IBM

IBM Directory Server Version 5.1: Administration Guide

IBM

IBM Directory Server Version 5.1: Administration Guide

Note

Before using this information and the product it supports, read the general information under Appendix F, “Notices” on page 281.

First Edition (November 2002)

This edition applies to version 5, release 1, of the IBM® Directory Server and to all subsequent releases and modifications until otherwise indicated in new editions.

Preface

This document contains the information that you need to administer the IBM Directory Server.

Contents

Part 1. Directory overview

Chapter 1. Defining a directory

A directory is a collection of information about objects arranged in a hierarchical structure. It is a specialized database, that enables users or applications to find resources that have the characteristics needed for a particular task.

If the name of an object is known, its characteristics can be retrieved. If the name of a particular individual object is not known, the directory can be searched for a list of objects that meet a certain requirement. Directories can usually be searched by specific criteria, not just by a predefined set of categories.

A directory is a specialized database that has characteristics that set it apart from general purpose relational databases. A characteristic of a directory is that it is accessed (read or searched) much more often than it is updated (written). Because directories must be able to support high volumes of read requests, they are typically optimized for read access. Because directories are not intended to provide as many functions as general-purpose databases, they can be optimized to economically provide more applications with rapid access to directory data in large distributed environments.

A directory can be centralized or distributed. If a directory is centralized, there is one directory server (or a server cluster) at one location that provides access to the directory. If the directory is distributed, there are more than one server, usually geographically dispersed, that provide access to the directory.

When a directory is distributed, the information stored in the directory can be partitioned or replicated. When information is partitioned, each directory server stores a unique and non-overlapping subset of the information. That is, each directory entry is stored by one and only one server. The technique to partition the directory is to use LDAP referrals. LDAP referrals allow the users to refer Lightweight Directory Access Protocol (LDAP) requests to either the same or different name spaces stored in a different (or same) server. When information is replicated, the same directory entry is stored by more than one server. In a distributed directory, some information may be partitioned, and some information may be replicated.

Directory clients and servers

Directories are usually accessed using the client-server model of communication. The client and server processes might or might not be on the same machine. A server is capable of serving many clients. An application that wants to read or write information in a directory does not access the directory directly. Instead, it calls a function or application programming interface (API) that causes a message to be sent to another process. This second process accesses the information in the directory on behalf of the requesting application. The results of the read or write are then returned to the requesting application.

An API defines the programming interface a particular programming language uses to access a service. The format and contents of the messages exchanged between client and server must adhere to an agreed upon protocol. LDAP defines a message protocol used by directory clients and directory servers. There is also an associated LDAP API for the C language and ways to access the directory from a Java™ application using the Java Naming and Directory Interface (JNDI).

Directory security

A directory should support the basic capabilities needed to implement a security policy. The directory might not directly provide the underlying security capabilities, but it might be integrated with a trusted network security service that provides the basic security services. First, a method is needed to authenticate users. Authentication verifies that users are who they say they are. A user name and password is a basic authentication scheme. Once users are authenticated, it must be determined if they have the authorization or permission to perform the requested operation on the specific object.

Authorization is often based on access control lists (ACLs). An ACL is a list of authorizations that may be attached to objects and attributes in the directory. An ACL lists what type of access each user or a group of users is allowed or denied. In order to make ACLs shorter and more manageable, users with the same access rights are often put into groups.

Chapter 2. The IBM Directory

The IBM Directory implements the Internet Engineering Task Force (IETF) LDAP V3 specifications. It also includes enhancements added by IBM in functional and performance areas. This version uses the IBM DB2® as the backing store to provide per LDAP operation transaction integrity, high performance operations, and on-line backup and restore capability. It interoperates with the IETF LDAP V3 based clients. Major features include:

- Administration and configuration for the IBM Directory is provided by a Graphical User Interface (GUI). The administration and configuration functions allow the administrator to:
  - Perform the initial setup of the directory
  - Change configuration parameters and options
  - Manage the daily operations of the directory, such as adding or editing objects, for example, object classes, attributes, and entries.
- A dynamically extensible directory schema – This means that administrators can define new attributes and object classes to enhance the directory schema. Changes can be made to the directory schema, too, which are subject to consistency checks. Users may dynamically modify the schema content without restarting the directory server. Since the schema itself is part of the directory, schema update operations are done through standard LDAP APIs. The major functions provided by LDAPv3 dynamic extensible schema are:
  - Queriable schema information through LDAP APIs
  - Dynamic schema changes through LDAP APIs
  - Server rootDSE
  - Dynamic configuration changes using LDAP APIs
- UTF-8 (Universal Character Set Transformation Format) – An IBM Directory Server supports data in multiple languages, and allows users to store, retrieve and manage information in a native language code page.
- Simple Authentication and Security Layer (SASL) – This support provides for additional authentication mechanisms. The Secure Sockets Layer (SSL) provides encryption of data and authentication using X.509v3 public-key certificates. A server may be configured to run with or without SSL support.
- Replication – Replication is supported, which makes additional read-only copies of the directory available, improving performance and reliability of the directory service.
- Referrals – Support for LDAP referrals, allowing directories to be distributed across multiple LDAP servers where a single server may only contain a subset of the whole directory data.
- Access control model – A powerful, easy-to-manage access control model is supported through ACLs.
- Change log
- Password policy
- Security audit logging

Part 2. Server Administration

Chapter 3. Web Administration graphical user interface (GUI)

The preferred method of administering the server is by using the Web Administration GUI.

Directory administration daemon

The directory administration daemon enables remote management of the IBM Directory Server. It must be installed on the machine where the IBM Directory Server is installed and be running continuously. The directory administration daemon accepts requests by way of LDAP extended operations and supports starting, stopping, restarting and status monitoring of the IBM Directory Server. By default, the IBM Directory administration daemon listens on two ports, port 3538 for non-SSL connections and port 3539 for SSL connections, if SSL communication is enabled.

Note: If you enable SSL communication, the directory administration daemon must be stopped and restarted for SSL to take effect. See "Configuring SSL settings" on page 43.

Setting up Web Administration

Unlike the previous releases in which the Server Administration GUI had to be installed on the same machine as the directory server, the IBM Directory Server Version 5.1 Web Administration tool is installed on an application server, such as the **embedded version of IBM WebSphere® Application Server - Express** included with the IBM Directory Server, and administered through a console. Servers that have been added to the console can utilize the Web Administration tool without having to have the tool installed.

Note: The **embedded version of IBM WebSphere Application Server - Express** does not support HP-UX systems. HP-UX must use another application server, such as Apache Tomcat/4.0.4.

Prerequisites

Before you can start using the Web Administration Tool, ensure that you have the completed the following prerequisites during the configuration of your server:

- Set the AdminDN and password.
- Configured a database.

See the *IBM Directory Server Version 5.1 Installation and Configuration Guide* for information on these tasks.

Notes:

1. The administration daemon must be running. Issue the command `ibmdiradm` to start the daemon, or additionally on Windows-based systems you can start the administration daemon using the Services panel.
2. If you have other Web server applications running, ensure that the Web Administration Tool is not running on the same port as the other Web servers.

Logging on to the console

If you have installed the Web Administration package start the application server. For the **embedded version of IBM WebSphere Application Server - Express** go to the directory where you installed the IBM Directory Server and issue the command:

For UNIX-based platforms

```
<IDSinstalldir>/ldap/appsrv/bin/startServer.sh server1
```

Note: For Solaris this is `opt/ibmldapc/appsrv/bin/startServer.sh` server1

For Windows-based platforms

```
<IDSinstalldir>\ldap\appsrv\bin\startServer.bat server1
```

Open a Web browser and type the following address: `http://localhost:9080/IDSWebApp/IDSjsp/Login.jsp`. The **IBM Directory Server Web Administration login page** panel is displayed.

1. Log in as **Console Admin**, the default selection in the **LDAP Hostname** field.
2. In the **Username** field type: **superadmin**.
3. In the **Password** field type: **secret**.
4. Click **Login**.

The **IBM Directory Server Web Administration Tool** panel is displayed.

Managing the console

At the **IBM Directory Server Web Administration Tool** panel:

1. Expand **Console administration** in the navigation area. Use

   Change console administrator login
   to change **superadmin** to a different administrator ID.

   a. Enter the new administrator ID.

      Note: Only one administrator ID is allowed. The **superadmin** ID is replaced by the new ID that you specified.

   b. Enter the current administrator password. The password, **secret**, is the same for the new administrator ID, until you change it

   Change console administrator password
   to change the administrator password, **secret**, to another password.

   a. Enter the current password.
   b. Enter the new password.
   c. Reenter the new password to confirm that their are no typographical errors.
   d. Click **OK**.

   Manage console servers
   to add, edit, or delete servers in the console. Click **Manage console servers** in the navigation area. A listing of server host names and port numbers is displayed. You can:

   Add a server to the console

   a. Click **Add**.
   b. Enter the host name address. For example *myserver.mycity.mycompany.com*
   c. Specify the port numbers or accept the defaults.

d. Specify if the server is SSL enabled.
e. Click **OK** to apply the changes or click **Cancel** to exit the panel without making any changes.

