255 sensitive
| attribute   racfNetviewOperatorClass   cis    _nocreate            4 sensitive
| attribute   racfNGMFADMKeyword         cis    _nocreate            3 sensitive
| attribute   racfOmvsGroupId            cis    _nocreate            4 sensitive
| attribute   racfOmvsHome               cis    _nocreate         1023 sensitive
| attribute   racfOmvsInitialProgram     cis    _nocreate         1023 sensitive
| attribute   racfOmvsUid                cis    _nocreate           10 sensitive
| attribute   racfOperatorClass          cis    _nocreate           62 sensitive
| attribute   racfOperatorIdentification cis    _nocreate            3 sensitive
| attribute   racfOperatorPriority       cis    _nocreate            3 sensitive
| attribute   racfOperatorReSignon       cis    _nocreate            7 sensitive
| attribute   racfOvmFileSystemRoot      cis    _nocreate         1023 sensitive
| attribute   racfOvmGroupId             cis    _nocreate            4 sensitive
| attribute   racfOvmHome                cis    _nocreate         1023 sensitive
| attribute   racfOvmHomeUUID            cis    _nocreate           36 sensitive
| attribute   racfOvmInitialProgram      cis    _nocreate         1023 sensitive
| attribute   racfOvmUid                 cis    _nocreate           10 sensitive
| attribute   racfOwner                  cis    _nocreate            8 sensitive
| attribute   racfPassword               cis    _nocreate            8 critical
| attribute   racfPasswordChangeDate     cis    _nocreate            6 critical
| attribute   racfPasswordInterval       cis    _nocreate            3 sensitive
| attribute   racfPrimaryLanguage        cis    _nocreate           15 sensitive
| attribute   racfProgrammerName         cis    _nocreate           20 sensitive
| attribute   racfResumeDate             cis    _nocreate            8 sensitive
| attribute   racfRevokeDate             cis    _nocreate            8 sensitive
| attribute   racfRoom                   cis    _nocreate           60 sensitive
| attribute   racfRoutcodeKeyword        cis    _nocreate            7 sensitive
| attribute   racfSecondaryLanguage      cis    _nocreate           15 sensitive
| attribute   racfSecurityCategoryList   cis    _nocreate           15 sensitive
| attribute   racfSecurityLabel          cis    _nocreate            8 sensitive
| attribute   racfSecurityLevel          cis    _nocreate           15 critical
| attribute   racfStorageKeyword         cis    _nocreate            4 sensitive
| attribute   racfSubGroupName           cis    _nocreate            8 sensitive
| attribute   racfSuperiorGroup          cis    _nocreate            8 sensitive
| attribute   racfTerminalTimeout        cis    _nocreate            4 sensitive
| attribute   racfUDKeyword              cis    _nocreate            3 sensitive
| attribute   racfWorkAttrAccountNumber  cis    _nocreate          255 sensitive
| attribute   racfWorkAttrUserName       cis    _nocreate           60 sensitive
| attribute   registrationServer         cis    registrationServe 1024 normal
| attribute   removable                  cis    removable            5 normal
| attribute   replaceable                cis    replaceable          5 normal
| attribute   reqIdentifier              cis    reqIdentifier      128 normal
| attribute   reqIdentifierName          cis    reqIdentifierName  128 normal
| attribute   required                   cis    required             5 normal
| attribute   requirementsDescription    cis    reqDescription     100 normal
```

| *Figure 39 (Part 11 of 14). The schema.IBM.at File*

```
| attribute   requiresDaughterBoard      cis    reqDaughterBd        5 normal
| attribute   revisionNumber             cis    revisionNumber      32 normal
| attribute   role                       cis    role               100 normal
| attribute   SAFAccountNumber           cis    _nocreate           40 sensitive
| attribute   SAFDefaultCommand          cis    _nocreate           80 sensitive
| attribute   SAFDefaultLoginProc        cis    _nocreate            8 sensitive
| attribute   SAFDefaultSysoutClass      cis    _nocreate            1 sensitive
| attribute   SAFDefaultUnit             cis    _nocreate            8 sensitive
| attribute   SAFDestination             cis    _nocreate            8 sensitive
| attribute   SAFDfpDataApplication      cis    _nocreate            8 sensitive
| attribute   SAFDfpDataClass            cis    _nocreate            8 sensitive
| attribute   SAFDfpManagementClass      cis    _nocreate            8 sensitive
| attribute   SAFDfpStorageClass         cis    _nocreate            8 sensitive
| attribute   SAFHoldClass               cis    _nocreate            1 sensitive
| attribute   SAFJobClass                cis    _nocreate            1 sensitive
| attribute   SAFLogonSize               cis    _nocreate            4 sensitive
| attribute   SAFMaximumRegionSize       cis    _nocreate            4 sensitive
| attribute   SAFMessageClass            cis    _nocreate            1 sensitive
| attribute   SAFTsoSecurityLabel        cis    _nocreate            8 sensitive
| attribute   SAFUserdata                cis    _nocreate            2 sensitive
| attribute   sapName
|             sap                        cis    sapName            256 normal
| attribute   sapPtr                     dn     sapPtr            1000 normal
| attribute   saveIdInAddressBook        cis    saveIdInAddressBo    5 normal
| attribute   saveIdInFile               cis    saveIdInFile         5 normal
| attribute   SCSIBus                    cis    SCSIBus             11 normal
| attribute   SCSIControllerIndex        cis    SCSIControlIndex    11 normal
| attribute   SCSILogicalUnit            cis    SCSILogicalUnit     11 normal
| attribute   SCSIPort                   cis    SCSIPort            11 normal
| attribute   SCSITargetID               cis    SCSITargetID        11 normal
| attribute   secondaryBusType           cis    secondaryBusType    32 normal
| attribute   secretaryName              cis    secretaryName      256 normal
| attribute   secretaryTelephoneNumber   tel    secTelNumber        32 normal
| attribute   secretKey                  bin    secretKey          256 critical
| attribute   sectorsPerCluster          cis    sectorsPerCluster   11 normal
| attribute   sectorsPerTrack            cis    sectorsPerTrack     11 normal
| attribute   securityBreach             cis    securityBreach      11 normal
| attribute   sequenceNumber             cis    sequenceNumber     250 normal
| attribute   serviceDescriptions        cis    serviceDesc        250 normal
| attribute   serviceHint                dn     serviceHint       1000 normal
| attribute   serviceName
|             svc                        cis    serviceName        256 normal
| attribute   servicePhilosophy          cis    servicePhilosophy   11 normal
| attribute   setDbQuota                 cis    setDbQuota          11 normal
| attribute   settingID
```

| *Figure 39 (Part 12 of 14). The schema.IBM.at File*

```
|             sid                          cis    settingID          256 normal
|  attribute  setWarningThreshold          cis    SETWARNINGTHRESH    11 normal
|  attribute  shortName                    cis    shortName          256 normal
|  attribute  size                         cis    size                11 normal
|  attribute  sizeStoredInPagingFiles      cis    sizeStoredInPagin   11 normal
|  attribute  skuNumber
|             sku                          ces    skuNumber           64 normal
|  attribute  slotLayout                   cis    slotLayout         256 normal
|  attribute  slotLocation                 cis    slotLocation        11 normal
|  attribute  slotNumber                   cis    slotNumber          11 normal
|  attribute  software
|             softwareName                 cis    software           256 normal
|  attribute  softwareElementID            ces    softwareElementID  256 normal
|  attribute  softwareElementState         cis    swElementState      11 normal
|  attribute  specialRequirements          cis    specialReq           5 normal
|  attribute  speed                        cis    speed               11 normal
|  attribute  startMode                    ces    startMode           10 normal
|  attribute  startupParameters            ces    startupParameters  256 normal
|  attribute  stepping                     cis    stepping           100 normal
|  attribute  storageExtentAccess          cis    stgExtentAccess     11 normal
|  attribute  storageExtentPurpose         ces    stgExtentPurpose   100 normal
|  attribute  subadminGroup                dn     subadminGroup     1000 sensitive
|  attribute  subcountryCode               cis    subcountryCode      16 normal
|  attribute  supportingFiles              cis    supportingFiles   1024 normal
|  attribute  supportsFileBasedCompression cis    supportsFileComp     5 normal
|  attribute  sys
|             systemName                   cis    sys                256 normal
|  attribute  sysplex                      cis    _nocreate            8 sensitive
|  attribute  systemRoles                  cis    systemRoles        256 normal
|  attribute  tag                          cis    tag                256 normal
|  attribute  taName                       cis    taName             128 normal
|  attribute  targetAdapter                dn     targetAdapter     1000 normal
|  attribute  targetGroup                  dn     targetGroup       1000 sensitive
|  attribute  targetLocation               cis    targetLocation     128 normal
|  attribute  targetService                dn     targetService     1000 normal
|  attribute  timeExpireLockout            cis    timeExpireLockout   11 normal
|  attribute  timeOfLastReset              cis    timeOfLastReset     30 normal
|  attribute  totalCylinders               cis    totalCylinders      11 normal
|  attribute  totalHeads                   cis    totalHeads          11 normal
|  attribute  totalSectors                 cis    totalSectors        11 normal
|  attribute  totalSwapSpaceSize           cis    totalSwapSpaceSz    11 normal
|  attribute  totalTracks                  cis    totalTracks         11 normal
|  attribute  totalWidth                   cis    totalWidth          11 normal
|  attribute  tracksPerCylinder            cis    tracksPerCylinder   11 normal
|  attribute  tsName                       cis    tsName             128 normal
```

| *Figure 39 (Part 13 of 14). The schema.IBM.at File*

```
| attribute   tsType                    cis    tsType             128 normal
| attribute   TypelessRDN               ces    TypelessRDN       1024 normal
| attribute   typeMaticDelay            cis    typeMaticDelay      11 normal
| attribute   typeMaticRate             cis    typeMaticRate       11 normal
| attribute   unitStatus                cis    unitStatus          64 normal
| attribute   upgradeMethod             cis    upgradeMethod       11 normal
| attribute   userid                    cis    userid             256 normal
| attribute   userPKCS12                bin    userPKCS12      250000 normal
| attribute   userState                 cis    userState          128 sensitive
| attribute   UserSuffix                dn     UserSuffix        1000 normal
| attribute   userType                  cis    userType           128 sensitive
| attribute   usingElementPtr           dn     usingElementPtr   1000 normal
| attribute   validValues               cis    validValues        128 normal
| attribute   vendor                    cis    vendor             256 normal
| attribute   version                   cis    version             64 normal
| attribute   verticalResolution        cis    vertRes             11 normal
| attribute   verticalSize              cis    verticalSize        11 normal
| attribute   videoArchitecture         cis    videoArchitecture   11 normal
| attribute   videoMemoryType           cis    videoMemoryType     11 normal
| attribute   videoMode                 cis    videoMode           11 normal
| attribute   videoProcessor            ces    videoProcessor     100 normal
| attribute   videoSubsystem            cis    videoSubsystem      11 normal
| attribute   visibleAlarm              cis    visibleAlarm         5 normal
| attribute   volumeName                cis    volumeName          16 normal
| attribute   volumeSerialNumber        cis    volSerialNbr        64 normal
| attribute   weight                    cis    weight              11 normal
| attribute   width                     cis    width               11 normal

| # ------------------------------------------------------------------
| # WARNING:  Do not alter the attribute type definitions in this file.
| # ------------------------------------------------------------------
```

| *Figure 39 (Part 14 of 14). The schema.IBM.at File*

The schema.IBM.oc File

```
# ------------------------------------------------------------------
# This file is shipped in code page IBM-1047 and must remain in
# code page IBM-1047.
# ------------------------------------------------------------------
#
# Licensed Materials - Property of IBM
# 5647-A01
# (C) Copyright IBM Corp. 1999
#
# ------------------------------------------------------------------
# This is the LDAP Server IBM-defined Object Class Definition file
#   for OS/390.
#
# WARNING:  Do not alter the object class definitions in this file.
# ------------------------------------------------------------------

objectclass   cacheObject
        requires
                objectClass
        allows
                ttl

objectclass   cimBIOSelement
        requires
                objectClass
        allows
                primaryBIOS

objectclass   cimCard
        requires
                objectClass
        allows
                hostingBoard,
                card,
                slotLayout,
                requiresDaughterBoard,
                specialRequirements,
                requirementsDescription

objectclass   cimChassis
        requires
                objectClass
        allows
                numberOfPowerCords,
                currentRequiredOrProduced,
```

Figure 40 (Part 1 of 31). The schema.IBM.oc File

```
|                 heatGeneration,
|                 chassisTypes

| objectclass   cimChip
|         requires
|                 objectClass
|         allows
|                 formFactor

| objectclass   cimConfiguration
|         requires
|                 objectClass
|         allows
|                 cid,
|                 usingElementPtr

| objectclass   cimController
|         requires
|                 objectClass
|         allows
|                 timeOfLastReset,
|                 protocolSupported,
|                 otherProtocolSupportedDescription,
|                 maxNumberControlled

| objectclass   cimDesktopMonitor
|         requires
|                 objectClass
|         allows
|                 displayType,
|                 bandWidth

| objectclass   cimDiskDrive
|         requires
|                 objectClass

| objectclass   cimDisketteDrive
|         requires
|                 objectClass

| objectclass   cimDisplay
|         requires
|                 objectClass

| objectclass   cimKeyboard
```

| *Figure 40 (Part 2 of 31). The schema.IBM.oc File*

```
        requires
                objectClass
        allows
                numberOfFunctionKeys,
                layout,
                userPassword

objectclass   cimLocation
        requires
                objectClass
        allows
                locationName,
                postalAddress,
                physicalPosition

objectclass   cimLogicalDevice
        requires
                objectClass
        allows
                deviceID,
                powerManagementSupported,
                powerManagementCapabilities

objectclass   cimLogicalDisk
        requires
                objectClass

objectclass   cimLogicalElement
        requires
                objectClass

objectclass   cimManagedElement
        requires
                objectClass
        allows
                description,
                caption

objectclass   cimManagedSystemElement
        requires
                objectClass
        allows
                installDate,
                configPtr
```

Figure 40 (Part 3 of 31). The schema.IBM.oc File

```
objectclass   cimMediaAccessDevice
        requires
                objectClass
        allows
                mediaAccessDeviceCapabilities,
                errorMethodology,
                compressionMethod,
                numberOfMediaSupported,
                maxMediaSize,
                defaultBlockSize,
                maxBlockSize,
                minBlockSize

objectclass   cimPCVideoController
        requires
                objectClass
        allows
                videoArchitecture,
                otherVideoArchitectureDescription,
                videoMode,
                numberOfColorPlanes

objectclass   cimPhysicalComponent
        requires
                objectClass
        allows
                removable,
                replaceable,
                hotSwappable

objectclass   cimPhysicalElement
        requires
                objectClass
        allows
                manufacturer,
                model,
                skuNumber,
                serialNumber,
                tag,
                version

objectclass   cimPhysicalFrame
        requires
                objectClass
        allows
```

Figure 40 (Part 4 of 31). The schema.IBM.oc File

```
                cableManagementStrategy,
                servicePhilosophy,
                serviceDescriptions,
                lockPresent,
                audibleAlarm,
                visibleAlarm,
                securityBreach,
                breachDescription

objectclass   cimPhysicalMemory
        requires
                objectClass
        allows
                formFactor,
                memoryType,
                totalWidth,
                dataWidth,
                speed,
                capacity,
                bankLabel,
                positionInRow,
                interleavePosition

objectclass   cimPhysicalPackage
        requires
                objectClass
        allows
                removable,
                replaceable,
                hotSwappable,
                height,
                depth,
                width,
                weight,
                isCompatible

objectclass   cimPointingDevice
        requires
                objectClass
        allows
                pointingType,
                numberOfButtons,
                headedness,
                pointingDeviceResolution
```

Figure 40 (Part 5 of 31). The schema.IBM.oc File

```
objectclass   cimPrinter
        requires
                objectClass
        allows
                printerStatus,
                detectedErrorState,
                paperSizesSupported,
                languagesSupported,
                jobCountSinceLastReset,
                timeOfLastReset,
                printerCapabilities,
                horizontalResolution,
                verticalResolution,
                queueName

objectclass   cimProcessor
        requires
                objectClass
        allows
                role,
                family,
                otherFamilyDescription,
                upgradeMethod,
                maxClockSpeed,
                currentClockSpeed,
                dataWidth,
                addressWidth,
                loadPercentage,
                stepping

objectclass   cimProduct
        requires
                objectClass
        allows
                identifyingNumber,
                skuNumber,
                vendor,
                version

objectclass   cimSCSIController
        requires
                objectClass
        allows
                protectionManagement,
                maxDataWidth,
```

Figure 40 (Part 6 of 31). The schema.IBM.oc File

```
|                 maxTransferRate,
|                 controllerTimeouts

| objectclass   cimSetting
|         requires
|                 objectClass
|         allows
|                 settingID

| objectclass   cimStorageExtent
|         requires
|                 objectClass
|         allows
|                 storageExtentPurpose,
|                 storageExtentAccess,
|                 errorMethodology,
|                 blockSize,
|                 numberOfBlocks

| objectclass   cimUserDevice
|         requires
|                 objectClass
|         allows
|                 isLocked

| objectclass   cimVideoController
|         requires
|                 objectClass
|         allows
|                 videoProcessor,
|                 videoMemoryType,
|                 numberOfVideoPages,
|                 maxMemorySupported,
|                 acceleratorCapabilities,
|                 currentBitsPerPixel,
|                 currentHorizontalResolution,
|                 currentVerticalResolution,
|                 maxRefreshRate,
|                 minRefreshRate,
|                 currentRefreshRate,
|                 currentScanMode,
|                 currentNumberOfColumns,
|                 currentNumberOfRows

| objectclass   container
```

| *Figure 40 (Part 7 of 31). The schema.IBM.oc File*

```
        requires
                objectClass,
                cn

objectclass   DB2Database
        requires
                objectClass,
                db2databaseName,
                db2nodePtr
        allows
                db2additionalParameters,
                db2ARLibrary,
                db2authenticationLocation,
                db2databaseAlias,
                db2databaseRelease,
                db2gwPtr,
                DCEPrincipalName

objectclass   DB2Node
        requires
                objectClass,
                db2nodeName
        allows
                db2instanceName,
                db2nodeAlias,
                db2Type,
                host,
                protocolInformation

objectclass   dceCellInfo
        requires
                objectClass,
                CDS-CELL,
                CDS-REPLICA

objectclass   dcObject
        requires
                objectClass,
                dc

objectclass   eAccount
        requires
                objectClass
        allows
                caption,
```

Figure 40 (Part 8 of 31). The schema.IBM.oc File

```
|                 userPassword,
|                 userCertificate,
|                 principalPtr

| objectclass   eApplicationSoftware
|         requires
|                 objectClass
|         allows
|                 appl,
|                 osType,
|                 otherOSTypeDescription,
|                 installSoftwarePtr,
|                 hostedSoftwarePtr

| objectclass   eApplicationSystem
|         requires
|                 objectClass
|         allows
|                 hostedSoftwarePtr,
|                 osPtr

| objectclass   eBIOSElement
|         requires
|                 objectClass
|         allows
|                 BIOSDate

| objectclass   eChassis
|         requires
|                 objectClass
|         allows
|                 modelNumber,
|                 modelSubNumber,
|                 connectorType,
|                 otherConnectorTypeDescription,
|                 physicalElementLocation

| objectclass   eComputerSystem
|         requires
|                 objectClass
|         allows
|                 host,
|                 location

| objectclass   eContactPerson
```

| *Figure 40 (Part 9 of 31). The schema.IBM.oc File*