Modify a server in the console

a. Select the radio button by the server you want to modify.
b. Click **Edit**.
c. You can change the port numbers.
d. You can change whether the server is SSL enabled.
e. Click **OK** to apply the changes or click **Cancel** to exit the panel without making any changes.

Remove a server from the console

a. Select the radio button by the server you want to remove.
b. Click **Delete**.
c. Click **OK** to delete the server or click **Cancel** to exit the panel without making any changes.

Manage console properties
to set the following:

- **Component management** - enables the selected components for all servers in the console. By default all the components are enabled.
- **Session properties** - sets the time out limit for the console session. The default setting is 60 minutes.
- **SSL key database** - enables you set up the console so that it can communicate with other LDAP servers using the Secure Sockets Layer (SSL), if necessary. Set the key database path and file name, the key password, the trusted database path and file name, the trusted password in the appropriate fields. Jks is the supported file type. See “Using gsk6ikm” on page 52 and “Secure Sockets Layer” on page 47 for information about key databases and SSL.

2. When you have finished setting up the console, click **Logout** to exit.

Starting the Web Administration

At the **Logout successful** panel that is displayed when you exit **Console administrator**, you see the message:

```
If you have been accidentally logged out then you will need to re-login
by clicking here.
```

Click the word **here** to go to the IBM Directory Server Web Administration Login Page.

Logging on to the Web Administration Tool

- At the **IBM Directory Server Web Administration login page** select the LDAP host name for your machine from the drop-down menu.
- Enter the DN and the password for that server (you set these up during the server installation process).
- Click **Login**.

After the tool has been started, the application window has five parts:

Banner area
The banner area located at the top of the panel contains the application name, IBM Directory Server Web Administration Tool, and the IBM Logo.

Navigation area
The navigation area, located on the left side of the panel, displays expandable categories for various server content tasks.

Work area
The work area displays the tasks associated with the selected task in the navigation area. For example, if **Managing server security** is selected in the navigation area, the work area displays the Server Security page and the tabs containing tasks related to setting up server security.

Server status area
The server status area, located at the top of the work area, indicates the status and the name of the server being administered. It also has two icon links, one to the Start/Stop/Restart procedure and the other to general help information. When you select a task from the navigation area, the name of the selected task, a link to the error log files, and a link to the task help are also displayed.

Task status area
The task status area, located beneath the work area, displays the status of the current task.

Chapter 4. Distinguished names (DNs)

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 order of the component attribute value pairs is important. The DN contains one component for each level of the directory hierarchy from the root down to the level where the entry resides. 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 first component of the DN is referred to as the Relative Distinguished Name (RDN). It identifies an entry distinctly from any other entries that 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″). These two example DNs are otherwise equivalent. The attribute:value pair making up the RDN for an entry must also be present in the entry. (This is not true of the other components of the DN.)

Follow this example to create a distinguished name in the ibmslapd.conf file:

```
DN: cn=Directory,cn=RDBM Backends,cn=IBM Directory,cn=Schemas,cn=Configuration
ibm-slapdDbName: <databasename>
ibm-slapdUserID: <username>
ibm-slapdDbUserPW: <password>
ibm-slapdDbInstance: <username>
```

Distinguished name syntax

The Distinguished Name (DN) syntax supported by this server is based on RFC 2253. The Backus Naur Form (BNF) syntax is defined as follows:

```
<name> ::= <name-component> ( <spaced-separator> )
         | <name-component> <spaced-separator> <name>

   <spaced-separator> ::= <optional-space>
                   <separator>
                   <optional-space>

   <separator> ::=  "," | ";"

   <optional-space> ::= ( <CR> ) *( " " )

   <name-component> ::= <attribute>
           | <attribute> <optional-space> "+"
             <optional-space> <name-component>

   <attribute> ::= <string>
           | <key> <optional-space> "=" <optional-space> <string>

   <key> ::= 1*( <keychar> ) | "OID." <oid> | "oid." <oid>
   <keychar> ::= letters, numbers, and space

   <oid> ::= <digitstring> | <digitstring> "." <oid>
   <digitstring> ::= 1*<digit>
   <digit> ::= digits 0-9
```

```
<string> ::= *( <stringchar> | <pair> )
         | '"' *( <stringchar> | <special> | <pair> ) '"'
         | "#" <hex>


<special> ::= "," | "=" | <CR> | "+" | "<" |  ">"
         | "#" | ";"

<pair> ::= "\" ( <special> | "\" | '"')
<stringchar> ::= any character except <special> or "\" or '"'


<hex> ::= 2*<hexchar>
<hexchar> ::= 0-9, a-f, A-F
```

A semicolon (;) character can be used to separate RDNs in a distinguished name, although the comma (,) character is the typical notation.

White-space characters (spaces) might be present on either side of the comma or semicolon. The white-space characters are ignored, and the semicolon is replaced with a comma.

In addition, space (′ ′ ASCII 32) characters may be present either before or after a ′+′ or ′=′. These space characters are ignored when parsing.

A value may be surrounded by double quotation (′″′ ACSII 34) characters, which are not part of the value. Inside the quoted value, the following characters can occur without any escaping:

- A space or ″#″ character occurring at the beginning of the string
- A space character occurring at the end of the string
- One of the characters ″′″, ″=″, ″+″, ″\″, ″<″, ″>″, or ″;″

Alternatively, a single character to be escaped may be prefixed by a backslash (′\′ ASCII 92). This method can be used to escape any of the characters listed previously and the double quotation marks (′″′ ASCII 34) character.

This notation is designed to be convenient for common forms of names. The following example is a distinguished name written using this notation. First is a name containing three components. The first of the components is a multivalued RDN. A multivalued RDN contains more than one attribute:value pair and can be used to distinctly identify a specific entry in cases where a simple CN value might be ambiguous:

```
OU=Sales+CN=J. Smith,O=Widget Inc.,C=US
```

DN escaping rules

A DN can contain special characters. These characters are **,** (comma), **=** (equals), **+** (plus), **<** (less than), **>** (greater than), **#** (number sign), **;** (semicolon), **** (backslash), and **“”** (quotation marks).

To escape these special characters or other characters in an attribute value in a DN string, use the following methods:

1. If a character to be escaped is one of special characters, precede it by a backslash (′\′ ASCII 92). This example shows a method of escaping a comma in an organization name:

   ```
   CN=L. Eagle,O=Sue\, Grabbit and Runn,C=GB
   ```

This is the preferred method.

2. Otherwise replace the character to be escaped by a backslash and two hex digits, which form a single byte in the code of the character. The code of the character **must** be in UTF-8 code set.

   ```
   CN=L. Eagle,O=Sue\2C Grabbit and Runn,C=GB
   ```

3. Surround the entire attribute value by “” (quotation marks) (ASCII 34), that are not part of the value. Between the quotation character pair, all characters are taken as is, except for the \ (backslash). The \ (backslash) can be used to escape a backslash (ASCII 92) or quotation marks (ASCII 34), any of the special characters previously mentioned, or hex pairs as in method 2. For example, to escape the quotation marks in `cn=xyz"qrs"abc`, it becomes `cn=xyz\"qrs\"abc` or to escape a `\`:

   ```
   "you need to escape a single backslash this way \\"
   ```

   Another example, `"\Zoo"` is illegal, because ’Z’ cannot be escaped in this context.

On the server end, when a DN is received in this form, the server reformats the DN using escape mechanisms number 1 and 2 for internal processing.

Pseudo DNs

Pseudo DNs are used in access control definition and evaluation. The LDAP/DB2 directory contains several pseudo DNs (for example, ″group:cn=this″ and ″access-id:cn=Anybody″), which are used to refer to large numbers of DNs that share a common characteristic, in relation to either the operation being performed or the object on which the operation is being performed.

Three pseudo DNs are supported by LDAP version 3:

- access-id: cn=this

  When specified as part of an ACL, this DN refers to the bindDN, which matches the DN on which the operation is performed. For example, if an operation is performed on the object ″cn=personA, ou=IBM, c=US″ and the bindDn is ″cn=personA, ou=IBM, c=US″, the permissions granted are a combination of those given to ″cn=this″ and those given to ″cn=personA, ou=IBM, c=US″.

- group: cn=anybody

  When specified as part of an ACL, this DN refers to all users, even those that are unauthenticated. Users cannot be removed from this group, and this group cannot be removed from the database.

- group: cn=Authenticated

  This DN refers to any DN that has been authenticated by the directory. The method of authentication is not considered.

  Note: ″cn=Authenticated″ refers to a DN that has been authenticated anywhere on the server, regardless of where the object representing the DN is located. It should be used with caution, however. For example, under one suffix, ″cn=Secret″ could be a node called ″cn=Confidential Material″ which has an aclentry of ″group:cn=Authenticated:normal:rsc″. Under another suffix, ″cn=Common″ could be the node ″cn=Public Material″. If these two trees reside on the same server, a bind to ″cn=Public Material″ would be considered authenticated, and would get permission to the normal class on the ″cn= Confidential Material″ object.

Some examples of pseudo DNs:

Example 1

Consider the following ACL for object: cn=personA, c=US AclEntry:

```
access-id: cn = this:critical:rwsc
AclEntry: group: cn=Anybody: normal:rsc
AclEntry: group: cn=Authenticated: sensitive:rcs
```

Table 1.