```
        requires
                objectClass
        allows
                internalTelephoneNumber,
                externalTelephoneNumber,
                cellularTelephoneNumber,
                pager,
                backupName,
                backupTelephoneNumber,
                managerName,
                managerTelephoneNumber,
                secretaryName,
                secretaryTelephoneNumber,
                mail,
                facsimileTelephoneNumber

objectclass   eController
        requires
                objectClass
        allows
                slotNumber,
                busAttributes,
                ioAccessSupported,
                hostBusType,
                maxScatter,
                maxCDB,
                addMajor,
                addMinor

objectclass   eDesktopMonitor
        requires
                objectClass
        allows
                displayType,
                horizontalSize,
                verticalSize,
                videoSubsystem,
                slotLocation

objectclass   eDiskDrive
        requires
                objectClass
        allows
                diskDriveIndex,
                bytesPerSector,
```

Figure 40 (Part 10 of 31). The schema.IBM.oc File

```
|                 interfaceType,
|                 partitions,
|                 sectorsPerTrack,
|                 totalCylinders,
|                 totalHeads,
|                 totalSectors,
|                 totalTracks,
|                 tracksPerCylinder,
|                 manufacturer,
|                 mediaLoaded,
|                 mediaType,
|                 model,
|                 SCSIBus,
|                 SCSILogicalUnit,
|                 SCSIPort,
|                 SCSITargetID,
|                 size,
|                 version,
|                 unitStatus,
|                 ansiLevel

| objectclass   eDominoAccount
|         requires
|                 objectClass,
|                 sn,
|                 userid
|         allows
|                 certificateExpirationDate,
|                 certifierId,
|                 certifierPassword,
|                 clienttypereg,
|                 createAddressBookEntry,
|                 createFullTextIndex,
|                 createIdFile,
|                 createMailDatabase,
|                 createNorthAmericanId,
|                 createNotesUser,
|                 description,
|                 fullName,
|                 givenName,
|                 idFilePath,
|                 idtype,
|                 initialPassword,
|                 initialPopulation,
|                 internetAddress,
```

| *Figure 40 (Part 11 of 31). The schema.IBM.oc File*

```
|                 l,
|                 localadmin,
|                 location,
|                 mail,
|                 mailDomain,
|                 mailFile,
|                 mailFileOwnerAccess,
|                 mailFileTemplate,
|                 mailProgram,
|                 mailServer,
|                 mailSystem,
|                 middleName,
|                 minPasswordLength,
|                 ou,
|                 overwriteaddressbook,
|                 overwriteidfile,
|                 profiles,
|                 registrationServer,
|                 saveIdInAddressBook,
|                 saveIdInFile,
|                 setDbQuota,
|                 setWarningThreshold,
|                 shortName

| objectclass   eDominoGroup
|         requires
|                 objectClass,
|                 groupid,
|                 listname
|         allows
|                 availablefordirsync,
|                 description,
|                 dominogroupmembers,
|                 GroupType,
|                 localadmin,
|                 owner,
|                 principalPtr

| objectclass   eDominoInitialPopulation
|         requires
|                 objectClass,
|                 AccountSuffix,
|                 GroupSuffix,
|                 LdifFileName,
|                 UserSuffix
```

| *Figure 40 (Part 12 of 31). The schema.IBM.oc File*

```
|          allows
|                  defaultpassword,
|                  initialPopulation,
|                  mailDomain

|  objectclass   eDominoUser
|          requires
|                  objectClass,
|                  domainuserid
|          allows
|                  accountHint,
|                  httppassword,
|                  httppasswordsync

|  objectclass   eGSOaccount
|          requires
|                  objectClass,
|                  cn
|          allows
|                  accountService,
|                  otherPrincipalPtr,
|                  secretKey

|  objectclass   eGSOApplicationSystem
|          requires
|                  objectClass
|          allows
|                  hostedSoftwarePtr,
|                  osPtr

|  objectclass   eGSOattachment
|          requires
|                  objectClass,
|                  propertyType

|  objectclass   eGSODomain
|          requires
|                  objectClass,
|                  cn
|          allows
|                  caption,
|                  description,
|                  targetLocation

|  objectclass   eGSOmachineProfile
```

| *Figure 40 (Part 13 of 31). The schema.IBM.oc File*

```
        requires
                objectClass,
                cn,
                osType
        allows
                caption,
                description,
                reqIdentifier

objectclass   eGSOSoftware
        requires
                objectClass

objectclass   eGSOuser
        requires
                objectClass,
                cn,
                userType,
                userState
        allows
                associatedName,
                subadminGroup,
                targetGroup

objectclass   eKeyboard
        requires
                objectClass
        allows
                countryCode,
                subcountryCode,
                codePage,
                typeMaticDelay,
                typeMaticRate

objectclass   eLocation
        requires
                objectClass
        allows
                companyName,
                postalAddress,
                st,
                postalCode,
                c,
                ou,
                roomNumber,
```

Figure 40 (Part 14 of 31). The schema.IBM.oc File

```
|                 buildingName,
|                 floor,
|                 city

|  objectclass   eLogicalDisk
|          requires
|                  objectClass
|          allows
|                  volumeName,
|                  compressed,
|                  driveType,
|                  fileSystem,
|                  maximumComponentLength,
|                  providerName,
|                  supportsFileBasedCompression,
|                  volumeSerialNumber,
|                  mediaType,
|                  sectorsPerCluster,
|                  bytesPerSector

|  objectclass   eManagedElement
|          requires
|                  objectClass

|  objectclass   eMotherBoard
|          requires
|                  objectClass
|          allows
|                  primaryBusType,
|                  revisionNumber,
|                  secondaryBusType

|  objectclass   eMotherBoardConfiguration
|          requires
|                  objectClass,
|                  settingID
|          allows
|                  internalCacheEnabled,
|                  externalCacheEnabled,
|                  planarSpeed

|  objectclass   eNFIApplication
|          requires
|                  objectClass
|          allows
```

| *Figure 40 (Part 15 of 31). The schema.IBM.oc File*

```
|                 nfiManagingID,
|                 nfiLogicalGroupID,
|                 nfiSFWCollectDateTime,
|                 nfiHDWCollectDateTime

| objectclass   eNFIChangeControlServer
|         requires
|                 objectClass
|         allows
|                 nfiCmWindowEnd,
|                 nfiCmWindowEndGMT,
|                 nfiCmWindowStart,
|                 nfiCmWindowStartGmt,
|                 nfiDistWindowEnd,
|                 nfiDistWindowStart,
|                 nfiDistWindowStartGMT,
|                 nfiMode,
|                 nfiProtocolType,
|                 nfiRen,
|                 nfiRgn,
|                 nfiServerName,
|                 nfiTargetType,
|                 nfiDistWindowEndGMT

| objectclass   eNFIOperatingSystem
|         requires
|                 objectClass
|         allows
|                 nfiMemoryDetected,
|                 nfiDedicatedIRQ,
|                 nfiSharedIRQ,
|                 nfiParallelPorts,
|                 nfiSerialPorts,
|                 nfiNVRAM,
|                 nfiBaseMemory,
|                 nfiBoardMemory,
|                 nfiAdapterMemory,
|                 nfiCacheableMemory,
|                 nfiReferenceDisk

| objectclass   eNFIPrinter
|         requires
|                 objectClass,
|                 portName
|         allows
```

| *Figure 40 (Part 16 of 31). The schema.IBM.oc File*

```
|                 driverName

|  objectclass   eNFIServer
|          requires
|                  objectClass
|          allows
|                  nfiInventoryServerID,
|                  nfiPreviousServerID,
|                  nfiRemoteServerID,
|                  nfiNetIdCPName,
|                  nfiLocalServerInv,
|                  nfiLocalDmMgr,
|                  nfiRemoteDmMgr,
|                  nfiNbrKeywords,
|                  nfiKeywords,
|                  nfiSnaAddress,
|                  nfiMode,
|                  networkAddress,
|                  nfiIPXAddress

|  objectclass   eNFISoftware
|          requires
|                  objectClass
|          allows
|                  serviceHint

|  objectclass   eNTAccount
|          requires
|                  objectClass,
|                  ntDomainUserID
|          allows
|                  codePage,
|                  eNTUserBadPwCount,
|                  eNTUserFlags,
|                  eNTUserLogonHours,
|                  eNTUserMaxStorage,
|                  eNTUserPasswordExpired,
|                  eNTUserPrimaryGroupId,
|                  eNTUserPriv,
|                  eNTUserUnitsPerWeek,
|                  ntDomainUserID,
|                  ntUserAcctExpires,
|                  ntuserauthflags,
|                  ntUserCountryCode,
|                  NTUserHomeDir,
```

Figure 40 (Part 17 of 31). The schema.IBM.oc File

```
|                 ntUserHomeDirDrive,
|                 ntUserLastLogoff,
|                 ntUserLastLogon,
|                 ntUserLogonServer,
|                 ntUserNumLogons,
|                 ntUserParms,
|                 ntUserProfile,
|                 ntUserScriptPath,
|                 ntUserUsrComment,
|                 ntUserWorkstations,
|                 principalPtr

| objectclass   eNTGroup
|         requires
|                 objectClass,
|                 eNTDomainGroupID
|         allows
|                 description,
|                 member,
|                 ntGroupID,
|                 eNTGroupAttributes,
|                 ntGroupCreateNewGroup,
|                 ntGroupDeleteGroup,
|                 ntGroupType,
|                 principalPtr

| objectclass   eNTInitialPopulation
|         requires
|                 objectClass,
|                 AccountSuffix,
|                 GroupSuffix,
|                 LdifFileName,
|                 ntDomain,
|                 UserSuffix
|         allows
|                 eNetworkHostName,
|                 eNetworkPort,
|                 ntDefaultPassword

| objectclass   eNTUser
|         requires
|                 objectClass,
|                 ntDomainUserID,
|                 uid
|         allows
```

| *Figure 40 (Part 18 of 31). The schema.IBM.oc File*

```
                eNTPasswordSync,
                principalName,
                accessHint,
                accountHint,
                caption,
                configPtr,
                description,
                userCertificate,
                secretKey

objectclass   eObjectDescription
        requires
                objectClass,
                cn
        allows
                caption,
                description,
                objectClassCaption,
                validValues,
                required,
                editable,
                msgFileName

objectclass   eOperatingSystem
        requires
                objectClass
        allows
                osType,
                otherOSTypeDescription,
                distributedOS,
                version,
                currentTimeZone,
                numberOfLicensedUsers,
                maxNumberOfProcesses,
                totalSwapSpaceSize,
                sizeStoredInPagingFiles,
                applSystemHint

objectclass   ePasswordGenerator
        requires
                objectClass,
                cn,
                secretKey
        allows
                caption,
```

Figure 40 (Part 19 of 31). The schema.IBM.oc File

```
|                 description,
|                 member

|  objectclass   ePasswordPolicy
|          requires
|                  objectClass,
|                  cn
|          allows
|                  passwordMinAge,
|                  passwordMaxAge,
|                  passwordMinLength,
|                  passwordMinAlphaChars,
|                  passwordMinOtherChars,
|                  passwordMaxRepeatedChars,
|                  passwordMinDiffChars,
|                  passwordRegistry,
|                  passwordCheckMethods,
|                  passwordDictFiles,
|                  passwordReuseNum,
|                  passwordTimeReuse,
|                  passwordExpireTime,
|                  maxFailedLogins,
|                  numberWarnDays,
|                  timeExpireLockout

|  objectclass   ePCVideoController
|          requires
|                  objectClass

|  objectclass   ePerson
|          requires
|                  objectClass
|          allows
|                  accessHint,
|                  accountHint,
|                  audio,
|                  businessCategory,
|                  c,
|                  carLicense,
|                  configPtr,
|                  departmentNumber,
|                  description,
|                  destinationIndicator,
|                  displayName,
|                  employeeNumber,
```

| *Figure 40 (Part 20 of 31). The schema.IBM.oc File*

```
                employeeType,
                facsimileTelephoneNumber,
                generationQualifier,
                givenName,
                homeFax,
                homePhone,
                homePostalAddress,
                initials,
                internationaliSDNNumber,
                jpegPhoto,
                l,
                labeledURI,
                mail,
                manager,
                middleName,
                mobile,
                o,
                organizationalStatus,
                otherMailbox,
                ou,
                pager,
                personalTitle,
                photo,
                physicalDeliveryOfficeName,
                postalAddress,
                postalCode,
                postOfficeBox,
                preferredDeliveryMethod,
                preferredLanguage,
                registeredAddress,
                roomNumber,
                secretary,
                seeAlso,
                st,
                street,
                telephoneNumber,
                teletexTerminalIdentifier,
                telexNumber,
                thumbNailLogo,
                thumbNailPhoto,
                title,
                uid,
                uniqueIdentifier,
                userCertificate,
                userPassword,
```

Figure 40 (Part 21 of 31). The schema.IBM.oc File

```
|                 userPKCS12,
|                 userSMIMECertificate,
|                 x121address,
|                 x500UniqueIdentifier

| objectclass   ePhysicalMemory
|         requires
|                 objectClass

| objectclass   ePointingDevice
|         requires
|                 objectClass
|         allows
|                 IRQNumber,
|                 doubleClickRate,
|                 headedness

| objectclass   eProcessor
|         requires
|                 objectClass
|         allows
|                 processorNumber,
|                 family

| objectclass   eProcessorCard
|         requires
|                 objectClass
|         allows
|                 family

| objectclass   eProperty
|         requires
|                 objectClass
|         allows
|                 binProperty,
|                 binPropertyType,
|                 cesProperty,
|                 cesPropertyType,
|                 cisProperty,
|                 cisPropertyType,
|                 propertyType

| objectclass   ePropertySet
|         requires
|                 objectClass
```

| *Figure 40 (Part 22 of 31). The schema.IBM.oc File*

```
        allows
                configPtr

objectclass   eSAP
        requires
                objectClass
        allows
                labeledURI,
                sapName,
                serviceHint

objectclass   eSCSIController
        requires
                objectClass
        allows
                SCSIControllerIndex,
                driverName,
                logicalUnit,
                deviceMap,
                hardwareVersion,
                manufacturer

objectclass   eService
        requires
                objectClass
        allows
                startMode,
                startupParameters,
                sapPtr,
                serviceName

objectclass   eSoftware
        requires
                objectClass
        allows
                software,
                softwareElementState,
                softwareElementID,
                vendor,
                identifyingNumber,
                version,
                osType,
                otherOSTypeDescription,
                manufacturer,
                buildNumber,
```

Figure 40 (Part 23 of 31). The schema.IBM.oc File

```
|                 serialNumber,
|                 codeSet,
|                 identificationCode,
|                 languageEdition,
|                 labeledURI,
|                 localPath,
|                 supportingFiles,
|                 applSoftwareHint,
|                 applSystemHint

| objectclass   eSoftwareMaintenanceUnit
|         requires
|                 objectClass
|         allows
|                 coverLetterStatus,
|                 actionPending,
|                 IPLAction,
|                 maintenanceUnitForSoftware

| objectclass   eSystem
|         requires
|                 objectClass
|         allows
|                 sys,
|                 systemRoles,
|                 primaryOwnerContact,
|                 primaryOwnerName,
|                 nameFormat

| objectclass   eTargetAdapter
|         requires
|                 objectClass,
|                 javaClassName,
|                 taName,
|                 tsType
|         allows
|                 caption,
|                 description,
|                 jarFileName,
|                 osType,
|                 propertyType,
|                 msgFileName

| objectclass   eTargetRecord
|         requires
```

| *Figure 40 (Part 24 of 31). The schema.IBM.oc File*

```
|                 objectClass,
|                 cn
|         allows
|                 uid,
|                 tsType,
|                 accountService,
|                 targetService,
|                 targetAdapter,
|                 reqIdentifier,
|                 optIdentifier,
|                 passwordGenerator,
|                 launchable,
|                 prereqTarget,
|                 autoInitiate,
|                 autoTerminate,
|                 secretKey

| objectclass   eTargetService
|         requires
|                 objectClass,
|                 tsName,
|                 tsType
|         allows
|                 caption,
|                 description,
|                 authenticationType,
|                 reqIdentifier,
|                 optIdentifier

| objectclass   eTargetServiceType
|         requires
|                 objectClass,
|                 tsType
|         allows
|                 caption,
|                 description,
|                 authenticationType,
|                 capability,
|                 reqIdentifierName,
|                 optIdentifierName,
|                 msgFileName

| objectclass   eUser
|         requires
|                 objectClass
```

Figure 40 (Part 25 of 31). The schema.IBM.oc File

```
|         allows
|                 principalName,
|                 userCertificate,
|                 configPtr,
|                 accessHint,
|                 accountHint

| objectclass   INamingService
|         requires
|                 objectClass,
|                 TypelessRDN,
|                 bt,
|                 oref
|         allows
|                 sequenceNumber

| objectclass   linkedContainer
|         requires
|                 objectClass
|         allows
|                 nextContainerDN

| objectclass   namedACL
|         requires
|                 objectClass

| objectclass   OS400Card
|         requires
|                 objectClass
|         allows
|                 OS400CardCategory,
|                 OS400CardFamilyLevel

| objectclass   OS400MCIApplication
|         requires
|                 objectClass
|         allows
|                 mciHDWCollectDateTime,
|                 mciHDWCollectVersion,
|                 mciSFWCollectDateTime,
|                 mciSFWCollectVersion,
|                 mciPTFCollectDateTime,
|                 mciPTFCollectVersion

| objectclass   OS400PTF
```

| *Figure 40 (Part 26 of 31). The schema.IBM.oc File*

```
|         requires
|                 objectClass
|         allows
|                 OS400PTFMax,
|                 OS400PTFMin,
|                 OS400PTFSaveFileStatus,
|                 OS400PTFSupersedingPTFId,
|                 OS400ProductID

| objectclass   OS400Software
|         requires
|                 objectClass
|         allows
|                 OS400ProductOption,
|                 OS400Level,
|                 OS400SupportedState

| objectclass   publisher
|         requires
|                 objectClass,
|                 publisherName
|         allows
|                 publisherType

| objectclass   racfbase
|         requires
|                 objectClass,
|                 sysplex

| objectclass   racfBaseCommon
|         requires
|                 objectClass
|         allows
|                 racfAuthorizationDate,
|                 racfOwner,
|                 racfInstallationData,
|                 racfDatasetModel

| objectclass   racfCicsSegment
|         requires
|                 objectClass
|         allows
|                 racfOperatorClass,
|                 racfOperatorIdentification,
|                 racfOperatorPriority,
```

Figure 40 (Part 27 of 31). The schema.IBM.oc File