User Binding as	Would receive
cn=personA, c=US	normal:rsc:sensitive:rcs:critical:rwsc
cn=personB, c=US	normal:rsc:sensitive:rsc
NULL (unauth.)	normal:rsc

In this example, personA receives permissions granted to the ″cn=this″ ID, and permissions given to both the ″cn=Anybody″ and ″cn=Authenticated″ pseudo DN groups.

Example 2

Consider the following ACL for object: cn=personA, c=US AclEntry: access-id:cn=personA, c=US: object:ad

```
AclEntry: access-id: cn = this:critical:rwsc
AclEntry: group: cn=Anybody: normal:rsc
AclEntry: group: cn=Authenticated: sensitive:rcs
```

For an operation performed on cn=personA, c=US:

Table 2.

User Binding as	Would receive
cn=personA, c=US	object:ad:critical:rwsc
cn=personB, c=US	normal:rsc:sensitive:rsc
NULL (unauth.)	normal:rsc

In this example, personA receives permissions granted to the ″cn=this″ ID, and those given to the DN itself ″cn=personA, c=US″. Note that the group permissions are not given because there is a more specific aclentry (″access-id:cn=personA, c=US″) for the bind DN (″cn=personA, c=US″).

Enhanced DN processing

A composite RDN of a DN may consist of multiple components connected by the '+' operators. The server enhances the support for searches on entries that have such a DN. A composite RDN can be specified in any order as the base for a search operation.

```
ldapsearch cn=mike+ou=austin,o=ibm,c=us
```

The server accepts DN normalization extended operations. DN normalization extended operations normalize DNs using the server schema. This extended operation might be useful for applications that use DNs. See the *IBM Directory Server Version 5.1 C-client Programming Reference* for more information.

Chapter 5. Basic server tasks

The following tasks can be performed by the directory administrator:

Changing an administrator distinguished name and password

The administrator name and password is usually set during the server installation and configuration process. However, you can change an administrator name and an administrator password by using either the Web Administration Tool or the command line.

Using Web Administration:

Click **User properties** in the navigation area of the Web Administration Tool. Two selections are displayed:

Change administrator login
Specify a new Administrator DN in the field and enter the current password. Click **OK** or click **Cancel** to return to the **Welcome** panel without making any changes.

Note: This selection is available only if you are logged in as the administrator. It is not available if you are logged in as a user.

Change password
To change the password for the currently logged-in DN, type your current password in the **Current password** field. Then type your new password in the **New password** field and type it again in the **Confirm new password** field and click **OK**. Click **Cancel** to return to the **Welcome** panel without making any changes.

Using the command line:

You can use either the **ldapcfg** command or the **ldapxcfg** utility from the command line.

Using the **ldapcfg** command:

```
ldapcfg -u <admindn> -p <adminpassword>
```

To use the ldapxcfg utility type **`ldapxcfg`** on a command line. When the IBM Directory Server Configuration Tool panel is displayed select **Administrator DN/password** and follow the directions. See the *IBM Directory Server Version 5.1 Installation and Configuration Guide* for additional information on using the ldapxcfg utility.

See Chapter 4, “Distinguished names (DNs)” on page 13 for more information about distinguished names.

Checking server status

You can check the status of the server by searching for the object classes under cn=monitor. To do this, use one of the following methods:

Using Web Administration:

Expand the Server administration category in the navigation area. Click **View server status**. If the directory server is running, the following information is displayed:

Hostname
The host name of the LDAP server.

Server status
The server is either **Running**, **Stopped**, or **Running configuration only mode**. You can determine the server status at any time by the three icons displayed in the left side corner of the server status area.

Start time
The time the server was started. The start time is in the format:

```
year-month-day hour:minutes:seconds GMT
```

Note: If expressed in local time the format is

```
day-month-date hour:minutes:seconds timezone year
```

Current time
The current time on the server. The current time is in the format:

```
year month day hour:minutes:seconds GMT
```

Note: If expressed in local time the format is

```
day month date hour:minutes:seconds timezone year
```

Total threads
The number of worker threads being used by the server.

Total threads blocked on write
The number of threads sending data back to the client.

Total threads blocked on read
The number of threads reading data from the client.

Number of connections
The number of currently active connections.

Total connections
The total number of connections since the server was started.

Number of operations initiated
The number of initiated requests since the server was started.

Number of operations completed
The number of completed requests since the server was started.

Number of searches initiated
The number of initiated searches since the server was started.

Number of searches completed
The number of completed searches since the server was started.

Number of entries sent
The number of entries sent by the server since the server was started.

Percentage of entry cache used
The percentage of entry cache currently used. This value does not display in configuration only mode.

Percentage of search filter cache used
: The percentage of search filter cache currently used. This value does not display in configuration only mode.

ACL cache
: A Boolean value indicating that the ACL cache is active (TRUE) or inactive (FALSE). This value does not display in configuration only mode.

Maximum ACL cache size
: The maximum number of entries in the ACL cache. This value does not display in configuration only mode.

Note: When a link is clicked in the navigation area a new connection is opened to the LDAP server. The number of connections increases, if a link is clicked repeatedly in the navigation area. These connections are closed only when the Java garbage collector begins its cleanup.

Using the command line:

```
ldapsearch -h <servername> -p <portnumber> -b cn=monitor -s base objectclass=*
```

This command returns the following information:

cn=monitor

version=IBM Directory, Version 5.1

total connections
: The total number of connections since the server was started.

current connections
: The number of active connections.

maxconnections
: The maximum number of active connections allowed.

writewaiters
: The number of threads sending data back to the client.

readwaiters
: The number of threads reading data from the client.

opsinitiated
: The number of initiated requests since the server was started.

livethreads
: The number of worker threads being used by the server.

opscompleted
: The number of completed requests since the server was started.

entriessent
: The number of entries sent by the server since the server was started.

searchesrequested
: The number of initiated searches since the server was started.

searchescompleted
: The number of completed searches since the server was started.

filter_cache_size
: The maximum number of filters allowed in the cache.

filter_cache_current
: The number of filters currently in the cache.

filter_cache_hit
The number of filters found in the cache.

filter_cache_miss
The number of filters not found in the cache.

filter_cache_bypass_limit
Search filters that return more entries than this limit are not cached.

entry_cache_size
The maximum number of entries allowed in the cache.

entry_cache_current
The number of entries currently in the cache.

entry_cache_hit
The number of entries found in the cache.

entry_cache_miss
The number of entries not found in the cache.

acl_cache
A Boolean value indicating that the ACL cache is active (TRUE) or inactive (FALSE).

acl_cache_size
The maximum number of entries in the ACL cache.

currenttime
The current time on the server. The current time is in the format:

```
year month day hour:minutes:seconds GMT
```

Note: If expressed in local time the format is

```
day month date hour:minutes:seconds timezone year
```

starttime
The time the server was started. The start time is in the format:

```
year month day hour:minutes:seconds GMT
```

Note: If expressed in local time the format is

```
day month date hour:minutes:seconds timezone year
```

en_currentregs
The current number of client registrations for event notification.

en_notificationssent
The total number of event notifications sent to clients since the server was started.

Checking connection status

Perform the following to check the connection status of the server.

Using Web Administration:

Click **View server connections** in the navigation area. A table containing the following information is displayed:

DN Specifies a list of all the DNs connected to the server.

Start time
Specifies the date and time when the connections were made.

Status Specifies whether the connection is active or idle. A connection is considered active if it has any operations in progress.

Operations initiated
Specifies the number of operations requested since the connection was established.

Operations completed
Specifies the number of operations that have been completed for each connection.

Notes:

1. This table displays up to 20 connections at a time.
2. When a link is clicked in the navigation area a new connection is opened to the LDAP server. The number of connections increases, if a link is clicked repeatedly in the navigation area. These connections are closed only when the Java garbage collector begins its cleanup.

Using the command line:

```
ldapsearch -D<adminDN> -w <adminPW> -h <servername> -p <portnumber>
          -b cn=connections,cn=monitor -s base objectclass=*
```

This command returns information in the following format:

```
cn=connections,cn=monitor
connection=1632 : 2002-10-05 19:18:21 GMT  : 1 : 1 : CN=ADMIN :
connection=1487 : 2002-10-05 19:17:01 GMT  : 1 : 1 : CN=ADMIN :
```

Starting and stopping the server

You can use either of the following methods to start or stop the server.

Using Web Administration:

Note: The Admin daemon must be running.

The current status of the server, either started, stopped or started in configuration mode, is indicated by the icons in the upper left-hand corner of the server status area. The current status is also described in the first sentence of the work area, for example

```
The Directory Server is currently running
```

1. If you have not done so already, click **Server Administration** in the Web Administration navigation area and then click **Start/Stop/Restart Server** in the expanded list.
2. The message area displays the current state of the server (stopped, running, or running in configuration only mode). Depending on the state of the server, running or stopped, buttons are enabled for you to change the state of the server.

Table 3.