```
|                 racfOperatorReSignon,
|                 racfTerminalTimeout

| objectclass   racfDCESegment
|         requires
|                 objectClass
|         allows
|                 racfDCEAutoLogin,
|                 racfDCEHomeCell,
|                 racfDCEHomeCellUUID,
|                 racfDCEPrincipal,
|                 racfDCEUUID

| objectclass   racfGroup
|         requires
|                 objectClass,
|                 racfid
|         allows
|                 racfSuperiorGroup,
|                 racfGroupNoTermUAC,
|                 racfSubGroupName,
|                 racfGroupUserAccess,
|                 racfGroupUserids

| objectclass   racfGroupOmvsSegment
|         requires
|                 objectClass
|         allows
|                 racfOmvsGroupId

| objectclass   racfGroupOvmSegment
|         requires
|                 objectClass
|         allows
|                 racfOvmGroupId

| objectclass   racfLanguageSegment
|         requires
|                 objectClass
|         allows
|                 racfPrimaryLanguage,
|                 racfSecondaryLanguage

| objectclass   racfNetviewSegment
|         requires
```

Figure 40 (Part 28 of 31). The schema.IBM.oc File

```
|                 objectClass
|         allows
|                 racfNetviewInitialCommand,
|                 racfDefaultConsoleName,
|                 racfCTLKeyword,
|                 racfMSGRCVRKeyword,
|                 racfNetviewOperatorClass,
|                 racfDomains,
|                 racfNGMFADMKeyword

| objectclass   racfOperparmSegment
|         requires
|                 objectClass
|         allows
|                 racfStorageKeyword,
|                 racfAuthKeyword,
|                 racfMformKeyword,
|                 racfLevelKeyword,
|                 racfMonitorKeyword,
|                 racfRoutcodeKeyword,
|                 racfLogCommandResponseKeyword,
|                 racfMGIDKeyword,
|                 racfDOMKeyword,
|                 racfKEYKeyword,
|                 racfCMDSYSKeyword,
|                 racfUDKeyword,
|                 racfMscopeSystems,
|                 racfAltGroupKeyword,
|                 racfAutoKeyword

| objectclass   racfProfileType
|         requires
|                 objectClass,
|                 profileType

| objectclass   racfUser
|         requires
|                 objectClass,
|                 racfid
|         allows
|                 racfAttributes,
|                 racfPassword,
|                 racfPasswordChangeDate,
|                 racfPasswordInterval,
|                 racfProgrammerName,
```

| *Figure 40 (Part 29 of 31). The schema.IBM.oc File*

```
|                 racfDefaultGroup,
|                 racfLastAccess,
|                 racfSecurityLabel,
|                 racfSecurityCategoryList,
|                 racfRevokeDate,
|                 racfResumeDate,
|                 racfLogonDays,
|                 racfLogonTime,
|                 racfClassName,
|                 racfConnectGroupName,
|                 racfConnectGroupAuthority,
|                 racfConnectGroupUACC,
|                 racfSecurityLevel

| objectclass   racfUserOmvsSegment
|         requires
|                 objectClass
|         allows
|                 racfOmvsUid,
|                 racfOmvsHome,
|                 racfOmvsInitialProgram

| objectclass   racfUserOvmSegment
|         requires
|                 objectClass
|         allows
|                 racfOvmUid,
|                 racfOvmHome,
|                 racfOvmInitialProgram,
|                 racfOvmFileSystemRoot,
|                 racfOvmHomeUUID

| objectclass   racfWorkAttrSegment
|         requires
|                 objectClass
|         allows
|                 racfWorkAttrUsername,
|                 racfBuilding,
|                 racfDepartment,
|                 racfRoom,
|                 racfAddressLine1,
|                 racfAddressLine2,
|                 racfAddressLine3,
|                 racfAddressLine4,
|                 racfWorkAttrAccountNumber
```

| *Figure 40 (Part 30 of 31). The schema.IBM.oc File*

```
| objectclass   resourceLimits
|         requires
|                 objectClass

| objectclass   SAFDfpSegment
|         requires
|                 objectClass
|         allows
|                 SAFDfpDataApplication,
|                 SAFDfpDataClass,
|                 SAFDfpManagementClass,
|                 SAFDfpStorageClass

| objectclass   SAFTsoSegment
|         requires
|                 objectClass
|         allows
|                 SAFAccountNumber,
|                 SAFDestination,
|                 SAFHoldClass,
|                 SAFJobClass,
|                 SAFMessageClass,
|                 SAFDefaultLoginProc,
|                 SAFLogonSize,
|                 SAFMaximumRegionSize,
|                 SAFDefaultSysoutClass,
|                 SAFUserdata,
|                 SAFDefaultUnit,
|                 SAFTsoSecurityLabel,
|                 SAFDefaultCommand

| # ------------------------------------------------------------------
| # WARNING:  Do not alter the object class definitions in this file.
| # ------------------------------------------------------------------
```

| *Figure 40 (Part 31 of 31). The schema.IBM.oc File*

The schema.user.at File

```
# ------------------------------------------------------------------
# This file is shipped in code page IBM-1047 and must remain in
# code page IBM-1047.
# ------------------------------------------------------------------
#/*************************************************************
# *
# * Licensed Materials - Property of IBM
# * 5647-A01
# * (C) Copyright IBM Corp. 1999
# *
# *************************************************************/
# ------------------------------------------------------------------
# This is the LDAP Server externally-defined and user-updateable
#   Attribute Type Definition file for OS/390.
#
# WARNING:  Do not alter the attribute type definitions in this file.
#
# This file can be MODIFIED to ADD attribute types specific to your
#   organization.
# ------------------------------------------------------------------

# NOTE: The LDAP Server depends upon the definitions of the commonName
#       attribute type.  Do not remove this attribute type from the
#       configuration of the LDAP Server.
attribute   cn
            commonName                cis    cn               128 normal
# NOTE: Before using this attribute, See the Migration
# section of the documentation.
#attribute   cn
#            commonName                cis    cn               256 normal

# NOTE: The LDAP Server depends upon the definitions of the member
#       attribute type.  Do not remove this attribute type from the
#       configuration of the LDAP Server.
attribute   member                    dn     member          1000 normall

# NOTE: The LDAP Server depends upon the definitions of the objectClass
#       attribute type.  Do not remove this attribute type from the
#       configuration of the LDAP Server.
attribute   objectClass               cis    objectClass      128 normal

# NOTE: The LDAP Server depends upon the definitions of the userPassword
#       attribute type.  Do not remove this attribute type from the
#       configuration of the LDAP Server.
attribute   userPassword              bin    userPassword     128 critical
```

Figure 41 (Part 1 of 7). The schema.user.at File

```
| attribute   abstract                  cis    abstract          500 normal
| attribute   actionDate                cis    actionDate         30 normal
| attribute   associatedDomain          cis    associatedDomain  128 normal
| attribute   associatedName            dn     associatedName   1000 normal
| attribute   audio                     bin    audio          250000 normal
| attribute   authorityRevocationList   bin    authRevocationLst 250000 critical
| attribute   billingAccount            cis    billingAccount     20 normal
| attribute   billingCountry            cis    billingCountry      2 normal
| attribute   buildingName              cis    buildingName      256 normal

| # NOTE: The LDAP Server depends upon the definitions of the businessCategory
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute   businessCategory          cis    businessCategory  128 normal

| attribute   c
|             countryName               cis    c                 128 normal
| attribute   caCertificate             bin    cACertificate  250000 critical
| attribute   carLicense                cis    carLicense        128 normal
| attribute   certificateRevocationList  bin    certRevocationLst 250000 critical
| attribute   changedSince              cis    changedSince       11 normal
| attribute   co
|             friendlyCountryName       cis    co                128 normal
| attribute   codePage                  cis    codePage           11 normal
| attribute   crossCertificatePair      bin    crossCertPair  250000 critical
| attribute   dc
|             domainComponent           cis    dc                 64 normal
| attribute   deliveryFormat            cis    deliveryFormat     11 normal
| attribute   deltaRevocationList       bin    deltRevocationLst 250000 critical
| attribute   departmentNumber          cis    departmentNumber  128 normal

| # NOTE: The LDAP Server depends upon the definitions of the description
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute   description               cis    description      1024 normal

| attribute   destinationIndicator      cis    destIndicator     128 normal
| attribute   directoryOperationName    dn     dirOperationNm   1000 normal
| attribute   directoryOperationString  cis    dirOperationStr  1024 normal
| attribute   ditRedirect               dn     dITRedirect      1000 normal
| attribute   dmdName                   cis    dmdName          1000 normal
| attribute   dn
|             distinguishedName         dn     dn               1000 normal
| attribute   dnQualifier               cis    dnQualifier       128 normal
```

| *Figure 41 (Part 2 of 7). The schema.user.at File*

```
| attribute   dnsRecord                  cis    dnsRecord         128 normal
| attribute   documentAuthor             dn     documentAuthor   1000 normal
| attribute   documentAuthorCommonName   cis    docAuthorCN       128 normal
| attribute   documentAuthorSurName      cis    docAuthorSN       128 normal
| attribute   documentIdentifier         cis    documentIdent     256 normal
| attribute   documentLocation           cis    documentLocation  256 normal
| attribute   documentPublisher          cis    documentPublisher 256 normal
| attribute   documentStore              cis    documentStore     128 normal
| attribute   documentTitle              cis    documentTitle     256 normal
| attribute   documentVersion            cis    documentVersion   256 normal
| attribute   drink
|             favouriteDrink             cis    drink             256 normal
| attribute   dsAQuality                 ces    dSAQuality       5000 normal
| attribute   emailFormat                cis    emailFormat        11 normal
| attribute   employeeNumber             cis    employeeNumber     20 normal
| attribute   employeeType               cis    employeeType      128 normal
| attribute   enhancedSearchGuide        ces    enhancedGuide    5000 normal
| attribute   facsimileTelephoneNumber
|             fax                        tel    fax                32 normal

| attribute   generationQualifier        cis    generationQualif   10 normal
| # NOTE: Before using this attribute, See the Migration
| # section of the documentation.
| #attribute   generationQualifier        cis    generationQualif   20 normal

| attribute   geographicalCoverage       cis    geogrCoverage     256 normal
| attribute   givenName                  cis    givenName         128 normal
| attribute   homeFax                    tel    homeFax            32 sensitive
| attribute   homePhone                  tel    homePhone          32 sensitive
| attribute   homePostalAddress          cis    homePostalAddress 500 sensitive
| attribute   host
|             hostName                   cis    host              256 normal

| attribute   houseIdentifier            cis houseIdentifier  32768  normal
| # NOTE: Before using this attribute, See the Migration
| # section of the documentation.
| #attribute   houseIdentifier           cis    houseIdentifier 32700 normal

| attribute   IGNCodePage                cis    IGNCodePage        10 normal
| attribute   iGNFlags                   cis    iGNFlags          128 normal
| attribute   info                       cis    info             2048 normal
| attribute   initials                   cis    initials           20 normal
| attribute   internationaliSDNNumber    ces    iSDNNumber         16 normal
| attribute   janetMailbox               cis    janetMailbox      256 normal
| attribute   javaClassName              ces    javaClassNm       256 normal
```

| *Figure 41 (Part 3 of 7). The schema.user.at File*

```
| attribute   javaSerializedObject       bin    javaSerObject   50000 normal
| attribute   jpegPhoto                  bin    jpegPhoto      250000 normal
| attribute   keywords                   cis    keywords          256 normal

| attribute   knowledgeInformation       cis    knowledgeInfo   32768 normal
| # NOTE: Before using this attribute, See the Migration
| # section of the documentation.
| #attribute   knowledgeInformation       cis    knowledgeInfo   32700 normal

| attribute   l
|             localityName               cis    l                 128 normal

| attribute   labeledURI                 ces    labeledURI        100 normal
| # NOTE: Before using this attribute, See the Migration
| # section of the documentation.
| #attribute   labeledURI                 ces    labeledURI      32700 normal

| attribute   languageCode               cis    languageCode        5 normal
| attribute   ldapBaseObject             cis    ldapBaseObject   1024 normal
| attribute   ldapFilter                 cis    ldapFilter       1024 normal
| attribute   ldapOperation              cis    ldapOperation      11 normal
| attribute   localUserid                cis    localUserid       256 normal
| attribute   mail
|             rfc822mailbox              cis    mail              256 normal
| attribute   mailPreferenceOption       cis    mailPrefOption     40 normal
| attribute   manager                    dn     manager          1000 normal
| attribute   memberCertificateDescription ces  memberCertificate 1000 normal
| attribute   membership                 cis    membership        256 normal
| attribute   memberURL                  ces    memberURL       32700 normal
| attribute   middleName                 cis    middleName        128 normal
| attribute   mobile
|             mobileTelephoneNumber      tel    mobile             32 normal
| attribute   name                       cis    name            32768 normal
| attribute   ntGroupCreateNewGroup      cis    ntGroupCreateNewG 1024 normal
| attribute   ntGroupDeleteGroup         cis    ntGroupDeleteGrou 1024 normal
| attribute   ntGroupID                  cis    ntGroupID        1024 normal
| attribute   ntGroupType                cis    ntGroupType        64 normal
| attribute   ntUserAcctExpires          cis    ntUserAcctExpires 1024 normal
| attribute   ntUserCountryCode          cis    ntUserCountryCode  64 normal
| attribute   NTUserHomeDir              cis    NTUserHomeDir   32700 normal
| attribute   ntUserHomeDirDrive         cis    ntUserHomeDirDriv  64 normal
| attribute   ntUserLastLogoff           cis    ntUserLastLogoff 1024 normal
| attribute   ntUserLastLogon            cis    ntUserLastLogon  1024 normal
| attribute   ntUserLogonServer          cis    ntUserLogonServer 1024 normal
| attribute   ntUserParms                cis    ntUserParms     32700 normal
```

Figure 41 (Part 4 of 7). The schema.user.at File

```
| attribute   ntUserProfile              cis    ntUserProfile    1024 normal
| attribute   ntUserScriptPath           cis    ntUserScriptPath 1024 normal
| attribute   ntUserUsrComment           cis    ntUserUsrComment 1024 normal

| # NOTE: The LDAP Server depends upon the definitions of the organizationName
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute   o
|             organizationName           cis    o                 128 normal

| attribute   obsoletedByDocument        cis    obsoletedByDoc    256 normal
| attribute   obsoletesDocument          cis    obsoletesDoc      256 normal
| attribute   organizationalStatus       cis    orgStatus         256 normal
| attribute   otherMailbox               cis    otherMailbox       40 normal

| # NOTE: The LDAP Server depends upon the definitions of the organizationalUnit
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute   ou
|             organizationalUnit         cis    ou                128 normal

| # NOTE: The LDAP Server depends upon the definitions of the owner
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute   owner                      dn     owner            1000 normal

| attribute   pager
|             pagerTelephoneNumber       tel    pager              32 normal
| attribute   passwordMaxAge             cis    passwordMaxAge     11 normal
| attribute   passwordMinAge             cis    passwordMinAge     11 normal
| attribute   passwordMinLength          cis    passwordMinLength  11 normal
| attribute   performanceFrequency       cis    perfFrequency       5 normal
| attribute   performedDate              cis    performedDate      30 normal
| attribute   personalSignature          bin    personalSignature 50000 normal
| attribute   personalTitle              cis    personalTitle      50 normal
| attribute   photo                      bin    photo          250000 normal
| attribute   physicalDeliveryOfficeName cis    physicalDelivery  128 normal
| attribute   postalAddress              cis    postalAddress     500 normal
| attribute   postalCode                 cis    postalCode         40 normal
| attribute   postOfficeBox              cis    postOfficeBox      40 normal
| attribute   preferredDeliveryMethod    cis    prefDeliveryMeth 1000 normal
| attribute   preferredLanguage          cis    preferredLanguage 128 normal
| attribute   preferredTechnicalFormats  cis    prefTechFormat    256 normal
| attribute   presentationAddress        ces    presentationAddr 1000 normal
| attribute   productOrService           cis    productOrService  256 normal
```

| *Figure 41 (Part 5 of 7). The schema.user.at File*

```
| attribute   protocolInformation        bin    protocolInfo     5000 normal
| attribute   publisherName              dn     publisherName    1000 normal

| attribute   registeredAddress          cis    registeredAddress 500 normal
| # NOTE: Before using this attribute, See the Migration
| # section of the documentation.
| #attribute   registeredAddress          cis    registeredAddress 5000 normal

| attribute   roleOccupant               dn     roleOccupant     1000 normal
| attribute   roomNumber                 cis    roomNumber        256 normal
| attribute   searchGuide                ces    searchGuide      5000 normal
| attribute   secretary                  dn     secretary        1000 normal

| # NOTE: The LDAP Server depends upon the definitions of the seeAlso
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute   seeAlso                    dn     seeAlso          1000 normal

| attribute   serialNumber               cis    serialNumber       64 normal
| attribute   singleLevelQuality         ces    singleLevelQual  5000 normal
| attribute   sizeLimit                  cis    sizeLimit          11 normal

| # NOTE: The LDAP Server depends upon the definitions of the surName
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute   sn
|             surName                    cis    sn                128 normal

| attribute   st
|             stateOrProvince            cis    st                128 normal
| attribute   street
|             streetAddress              cis    street            128 normal
| attribute   subject                    cis    subject           100 normal
| attribute   subtreeMaximumQuality      ces    subtreeMaxQuality 5000 normal
| attribute   subtreeMinimumQuality      ces    subtreeMinQuality 5000 normal
| attribute   supportedAlgorithms        bin    supportedAlgor 250000 normal
| attribute   supportedApplicationContext cis    supportAppContext 1000 normal

| # NOTE: The LDAP Server depends upon the definitions of the telephoneNumber
| #       attribute type.  Do not remove this attribute type from the
| #       configuration of the LDAP Server.
| attribute   telephoneNumber            tel    telephoneNumber    32 normal

| attribute   teletexTerminalIdentifier  cis    teletexTerminalId 1000 normal
| attribute   telexNumber
```

| *Figure 41 (Part 6 of 7). The schema.user.at File*

```
|             telexeNumber               cis    telexNumber        28 normal
|  attribute   textEncodedOrAddress      cis    textEncodedOrAddr 256 normal
|  attribute   thumbNailLogo             bin    thumbnailLogo  250000 normal
|  attribute   thumbNailPhoto            bin    thumbnailPhoto 250000 normal
|  attribute   title                     cis    title             128 normal
|  attribute   ttl
|              timeToLive                cis    ttl                11 normal
|  attribute   uid                       cis    uid               256 normal
|  attribute   uniqueIdentifier          cis    uniqueIdentifier  128 normal
|  attribute   uniqueMember              dn     uniqueMember     1000 normal
|  attribute   updatedByDocument         cis    updatedByDocument 256 normal
|  attribute   updatesDocument           cis    updatesDocument   256 normal
|  attribute   url                       ces    url               100 normal
|  attribute   userCertificate           bin    userCertificate 250000 critical
|  attribute   userClass                 cis    userClass         256 normal
|  attribute   userSMIMECertificate      bin    userSMIMECertific 250000 normal
|  attribute   validFrom                 cis    validFrom          30 normal
|  attribute   validTo                   cis    validTo            30 normal
|  attribute   videoTelephoneNumber      tel    videoTelNbr        32 normal
|  attribute   wWWURL                    ces    wWWURL            512 normal
|  attribute   x121address               ces    x121Address        15 normal
|  attribute   x500UniqueIdentifier      bin    x500UniqueId      128 normal

|  # ------------------------------------------------------------------
|  # WARNING:  Do not alter the attribute type definitions in this file.
|  #
|  # This file can be MODIFIED to ADD attribute types specific to your
|  #   organization.
|  # ------------------------------------------------------------------
```

| *Figure 41 (Part 7 of 7). The schema.user.at File*

The schema.user.oc File

```
# ------------------------------------------------------------------
# This file is shipped in code page IBM-1047 and must remain in
# code page IBM-1047.
# ------------------------------------------------------------------
#/*************************************************************
# *
# * Licensed Materials - Property of IBM
# * 5647-A01
# * (C) Copyright IBM Corp. 1999
# *
# *************************************************************/
# ------------------------------------------------------------------
# This is the LDAP Server externally-defined and user-updateable
#   Object Class Definition file for OS/390.
#
# WARNING:  Do not alter the object class definitions in this file.
#
# This file can be MODIFIED to ADD object classes specific to your
#   organization.
# ------------------------------------------------------------------

objectclass   account
        requires
                objectClass,
                uid
        allows
                description,
                host,
                l,
                o,
                ou,
                seeAlso

objectclass   applicationEntity
        requires
                objectClass,
                cn,
                presentationAddress
        allows
                supportedApplicationContext,
                seeAlso,
                ou,
                o,
                l,
                description
```

Figure 42 (Part 1 of 25). The schema.user.oc File

```
| objectclass   applicationProcess
|         requires
|                 objectClass,
|                 cn
|         allows
|                 description,
|                 l,
|                 ou,
|                 seeAlso

| objectclass   BBOPIRAliasDef
|         requires
|                 objectClass,
|                 TypelessRDN,
|                 bbop_def_kind
|         allows
|                 bbop_abs_name,
|                 bbop_def_in,
|                 bbop_id,
|                 bbop_name,
|                 bbop_orig_td,
|                 bbop_refer,
|                 bbop_version

| objectclass   BBOPIRAttributeDef
|         requires
|                 objectClass,
|                 TypelessRDN,
|                 bbop_def_kind
|         allows
|                 bbop_abs_name,
|                 bbop_def_in,
|                 bbop_id,
|                 bbop_mode,
|                 bbop_name,
|                 bbop_type_def,
|                 bbop_version

| objectclass   BBOPIRConstantDef
|         requires
|                 objectClass,
|                 bbop_def_kind,
|                 TypelessRDN
|         allows
```

| *Figure 42 (Part 2 of 25). The schema.user.oc File*

```
|                 bbop_abs_name,
|                 bbop_def_in,
|                 bbop_id,
|                 bbop_name,
|                 bbop_type_def,
|                 bbop_value,
|                 bbop_version

| objectclass   BBOPIREnumDef
|         requires
|                 objectClass,
|                 TypelessRDN,
|                 bbop_def_kind
|         allows
|                 bbop_abs_name,
|                 bbop_def_in,
|                 bbop_id,
|                 bbop_members,
|                 bbop_name,
|                 bbop_refer,
|                 bbop_version

| objectclass   BBOPIRExceptionDef
|         requires
|                 objectClass,
|                 TypelessRDN,
|                 bbop_def_kind
|         allows
|                 bbop_abs_name,
|                 bbop_def_in,
|                 bbop_id,
|                 bbop_members,
|                 bbop_name,
|                 bbop_refer,
|                 bbop_version

| objectclass   BBOPIRInterfaceDef
|         requires
|                 objectClass,
|                 TypelessRDN,
|                 bbop_def_kind
|         allows
|                 bbop_abs_name,
|                 bbop_base_int,
|                 bbop_def_in,
```

| *Figure 42 (Part 3 of 25). The schema.user.oc File*

```
|                 bbop_id,
|                 bbop_name,
|                 bbop_refer,
|                 bbop_version

| objectclass   BBOPIRModuleDef
|         requires
|                 objectClass,
|                 TypelessRDN,
|                 bbop_def_kind
|         allows
|                 bbop_id,
|                 bbop_name,
|                 bbop_version,
|                 bbop_def_in,
|                 bbop_abs_name

| objectclass   BBOPIROperationDef
|         requires
|                 objectClass,
|                 TypelessRDN,
|                 bbop_def_kind
|         allows
|                 bbop_id,
|                 bbop_name,
|                 bbop_version,
|                 bbop_def_in,
|                 bbop_abs_name,
|                 bbop_mode,
|                 bbop_res_def,
|                 bbop_params,
|                 bbop_contexts,
|                 bbop_except

| objectclass   BBOPIRRepository
|         requires
|                 objectClass,
|                 TypelessRDN,
|                 bbop_def_kind

| objectclass   BBOPIRRepositoryId
|         requires
|                 objectClass,
|                 TypelessRDN,
|                 bbop_def_kind
```

| *Figure 42 (Part 4 of 25). The schema.user.oc File*

```
|         allows
|                 bbop_abs_name,
|                 bbop_pkstring

| objectclass   BBOPIRStructDef
|         requires
|                 objectClass,
|                 TypelessRDN,
|                 bbop_def_kind
|         allows
|                 bbop_id,
|                 bbop_name,
|                 bbop_version,
|                 bbop_def_in,
|                 bbop_abs_name,
|                 bbop_members,
|                 bbop_refer

| objectclass   BBOPIRUnionDef
|         requires
|                 objectClass,
|                 TypelessRDN,
|                 bbop_def_kind
|         allows
|                 bbop_id,
|                 bbop_name,
|                 bbop_version,
|                 bbop_abs_name,
|                 bbop_def_in,
|                 bbop_disc_td,
|                 bbop_members,
|                 bbop_refer

| objectclass   certificationAuthority
|         requires
|                 objectClass,
|                 authorityRevocationList,
|                 caCertificate,
|                 certificateRevocationList
|         allows
|                 crossCertificatePair

| objectclass   certificationAuthority-V2
|         requires
|                 objectClass,
```

| *Figure 42 (Part 5 of 25). The schema.user.oc File*

```
|                 authorityRevocationList,
|                 certificateRevocationList,
|                 caCertificate
|         allows
|                 crossCertificatePair,
|                 deltaRevocationList

| objectclass   country
|         requires
|                 objectClass,
|                 c
|         allows
|                 searchGuide,
|                 description

| objectclass   cRLDistributionPoint
|         requires
|                 objectClass,
|                 cn
|         allows
|                 certificateRevocationList,
|                 authorityRevocationList,
|                 deltaRevocationList

| objectclass   cRLDistributionPoint
|         requires
|                 objectClass,
|                 cn
|         allows
|                 certificateRevocationList,
|                 authorityRevocationList,
|                 deltaRevocationList

| objectclass   device
|         requires
|                 objectClass,
|                 cn
|         allows
|                 description,
|                 l,
|                 o,
|                 ou,
|                 owner,
|                 seeAlso,
|                 serialNumber
```

| *Figure 42 (Part 6 of 25). The schema.user.oc File*

```
| objectclass   directoryOperation
|         requires
|                 objectClass,
|                 cn
|         allows
|                 ldapOperation,
|                 ldapBaseObject,
|                 ldapFilter,
|                 directoryOperationString,
|                 iGNFlags,
|                 description,
|                 sizeLimit

| objectclass   directoryOperationSchedule
|         requires
|                 objectClass,
|                 cn
|         allows
|                 description,
|                 directoryOperationName,
|                 deliveryFormat,
|                 emailFormat,
|                 performanceFrequency,
|                 actionDate,
|                 performedDate,
|                 changedSince,
|                 validFrom,
|                 validTo,
|                 iGNFlags,
|                 sizeLimit,
|                 mail

| objectclass   dmd
|         requires
|                 objectClass,
|                 dmdName
|         allows
|                 userPassword,
|                 searchGuide,
|                 seeAlso,
|                 businessCategory,
|                 x121address,
|                 registeredAddress,
|                 destinationIndicator,
```

| *Figure 42 (Part 7 of 25). The schema.user.oc File*

```
                preferredDeliveryMethod,
                telexNumber,
                teletexTerminalIdentifier,
                telephoneNumber,
                internationaliSDNNumber,
                facsimileTelephoneNumber,
                street,
                postalAddress,
                postalCode,
                postOfficeBox,
                physicalDeliveryOfficeName,
                st,
                l,
                description

objectclass   dNSDomain
        requires
                objectClass,
                dc
        allows
                associatedName,
                businessCategory,
                dnsRecord,
                description,
                destinationIndicator,
                facsimileTelephoneNumber,
                internationaliSDNNumber,
                l,
                o,
                physicalDeliveryOfficeName,
                postalAddress,
                postalCode,
                postOfficeBox,
                preferredDeliveryMethod,
                registeredAddress,
                searchGuide,
                seeAlso,
                st,
                street,
                telephoneNumber,
                teletexTerminalIdentifier,
                telexNumber,
                userPassword,
                x121address
```

Figure 42 (Part 8 of 25). The schema.user.oc File

```
objectclass   document
        requires
                objectClass,
                documentIdentifier
        allows
                abstract,
                audio,
                documentAuthorCommonName,
                documentAuthorSurName,
                cn,
                ditRedirect,
                description,
                documentAuthor,
                documentLocation,
                documentPublisher,
                documentStore,
                documentTitle,
                documentVersion,
                info,
                jpegPhoto,
                keywords,
                l,
                lastModifiedBy,
                lastModifiedTime,
                manager,
                o,
                obsoletedByDocument,
                obsoletesDocument,
                ou,
                photo,
                seeAlso,
                subject,
                uniqueIdentifier,
                updatedByDocument,
                updatesDocument
# NOTE: Before using this object class, See the Migration
# section of the documentation.
#objectclass   document
#        requires
#                objectClass,
#                documentIdentifier
#        allows
#                abstract,
#                audio,
#                documentAuthorCommonName,
```

Figure 42 (Part 9 of 25). The schema.user.oc File

```
#                documentAuthorSurName,
#                cn,
#                ditRedirect,
#                description,
#                documentAuthor,
#                documentLocation,
#                documentPublisher,
#                documentTitle,
#                documentStore,
#                documentVersion,
#                info,
#                jpegPhoto,
#                keywords,
#                l,
#                manager,
#                o,
#                obsoletedByDocument,
#                obsoletesDocument,
#                ou,
#                photo,
#                seeAlso,
#                subject,
#                uniqueIdentifier,
#                updatedByDocument,
#                updatesDocument

objectclass   documentSeries
        requires
                objectClass,
                cn
        allows
                description,
                l,
                o,
                ou,
                seeAlso,
                telephoneNumber

objectclass   domain
        requires
                objectClass,
                dc
        allows
                userPassword,
                searchGuide,
```

Figure 42 (Part 10 of 25). The schema.user.oc File

```
                seeAlso,
                businessCategory,
                x121address,
                registeredAddress,
                destinationIndicator,
                preferredDeliveryMethod,
                telexNumber,
                teletexTerminalIdentifier,
                telephoneNumber,
                internationaliSDNNumber,
                facsimileTelephoneNumber,
                street,
                postOfficeBox,
                postalAddress,
                postalCode,
                physicalDeliveryOfficeName,
                st,
                l,
                description,
                o,
                associatedName

objectclass   domainRelatedObject
        requires
                objectClass,
                associatedDomain

objectclass   dSA
        requires
                objectClass,
                presentationAddress,
                cn
        allows
                knowledgeInformation
# NOTE: Before using this object class, See the Migration
# section of the documentation.
#objectclass   dSA
#        requires
#                objectClass
#        allows
#                knowledgeInformation

objectclass   friendlyCountry
        requires
                objectClass,
```

Figure 42 (Part 11 of 25). The schema.user.oc File

```
|                 c,
|                 co
|         allows
|                 description,
|                 searchGuide

|  objectclass   groupOfCertificates
|         requires
|                 objectClass,
|                 cn
|         allows
|                 businessCategory,
|                 description,
|                 o,
|                 ou,
|                 owner,
|                 seeAlso,
|                 memberCertificateDescription

|  objectclass   groupOfNames
|         requires
|                 objectClass,
|                 cn,
|                 member
|         allows
|                 businessCategory,
|                 seeAlso,
|                 owner,
|                 ou,
|                 o,
|                 description

|  objectclass   groupOfUniqueNames
|         requires
|                 objectClass,
|                 cn,
|                 uniqueMember
|         allows
|                 businessCategory,
|                 seeAlso,
|                 owner,
|                 ou,
|                 o,
|                 description
```

| *Figure 42 (Part 12 of 25). The schema.user.oc File*

```
objectclass   groupOfURLs
        requires
                objectClass,
                cn
        allows
                memberURL,
                businessCategory,
                description,
                o,
                ou,
                owner,
                seeAlso

objectclass   iGNObject
        requires
                objectClass
        allows
                billingAccount,
                billingCountry,
                geographicalCoverage,
                IGNCodePage,
                iGNFlags,
                info,
                languageCode,
                membership,
                productOrService,
                wWWURL

objectclass   iGNPerson
        requires
                objectClass,
                uniqueIdentifier
        allows
                generationQualifier,
                givenName,
                initials,
                localUserid,
                mailPreferenceOption,
                preferredTechnicalFormats,
                textEncodedOrAddress,
                uid,
                videoTelephoneNumber,
                businessCategory,
                mail,
                mobile,
```

Figure 42 (Part 13 of 25). The schema.user.oc File

```
                pager

objectclass   inetOrgPerson
        requires
                objectClass,
                sn,
                cn
        allows
                audio,
                businessCategory,
                carLicense,
                departmentNumber,
                employeeNumber,
                employeeType,
                givenName,
                homePhone,
                homePostalAddress,
                initials,
                jpegPhoto,
                labeledURI,
                mail,
                manager,
                mobile,
                pager,
                photo,
                preferredLanguage,
                roomNumber,
                secretary,
                uid,
                userCertificate,
                userSMIMECertificate,
                x500UniqueIdentifier

objectclass   javaContainer
        requires
                objectClass,
                cn

objectclass   javaObject
        requires
                objectClass
        allows
                javaClassName,
                javaSerializedObject
```

Figure 42 (Part 14 of 25). The schema.user.oc File

```
objectclass   labeledURIObject
        requires
                objectClass
        allows
                labeledURI

objectclass   liOrganization
        requires
                objectClass,
                o
        allows
                businessCategory,
                description,
                destinationIndicator,
                facsimileTelephoneNumber,
                internationaliSDNNumber,
                l,
                physicalDeliveryOfficeName,
                postalAddress,
                postalCode,
                postOfficeBox,
                preferredDeliveryMethod,
                registeredAddress,
                searchGuide,
                seeAlso,
                st,
                street,
                telephoneNumber,
                teletexTerminalIdentifier,
                telexNumber,
                userPassword,
                x121address,
                mail,
                labeledURI,
                c,
                uniqueIdentifier,
                otherMailbox,
                thumbNailLogo,
                manager

objectclass   liPerson
        requires
                objectClass,
                cn,
                sn
```

Figure 42 (Part 15 of 25). The schema.user.oc File

```
        allows
                description,
                seeAlso,
                telephoneNumber,
                userPassword,
                mail,
                userCertificate,
                labeledURI,
                givenName,
                generationQualifier,
                o,
                l,
                c,
                personalTitle,
                initials,
                middleName,
                uniqueIdentifier,
                homePhone,
                homeFax,
                homePostalAddress,
                thumbNailLogo,
                title,
                facsimileTelephoneNumber,
                mobile,
                pager,
                postalAddress,
                ou,
                roomNumber,
                otherMailbox,
                telexNumber,
                thumbNailPhoto,
                secretary,
                manager

objectclass   locality
        requires
                objectClass
        allows
                street,
                seeAlso,
                searchGuide,
                st,
                l,
                description
```

Figure 42 (Part 16 of 25). The schema.user.oc File

```
| objectclass   newPilotPerson
|         requires
|                 objectClass,
|                 sn,
|                 cn
|         allows
|                 businessCategory,
|                 description,
|                 drink,
|                 homePhone,
|                 homePostalAddress,
|                 janetMailbox,
|                 mail,
|                 mailPreferenceOption,
|                 mobile,
|                 organizationalStatus,
|                 otherMailbox,
|                 pager,
|                 personalSignature,
|                 personalTitle,
|                 preferredDeliveryMethod,
|                 roomNumber,
|                 secretary,
|                 seeAlso,
|                 telephoneNumber,
|                 textEncodedOrAddress,
|                 uid,
|                 userClass,
|                 userPassword

| objectclass   oldQualityLabelledData
|         requires
|                 objectClass,
|                 singleLevelQuality
|         allows
|                 subtreeMaximumQuality,
|                 subtreeMinimumQuality

| objectclass   organization
|         requires
|                 objectClass,
|                 o
|         allows
|                 businessCategory,
|                 description,
```

| *Figure 42 (Part 17 of 25). The schema.user.oc File*

```
|                 destinationIndicator,
|                 facsimileTelephoneNumber,
|                 internationaliSDNNumber,
|                 l,
|                 physicalDeliveryOfficeName,
|                 postOfficeBox,
|                 postalAddress,
|                 postalCode,
|                 preferredDeliveryMethod,
|                 registeredAddress,
|                 searchGuide,
|                 seeAlso,
|                 st,
|                 street,
|                 telephoneNumber,
|                 teletexTerminalIdentifier,
|                 telexNumber,
|                 userPassword,
|                 x121Address

| objectclass   organizationalPerson
|         requires
|                 objectClass,
|                 sn,
|                 cn
|         allows
|                 description,
|                 destinationIndicator,
|                 facsimileTelephoneNumber,
|                 internationaliSDNNumber,
|                 l,
|                 ou,
|                 physicalDeliveryOfficeName,
|                 postOfficeBox,
|                 postalAddress,
|                 postalCode,
|                 preferredDeliveryMethod,
|                 registeredAddress,
|                 seeAlso,
|                 st,
|                 street,
|                 telephoneNumber,
|                 teletexTerminalIdentifier,
|                 telexNumber,
|                 title,
```

| *Figure 42 (Part 18 of 25). The schema.user.oc File*

```
|                 userPassword,
|                 x121Address
| # NOTE: Before using this object class, See the Migration
| # section of the documentation.
| #objectclass   organizationalPerson
| #        requires
| #                objectClass,
| #                sn,
| #                cn
| #        allows
| #                title,
| #                x121address,
| #                registeredAddress,
| #                destinationIndicator,
| #                preferredDeliveryMethod,
| #                telexNumber,
| #                teletexTerminalIdentifier,
| #                telephoneNumber,
| #                internationaliSDNNumber,
| #                facsimileTelephoneNumber,
| #                street,
| #                postalAddress,
| #                postalCode,
| #                postOfficeBox,
| #                physicalDeliveryOfficeName,
| #                ou,
| #                st,
| #                l,
| #                description,
| #                seeAlso

| objectclass   organizationalRole
|         requires
|                 objectClass,
|                 cn
|         allows
|                 description,
|                 destinationIndicator,
|                 facsimileTelephoneNumber,
|                 internationaliSDNNumber,
|                 l,
|                 ou,
|                 physicalDeliveryOfficeName,
|                 postalAddress,
|                 postalCode,
```

| *Figure 42 (Part 19 of 25). The schema.user.oc File*