Server status	**Buttons available**
Stopped	Start, Close
Running	Stop, Restart, Close
Running in configuration only mode	Stop, Restart, Close

- If the server is running, click **Stop** to stop the server or **Restart** to stop and then start the server.
- If the server is stopped, click **Start** to start the server.
- Click **Close** to return to the Introduction panel.

3. A message displays when the server successfully starts or stops.

If you need to perform server configuration maintenance, select the **Start / Restart in configuration only mode** check box. In this mode only the system administrator can bind to the server. All other connections are refused until the server is restarted with DB2 backends enabled (the **Start / Restart in configuration only mode** check box deselected). See "Configuration only mode" for additional information.

Note: Configuration maintenance can be done while the server is running.

Using the command line:

Use the following command to start and stop the server:

Note: The admin daemon must be running.

```
ibmdirctl [-h <hostname>] [-D <adminDN>] [-w <password>] [-p <portnumber>]
  start|stop|restart|status -- [ibmslapd options]
```

See "ibmdirctl" on page 222 for additional information.

For Windows® systems use the previous command or:

1. From the desktop, double-click the **My Computer** icon.
2. Double-click the **Control Panel** icon.
3. Double-click the **Services** icon.
4. To start the server select IBM Directory V5.1 and click **Start**.
5. To stop the server select IBM Directory V5.1 and click **Stop**.

Configuration only mode

The IBM Directory Server supports LDAP access to the server's configuration settings. An administrator can use LDAP protocol to query and update the configuration for the server. This feature enables remote administration. In order for this access to be more robust and reliable, the server does not depend on successful initialization of the database backends. It is possible to start the server in configuration only mode with only the cn=configuration suffix active. In other words, as long as the configuration backend is available, the server starts and accepts LDAP requests. Configuration only mode gives an administrator remote access to the server even when errors are encountered during startup.

The following features are supported in configuration only mode:

- Access to the configuration file and log files.
- Auditing
- Event notification
- Kerberos
- SASL
- SSL

The following features are not supported in configuration only mode:

- Access to the database

- Changelog
- Password policy
- Replication
- Schema changes
- Transactions

Minimum requirements for configuration only mode

- The configuration file must be in the correct LDIF format and the server must be able to locate and read the file.
- The server must be able to read and load the schema according to the configuration file.
- The server must be able to load the configuration plugin.

How to start in configuration only mode

- Any failure during server startup causes the server to start in configuration only mode.
- Check the **Configuration only mode** when starting the server through the Web Administration Tool.
- Specify -a or -A on server startup.

```
ibmslapd -a
```

or

```
ibmdirctl -h <hostname> -D <adminDN> -w <password>-p <portnumber>
  start -- -a
```

Note: The -n or -N options prevent the server from starting, if the server is unable to start with the database backends (not in configuration only mode).

How to verify that the server is running in configuration only mode

- If the server has started in configuration only mode the || icon located between the stop and start icons is highlighted.
- Issue a search of the root dse for the attribute **ibm-slapdisconfigurationmode**. If set to true, the server is running in configuration only mode.

```
ldapsearch -s base -b " " objectclass=* ibm-slapdisconfigurationmode
```

Chapter 6. Setting server properties

This section discusses how to set up the various properties for your server.

Note: While the Web Administration Tool is the preferred method, updates to the server configuration file can be made through LDAP. The LDAP modify requests can be generated by:

- A C-application using the C-client provided with the IBM Directory Server
- A Java application using JNDI
- Any other interface that generates a standard V3 LDAP.

Examples that are provided use the ldapmodify command line utility.

ldapmodify can be run either in interactive mode or with inputs specified in a file. For most examples in this guide, the file contents to be used with the **ldapmodify** command are supplied. The general form of the command to use with these files is:

```
ldapmodify -D <adminDN> –w <password> –i <filename>
```

To update the server configuration settings dynamically, you need to issue the following **ldapexop** commands. This command updates all configuration settings that are dynamic:

```
ldapexop -D cn=root -w root -op readconfig -scope entire
```

This command updates a single setting.

```
ldapexop -D cn=root -w root -op readconfig -scope single <entry DN>
         <attribute>
```

The **ldapexop** command only updates those attributes that are dynamic. For other changes to take effect you must stop and restart the server. See “Dynamically-changed attributes” on page 278 for a list of the attributes that can be updated dynamically. See Chapter 15, “Command line utilities” on page 191 for more information about the **ldapmodify** and **ldapexop** commands.

Click the **Manage server properties** tab in the Web Administration navigation area to display the **Manage server properties** panel. This panel is displayed with the **General** tab preselected. The General panel has two read-only information fields which display the host name of the server and the version level of the IBM Directory Server that is installed on the machine.

This panel also has two modifiable required fields, **Unsecure port** (default value is 389) and **Secure port** (default value 636) that display the respective current port numbers. If you want to change the port settings, enter a number ranging from 1 through 65535.

You must restart the server for changes to take effect.

Notes:

1. Only the administrator is allowed to update the server configuration settings.
2. If you change the port setting for the server, you must also change the port settings for the server in the console. See “Managing the console” on page 10.

Setting Performance

Note: For the latest tuning information, see the IBM Directory Version 5.1 Tuning Guide located on the IBM Directory Server Web site located at http://www.ibm.com/software/network/directory/library/.

To change the search limits, and connections settings to enhance performance do the following:

Using Web Administration:

Expand the **Manage server properties** category in the navigation area of the Web Administration Tool, select the **Performance** tab.

1. Specify the **Maximum number of database connections**. This sets the number of DB2 connections used by the server. If your LDAP server receives a high volume of client requests or clients are receiving ″connection refused″ errors, you might see better results by increasing the setting the number of connections made to DB2 by the server. While there are no longer any limitations upon the number of connections you specify, each connection does consume resources. Consult the *IBM Directory Server Version 5.1 Tuning Guide* for the latest tuning recommendations for you system.
2. Select **Cache ACL information** to use the following ACL cache settings. This option must be selected in order for the other cache setting options on this panel to take effect.
3. Specify the **Maximum number of elements in ACL cache**. The default is 25,000.
4. Specify the **Maximum number of elements in entry cache**. The default is 25,000.
5. Specify the **Maximum number of elements in search filter cache**. The default is 25,000. The search filter cache consists of actual queries on the requested attribute filters and resulting entry identifiers that matched. On an update operation, all filter cache entries are invalidated.
6. Specify the **Maximum number of elements from a single search added to search filter cache**. If you select **Elements**, you must enter a number. The default is 100. Otherwise, select **Unlimited**. Search entries that match more entries that the number specified here are not added to the search filter cache.
7. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.
8. If you are setting the number of database connections, you must restart the server for the changes to take effect. If you were modifying only the settings, the server does not need to be restarted.

Using the command line:

To perform the same operations using the command line, issue the following command:

```
ldapmodify -D <AdminDN> -w<Adminpassword> -i<filename>
```

where <*filename*> contains:

```
dn: cn=Directory,cn=RDBM Backends,cn=IBM Directory,cn=Schemas,cn=Configuration
changetype: modify
replace: ibm-slapdDbConnections
ibm-slapdDbConnections:  15

dn: cn=Front End, cn=Configuration
changetype: modify
```

```
replace: ibm-slapdACLCache
ibm-slapdACLCache: TRUE
-
replace: ibm-slapdACLCacheSize
ibm-slapdACLCacheSize: 25000
-
replace: ibm-slapdEntryCacheSize
ibm-slapdEntryCacheSize: 25000
-
replace: ibm-slapdFilterCacheSize
ibm-slapdFilterCacheSize: 25000
-
replace: ibm-slapdFilterCacheBypassLimit
ibm-slapdFilterCacheBypassLimit: 100
```

To update the settings dynamically issue the following **ldapexop** command:

```
ldapexop -D cn=root -w root -op readconfig -scope entire
```

The **ldapexop** command only updates those attributes that are dynamic. For other changes to take effect you must stop and restart the server. See "Dynamically-changed attributes" on page 278 for a list of the attributes that can be updated dynamically.

Setting Searches

You can set search parameters to control users' search capabilities, such as paged and sorted searching.

Paged results allow you to manage the amount of data returned from a search request. You can request a subset of entries (a page) instead of receiving all the results at once. Subsequent search requests display the next page of results until the operation is canceled or the last result is returned. Sorted search allows a client to receive search results sorted by a list of criteria, where each criteria represents a sort key. This moves the responsibility of sorting from the client application to the server, where it might be done more efficiently.

Using Web Administration:

Expand the **Manage server properties** category in the navigation area of the Web Administration Tool, select the **Search settings** tab.