```
|                 postOfficeBox,
|                 preferredDeliveryMethod,
|                 registeredAddress,
|                 roleOccupant,
|                 seeAlso,
|                 st,
|                 street,
|                 telephoneNumber,
|                 teletexTerminalIdentifier,
|                 telexNumber,
|                 x121address

| objectclass   organizationalUnit
|         requires
|                 objectClass,
|                 ou
|         allows
|                 userPassword,
|                 searchGuide,
|                 seeAlso,
|                 businessCategory,
|                 x121address,
|                 registeredAddress,
|                 destinationIndicator,
|                 preferredDeliveryMethod,
|                 telexNumber,
|                 teletexTerminalIdentifier,
|                 telephoneNumber,
|                 internationaliSDNNumber,
|                 facsimileTelephoneNumber,
|                 street,
|                 postOfficeBox,
|                 postalCode,
|                 postalAddress,
|                 physicalDeliveryOfficeName,
|                 st,
|                 l,
|                 description

| # NOTE: The LDAP Server depends upon the definitions of the person
| #       object class.  Do not remove this object class from the
| #       configuration of the LDAP Server.
| objectclass   person
|         requires
|                 objectClass,
```

| *Figure 42 (Part 20 of 25). The schema.user.oc File*

```
|                 cn,
|                 sn
|         allows
|                 userPassword,
|                 telephoneNumber,
|                 seeAlso,
|                 description

| objectclass   pilotDSA
|         requires
|                 objectClass,
|                 dsAQuality

| objectclass   pilotObject
|         requires
|                 objectClass
|         allows
|                 audio,
|                 ditRedirect,
|                 info,
|                 jpegPhoto,
|                 lastModifiedBy,
|                 lastModifiedTime,
|                 manager,
|                 photo,
|                 uniqueIdentifier
| # NOTE: Before using this object class, See the Migration
| # section of the documentation.
| #objectclass   pilotObject
| #        requires
| #                objectClass,
| #                cn
| #        allows
| #                audio,
| #                ditRedirect,
| #                info,
| #                jpegPhoto,
| #                manager,
| #                photo,
| #                uniqueIdentifier

| objectclass   pilotDSA
|         requires
|                 objectClass,
|                 dsAQuality
```

Figure 42 (Part 21 of 25). The schema.user.oc File

```
objectclass   pilotOrganization
        requires
                objectClass,
                ou,
                o
        allows
                buildingName,
                businessCategory,
                description,
                destinationIndicator,
                facsimileTelephoneNumber,
                internationaliSDNNumber,
                l,
                physicalDeliveryOfficeName,
                postalAddress,
                postalCode,
                postOfficeBox,
                preferredDeliveryMethod,
                registeredAddress,
                searchGuide,
                seeAlso,
                st,
                street,
                telephoneNumber,
                teletexTerminalIdentifier,
                telexNumber,
                userPassword,
                x121address

objectclass   qualityLabelledData
     requires
          objectClass,
          singleLevelQuality
     allows
          subtreeMaximumQuality,
          subtreeMinimumQuality

objectclass   qualityLabelledData
        requires
                objectClass,
                singleLevelQuality
        allows
                subtreeMaximumQuality,
                subtreeMinimumQuality
```

Figure 42 (Part 22 of 25). The schema.user.oc File

```
| objectclass   residentialPerson
|         requires
|                 objectClass,
|                 sn,
|                 cn
|         allows
|                 businessCategory,
|                 description,
|                 destinationIndicator,
|                 facsimileTelephoneNumber,
|                 internationaliSDNNumber,
|                 l,
|                 physicalDeliveryOfficeName,
|                 postOfficeBox,
|                 postalAddress,
|                 postalCode,
|                 preferredDeliveryMethod,
|                 registeredAddress,
|                 seeAlso,
|                 st,
|                 street,
|                 telephoneNumber,
|                 teletexTerminalIdentifier,
|                 telexNumber,
|                 userPassword,
|                 x121Address
| # NOTE: Before using this object class, See the Migration
| # section of the documentation.
| #objectclass   residentialPerson
| #        requires
| #                objectClass,
| #                cn,
| #                l,
| #                sn
| #        allows
| #                businessCategory,
| #                x121address,
| #                registeredAddress,
| #                destinationIndicator,
| #                preferredDeliveryMethod,
| #                telexNumber,
| #                teletexTerminalIdentifier,
| #                telephoneNumber,
| #                internationaliSDNNumber,
```

| *Figure 42 (Part 23 of 25). The schema.user.oc File*

```
#                facsimileTelephoneNumber,
#                street,
#                postalAddress,
#                postalCode,
#                postOfficeBox,
#                physicalDeliveryOfficeName,
#                st,
#                description,
#                seeAlso,
#                userPassword

objectclass   rFC822LocalPart
        requires
                objectClass,
                dc
        allows
                associatedName,
                businessCategory,
                cn,
                description,
                destinationIndicator,
                facsimileTelephoneNumber,
                internationaliSDNNumber,
                l,
                o,
                physicalDeliveryOfficeName,
                postalAddress,
                postalCode,
                postOfficeBox,
                preferredDeliveryMethod,
                registeredAddress,
                searchGuide,
                seeAlso,
                sn,
                st,
                street,
                telephoneNumber,
                teletexTerminalIdentifier,
                telexNumber,
                userPassword,
                x121address

objectclass   room
        requires
                objectClass,
```

Figure 42 (Part 24 of 25). The schema.user.oc File

```
                cn
        allows
                description,
                roomNumber,
                seeAlso,
                telephoneNumber

objectclass   strongAuthenticationUser
        requires
                objectClass,
                userCertificate

# NOTE: The LDAP Server depends upon the definitions of the top
#       object class.  Do not remove this object class from the
#       configuration of the LDAP Server.
objectclass   top
        requires
                objectClass

objectclass   userSecurityInformation
        requires
                objectClass
        allows
                supportedAlgorithms

# ---------------------------------------------------------------------
# WARNING:  Do not alter the object class definitions in this file.
#
# This file can be MODIFIED to ADD object classes specific to your
#   organization.
# ---------------------------------------------------------------------
```

Figure 42 (Part 25 of 25). The schema.user.oc File

Appendix B. Sample JCL

This appendix shows the following JCL samples:

- "Sample JCL for the LDAP Server"
- "Sample JCL for ldif2db" on page 365
- "Sample JCL for db2ldif" on page 367

Sample JCL for the LDAP Server

Following is the sample JCL provided for the LDAP Server (LDAPSRV PROC JCL).

```
//*************************************************************//
//*
//* Licensed Materials - Property of IBM
//* 5647-A01
//* (C) Copyright IBM Corp.  1997, 1998
//*
//*************************************************************//
//*************************************************************//
//*
//* Procedure for starting the LDAPSRV server
//*
//* To start server using configuration file /etc/ldap/slapd.conf
//* specify:
//*  s ldapsrv
//*
//* To start server using alternate configuration file or
//* other parameters specify:
//*  s ldapsrv,parms='options'
//* where options can be:
//*           -f filename  # alternate configuration file
//*           -d level     # debug level (65535 turns on all debugs)
//*           -p portno    # non-secure port number
//*           -s portno    # secure port number
//*
//* An alternative to the -f option is to define a CONFIG DD.
//* The remaining options are optional.  If not set, message/debug
//* levels are set to 0, non-secure port number will be 389, and
//* secure port number will be 636.  NOTE:  use of these low port
//* numbers will require that the LDAPSRV server run under a userid
```

Figure 43 (Part 1 of 2). Sample JCL for the LDAP Server

```
//* that has OpenEdition UID 0.
//*
//* In the JCL below, GLDHLQ refers to the high level qualifier
//* that was used to install the LDAP Server datasets.  This will
//* have to be customized for the installation.
//*
//*************************************************************//
//*************************************************************//
//LDAPSRV  PROC REGSIZE=64M,
//*-------------------------------------------------------------------
//* CUSTOMIZABLE SYMBOLIC PARAMETERS
//*-------------------------------------------------------------------
//  PARMS='',
//  OUTCLASS='A',
//  GLDHLQ='XXXXXX'
//*-------------------------------------------------------------------
//GO       EXEC PGM=GLDSLAPD,REGION=&REGSIZE,TIME=1440,
//         PARM=('/&PARMS >DD:SLAPDOUT 2>&1')
//*-------------------------------------------------------------------
//* STEPLIB must be customized based on install HLQ.
//*-------------------------------------------------------------------
//STEPLIB  DD DSN=&GLDHLQ..SGLDLNK,DISP=SHR
|//*-------------------------------------------------------------------
|//* Fill in and uncomment the following DD if the libraries for DB2
|//* are not in LINKLST or LPA on the system.  Modify <DB2HLQ>
|//* to be the high level qualifier of the DB2 installation
|//* on the system.
|//*-------------------------------------------------------------------
|//*         DD DSN=<DB2HLQ>.SDSNLOAD,DISP=SHR
//*-------------------------------------------------------------------
//* CONFIG can be used to specify the LDAP server config file.
//* If this DD is used, the name of the dataset must be customized
//* for the installation.
//*-------------------------------------------------------------------
//*CONFIG   DD DSN=<CONFIG.FILE.DATASET>,DISP=SHR
//*-------------------------------------------------------------------
//* ENVVAR can be used to specify any environment variables
//* If this DD is used, the name of the dataset must be customized
//* for the installation.
//*-------------------------------------------------------------------
//*ENVVAR   DD DSN=<ENVVAR.FILE.DATASET>,DISP=SHR
//*-------------------------------------------------------------------
//* DSNAOINI can be used to specify the dsnaoini dataset required by DB2.
//* Alternatively, the dsnaoini data set can be specified in the
//* configuration file.
//* If this DD is used, the name of the dataset must be customized
//* for the installation.
//*-------------------------------------------------------------------
//*DSNAOINI DD DSN=<DSNAOINI.DATASET>,DISP=SHR
//SLAPDOUT DD SYSOUT=&OUTCLASS
//SYSOUT  DD SYSOUT=&OUTCLASS
//SYSUDUMP DD SYSOUT=&OUTCLASS
//CEEDUMP  DD SYSOUT=&OUTCLASS
```

Figure 43 (Part 2 of 2). Sample JCL for the LDAP Server

Sample JCL for ldif2db

```
//LDIF2DBA JOB MSGCLASS=H
//*************************************************************//
//*
//* Licensed Materials - Property of IBM
//* 5647-A01
//* (C) Copyright IBM Corp.  1997, 1998
//*
//*************************************************************//
//*************************************************************//
//*
//* Procedure for running LDIF2DB from batch
//*
//* To start LDIF2DB using configuration file /etc/ldap/slapd.conf
//* just submit the job.
//*
//* To start LDIF2DB using an alternate configuration file or
//* other parameters specify one or more of the following
//* options in the PARMS substitution parameter in the
//* EXEC line below:
//*
//* The options can be:
//*           -f filename  # alternate configuration file
//*           -i filename  # file from which to take input
//*
//* An alternative to the -f option is to define a CONFIG DD
//* in the PROC.
//*
//* An alternative to the -i option is to define the SYSIN DD
//* card to a dataset that contains the input.
//*
//* Both the -f and -i options are optional.  If the -f option is
//* not specified and the CONFIG DD card is not specified, the
//* HFS file '/etc/ldap/slapd.conf' will be used for the config file.
//* If the -i option is not specified, then the SYSIN DD card is
//* is used for the input to the program.
//*
//* In the JCL below, GLDHLQ refers to the high level qualifier
//* that was used to install the LDAP Server datasets.  This will
//* have to be customized for the installation.
//*
//*************************************************************//
//*************************************************************//
```

Figure 44 (Part 1 of 2). Sample JCL for ldif2db

```
//LDIF2DB PROC REGSIZE=2048K,
//*-------------------------------------------------------------------
//* CUSTOMIZABLE SYMBOLIC PARAMETERS
//* Customize the parameters here for desired behavior.
//*-------------------------------------------------------------------
//  PARMS='',
//  GLDHLQ='XXXXXX'
//  OUTCLASS='A',
//LDIF2DB  EXEC PGM=GLDLD2DB,REGION=&REGSIZE,
//         PARM=('/&PARMS')
//*-------------------------------------------------------------------
//* STEPLIB must be customized based on install HLQ.
//*-------------------------------------------------------------------
//STEPLIB  DD DSN=&GLDHLQ..SGLDLNK,DISP=SHR
//*-------------------------------------------------------------------
//* Fill in and uncomment the following DD if the libraries for DB2
//* are not in LINKLST or LPA on the system.  Modify <DB2HLQ>
//* to be the high level qualifier of the DB2 installation
//* on the system.
//*-------------------------------------------------------------------
//*        DD DSN=<DB2HLQ>.SDSNLOAD,DISP=SHR
//*-------------------------------------------------------------------
//* DSNAOINI can be used to specify the DSNAOINI file needed for DB2.
//* If this DD is not used, the DSNAOINI setting must be in the
//* LDAP server configuration file.
//* If this DD is used, the name of the dataset must be customized
//* for the installation.
//*-------------------------------------------------------------------
//*DSNAOINI DD DSN=<DB2.DSNAOINI.DATASET>,DISP=SHR
//*-------------------------------------------------------------------
//* CONFIG can be used to specify the LDAP server config file.
//* If this DD is used, the name of the dataset must be customized
//* for the installation.
//*-------------------------------------------------------------------
//*CONFIG   DD DSN=<CONFIG.FILE.DATASET>,DISP=SHR
//*-------------------------------------------------------------------
//* ENVVAR can be used to specify any environment variables
//* If this DD is used, the name of the dataset must be customized
//* for the installation.
//*-------------------------------------------------------------------
//*ENVVAR   DD DSN=<ENVVAR.FILE.DATASET>,DISP=SHR
//*-------------------------------------------------------------------
//* SYSIN can be used to specify the input to this command.
//* If this DD is used, the name of the dataset must be customized
//* for the installation.
//*-------------------------------------------------------------------
//*SYSIN    DD DSN=<INPUT.LDIF.DATASET>,DISP=SHR
//SYSPRINT DD SYSOUT=&OUTCLASS
//CEEDUMP  DD SYSOUT=&OUTCLASS
//SYSERR   DD SYSOUT=&OUTCLASS
//STDOUT   DD SYSOUT=&OUTCLASS
//         PEND
//*-------------------------------------------------------------------
//GO       EXEC LDIF2DB
```

Figure 44 (Part 2 of 2). Sample JCL for ldif2db

Sample JCL for db2ldif

```
//DB2LDIFA JOB MSGCLASS=H
//*************************************************************//
//*
//* Licensed Materials - Property of IBM
//* 5647-A01
//* (C) Copyright IBM Corp.  1997, 1998
//*
//*************************************************************//
//*************************************************************//
//*
//* Procedure for running DB2LDIF from batch
//*
//* To start DB2LDIF using configuration file /etc/ldap/slapd.conf
//* just submit the job.
//*
//* To start DB2LDIF using an alternate configuration file or
//* other parameters specify one or more of the following
//* options in the PARMS substitution parameter in the
//* EXEC line below:
//*
//* The options can be:
//*           -f filename  # alternate configuration file
//*           -o filename  # file in which to store the output
//*           -s subtreeDN # root of the subtree to extract
//*
//* An alternative to the -f option is to define a CONFIG DD
//* in the PROC.
//*
//* An alternative to the -o option is to define the SYSPRINT DD
//* card to a dataset that will store the output.
//*
//* The -f and -o, and -s options are optional.  If the -f option is
//* not specified and the CONFIG DD card is not specified, the
//* HFS file '/etc/ldap/slapd.conf' will be used for the config file.
//* If the -o option is not specified, then the SYSPRINT DD card is
//* is used for the output from the program.  If the -s option is
//* not specified, all data held by the LDAP server will be printed.
//*
//* In the JCL below, GLDHLQ refers to the high level qualifier
//* that was used to install the LDAP Server datasets.  This will
//* have to be customized for the installation.
//*
//*************************************************************//
//*************************************************************//
//DB2LDIF PROC REGSIZE=2048K,
//*-------------------------------------------------------------------
//* CUSTOMIZABLE SYMBOLIC PARAMETERS
//* Customize the parameters here for desired behavior.
//*-------------------------------------------------------------------
//  PARMS='',
```

Figure 45 (Part 1 of 2). Sample JCL for db2ldif

```
//  GLDHLQ='XXXXXX'
//  OUTCLASS='A',
//DB2LDIF  EXEC PGM=GLDDB2LD,REGION=&REGSIZE,
//         PARM=('/&PARMS')
//*-------------------------------------------------------------------
//* STEPLIB must be customized based on install HLQ.
//*-------------------------------------------------------------------
//* Fill in and uncomment the following DD if the libraries for DB2
//* are not in LINKLST or LPA on the system.  Modify <DB2HLQ>
//* to be the high level qualifier of the DB2 installation
//* on the system.
//*-------------------------------------------------------------------
//*        DD DSN=<DB2HLQ>.SDSNLOAD,DISP=SHR
//*-------------------------------------------------------------------
//STEPLIB  DD DSN=&GLDHLQ..SGLDLNK,DISP=SHR
//*-------------------------------------------------------------------
//* DSNAOINI can be used to specify the DSNAOINI file needed for DB2.
//* If this DD is not used, the DSNAOINI setting must be in the
//* LDAP server configuration file.
//* If this DD is used, the name of the dataset must be customized
//* for the installation.
//*-------------------------------------------------------------------
//*DSNAOINI DD DSN=<DB2.DSNAOINI.DATASET>,DISP=SHR
//*-------------------------------------------------------------------
//* CONFIG can be used to specify the LDAP server config file.
//* If this DD is used, the name of the dataset must be customized
//* for the installation.
//*-------------------------------------------------------------------
//*CONFIG   DD DSN=<CONFIG.FILE.DATASET>,DISP=SHR
//*-------------------------------------------------------------------
//* ENVVAR can be used to specify any environment variables
//* If this DD is used, the name of the dataset must be customized
//* for the installation.
//*-------------------------------------------------------------------
//*ENVVAR   DD DSN=<ENVVAR.FILE.DATASET>,DISP=SHR
//*-------------------------------------------------------------------
//* SYSPRINT can be used to specify the location for output
//* from this command.
//* If this DD is used, the name of the dataset must be customized
//* for the installation.
//*-------------------------------------------------------------------
//*SYSPRINT DD DSN=<OUTPUT.LDIF.DATASET>,DISP=NEW
//SYSPRINT DD SYSOUT=&OUTCLASS
//CEEDUMP  DD SYSOUT=&OUTCLASS
//SYSERR   DD SYSOUT=&OUTCLASS
//STDOUT   DD SYSOUT=&OUTCLASS
//         PEND
//*-------------------------------------------------------------------
//GO       EXEC DB2LDIF
```

Figure 45 (Part 2 of 2). Sample JCL for db2ldif

Appendix C. Sample LDIF Input File

The following sample LDIF input file can be found in the **/usr/lpp/ldap/examples/sample_server** directory and is called **sample.ldif**.