1. Set the **Search size limit**. Click either the **Entries** or the **Unlimited** radio button. If you select **Entries**, you need to specify in the field the maximum number of entries a search returns. If more entries fit the search criteria, they are not returned. This limit does not apply to the administrator.
2. Set the **Search time limit**. Click either the **Seconds** or the **Unlimited** radio button. If you select **Entries**, you need to specify in the field the maximum amount of time the server spends processing the request. This limit does not apply to the administrator.
3. To restrict search sorting capabilities to administrators, select the **Only allow administrators to sort searches** checkbox.
4. To restrict search paging capabilities to administrators, select the **Only allow administrators to page searches** checkbox.
5. To restrict ACL access to administrators and owners, select the **Only allow administrators and owners to see ACLs on entries** checkbox

6. Specify the number of entries to display per page of paged search results. This value determines the maximum number of entries allowed per page. If the number of entries per page of paged searches exceeds the search size limit, an error occurs.
7. Specify in seconds the time to wait (idle time out) for paged searches. Paged searches require an open connection between the LDAP server and the DB2 database where the LDAP data is stored. The idle time out administrative limit is designed to age out DB2 database connections held open for paged results search requests.
8. Specify the maximum number of concurrent paged searches allowed by the server at any given time.
9. Specify the maximum number of attributes used for sorting in sorted searches.
10. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.

See “Extended controls for search” on page 29 for additional information about searches.

Using the command line:

To perform the same operations using the command line, issue the following command:

```
ldapmodify -D <AdminDN> -w<Adminpassword> -i<filename>
```

where *<filename>* contains:

```
dn: cn=Configuration
changetype: modify
replace: ibm-slapdTimeLimit
ibm-slapdTimeLimit:  900
-
replace: ibm-slapdSizeLimit
ibm-slapdSizeLimit:  500

dn: cn=Directory,cn=RDBM Backends,cn=IBM Directory,cn=Schemas,cn=Configuration
changetype: modify
replace: ibm-slapdPagedResAllowNonAdmin
ibm-slapdPagedResAllowNonAdmin: TRUE
-
replace: ibm-slapdPagedResLmt
ibm-slapdPagedResLmt: 3
-
replace: ibm-slapdPageSizeLmt
ibm-slapdPageSizeLmt: 201
-
replace: ibm-slapdSortKeyLimit
ibm-slapdSortKeyLimit: 3
-
replace:
ibm-slapdSortSrchAllowNonAdmin: TRUE

dn: cn=Front End, cn=Configuration
changetype: modify
replace: ibm-slapdIdleTimeOut
ibm-slapdIdleTimeOut:  300
```

To update the settings dynamically issue the following **ldapexop** command:

```
ldapexop -D cn=root -w root -op readconfig -scope entire
```

See “ldapsearch” on page 214 for information on how to perform searches using the command line.

Extended controls for search

The search function searches for a filter match on only the first 240 bytes of an attribute if indexing is enabled for that attribute. Additionally, if sort is specified on a search request, the server sorts the entries found by the search using only the first 240 bytes. Any end user or client application needs to take into consideration that a match for a search filter that exists in a value after the first 240 bytes might not be returned to the client depending on whether indexing is enabled for that table.

Note: This restriction is specific to the IBM Directory Server. IBM LDAP servers on other platforms, including z/OS™ and OS/400® might have different restrictions. Consult the documentation for each platform to determine restrictions.

The administrator can tell if indexing has been enabled for an attribute by looking at the attribute definition in the Web Administration tool, or by looking at the attribute definition returned by a search of cn=schema. When viewing an attribute definition in the Web Administration tool, the IBM attributes fields includes one of the following index keywords: APPROX, EQUALITY, ORDERING, or SUBSTR. If the ldapsearch utility is used, the ibmattributetypes value contains the same keywords. For example, the 'cn' attribute has all indexes defined:

```
attributetypes=( 2.5.4.3 NAME ( 'cn' 'commonName' ) DESC 'This is the X.500
                commonName attribute, which contains a name of an object.
                If the object corresponds to a person, it is typically the
                persons full name.' SUP 2.5.4.41 EQUALITY 2.5.13.2
                ORDERING 2.5.13.3 SUBSTR 2.5.13.4 )
ibmattributetypes=( 2.5.4.3 DBNAME ( 'cn' 'cn' ) ACCESS-CLASS NORMAL LENGTH
                   256 EQUALITY ORDERING SUBSTR APPROX )
```

Sorted search control

Sorted Search Results provides sort capabilities for LDAP clients that have limited or no sort functionality. Sorted Search Results allows an LDAP client to receive search results sorted based on a list of criteria, where each criteria represents a sort key. The sort criteria includes attribute types, matching rules, and descending order. The server uses this criteria to sort search results before returning them. This moves the responsibility of sorting from the client application to the server, where it might be done much more efficiently. For example, a client application might want to sort the list of employees at their Grand Cayman site by surname, common name, and telephone number. Instead of building the search list twice so it can be sorted (once at the server and then again at the client after all the results are returned), the search list is built only once, and then sorted, before returning the results to the client application.

The server sorts search entries based on attributes and by default allows a maximum of three sort keys (attribute names) per search operation. To change the value of this administrative limit, change the following line, **ibm-slapdSortKeyLimit: 3**, in the ibmslapd.conf file. See “Setting Searches” on page 27 for information on how to do this. If the line does not exist, add it to set the new maximum (if the line does not exist, the server is using the default value). Additionally, because sorting search results before returning them uses more server resources than simply returning them, by default only requests from users binding with Administrator authority are honored by the server. If you would like the server to honor requests from all others, including those binding anonymously, you can configure the server to do so. To honor sorted search requests submitted using non-Administrator bind, change the following line, **ibm-slapdSortSrchAllowNonAdmin: FALSE**, in the ibmslapd.conf file. See “Setting

Searches" on page 27. If the line does not exist, add it with a value of TRUE to allow non-Administrator bind. See"Adding search attributes example" on page 32.

The LDAP server returns all referrals to the client at the end of a search request, the same as is done today. It is up to the application using the client services to decide whether to set the criticality of the sorted search request, and to handle a lack of support of those controls on referral servers as appropriate based on the application. Additionally, the LDAP server does not ensure that the referral server supports the sorted search control. Multiple lists could be returned to the client application, some not sorted. It is the client application's decision as to how to best present this information to the end user. Possible solutions include: combine all referral results before presenting to the end user; show multiple lists and the corresponding referral server host name; take no extra steps and show all results to the end user as they are returned from the server. The client application must turn off referrals to get one truly sorted list, otherwise when chasing referrals with sorted search controls specified, unpredictable results might occur.

It is important to note when taking advantage of the server sorted search results that:

- The server takes advantage of the underlying DB2 database to perform sorting of search results. This means that there might be different sorted search results based on the data code page for the database (especially if your database code page is UTF-8).
- Matching rules specified for a sort key attribute are ignored by the server. At this time, matching rules are not supported by the server.
- There is no support for multi-server sorting (referrals). The server cannot guarantee that referred servers support sorted search results.

More information about the server side sorted search control can be found in RFC 2891. The control OID for sorted search results is 1.2.840.113556.1.4.473, and is included in the RootDSE information as a supported control.

Simple paged results

Simple Paged Results provides paging capabilities for LDAP clients that want to receive just a subset of search results (page) instead of the entire list. The next page of entries is returned to the client application for each subsequent paged results search request submitted by the client until the operation is canceled or the last result is returned. The server ignores a simple paged results request if the page size is greater than or equal to the sizeLimit value for the server because the request can be satisfied in a single operation.

Because paging of search results holds server resources throughout the life of the simple paged results request, there are several new administrative limits employed to ensure that server resources cannot be abused, or misused, through the use of simple paged results search requests.

ibm-slapdPagedResAllowNonAdmin

By default only requests from users binding with Administrator authority is honored by the server. If you would like the server to honor requests from all others, including those binding anonymously, you can configure the server to do so. To honor simple paged results search requests submitted using non-Administrator bind, change the following line, **ibm-slapdPagedResAllowNonAdmin: FALSE**, in the ibmslapd.conf file. See "Setting Searches" on page 27. If the line does not exist, add it with a value of TRUE to allow non-Administrator bind. See"Adding search attributes example" on page 32.

ibm-slapdPagedResLmt

By default, the server allows a maximum of three outstanding simple paged results operations at any given time. To ensure the fastest response for subsequent simple paged results request, the server holds a database connection open throughout the life of the search request until the user cancels the simple paged results request, or the last result is returned to the client application. This administrative limit is designed to ensure that other operations being handled by the server are not denied service because all database connections are in use by outstanding simple paged results search requests. It is recommended that you set the **ibm-slapdPagedResLmt** value lower that the maximum number of database connections for your server. To change the value of this administrative limit, change the following line, **ibm-slapdPagedResLmt: 3**, in the ibmslapd.conf file. See "Setting Searches" on page 27. If the line does not exist add it to set the new maximum (if the line does not exist, the server is using the default value). See"Adding search attributes example" on page 32.

ibm-slapdPagedSizeLmt

By default, the server returns a maximum of 50 search results for a single page. If you would like to set a different page size maximum, you can change the following line, **ibm-slapdPagedSizeLmt: 50**, in the ibmslapd.conf file. See "Setting Searches" on page 27. If the line does not exist, add it with the value of the new maximum page size. See"Adding search attributes example" on page 32.

ibm-slapdIdleTimeOut

The idle time out administrative limit is designed to age out DB2 database connections held open for simple paged results search requests. The default idle time for simple paged results request is 500 seconds. For example, if a client application were to pause for 510 seconds between pages, the server would age out the request in order to free the database connection for use by other server operations. The server returns the appropriate error to the client application for the next simple paged results request submitted, at which point the client application needs to restart the simple paged results request. The idle timer for each simple paged results request is restarted after every page returned to the client application. The server checks for aged out simple paged results request every 5 seconds, so if you set the value of **ibm-slapdIdleTimeOut** value lower than 5 seconds, you still have to wait 5 seconds for the simple paged results requests to be aged out. To change the value of this administrative limit, change the following line, **ibm-slapdIdleTimeOut: 300**, in the ibmslapd.conf file. See "Setting Searches" on page 27. If the line does not exist add it to set the new maximum (if the line does not exist, the server is using the default value). See"Adding search attributes example" on page 32.