```
dn: o=Your Company, c=US
objectclass: organization
o: Your Company

dn: cn=LDAP Administrator, o=Your Company, c=US
objectclass: organizationalPerson
cn: LDAP Administrator
sn: Administrator
userPassword: secret

dn: ou=Home Town, o=Your Company, c=US
ou: Home Town
objectclass: organizationalUnit
seealso: cn=Linda Carlesberg, ou=Home Town, o=Your Company, c=US

dn: ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
ou: In Flight Systems
objectclass: organizationalUnit
description: main product:Course Maker
businessCategory: aircraft
seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US

dn: ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
ou: Home Entertainment
objectclass: organizationalUnit
description: main product:TV Connection
businessCategory: Home Entertainment

dn: ou=Groups, o=Your Company, c=US
objectclass: organizationalUnit
ou: Groups

dn: cn=Bowling team, ou=Groups, o=Your Company, c=US
objectclass: groupOfNames
description: IBM Home Town Bowling Team
cn: Bowling team
```

Figure 46 (Part 1 of 12). Sample LDIF Input File

```
| owner: cn=Mary Burnnet, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| member: cn=Mary Burnnet, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| member: cn=Michael Campbell+postalcode=4609, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| member: cn=Eddie Catu, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| member: cn=Melinda Charles, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| member: cn=Al Edwards, ou=Widget Division, ou=Home Town, o=Your Company, c=US

| dn: ou=Widget Division, ou=Home Town, o=Your Company, c=US
| ou: Widget Division
| objectclass: organizationalUnit
| description: main product:Orange Widget Delux
| businessCategory: home entertainment

| dn: cn=Mary Burnnet, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Mary Burnnet
| sn: Burnnet
| telephonenumber: 1-812-855-5923
| internationaliSDNNumber: 755-5923
| facsimiletelephonenumber: 1-812-855-5923
| title: ISO Deputy, Qual. Tech
| postalcode: 1515
| seealso: cn=Linda Carlesberg, ou=Home Town, o=Your Company, c=US

| dn: cn=David Campbell, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: David Campbell
| sn: Campbell
| telephonenumber: 1-812-855-7509
| internationaliSDNNumber: 755-7509
| title: Mfg. Assembly
| seealso: cn=Mary Burnnet, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| postalcode: 1514

| dn: cn=James Campbell, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: James Campbell
| sn: Campbell
| telephonenumber: 1-812-855-8861
| internationaliSDNNumber: 755-8861
| facsimiletelephonenumber: 1-812-855-5237
| title: Home Town Adaptive Technology Center Accessibility
| seealso: cn=Mary Burnnet, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| postalcode: 4503
| telexnumber: 1-812-343-7700
```

| *Figure 46 (Part 2 of 12). Sample LDIF Input File*

```
| facsimiletelephonenumber: 755-5237

| dn: cn=Michael Campbell, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Michael Campbell
| sn: Campbell
| telephonenumber: 1-812-855-5838
| internationaliSDNNumber: 755-5838
| postalcode: 4681

| dn: cn=Michael Campbell+postalcode=4609, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Michael Campbell
| sn: Campbell
| telephonenumber: 1-812-855-7743
| title: Drill Department
| postalcode: 4609

| dn: cn=Bob Campbell, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Bob Campbell
| sn: Campbell
| telephonenumber: 1-812-855-8541
| internationaliSDNNumber: 755-8541
| title: Mechanical Ana. Thermal
| seealso: cn=Mary Burnnet, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| postalcode: 4502

| dn: cn=Bonnie Daniel, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Bonnie Daniel
| sn: Daniel
| telephonenumber: 1-812-855-7453
| internationaliSDNNumber: 755-7453
| title: RISC Manufacturing
| seealso: cn=Mary Burnnet, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| postalcode: 1515

| dn: cn=Brenda England, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Brenda England
| sn: England
| telephonenumber: 1-812-855-5231
| internationaliSDNNumber: 755-5231
| facsimiletelephonenumber: 1-812-855-5237
```

| *Figure 46 (Part 3 of 12). Sample LDIF Input File*

```
| title: Assistant to Dr. Campbell
| postalcode: 4503
| facsimiletelephonenumber: 755-5237

| dn: cn=David Delbert, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: David Delbert
| sn: Delbert
| telephonenumber: 1-812-855-8504
| internationaliSDNNumber: 523-8504
| facsimiletelephonenumber: 1-812-855-6040
| title: SWAT OS/2 Analyst
| seealso: cn=Mary Burnnet, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| postalcode: 2901
| telexnumber: 1-800-546-4646
| facsimiletelephonenumber: 755-6040

| dn: cn=Al Edwards, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Al Edwards
| sn: Edwards
| telephonenumber: 1-812-855-1370
| internationaliSDNNumber: 755-1370
| facsimiletelephonenumber: 1-812-855-5004
| title: Site Occupancy Planner
| seealso: cn=Mary Burnnet, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| postalcode: 4713

| dn: cn=Arthur Edwards, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Arthur Edwards
| sn: Edwards
| telephonenumber: 1-812-855-8474
| internationaliSDNNumber: 523-8474
| facsimiletelephonenumber: 1-812-855-6040
| title: PSP Enterprise Customer Support OS/2 SWAT Team
| seealso: cn=Mary Burnnet, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| postalcode: 2901
| telexnumber: 1-800-546-4646
| facsimiletelephonenumber: 755-6040

| dn: cn=Curtis Edwards Jr, ou=Widget Division, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Curtis Edwards Jr
| sn: Edwards
```

| *Figure 46 (Part 4 of 12). Sample LDIF Input File*

```
telephonenumber: 1-812-855-8053
internationaliSDNNumber: 755-8053
facsimiletelephonenumber: 1-812-855-7101
title: EMC TEST
postalcode: 4502
facsimiletelephonenumber: 755-7101

dn: cn=Cynthia Flowers, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
objectclass: organizationalPerson
cn: Cynthia Flowers
sn: Flowers
telephonenumber: 1-812-855-8609
internationaliSDNNumber: 755-8609
facsimiletelephonenumber: 1-812-855-8712
title: Software Contracts
postalcode: 1725
facsimiletelephonenumber: 755-8712
seealso: cn=Linda Carlesberg, ou=Home Town, o=Your Company, c=US

dn: cn=Doug Edwards, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
objectclass: organizationalPerson
cn: Doug Edwards
sn: Edwards
telephonenumber: 1-812-855-8386
internationaliSDNNumber: 755-8386
facsimiletelephonenumber: 1-812-855-8199
title: Panel Finance / Accounting
seealso: cn=Cynthia Flowers, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
postalcode: 4604
facsimiletelephonenumber: 755-8199

dn: cn=Jeffrey James, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
objectclass: organizationalPerson
cn: Jeffrey James
sn: James
telephonenumber: 1-812-855-7551
internationaliSDNNumber: 755-7551
facsimiletelephonenumber: 1-812-855-7193
title: programmer
seealso: cn=Cynthia Flowers, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
postalcode: 1033
facsimiletelephonenumber: 755-7193

dn: cn=Ron Edwards, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
objectclass: organizationalPerson
```

Figure 46 (Part 5 of 12). Sample LDIF Input File

```
| cn: Ron Edwards
| sn: Edwards
| telephonenumber: 1-812-855-4021
| internationaliSDNNumber: 755-4021
| facsimiletelephonenumber: 1-812-855-5454
| title: DEPT TECH
| seealso: cn=Cynthia Flowers, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
| postalcode: 4601
| telexnumber: 1-812-474-3783

| dn: cn=Jerry Chevy, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Jerry Chevy
| sn: Chevy
| telephonenumber: 1-812-855-7562
| internationaliSDNNumber: 755-7562
| facsimiletelephonenumber: 1-812-855-5004
| title: SITE STRATEGIC PLANNER
| seealso: cn=Cynthia Flowers, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
| postalcode: 4713
| facsimiletelephonenumber: 755-5004

| dn: cn=Marvin McGee, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Marvin McGee
| sn: McGee
| telephonenumber: 1-812-855-9797
| internationaliSDNNumber: 755-9797
| facsimiletelephonenumber: 1-812-855-5004
| seealso: cn=Cynthia Flowers, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
| postalcode: 4713
| facsimiletelephonenumber: 755-5004

| dn: cn=Marshall Riely, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Marshall Riely
| sn: Riely
| telephonenumber: 1-812-855-7218
| internationaliSDNNumber: 755-7218
| title: auto. equip. maint. spec.
| seealso: cn=Cynthia Flowers, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
| postalcode: 4601
| telexnumber: 1-812-480-7509

| dn: cn=James Giliam, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
```

Figure 46 (Part 6 of 12). Sample LDIF Input File

```
| objectclass: organizationalPerson
| cn: James Giliam
| sn: Giliam
| telephonenumber: 1-812-855-5386
| internationaliSDNNumber: 755-5386
| facsimiletelephonenumber: 1-812-855-5824
| title: Project Management
| seealso: cn=Cynthia Flowers, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
| postalcode: 9635

| dn: cn=Al Garcia, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Al Garcia
| sn: Garcia
| telephonenumber: 1-812-855-7579
| internationaliSDNNumber: 755-7095
| title: LEAD TA / MAINTENANCE
| seealso: cn=Cynthia Flowers, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
| postalcode: 1377

| dn: cn=Ben Garcia Jr, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Ben Garcia Jr
| sn: Garcia
| telephonenumber: 1-812-855-3674
| internationaliSDNNumber: 523-3674
| facsimiletelephonenumber: 1-812-855-1077
| title: OS/2 LAN Server Support
| seealso: cn=Cynthia Flowers, ou=Home Entertainment, ou=Home Town, o=Your Company, c=US
| postalcode: 2901
| telexnumber: 1-812-474-3111
| facsimiletelephonenumber: 755-1077

| dn: cn=Becky Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Becky Garcia
| sn: Garcia
| telephonenumber: 1-812-855-5366
| internationaliSDNNumber: 755-5366
| facsimiletelephonenumber: 1-812-855-7961
| title: Flight Manager in Professional Certification
| postalcode: 9635
| facsimiletelephonenumber: 755-7961
| seealso: cn=Linda Carlesberg, ou=Home Town, o=Your Company, c=US
```

| *Figure 46 (Part 7 of 12). Sample LDIF Input File*

```
| dn: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Maria Garcia
| sn: Garcia
| telephonenumber: 1-812-855-8717
| internationaliSDNNumber: 755-8717
| title: SUPPLEMENTAL
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| postalcode: 4633

| dn: cn=Bob Garcia, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Bob Garcia
| sn: Garcia
| telephonenumber: 1-812-855-4553
| internationaliSDNNumber: 755-4553
| title: Preload Test
| seealso: cn=Linda Carlesberg, ou=Home Town, o=Your Company, c=US
| postalcode: 9340

| dn: cn=Ricardo Garcia, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Ricardo Garcia
| sn: Garcia
| telephonenumber: 1-812-855-8278
| internationaliSDNNumber: 755-8278
| postalcode: 1365

| dn: cn=Amy Nguyen, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Amy Nguyen
| sn: Nguyen
| telephonenumber: 1-812-855-7189
| internationaliSDNNumber: 755-7189
| facsimiletelephonenumber: 1-812-855-8199
| title: Coop Dept 32E
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| postalcode: 4604

| dn: cn=James Nguyen, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: James Nguyen
| sn: Nguyen
| telephonenumber: 1-812-855-4156
| internationaliSDNNumber: 755-4156
```

| *Figure 46 (Part 8 of 12). Sample LDIF Input File*

```
| title: AIX Network Device Drivers
| postalcode: 9551
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US

| dn: cn=Henry Nguyen, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Henry Nguyen
| sn: Nguyen
| telephonenumber: 1-812-855-4028
| internationaliSDNNumber: 755-4028
| facsimiletelephonenumber: 1-812-855-9087
| title: AIX Support
| postalcode: 9551
| facsimiletelephonenumber: 755-9087

| dn: cn=Kyle Nguyen, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Kyle Nguyen
| sn: Nguyen
| telephonenumber: 1-812-855-8974
| internationaliSDNNumber: 755-8974
| title: System Support
| seealso: cn=Linda Carlesberg, ou=Home Town, o=Your Company, c=US
| postalcode: 9810
| telexnumber: 1-812-397-1205

| dn: cn=Wayne Nguyen, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Wayne Nguyen
| sn: Nguyen
| telephonenumber: 1-812-855-5052
| internationaliSDNNumber: 755-5052
| title: Object technology consultant
| postalcode: 1003

| dn: cn=Jason Li, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Jason Li
| sn: Li
| telephonenumber: 1-812-855-1466
| internationaliSDNNumber: 755-1466
| facsimiletelephonenumber: 1-812-855-3882
| title: Internet Solutions; HA
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| postalcode: 9584
```

| *Figure 46 (Part 9 of 12). Sample LDIF Input File*

```
| facsimiletelephonenumber: 755-3882

| dn: cn=Melinda Charles, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Melinda Charles
| sn: Charles
| telephonenumber: 1-812-855-5489
| internationaliSDNNumber: 755-5489
| facsimiletelephonenumber: 1-812-855-7670
| title: Integrated Procurement Solutions Home Town
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| postalcode: 1109
| facsimiletelephonenumber: 755-7670

| dn: cn=Bill Keller Jr., ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Bill Keller Jr.
| sn: Keller
| telephonenumber: 1-812-855-5245
| internationaliSDNNumber: 755-5245
| facsimiletelephonenumber: 1-812-855-8138
| title: Returned parts inventory control
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| postalcode: 1505
| telexnumber: 1-800-563-7138
| facsimiletelephonenumber: 755-8138

| dn: cn=Cynthia Smith, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Cynthia Smith
| sn: Smith
| telephonenumber: 1-812-855-8301
| internationaliSDNNumber: 755-8301
| facsimiletelephonenumber: 1-812-855-6074
| title: Electrical Analysis
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| postalcode: 9812
| facsimiletelephonenumber: 755-6074

| dn: cn=Donald Sinclar, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Donald Sinclar
| sn: Sinclar
| telephonenumber: 1-812-855-8840
| internationaliSDNNumber: 755-8840
```

| *Figure 46 (Part 10 of 12). Sample LDIF Input File*

```
| facsimiletelephonenumber: 1-812-855-8138
| title: Mgr. Returned RISC Mach. / Recon / Scrap
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| postalcode: 1514
| facsimiletelephonenumber: 755-8138

| dn: cn=Ben Catu, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Ben Catu
| sn: Catu
| telephonenumber: 1-812-855-6218
| internationaliSDNNumber: 755-6218
| postalcode: 9811
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US

| dn: cn=Eddie Catu, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| description: This entry has every attribute for this class
| cn: Eddie Catu
| sn: Catu
| description: He puts it all together
| destinationindicator: final assembly
| facsimiletelephonenumber: 1-812-855-8985
| internationaliSDNNumber: 755-8498
| l: North America
| physicaldeliveryofficename: doc 17
| postofficebox: 1420
| postaladdress: 2183 Tamil ln. Home Town TX 78659
| postalcode: 1800
| preferreddeliveryMethod: UPS
| registeredaddress: Catu.ring2.austin.ibm.com
| st: TX
| street: 2183 Tamil ln.
| telephonenumber: 1-812-855-8498
| teletexterminalidentifier: 755-8498
| telexnumber: 755-8498
| title: Assembly
| userpassword:: Y2hhbmdlbLiK
| x121Address: 198.176.123.101
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US

| dn: cn=Jesse Catu, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Jesse Catu
| sn: Catu
```

| *Figure 46 (Part 11 of 12). Sample LDIF Input File*

```
| telephonenumber: 1-812-855-7748
| internationaliSDNNumber: 755-7748
| postalcode: 1354
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US

| dn: cn=Joe Simms, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Joe Simms
| sn: Simms
| telephonenumber: 1-812-855-8395
| internationaliSDNNumber: 755-8395
| postalcode: 1514
| telexnumber: 1-812-495-7860
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US

| dn: cn=Judy Simms, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Judy Simms
| sn: Simms
| telephonenumber: 1-812-855-7352
| postalcode: 1343
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US

| dn: cn=Linda Carlesberg, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Linda Carlesberg
| sn: Carlesberg
| telephonenumber: 1-812-855-5492
| internationaliSDNNumber: 755-5492
| title: Purchasing Services
| seealso: cn=Cindy Jeffers, o=Your Company, c=US
| postalcode: 1109
| facsimiletelephonenumber: 755-8985

| dn: cn=Robert Dean, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
| objectclass: organizationalPerson
| cn: Robert Dean
| sn: Dean
| telephonenumber: 1-812-855-5703
| internationaliSDNNumber: 755-5703
| facsimiletelephonenumber: 1-812-855-5704
| postalcode: 1701
| facsimiletelephonenumber: 755-5704
| seealso: cn=Maria Garcia, ou=In Flight Systems, ou=Home Town, o=Your Company, c=US
```

| *Figure 46 (Part 12 of 12). Sample LDIF Input File*

Appendix D. Example Program to Search Entries Using LDAP

The following program is an example of searching entries using the LDAP APIs. The example program can also be found in the **/usr/lpp/ldap/examples** directory. (The LDAP APIs are documented in the *OS/390 LDAP Client Application Development Guide and Reference*.)

Note the following regarding the **ldapsearch.c** example program and all program source shipped in **/usr/lpp/ldap/examples**:

- The example source code as shipped with the LDAP Server is only compilable from the OS/390 shell environment. As shipped, the code is not compilable from the batch environment.
- If compilation from a batch environment is required, compilation flags and libraries required can be found in the Makefile.
- Be aware that there are lines in the example code that exceed 80 characters in length. If the modules are placed into datasets, the datasets must be allocated such that these lines are not truncated.
- See the *OS/390 UNIX System Services Command Reference* for more details about running the **c89** program from the OS/390 shell and from batch.

```
??=ifdef __COMPILER_VER__
??=pragma filetag ("IBM-1047")
??=endif



/*********************************************************************/
/* THIS FILE CONTAINS SAMPLE CODE.  IBM PROVIDES THIS CODE ON AN     */
/* 'AS IS' BASIS WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS        */
/* OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES  */
/* OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.           */
/*********************************************************************/

/*
 * Copyright (c) 1995 Regents of the University of Michigan.
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms are permitted
 * provided that this notice is preserved and that due credit is given
 * to the University of Michigan at Ann Arbor. The name of the University
 * may not be used to endorse or promote products derived from this
 * software without specific prior written permission. This software
 * is provided "as is" without express or implied warranty.
 */

/* ldapsearch.c - simple program to search, list, or read entries
 *                using LDAP
 */

#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <strings.h>
#include <ctype.h>
#include <ldap.h>
#include <line64.h>
#include <locale.h>
```

```
| #ifndef TRUE
|     #define TRUE 1
| #endif

| #ifndef FALSE
|     #define FALSE 0
| #endif

| #define DEFSEP      "="

| static int  rebindproc( LDAP *ld, char **dnp, char **pwp, int *methodp, int freeit );
| static int  dosearch( LDAP *, char *, int, char **, int, char *, char *);
| static void print_entry( LDAP *, LDAPMessage *, int);
| static int  write_ldif_value( char *, char *, unsigned long );
| static void usage( char *s );
| static int  write_ldif_value_or_bvalue( char *, char *, unsigned long, char *, unsigned long );

| static char *prog = NULL;
| static char *binddn = NULL;
| static char *passwd = NULL;
| static char *base = NULL;
| static char *ldaphost = "localhost";
| static int  ldapport = LDAP_PORT;
| static char *sep = DEFSEP;
| static int  verbose, not, allow_binary, vals2tmp, ldif;
| static int  ldapversion = LDAP_VERSION2;


| main( int argc, char **argv )
| {
|     char *infile, *filtpattern, **attrs, line[ BUFSIZ ];
|     char *optpattern = "ZnvtRMABLD
| :V:s:f:h:b:d:p:F:a:w:l:z:S:K:P:N:";
|     FILE * fp;
|     int  rc, i, first, scope, deref, attrsonly, port = 0;
|     int  timelimit, sizelimit, authmethod;
|     int  follow_referrals;
|     LDAP * ld;
|     extern char *optarg;
|     extern int  optind;
|     int  debugLevel = 0;
|     int  debugSpecified = FALSE;
|     int  ssl = FALSE;
|     char *keyfile = NULL, *keyfile_pw = NULL, *keyfile_dn = NULL;
|     int  failureReasonCode;
|     FILE * cf_fd;
|     char *mechanism = "EXTERNAL";
|     int  sasl_bind = FALSE;
|     struct berval **servercred = NULL;
|     int  manageDsa = FALSE;
|     LDAPControl manageDsaIT = { "2.16.840.1.113730.3.4.2",   /*OID*/
|         {0, NULL},                  /*no value*/
|         LDAP_OPT_ON                 /*critical*/
|     };
|     LDAPControl *M_controls[2] = { NULL, NULL};

|     M_controls[0] = &manageDsaIT;
```

```
|      if (prog = strrchr(argv[0], '/'))   /* Strip off any path info
|                                          * on program name
|                                          */
|          ++prog;
|      else
|          prog = argv[0];

|      setlocale(LC_ALL, "");

|      infile = NULL;
|      deref = verbose = allow_binary = not = vals2tmp = attrsonly = ldif = 0;
|      follow_referrals = LDAP_OPT_ON;        /* default to chase referrals */
|      sizelimit = timelimit = 0;
|      scope = LDAP_SCOPE_SUBTREE;

|      while (( i = getopt( argc, argv, optpattern )) != EOF ) {

|          switch ( i ) {
|          case 'V':   /* use version 3 functions */
|              ldapversion = atoi(optarg);
|              if ( (ldapversion != LDAP_VERSION2) &&
|                   (ldapversion != LDAP_VERSION3) ) {
|                  fprintf(stderr, "Incorrect LDAP protocol version selected.\n");
|                  fprintf(stderr, "Supported values are 2 and 3\n");
|                  usage( prog );
|                  exit( 1 );
|              }
|              break;
|       case 'S':   /* use Sasl Bind functions */
|          if ( strncasecmp( optarg, "external", 8 ) == 0 ) {
|                  sasl_bind = TRUE;
|         }
|         else {
|        fprintf( stderr, "only supported mechanism is EXTERNAL\n" );
|        usage( prog );
|                  exit( 1 );
|         }
|         break;
|          case 'n':   /* do Not do any searches */
|              not = TRUE;
|              break;
|          case 'v':   /* verbose mode */
|              verbose = TRUE;
|              break;
|          case 'd':
|              debugLevel = atoi( optarg );
|              debugSpecified = TRUE;
|              break;
|          case 't':   /* write attribute values to /tmp files */
|              vals2tmp = TRUE;
|              break;
|          case 'R':   /* don't automatically chase referrals */
|              follow_referrals = LDAP_OPT_OFF;
|              break;
|          case 'M':       /* manage referral objects as normal entries */
|              manageDsa = TRUE;
|              break;
```

```
|          case 'A':   /* retrieve attribute names only -- no values */
|              attrsonly = TRUE;
|              break;
|          case 'L':   /* print entries in LDIF format */
|              ldif = TRUE;
|              /* fall through -- always allow binary when outputting LDIF */
|          case 'B':   /* allow binary values to be printed */
|              allow_binary = TRUE;
|              break;
|          case 's':   /* search scope */
|              if ( strncasecmp( optarg, "base", 4 ) == 0 ) {
|                  scope = LDAP_SCOPE_BASE;
|              }
|              else if ( strncasecmp( optarg, "one", 3 ) == 0 ) {
|                  scope = LDAP_SCOPE_ONELEVEL;
|              }
|              else if ( strncasecmp( optarg, "sub", 3 ) == 0 ) {
|                  scope = LDAP_SCOPE_SUBTREE;
|              }
|              else {
|                  fprintf( stderr, "scope should be base, one, or sub\n" );
|                  usage( prog );
|                  exit( 1 );
|              }
|              break;

|          case 'a':   /* set alias deref option */
|              if ( strncasecmp( optarg, "never", 5 ) == 0 ) {
|                  deref = LDAP_DEREF_NEVER;
|              }
|              else if ( strncasecmp( optarg, "search", 5 ) == 0 ) {
|                  deref = LDAP_DEREF_SEARCHING;
|              }
|              else if ( strncasecmp( optarg, "find", 4 ) == 0 ) {
|                  deref = LDAP_DEREF_FINDING;
|              }
|              else if ( strncasecmp( optarg, "always", 6 ) == 0 ) {
|                  deref = LDAP_DEREF_ALWAYS;
|              }
|              else {
|                  fprintf( stderr, "alias deref should be never, search,"
|                                                  " find, or always\n" );
|                  usage( prog );
|                  exit( 1 );
|              }
|              break;

|          case 'F':   /* field separator */
|              sep = strdup( optarg );
|              break;
|          case 'f':   /* input file */
|              infile = strdup( optarg );
|              break;
|          case 'h':   /* ldap host */
|              ldaphost = strdup( optarg );
|              break;
|          case 'b':   /* searchbase */
|              base = strdup( optarg );
```

```
|             break;
|         case 'D':   /* bind DN */
|             binddn = strdup( optarg );
|             break;
|         case 'p':   /* ldap port */
|             ldapport = atoi( optarg );
|             port = 1;
|             break;
|         case 'w':   /* bind password */
|             passwd = strdup( optarg );
|             break;
|         case 'l':   /* time limit */
|             timelimit = atoi( optarg );
|             break;
|         case 'z':   /* size limit */
|             sizelimit = atoi( optarg );
|             break;
|         case 'K':
|             keyfile = strdup( optarg );
|             break;
|         case 'P':
|             keyfile_pw = strdup( optarg );
|             break;
|         case 'Z':
|             ssl = TRUE;
|             break;
|         case 'N':
|             keyfile_dn = strdup( optarg );
|             break;
|         default:
|             usage( prog );
|             exit( 1 );
|         }
|     }

|     if ( manageDsa && (ldapversion == LDAP_VERSION2)) {
|         fprintf( stderr, "-M option requires version 3. -M ignored.\n");
|     }

|     if (( base == NULL )) {
|         base = getenv( "LDAP_BASEDN" );
|         if (base != NULL) {
|             base = strdup(base);
|         }
|         /* if NULL will start at top */
|     }

|     if ( argc - optind < 1 ) {
|         usage( prog );
|     }
|     filtpattern = strdup( argv[ optind ] );
|     if ( argv[ optind + 1 ] == NULL ) {
|         attrs = NULL;
|     }
|     else {
|         attrs = &argv[ optind + 1 ];
|     }
|     if ( infile != NULL ) {
```

```
|          if ( infile[0] == '-' && infile[1] == '\0' ) {
|              fp = stdin;
|          }
|          else if (( fp = fopen( infile, "r" )) == NULL ) {
|              perror( infile );
|              exit( 1 );
|          }
|      }

|      if ( !not ) {
|          if (ssl) {
|              if (!port) {
|                  ldapport = LDAPS_PORT;
|              }

|              if ( keyfile == NULL ) {
|                  keyfile = getenv("SSL_KEYRING");
|                  if (keyfile != NULL) {
|                      keyfile = strdup(keyfile);
|                  }
|              }

|              if (verbose) {
|                  printf( "ldap_ssl_client_init( %s, %s, 0,"
|                                                  " &failureReasonCode )\n",
|                          ((keyfile) ? keyfile : "NULL"),
|                          ((keyfile_pw) ? keyfile_pw : "NULL"));
|              }
|              rc = ldap_ssl_client_init( keyfile, keyfile_pw, 0,
|                                          &failureReasonCode ) ;
|              if (rc != LDAP_SUCCESS) {
|                  fprintf( stderr,
|                           "ldap_ssl_client_init failed! rc == %d,"
|                                                  " failureReasonCode == %d\n",
|                           rc, failureReasonCode );
|                  exit( 1 ) ;
|              }
|              if (verbose) {
|                  printf("ldap_ssl_init( %s, %d, %s )\n", ldaphost, ldapport,
|                         ((keyfile_dn) ? keyfile_dn : "NULL"));
|              }
|              ld = ldap_ssl_init( ldaphost, ldapport, keyfile_dn ) ;
|              if (ld == NULL) {
|                  fprintf( stderr, "ldap_ssl_init failed\n" ) ;
|                  perror( ldaphost ) ;
|                  exit( 1 ) ;
|              }
|          }
|          else {
|              if (verbose) {
|                  printf("ldap_init(%s, %d) \n", ldaphost, ldapport);
|              }
|              if ((ld = ldap_init(ldaphost, ldapport)) == NULL) {
|                  perror(ldaphost);
|                  exit(1);
|              }
|          }
```

```
|          ldap_set_option_np(ld, LDAP_OPT_PROTOCOL_VERSION, ldapversion);
|          if ( debugSpecified ) {
|             ldap_set_option_np(ld, LDAP_OPT_DEBUG, debugLevel);
|          }
|          ldap_set_option_np(ld, LDAP_OPT_DEREF, deref);
|          ldap_set_option_np(ld, LDAP_OPT_REFERRALS, follow_referrals);
|          ldap_set_option_np( ld, LDAP_OPT_TIMELIMIT, timelimit);
|          ldap_set_option_np( ld, LDAP_OPT_SIZELIMIT, sizelimit);
|          if ( manageDsa ) {
|              ldap_set_option_np( ld, LDAP_OPT_SERVER_CONTROLS, M_controls);
|          }

|          if ( binddn != NULL ) {
|              ldap_set_rebind_proc( ld, (LDAPRebindProc)rebindproc );
|          }

|          if ( ldapversion != LDAP_VERSION3 ]]
|               (ldapversion == LDAP_VERSION3 && binddn != NULL
|                && sasl_bind == FALSE) ) {
|             /*
|              * When running LDAP Version 3 protocol, bind only if
|              * a bind DN was specified.
|              */
|             authmethod = LDAP_AUTH_SIMPLE;
|             if ( ldap_bind_s( ld, binddn, passwd, authmethod )
|                                                 != LDAP_SUCCESS ) {
|                 ldap_perror( ld, "ldap_bind" );
|                 exit( 1 );
|             }
|          }
|     else if ( ldapversion == LDAP_VERSION3 && sasl_bind == TRUE ) {
|          if ( ldap_sasl_bind_s(ld, NULL, mechanism, NULL, NULL, NULL,
|                  servercred) != LDAP_SUCCESS ) {
|            ldap_perror( ld, "ldap_sasl_bind_s" );
|            exit( 1 );
|        }
|     }
|      } /* ! not */

|      if ( verbose ) {
|          printf( "filter pattern: %s\nreturning: ", filtpattern );
|          if ( attrs == NULL ) {
|              printf( "ALL" );
|          }
|          else {
|              for ( i = 0; attrs[ i ] != NULL; ++i ) {
|                  printf( "%s ", attrs[ i ] );
|              }
|          }
|          putchar( '\n' );
|      }

|      if ( infile == NULL ) {
|          rc = dosearch( ld, base, scope, attrs, attrsonly, filtpattern, NULL );
|      }
|      else {
|          rc = LDAP_SUCCESS;
|          first = 1;
```

```
|         while ( rc == LDAP_SUCCESS &&
|                 fgets( line, sizeof( line ), fp ) != NULL ) {
|             line[ strlen( line ) - 1 ] = '\0';
|             if ( !first ) {
|                 putchar( '\n' );
|             }
|             else {
|                 first = 0;
|             }
|             rc = dosearch( ld, base, scope, attrs, attrsonly, filtpattern, line );
|         }
|         if ( fp != stdin ) {
|             fclose( fp );
|         }
|     }

|     if ( !not ) {
|         ldap_set_option_np( ld, LDAP_OPT_SERVER_CONTROLS, NULL);
|         ldap_unbind( ld );
|     }

|     exit( rc );
| }

| static void usage( char *s )
| {
|     fprintf( stderr, "usage: %s [options] filter [attributes...]\nwhere:\n", s );
|     fprintf( stderr, "    filter\tRFC-1558 compliant LDAP search filter\n" );
|     fprintf( stderr, "    attributes\twhitespace-separated list of"
|                                         " attributes to retrieve\n" );
|     fprintf( stderr, "\t\t(if no attribute list is given, all are"
|                                         " retrieved)\n" );
|     fprintf( stderr, "options:\n" );
|     fprintf( stderr, "    -?\t\tprint this text\n" );
|     fprintf( stderr, "    -V version\tselect LDAP protocol version"
|                                         " (2 or 3; default is 2)\n");
|     fprintf( stderr, "    -S mechanism select SASL bind mechanism"
|                                 " (only supported mechanism is EXTERNAL)\n");
|     fprintf( stderr, "    -n\t\tshow what would be done but don't actually"
|                                         " search\n" );
|     fprintf( stderr, "    -v\t\trun in verbose mode (diagnostics to standard"
|                                         " output)\n" );
|     fprintf( stderr, "    -t\t\twrite values to files in /tmp\n" );
|     fprintf( stderr, "    -A\t\tretrieve attribute names only (no values)\n" );
|     fprintf( stderr, "    -B\t\tdo not suppress printing of non-printable"
|                                         " values\n" );
|     fprintf( stderr, "    -L\t\tprint entries in LDIF format"
|                                         " (-B is implied)\n" );
|     fprintf( stderr, "    -R\t\tdo not automatically follow referrals\n" );
|     fprintf( stderr, "    -M\t\tManage referral objects as normal entries."
|                                         " (requires -V 3)\n" );
|     fprintf( stderr, "    -d level\tset LDAP debugging level to 'level'\n" );
|     fprintf( stderr, "    -F sep\tprint 'sep' instead of '=' between"
|                                       " attribute names and values\n" );
|     fprintf( stderr, "    -f file\tperform sequence of searches listed in"
|                                       " 'file'. ('-' implies stdin)\n" );
|     fprintf( stderr, "    -b basedn\tbase dn for search. LDAP_BASEDN in"
|                                       " environment is default\n" );
```

```
|      fprintf( stderr, "    -s scope\tone of base, one, or sub"
|                                        " (search scope)\n" );
|      fprintf( stderr, "    -a deref\tone of never, always, search, or"
|                                        " find (alias dereferencing)\n" );
|      fprintf( stderr, "    -l time lim\ttime limit (in seconds) for search\n" );
|      fprintf( stderr, "    -z size lim\tsize limit (in entries) for search\n" );
|      fprintf( stderr, "    -D binddn\tbind dn\n" );
|      fprintf( stderr, "    -w passwd\tbind passwd (for simple"
|                                             " authentication)\n" );
|      fprintf( stderr, "    -h host\tldap server\n" );
|      fprintf( stderr, "    -p port\tport on ldap server\n" );
|      fprintf( stderr, "    -Z\t\tuse a secure ldap connection for search\n");
|      fprintf( stderr, "    -K keyfile\tfile to use for keys/certificates\n");
|      fprintf( stderr, "    -P key_pw\t keyfile password\n");
|      fprintf( stderr, "    -N key_dn\t Certificate Name in keyfile\n");
|  }

|  static int  dosearch( LDAP *ld, char *base, int scope, char **attrs,
|                        int attrsonly, char *filtpatt, char *value )
|  {
|      char filter[ BUFSIZ ], **val;
|      int  rc, first, matches;
|      int  references;
|      char **referrals = NULL;
|      int  errcode;
|      char *matched, *errmsg;
|      LDAPMessage * res, *e;
|      int  msgidp;

|      if ( value ) {
|          sprintf( filter, filtpatt, value );
|      }
|      else {
|          strncpy ( filter, filtpatt, BUFSIZ - 1 );
|      }

|      if ( verbose ) {
|          printf( "filter is: (%s)\n", filter );
|      }

|      if ( not ) {
|          return ( LDAP_SUCCESS );
|      }
|      if ( ldap_search( ld, base, scope, filter, attrs, attrsonly ) == -1 ) {
|          ldap_perror( ld, "ldap_search" );
|          return ( ldap_get_errno( ld ) );
|      }

|      matches = 0;
|      references = 0;
|      first = 1;

|      for ( ; ; ) {
|          rc = ldap_result( ld, LDAP_RES_ANY, 0, NULL, &res );
|          if ( rc == LDAP_RES_SEARCH_ENTRY ) {
|              matches++;
|              e = ldap_first_entry( ld, res );
|              if ( !first ) {
```

```
                putchar( '\n' );
            }
            else {
                first = 0;
            }
            print_entry( ld, e, attrsonly );
            ldap_msgfree( res );
        }
        else if ( rc == LDAP_RES_SEARCH_REFERENCE ) {
            references++;
            /* parse and free the search reference */
            ldap_parse_reference_np( ld, res, &referrals, NULL, 1 );
            if ( referrals != NULL ) {
                int i;
                for ( i = 0; referrals[i] != NULL; i++) {
                    fprintf( stderr,
                             (i == 0) ? "Unfollowed search reference: %s\n" :
                             "                                %s\n",
                             referrals[i]);
                }
                fflush( stderr );
                ldap_value_free( referrals );
                referrals = NULL;
            }
        }
        else {
            /* must be a search result */
            break;
        }
    } /* end for */

    if ( rc == -1 ) {
        ldap_perror( ld, "ldap_result" );
        return ( rc );
    }

    if (ldapversion > LDAP_VERSION2) {
        if ( ( rc = ldap_parse_result( ld, res, &errcode, &matched, &errmsg,
                                       &referrals, NULL, 1 ) )
                                                    != LDAP_SUCCESS ) {
            fprintf( stderr, "ldap_search: error parsing result: %d, %s\n",
                     rc, ldap_err2string( rc ) );
        }
        else {
            if ( errcode != LDAP_SUCCESS ) {
                fprintf( stderr, "ldap_search: %s\n",
                                 ldap_err2string( errcode ) );
                if ( matched != NULL ) {
                    if ( *matched != '\0' )
                        fprintf( stderr, "ldap_search: matched: %s\n",
                                         matched );
                    ldap_memfree( matched );
                }
                if ( errmsg != NULL ) {
                    if ( *errmsg != '\0' )
                        fprintf( stderr, "ldap_search: additional info: %s\n",
                                         errmsg );
                    ldap_memfree( errmsg );
```

```
                }
            }
            if ( referrals != NULL ) {
                int i;
                for ( i = 0; referrals[i] != NULL; i++) {
                    fprintf( stderr, "%s %s\n",
                             (i == 0) ? "Unfollowed referral:" :
                             "                    ",
                             referrals[i]);
                }
                ldap_value_free( referrals );
                referrals = NULL;
            }
        }
        fflush( stderr );
    }
    else {
        if (( rc = ldap_result2error( ld, res, 1 )) != LDAP_SUCCESS ) {
            ldap_perror( ld, "ldap_search" );
        }
    }

    if ( verbose ) {
        printf( "%d matches\n", matches );
        if (references > 0) {
            printf( "%d unfollowed references\n", references );
        }
    }

    return ( rc );
}


static void print_entry( LDAP *ld, LDAPMessage *entry, int attrsonly)
{
    char    *a, *dn, tmpfname[ 64 ];
    int i, j, printable = TRUE;
    BerElement      * ber;
    struct berval **bvals;
    FILE        * tmpfp;
    char    **vals = NULL;

    dn = ldap_get_dn( ld, entry );
    if ( ldif ) {
        write_ldif_value( "dn", dn, strlen( dn ));
    }
    else {
        printf( "%s\n", dn );
    }
    ldap_memfree( dn );

    for ( a = ldap_first_attribute( ld, entry, &ber ); a != NULL;
        a = ldap_next_attribute( ld, entry, ber ) ) {
        if ( attrsonly ) {
            if ( ldif ) {
                write_ldif_value( a, "", 0 );
            }
            else {
```