The LDAP server returns all referrals to the client at the end of a search request, the same as a search without any controls. That means that if the server has 10 pages of results returned, all the referrals are returned on the 10th page, not at the end of each page. When chasing referrals, the client application needs to send in an initial paged results request, with the cookie set to null, to each of the referral servers. It is up to the application using the client services to decide whether or not to set the criticality as to the support of paged results, and to handle a lack of support of this control on referral servers as appropriate based on the application. Additionally, the LDAP server does not ensure that the referral server supports paged results controls. Multiple lists could be returned to the client application, some not paged. It is at the client application's decision as to how to best present this information to the end user. Possible solutions include: combine all referral results before presenting to the end user; show multiple lists and the corresponding referral server host name; take no extra steps and show all results to the end user as they are returned from the server. The client application must turn off referrals to get one truly paged list, otherwise when chasing referrals with the paged results search control specified, unpredictable results may occur.

More information about the server side simple paged results control can be found in RFC 2686. The control OID for simple paged results is 1.2.840.113556.1.4.319, and is included in the RootDSE information as a supported control.

Adding search attributes example

```
ldapmodify -D <AdminDN> -w<Adminpassword> -i<filename>
```

where <*filename*> contains:

```
dn: cn=Directory,cn=RDBM Backends,cn=IBM Directory,cn=Schemas,cn=Configuration
changetype: add
add: ibm-slapdSortSrchAllowNonAdmin
ibm-slapdSortSrchAllowNonAdmin: TRUE
-
add: ibm-slapdSortKeyLimit
ibm-slapdSortKeyLimit: 3
-
add: ibm-slapdPagedResAllowNonAdmin
ibm-slapdPagedResAllowNonAdmin: TRUE
-
add: ibm-slapdPagedResLmt
ibm-slapdPagedResLmt: 3
-
add: ibm-slapdPagedSizeLmt
ibm-slapdPagedSizeLmt: 50
-
add: ibm-slapdIdleTimeOut
ibm-slapdIdleTimeOut: 300
```

To update the settings dynamically issue the following **ldapexop** command:

```
ldapexop -D cn=root -w root -op readconfig -scope entire
```

Transaction Support

Transaction processing enables an application to group a set of entry updates together in one operation. Normally each individual LDAP operation is treated as a separate transaction with the database. Grouping operations together is useful when one operation is dependent on another operation because if one of the operations fails, the entire transaction fails. Transaction settings determine the limits on the transaction activity allowed on the server.

Enabling transaction support

To enable transaction support:

Using Web Administration:

Expand the **Manage server properties** category in the navigation area of the Web Administration Tool, select the **Transactions** tab.

1. Select the **Enable transaction processing** check box to enable transaction processing. If **Enable transaction processing** is disabled, all other options on this panel, such as **Maximum number of operations per transaction** and **Pending time limit**, are ignored by the server.

2. Set the **Maximum number of transactions**. Click either the **Transactions** or the **Unlimited** radio button. If you select **Transactions**, you need to specify in the field the maximum number of transactions. The maximum number of transactions is 2,147,483,647. The default setting is 20 transactions.
3. Set the **Maximum number of operations per transaction**. Click either the **Operations** or the **Unlimited** radio button. If you select **Operations**, you need to specify in the field the maximum number of operations allowed for each transaction. The maximum number of operations is 2,147,483,647. The smaller the number, the better the performance. The default is 5 operations.
4. Set the **Pending time limit**. This selection sets the maximum timeout value of a pending transaction in seconds. Click either the **Seconds** or the **Unlimited** radio button. If you select **Seconds**, you net to specify in the field the maximum number of seconds allowed for each transaction. The maximum number of seconds is 2,147,483,647. Transactions left uncompleted for longer than this time are cancelled (rolled back). The default is 300 seconds.
5. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.
6. If you have enabled transaction support, you must restart the server for the changes to take effect. If you were modifying only the settings, the server does not need to be restarted.

Using the command line:

To perform the same operations using the command line, issue the following command:

```
ldapmodify -D <AdminDN> -w<Adminpassword> -i<filename>
```

where *<filename>* contains:

```
dn: cn=Transaction,cn=Configuration
changetype: modify
replace: ibm-slapdTransactionEnable
ibm-slapdTransactionEnable: TRUE
-
replace: ibm-slapdMaxNumOfTransactions
ibm-slapdMaxNumOfTransactions: 20
-
replace: ibm-slapdMaxOpPerTransaction
ibm-slapdMaxOpPerTransaction: 5
-
replace: ibm-slapdMaxTimeLimitOfTransactions
ibm-slapdMaxTimeLimitOfTransactions: 300
```

To update the settings dynamically issue the following **ldapexop** command:

```
ldapexop -D cn=root -w root -op readconfig -scope entire
```

The **ldapexop** command only updates those attributes that are dynamic. For other changes to take effect you must stop and restart the server. See "Dynamically-changed attributes" on page 278 for a list of the attributes that can be updated dynamically.

Disabling transaction support

To disable transaction processing:

Using Web Administration:

1. Deselect the **Enable transaction processing** check box to enable transaction processing.

2. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.
3. You must restart the server for the changes to take effect.

Using the command line:

To perform the same operations using the command line, issue the following command:

```
ldapmodify -D <AdminDN> -w<Adminpassword> -i<filename>
```

where *<filename>* contains:

```
dn: cn=Transaction,cn=Configuration
changetype: modify
replace: ibm-slapdTransactionEnable
ibm-slapdTransactionEnable: False
```

You must restart the server for the changes to take effect.

See the *IBM Directory Server Version 5.1 C-Client SDK Programming Reference* for more information about transaction support.

Event notification

The event notification function allows a server to notify a registered client that an entry in the directory tree has been changed, added or deleted. This notification is in the form of an unsolicited message.

When an event occurs, the server sends a message to the client as an LDAP v3 unsolicited notification. The messageID is 0 and the message is in the form of an extended operation response. The responseName field is set to the registration OID. The response field contains the unique registration ID and a timestamp for when the event occurred. The time field is in UTC time format.

Note: When a transaction occurs, the event notifications for the transaction steps cannot be sent until the entire transaction is completed.

Enabling event notification

To enable event notification:

Using Web Administration:

Expand the **Manage server properties** category in the navigation area of the Web Administration Tool, select the **Event notification** tab.

1. Select the **Enable event notification** check box to enable event notification. If **Enable event notification** is disabled, the server ignores all other options on this panel.
2. Set the **Maximum registrations per connection**. Click either the **Registrations** or the **Unlimited** radio button. If you select **Registrations**, you need to specify in the field the maximum number of registrations allowed for each connection. The maximum number of transactions is 2,147,483,647. The default setting is 100 registrations.
3. Set the **Maximum registrations total**. This selection sets how many registrations the server can have at any one time. Click either the **Registrations** or the **Unlimited** radio button. If you select **Registrations**, you need to specify in the

field the maximum number of registrations allowed for each connection. The maximum number of transactions is 2,147,483,647. The default number of registrations is **Unlimited**.
4. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.
5. If you have enabled event notification, you must restart the server for the changes to take effect. If you were modifying only the settings, the server does not need to be restarted.

Using the command line:

To perform the same operations using the command line, issue the following command:

```
ldapmodify -D <AdminDN> -w<Adminpassword> -i<filename>
```

where <*filename*> contains:

```
dn: cn=Event Notification,cn=Configuration
changetype: modify
replace: ibm-slapdEnableEventNotification
ibm-slapdEnableEventNotification:  TRUE
-
replace: ibm-slapdMaxEventsPerConnection
ibm-slapdMaxEventsPerConnection:  100
-
replace: ibm-slapdMaxEventsTotal
ibm-slapdMaxEventsTotal:  0
```

If you have enabled event notification, you must restart the server for the changes to take effect. If you were modifying only the settings, the server does not need to be restarted.

To update the settings dynamically issue the following **ldapexop** command:

```
ldapexop -D cn=root -w root -op readconfig -scope entire
```

The **ldapexop** command only updates those attributes that are dynamic. For other changes to take effect you must stop and restart the server. See "Dynamically-changed attributes" on page 278 for a list of the attributes that can be updated dynamically.

Disabling event notification

To disable event notification:

Using Web Administration:

1. Deselect the **Enable event notification** check box to enable transaction processing.
2. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.
3. You must restart the server for the changes to take effect.

Using the command line:

To perform the same operations using the command line, issue the following command:

```
ldapmodify -D <AdminDN> -w<Adminpassword> -i<filename>
```

where <*filename*> contains:

```
dn: cn=Event Notification,cn=Configuration
changetype: modify
replace: ibm-slapdEnableEventNotification
ibm-slapdEnableEventNotification:  FALSE
```

You must restart the server for the changes to take effect.