```
                printf( "%s\n", a );
            }
        }
        else if (( bvals = ldap_get_values_len( ld, entry, a )) != NULL ) {
            vals = ldap_get_values( ld, entry, a);
            for ( i = 0; bvals[i] != NULL; i++) {
                if ( vals2tmp ) {
                    sprintf( tmpfname, "/tmp/ldapsearch-%s-XXXXXX", a );
                    tmpfp = NULL;

                    if ( mktemp( tmpfname ) == NULL ) {
                        perror( tmpfname );
                    }
                    else if (( tmpfp = fopen( tmpfname, "w")) == NULL ) {
                        perror( tmpfname );
                    }
                    else if ( fwrite( bvals[ i ]->bv_val,
                                      bvals[ i ]->bv_len, 1, tmpfp ) == 0 ) {
                        perror( tmpfname );
                    }
                    else if ( ldif ) {
                        write_ldif_value( a, tmpfname, strlen( tmpfname ));
                    }
                    else {
                        printf( "%s%s%s\n", a, sep, tmpfname );
                    }

                    if ( tmpfp != NULL ) {
                        fclose( tmpfp );
                    }
                }
                else {

                    int value_len = bvals[ i ]->bv_len;
                    char    *str_value = vals[ i ];
                    if ( ldif ) {
                        write_ldif_value_or_bvalue( a,
                                                    str_value,
                                                    value_len,
                                                    bvals[ i ]->bv_val,
                                                    value_len );
                    }
                    else {
                        printable = TRUE;
                        if (strlen(str_value) == value_len) {
                            for ( j = 0; j < value_len; j++) {
                                if ( !isprint( str_value[ j ] )) {
                                    printable = FALSE;
                                    break;
                                }
                            }
                        }

                        printf( "%s%s%s\n", a, sep,
                                printable ? str_value :
                                (allow_binary ? bvals[ i ]->bv_val :
                                 "NOT Printable") );
                    }
```

```
                }
            }
            ldap_value_free_len( bvals );
            ldap_value_free( vals );
        }
        ldap_memfree( a );
    }
}


static int
write_ldif_value_or_bvalue( char *type, char *value, unsigned long vallen,
                            char *bvalue, unsigned long bvallen)
{
    char    *ldif;

    if ( ( ldif = ldif_type_and_value_or_bvalue( type, value, (int)vallen,
                                                 bvalue,  (int)bvallen ) )
                                                                == NULL ) {
        return ( -1 );
    }

    fputs( ldif, stdout );
    free( ldif );

    return ( 0 );
}


static int
write_ldif_value( char *type, char *value, unsigned long vallen )
{
    char    *ldif;

    if (( ldif = ldif_type_and_value( type, value, (int)vallen )) == NULL ) {
        return ( -1 );
    }

    fputs( ldif, stdout );
    free( ldif );

    return ( 0 );
}


static int  rebindproc( LDAP *ld, char **dnp, char **pwp, int *methodp,
                        int freeit )
{
    if ( !freeit ) {
        *methodp = LDAP_AUTH_SIMPLE;
        if ( binddn != NULL ) {
            *dnp = strdup( binddn );
            *pwp = strdup ( passwd );
        }
        else {
            *dnp = NULL;
            *pwp = NULL;
        }
```

```
|      }
|      else {
|          free ( *dnp );
|          free ( *pwp );
|      }
|      return ( LDAP_SUCCESS );
|  }
```

Appendix E. Sample Makefile

Following is a sample Makefile.

```
#  THIS FILE CONTAINS SAMPLE CODE.  IBM PROVIDES THIS CODE ON AN
#  'AS IS' BASIS WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS
#  OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
#  OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.
#
CFLAGS = -W0,DLL -Dmvs -D_OPEN_THREADS -DMVS_PTHREADS -D_ALL_SOURCE -DEBCDIC_PLATFORM -D_LONGMAP
CFLAGS +=-I/usr/include -I.
SIDEFILE=/usr/lib/GLDCLDAP.x
LIBS = $(SIDEFILE)
OBJS2 = line64.o
MODS = ldapsearch ldapdelete ldapmodify ldapmodrdn sdelete ldapadd
default:  $(MODS)
ldapsearch: ldapsearch.o $(OBJS2)
   c89 -o ldapsearch ldapsearch.o $(OBJS2) $(LIBS)
ldapdelete: ldapdelete.o
   c89 -o ldapdelete ldapdelete.o $(LIBS)
ldapmodify: ldapmodify.o $(OBJS2)
   c89 -o ldapmodify ldapmodify.o $(OBJS2) $(LIBS)
ldapmodrdn: ldapmodrdn.o
   c89 -o ldapmodrdn ldapmodrdn.o $(LIBS)
sdelete: sdelete.o
   c89 -o sdelete sdelete.o $(LIBS)
ldapadd: ldapmodify
   ln -s ./ldapmodify ldapadd
clean:
   rm -f *.o
clobber: clean
   rm -f $(MODS)
```

Figure 47. Sample Makefile

Appendix F. Supported Server Controls

The sections that follow describe the supported server controls.

manageDsaIT

Name: manageDsaIT

Description: Used on a request to suppress referral processing, thereby allowing the client to manipulate referral objects.

Assigned Object Identifier: 2.16.840.1.113730.3.4.2

Target of Control: Server

Control Criticality: Critical

Values: There is no value; the **controlValue** field is absent.

Detailed Description: This control is valid when sent on a client's search, compare, add, delete, modify, or modify DN request. The presence of the control indicates that the server should not return referrals or search continuation references to the client. This allows the client to read or modify referral objects.

authenticateOnly

Name: authenticateOnly

Description: Used on an LDAP bind operation to indicate to the LDAP Server that it should not attempt to find any group membership information for the client's bind DN.

Assigned Object Identifier: 1.3.18.0.2.10.2

Target of Control: Server

Control Criticality: Critical at client's option

Values: There is no value; the **controlValue** field is absent.

Detailed Description: This control is valid when sent on an LDAP client's bind request to the LDAP Server. The presence of this control on the bind request overrides extended group membership searching and default group membership gathering, and causes the LDAP Server to only authenticate the client's bind DN and not gather any group information at all. This control is intended for a client who does not care about group memberships and subsequent complete authorization checking using groups, but is using the bind only for authentication to the LDAP Server and fast bind processing.

Appendix G. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10594-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

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

The following paragraph does not apply 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: but do not indicate that it is the legal department.

IBM Corporation
Mail Station P300
522 South Road
Poughkeepsie, NY 12601-5400
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 information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

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

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 illustrates 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. 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 IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

Trademarks

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

IBM	DATABASE 2	OS/2
AIX	DB2	OS/390
AIX/6000	DRDA	OS/400
BookManager	Library Reader	Parallel Sysplex
CICS	OpenEdition	RACF

Lotus is a trademark of Lotus Development Corporation 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 in the United States, other countries, or both and is licensed exclusively through The Open Group.

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

Glossary

This glossary defines new LDAP Server terms and abbreviations used in this book If you do not find the term you are looking for, refer to the index or to the *IBM Dictionary of Computing*, SC20-1699.

This glossary includes terms and definitions from:

- *IBM Dictionary of Computing*, SC20-1699.
- *Information Technology—Portable Operating System Interface (POSIX),* from the POSIX series of standards for applications and user interfaces to open systems, copyrighted by the Institute of Electrical and Electronics Engineers (IEEE).
- *American National Standard Dictionary for Information Systems,* ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI).
- *Information Technology Vocabulary,* developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1.SC1).
- *CCITT Sixth Plenary Assembly Orange Book, Terms and Definitions* and working documents published by the International Telecommunication Union, Geneva, 1978.
- Open Software Foundation (OSF).

A

access control. Ensuring that the resources of a computer system can be accessed only by authorized users in authorized ways.

access control list (ACL). Data that controls access to a protected object. An ACL specifies the privilege attributes needed to access the object and the permissions that may be granted, to the protected object, to principals that possess such privilege attributes.

ACL. Access control list.

attribute. Information of a particular type concerning an object and appearing in an entry that describes the object in the directory information base (DIB). It denotes the attribute's type and a sequence of one or more attribute values, each accompanied by an integer denoting the value's syntax.

B

backend. A subsystem of the LDAP Server which implements access to a persistent storage mechanism for information.

C

certificate. Used to prove your identity. A secure server must have a certificate and a public-private key pair. A certificate is issued and signed by a Certificate Authority (CA).

CKDS. Cryptographic Key Data Set.

client. A computer or process that accesses the data, services, or resources of another computer or process on the network. Contrast with *server*.

cipher. A method of transforming text in order to conceal its meaning.

configuration. The manner in which the hardware and software of an information processing system are organized and interconnected.

Cryptographic Key Data Set (CKDS). (1) A data set that contains the encrypting keys used by an installation. (2) In ICSF, a VSAM data set that contains all the cryptographic keys. Besides the encrypted key value, an entry in the cryptographic key data set contains information about the key.

cryptography. (1) The transformation of data to conceal its meaning. (2) In computer security, the principles, means, and methods for encrypting plaintext and decrypting ciphertext. (3) In ICSF, the use of cryptography is extended to include the generation and verification of MACs, the generation of MDCs and other one-way hashes, the generation and verification of PINs, and the generation and verification of digital signatures.

D

daemon. A long-lived process that runs unattended to perform continuous or periodic system-wide functions such as network control. Some daemons are triggered automatically to perform their task; others operate periodically.

Data Encryption Standard (DES). In computer security, the National Institute of Standards and Technology (NIST) Data Encryption Standard, adopted by the U.S. government as Federal Information

Processing Standard (FIPS) Publication 46, which allows only hardware implementations of the data encryption algorithm.

data hierarchy. A data structure consisting of sets and subsets such that every subset of a set is of lower rank than the data of the set.

data model. (1) A logical view of the organization of data in a database. (2) In a database, the user's logical view of the data in contrast to the physically stored data, or storage structure. (3) A description of the organization of data in a manner that reflects information structure of an enterprise.

database. A collection of data with a given structure for accepting, storing, and providing, on demand, data for multiple users.

Database 2 (DB2). An IBM relational database management system.

DB2. Database 2.

DES. Data Encryption Standard (DES).

directory. (1) A logical unit for storing entries under one name (the directory name) in a CDS namespace. Each physical instance of a directory is called a replica. (2) A collection of open systems that cooperates to hold a logical database of information about a set of objects in the real world.

directory schema. The set of rules and constraints concerning directory information tree (DIT) structure, object class definitions, attribute types, and syntaxes that characterize the directory information base (DIB).

directory service. The directory service is a central repository for information about resources in a distributed system.

distinguished name (DN). One of the names of an object, formed from the sequence of RDNs of its object entry and each of its superior entries.

DN. Distinguished name.

E

environment variable. A variable included in the current software environment that is available to any called program that requests it.

I

ICSF. Integrated Cryptographic Service Facility.

Integrated Cryptographic Service Facility (ICSF). A licensed program that runs under MVS/System Product 3.1.3, or higher, or OS/390 Release 1, or higher, and provides access to the hardware cryptographic feature for programming applications. The combination of the hardware cryptographic feature and ICSF provides secure high-speed cryptographic services.

J

JCL. Job control language.

Job control language (JCL). A control language used to identify a job to an operating system and to describe the job's requirements.

K

key generator utility program (KGUP). A program that processes control statements for generating and maintaining keys in the cryptographic key data set.

KGUP. Key generator utility program.

L

LDAP. Lightweight Directory Access Protocol.

Lightweight Directory Access Protocol (LDAP). A client/server protocol for accessing a directory service.

M

master replica. The first instance of a specific directory in the namespace. After copies of the directory have been made, a different replica can be designated as the master, but only one master replica of a directory can exist at a time.

MD5. Message Digest 5. A hash algorithm.

MKKF. Make Key File.

MKKF utility. A command-line utility used to create public/private key pairs and certificate requests, receive certificate requests into a key ring, and manage keys in a key ring.

O

object class. An identified family of objects that share certain characteristics. An object class can be specific to one application or shared among a group of applications. An application interprets and uses an entry's class-specific attributes based on the class of the object that the entry describes.

OCSF. Open Cryptographic Services Facility.

ODBC. Open database connectivity.

Open Cryptographic Services Facility (OCSF). A derivative of the IBM Keyworks technology which is an implementation of the Common Data Security Architecture (CDSA) for applications running in the UNIX System Services environment.

OS/390 Cryptographic Services. An OS/390 offering that supplies a set of interfaces for cryptographic functions.

P

private key. Used for the encryption of data. A secure server keeps its private key secret. A secure server sends clients its public key so they can encrypt data to the server. The server then decrypts the data with its private key.

public key. Used for the encryption of data. A secure server makes its public key widely available so that its clients can encrypt data to send to the server. The server then decrypts the data with its private key.

R

RACF. Resource Access Control Facility.

RDN. Relative distinguished name.

referral. An outcome that can be returned by a directory system agent that cannot perform an operation itself. The referral identifies one or more other directory system agents more able to perform the operation.

relative distinguished name (RDN). A component of a DN. It identifies an entry distinctly from any other entries which have the same parent.

replica. A directory in the CDS namespace. The first instance of a directory in the namespace is the master replica. See *master replica*.

replication. The making of a shadow of a database to be used by another node. Replication can improve availability and load-sharing.

Resource Access Control Facility (RACF). An IBM licensed program, that provides for access control by identifying and verifying the users to the system, authorizing access to protected resources, and logging the detected unauthorized access to protected resources.

S

SASL. Simple Authentication Security Layer.

schema. See *directory schema*.

Secure Sockets Layer (SSL) security. A facility used to protect LDAP access.

server. On a network, the computer that contains programs, data, or provides the facilities that other computers on the network can access. Contrast with *client*.

SHA. Secure Hash Algorithm. A hash algorithm required for use with the Digital Signature Standard.

Simple Authentication Security Layer (SASL). Refers to a method of binding using SSL authentication and the client's certificate identity.

SLAPD. A stand-alone LDAP daemon.

SPUFI. SQL Processor Using File Input.

SQL Processor Using File Input (SPUFI). A facility of the TSO attachment subcomponent that enables the DB2I user to run SQL statements without embedding them in an application program.

SQL. Structured Query Language.

SSL. Secure Sockets Layer.

Structured Query Language (SQL). A standardized language for defining and manipulating data in a relational database.

T

thread. A single sequential flow of control within a process.

Time Sharing Option (TSO). An operating system option that provides interactive time sharing from remote terminals.

TSO. Time Sharing Option.

U

UCS Transformation Format (UTF). The LDAP Version 3 protocol specifies that data is passed between client and server in the UTF-8 character set.

UTF. UCS Transformation Format.

X

X.500. The CCITT/ISO standard for the open systems interconnection (OSI) application-layer directory. It allows users to register, store, search, and retrieve information about any objects or resources in a network or distributed system.

Bibliography

This bibliography provides a list of publications that are useful when implementing the LDAP Server product. The complete title, order number, and a brief description is given for each publication.

IBM OS/390 Security Server Publications

- *OS/390 LDAP Client Application Development Guide and Reference*, SC24-5878

  This book describes the Lightweight Directory Access Protocol (LDAP) client APIs that you can use to develop LDAP applications.

- *OS/390 Security Server (RACF) Command Language Reference*, SC28-1919

  This book describes the syntax and the functions of the commands for RACF. It is intended for RACF-defined users who are responsible for creating, updating, or maintaining the profiles for users, groups, data sets, and general resources in the RACF database.

IBM C/C++ Language Publication

- *IBM OS/390 C/C++ Programming Guide*, SC09-2362

  This book describes how to develop applications in the C/C++ language in OS/390.

IBM DB2 Publications

- *DB2 for OS/390 Call Level Interface Guide and Reference*, SC26-8959

  This book provides the information necessary to write applications using DB2 Call Level Interface to access IBM DATABASE 2 servers as well as any database that supports DRDA® level 1 or DRDA level 2 protocols. This book should also be used as a supplement when writing portable ODBC applications that can be run in a native DB2 for OS/390 environment using the DB2 Call Level Interface.

- *DB2 for OS/390 Application Programming and SQL Guide*, SC26-8958

  This book discusses how to design and write application programs that access DB2 for OS/390 (DB2), a highly flexible relational database management system (DBMS).

- *DB2 for OS/390 Messages and Codes*, GC26-8979

  This book lists messages and codes issued by DB2, with explanations and suggested responses.

The HTTP address for the IBM DB2 publications is:

`http://www.software.ibm.com/data/db2/os390/library.html`

IBM OS/390 Cryptographic Services Publications

- *OS/390 Cryptographic Services System Secure Sockets Layer Programming Guide and Reference*, SC24-5877

  Contains guidance and reference information that an application programmer needs to use the System Secure Sockets Layer (SSL) callable programming interfaces. SSL is a communications layer that provides data privacy and integrity, as well as server and client authentication based on public key certificates.

- *OS/390 Open Cryptographic Services Facility Application Developer's Guide and Reference*, SC24-5875

Provides an overview of the Open Cryptographic Services Facility (OCSF). It explains how to integrate OCSF into applications and contains a sample OCSF application. It also defines the interfaces that application developers employ to access security services provided by the OCSF framework and service provider modules. Specific information about the individual service providers is also described.

- *OS/390 ICSF Administrator's Guide*, SC23-3975

  Explains how to administer ICSF on CMOS and bipolar hardware. It describes managing cryptographic keys, entering keys into the Cryptographic Key Data Set (CKDS) and using the key generator utility program (KGUP).

Index

Special Characters

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Readers' Comments

OS/390®
Security Server
LDAP Server
Administration and Usage Guide

Publication No. SC24-5861-04

You may use this form to report errors, to suggest improvements, or to express your opinion on the appearance, organization, or completeness of this book.

Date: _____

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

Note

Report system problems to your IBM representative or the IBM branch office serving you.
U.S. customers can order publications by calling the IBM Software Manufacturing Solutions at **1-800-879-2755**.

In addition to using this postage-paid form, you may send your comments by:

FAX	1-607-752-2327	Internet	pubrcf@vnet.ibm.com
IBM Mail	USIB2L8Z@IBMMAIL	IBMLink	GDLVME(PUBRCF)

Would you like a reply? ___YES ___ NO If yes, please tell us the type of response you prefer.

___ Electronic address: _____

___ FAX number: _____

___ Mail: (Please fill in your name and address below.)

Name _____ Address _____

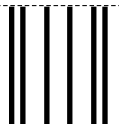
Company or Organization _____

Phone No. _____

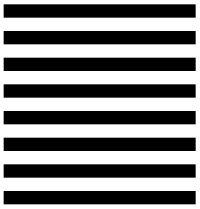
Readers' Comments
SC24-5861-04

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

Department G60
International Business Machines Corporation
Information Development
1701 North Street
ENDICOTT NY 13760-5553

Fold and Tape **Please do not staple** Fold and Tape

SC24-5861-04

Cut or Fold
Along Line

Program Number: 5647-A01

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC24-5861-04

Spine information:

OS/390 Security Server

LDAP Server Administration and Usage Guide