See the *IBM Directory Server Version 5.1 C-Client SDK Programming Reference* for more information about event notification.

Understanding suffixes

A suffix is a DN that identifies the top entry in a locally held directory hierarchy. Because of the relative naming scheme used in LDAP, this DN is also the suffix of every other entry within that directory hierarchy. A directory server can have multiple suffixes, each identifying a locally held directory hierarchy, for example, o=ibm,c=us.

Note: The specific entry that matches the suffix must be added to the directory.

Entries to be added to the directory must have a suffix that matches the DN value, such as 'ou=Marketing,o=ibm,c=us'. If a query contains a suffix that does not match any suffix configured for the local database, the query is referred to the LDAP server that is identified by the default referral. If no LDAP default referral is specified, an Object does not exist result is returned.

Creating or adding suffixes

To create or add a suffix use one of the following methods.

Using Web Administration:

Expand the **Manage server properties** category in the navigation area of the Web Administration Tool, select the **Suffixes** tab.

1. Enter the Suffix DN, for example, **c=Italy**. The maximum is 1000 characters for a suffix.
2. Click **Add**.
3. Repeat this process for as many suffixes as you want to add.
4. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.

Using the command line:

To perform the same operations using the command line, issue the following command:

```
ldapmodify -D <AdminDN> -w<Adminpassword> -i<filename>
```

where <*filename*> contains:

```
DN: cn=Directory, cn=RDBM Backends, cn=IBM Directory, cn=Schemas, cn=Configuration
changetype: modify
add: ibm-slapdSuffix
ibm-slapdSuffix: <suffixname>
ibm-slapdSuffix: <suffix2>
ibm-slapdSuffix: <suffix3>
```

To update the settings dynamically issue the following **ldapexop** command:

```
ldapexop -D cn=root -w root -op readconfig -scope single "cn=Directory,
    cn=RDBM Backends,cn=IBM Directory,cn=Schemas,cn=Configuration" ibm-slapdSuffix
```

Removing a suffix

To remove a suffix use one of the following methods:

Using Web Administration:

Expand the **Manage server properties** category in the navigation area of the Web Administration Tool, select the **Suffixes** tab.

1. From the **Current suffix DNs** section, select the suffixes you want to remove.
2. Click **Remove**.
3. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.

Using the command line:

To perform the same operations using the command line, issue the following command:

```
ldapmodify -D <AdminDN> -w<Adminpassword> -i<filename>
```

where *<filename>* contains:

```
DN: cn=Directory, cn=RDBM Backends, cn=IBM Directory, cn=Schemas, cn=Configuration
changetype: modify
delete: ibm-slapdSuffix
ibm-slapdSuffix: <suffixname>
ibm-slapdSuffix: <suffix2>
ibm-slapdSuffix: <suffix3>
```

You must restart the server for the change to take effect.

To update the settings dynamically issue the following **ldapexop** command:

```
ldapexop -D cn=root -w root -op readconfig -scope single "cn=Directory,
    cn=RDBM Backends,cn=IBM Directory,cn=Schemas,cn=Configuration" ibm-slapdSuffix
```

Note: You can also use the configuration utilities, **ldapcfg**, **ldapucfg**, and **ldapxcfg** to add and remove suffixes. See the *IBM Directory Server Version 5.1 Installation and Configuration Guide* for more information about these utilities.

Using referrals

Referrals provide a way for servers to refer clients to additional directory servers. A referral specifies the URL of an alternate LDAP server. This alternate server handles any requests for objects that are not found within any of the subtrees of the current LDAP server. 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

Some of the advantages of using referrals are the ability to:

- 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

Note: On the Linux, Solaris, and HP-UX platforms, if a client hangs while chasing referrals, ensure that the environment variable LDAP_LOCK_REC has been set in your system environment. No specific value is required.

```
set LDAP_LOCK_REC=anyvalue
```

See “Referrals” for more detailed information and information on how to create referrals manually.

Using Web Administration:

Using the Web Administration utility is the recommended method to create and remove referrals.

Creating referrals

Expand the **Manage server properties** category in the navigation area of the Web Administration Tool, select the **Referrals** tab.

1. Enter a referral URL beginning with the initial value **ldap://**. The maximum length is 32700 characters.
2. Click **Add**.
3. Repeat this process for as many referrals as you want to add.
4. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel and return to the Introduction panel without making any changes.

You must restart the server for the changes to take effect.

Removing referrals

Expand the **Manage server properties** category in the navigation area of the Web Administration Tool, select the **Referrals** tab.

1. From the **Current referrals** section, select the referral you want to remove.
2. Click **Remove**.
3. A confirmation panel is displayed. Click **OK** to remove the referral or click **Cancel** to return to the previous panel without making any changes.
4. Repeat this process for as many referrals as you want to remove.
5. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel and return to the Introduction panel without making any changes.

You must restart the server for the changes to take effect.

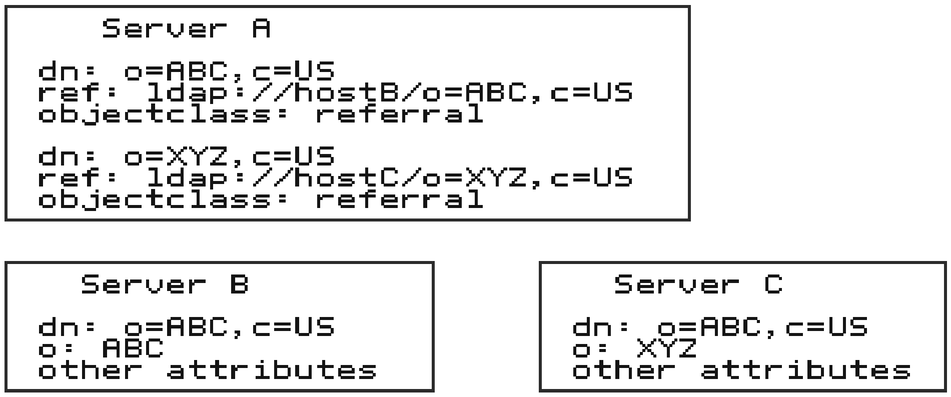
Referrals

This section describes how to use the referral object class and the ref attribute to construct entries in an LDAP directory containing references to other LDAP directories. This section also describes how to associate multiple servers using referrals and an example of this.

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 an entry maintained in the referenced server.

Creating entries: Following is an example configuration that illustrates the use of the ref attribute.



```
Server A

dn: o=ABC,c=US
ref: ldap://hostB/o=ABC,c=US
objectclass: referral

dn: o=XYZ,c=US
ref: ldap://hostC/o=XYZ,c=US
objectclass: referral
```

```
Server B

dn: o=ABC,c=US
o: ABC
other attributes
```

```
Server C

dn: o=ABC,c=US
o: XYZ
other attributes
```

Figure 1. Example of using the referral 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 a reference to Server C.

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 (closer to the root of the hierarchy) information and set the default referral to point back to its parent server.

Associating servers with referrals

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.

Note: Referral objects can be seen from command line LDAP utilities by specifying the **-M** option.

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 that 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 the value of the objectclass ″referral″.

ref: Specifies the LDAP Web address of the server. This Web address consists of the ldap:// identifier, the hostname:port, and a DN. The identifier can be either a host name string or a TCP/IP address. The DN requires a slash (/) before it to delimit it from the hostname:port, and should match the DN of the referral object. It is highly recommended that the DN specified in the value of the referral attribute match the DN of the referral object. Typically, it is an entry in a naming context at or below the naming context held by the referencing server.

```
dn:           o=IBM,c=US
objectclass:  referral
ref:          ldap://9.130.25.51:389/o=IBM,c=US
```

Defining the default referral: Define a default referral to reference a directory on another 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 that possibly serves a disjoint portion of the namespace

Note: The default referral LDAP URL does not include the DN portion. It includes only the ldap:// identifier and the hostname:port.

For example:

```
# referral
dn: cn=Referral, cn=Configuration
objectclass: top
objectclass: ibm-slapdReferral
cn: Referral
ibm-slapdReferral: ldap://dcecds3.endicott.ibm.com:389
ibm-slapdReferral: ldap://<additional hostname:port>
ibm-slapdReferral: ldap://<additional hostname:port>
ibm-slapdReferral: ldap://<additional hostname:port>
```

Deleting default referrals: To delete a single default referral, for example, austin.ibm.com:389, issue the command:

```
ldapmodify -D <adminDN> -w <adminPW> -f <filename>
```

where *<filename>* contains:

```
dn: cn=referral, cn= configuration
changetype: modify
delete: ibm-slapdReferral
ibm-slapdReferral: ldap://referral.austin.ibm.com:398
```

To delete all default referrals:

```
ldapdelete -D <adminDN> -w <adminPW> "cn=referral,cn=configuration"
```

Binding with a distributed namespace

When performing searches, the same DN that was used to bind or log in to the original server is used to bind to the referred-to server, unless the IBM Directory application is designed to modify the bind DN and credentials. The correct access must be set up for the same DN to be able to bind to both servers for chasing the referrals. See “Logging on to the Web Administration Tool” on page 11 for additional information.

An example of distributing the namespace through referrals

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 Raleigh, IBM HQ
   ```
2. Set up multiple servers, each containing portions of the namespace.

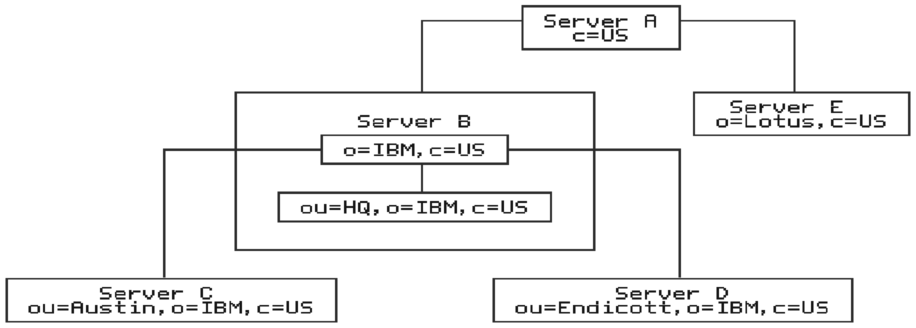


Figure 2. Setting up the servers

Server descriptions:

Server A
A server used to locate other servers in the U.S. With no other knowledge, clients can come here first to locate information for anyone in the U.S.

Server B
A hub for all data pertaining to IBM in the U.S. Holds all HQ information directly. Holds all knowledge (referrals) of where other IBM data resides.

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 descendants in other servers.

```
dn: o=IBM,c=US                                 ←——→Pointer to Server B
objectClass: referral
ref: ldap://ibm.com:389/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 3. Server A database (LDIF input)

Servers can also define a default referral, which is used to point to a ″more knowledgeable″ server for anything that is not underneath them in the namespace.

Note: The default referral LDAP Web address does not include the DN portion.

Following is an arrangement of the same five servers, showing the referral objects in the database as well as the default referrals that are used for superior references.

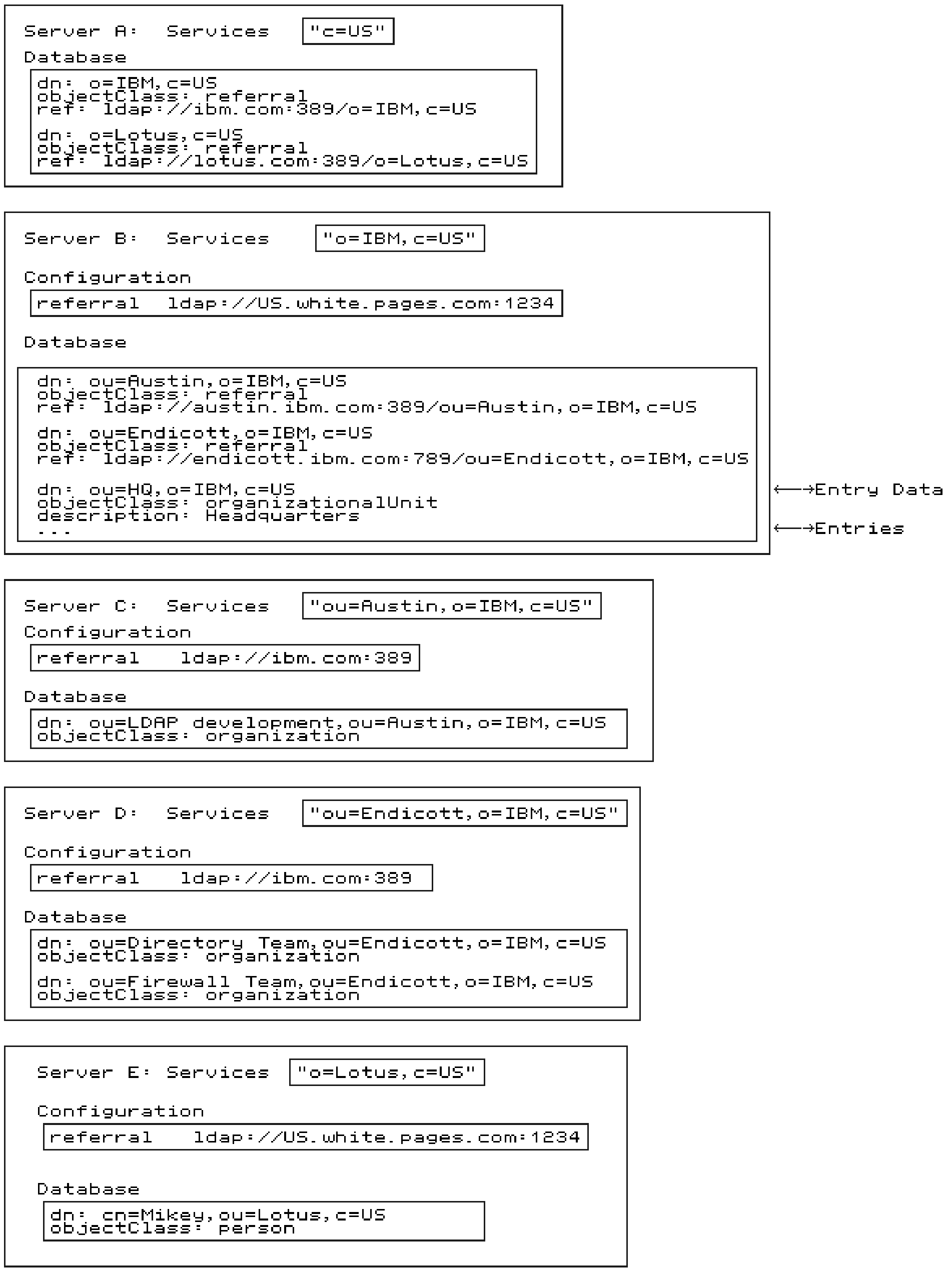

```
Server A:  Services   "c=US"
Database
dn: o=IBM,c=US
objectClass: referral
ref: ldap://ibm.com:389/o=IBM,c=US

dn: o=Lotus,c=US
objectClass: referral
ref: ldap://lotus.com:389/o=Lotus,c=US

Server B:  Services   "o=IBM,c=US"

Configuration
referral  ldap://US.white.pages.com:1234

Database

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

dn: ou=HQ,o=IBM,c=US                                     ←→Entry Data
objectClass: organizationalUnit
description: Headquarters
...                                                      ←→Entries

Server C:  Services   "ou=Austin,o=IBM,c=US"
Configuration
referral   ldap://ibm.com:389

Database
dn: ou=LDAP development,ou=Austin,o=IBM,c=US
objectClass: organization

Server D:  Services   "ou=Endicott,o=IBM,c=US"

Configuration
referral   ldap://ibm.com:389

Database
dn: ou=Directory Team,ou=Endicott,o=IBM,c=US
objectClass: organization
dn: ou=Firewall Team,ou=Endicott,o=IBM,c=US
objectClass: organization

Server E: Services  "o=Lotus,c=US"

Configuration
referral   ldap://US.white.pages.com:1234

Database
dn: cn=Mikey,ou=Lotus,c=US
objectClass: person
```

Figure 4. Referral example summary

Chapter 7. Securing the directory

This section describes the steps necessary for keeping the data in your directory secure.

Setting Secure Sockets Layer (SSL)

The IBM Directory Server has the ability to protect LDAP access by encrypting data with Secure Sockets Layer (SSL) security. When using SSL to secure LDAP communications with the IBM Directory, both server authentication and client authentication are supported. To use SSL you must have GSKit installed on your system. See “Secure Sockets Layer” on page 47 and “Using gsk6ikm” on page 52 for more information.

Configuring SSL settings

Expand the **Manage security properties** category in the navigation area of the Web Administration Tool, select the **SSL settings** tab.

1. Enable the type of SSL connections, select one of the following radio buttons:

   SSL Off
   Clients are permitted to conduct only unsecure communications.

   SSL On
   Clients are permitted to conduct either secure or unsecure communications.

   SSL Only
   Clients are not permitted to conduct unsecured communications. This is the most secure way to configure your server.
2. Select the authentication method.

   Note: You must distribute the server certificate to each client. For server and client authentication you also must add the certificate for each client to the server’s key database.

   Select the radio button for either:

   Server authentication

   For server authentication the IBM Directory Server supplies the client with the IBM Directory 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 IBM Directory Server and the client application.

   For server authentication to work, the IBM Directory Server must have a private key and associated server certificate in the server’s key database file.

   Server and client authentication
   This type of authentication provides for two-way authentication between the LDAP client and the LDAP server. With client authentication, the LDAP client must have a digital certificate (based on the X.509 standard). This digital certificate is used to authenticate the LDAP client to the IBM Directory Server. See “Client authentication” on page 51.

3. Specify the secure port number to use. The default port is 636.
4. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.
5. You must stop and restart both the IBM Directory Server and the administration daemon for the changes to take effect.
   a. Stop the server.
   b. Stop the administration daemon.
      - For Unix-based systems issue the commands:
        ```
        PS -ef | grep ibmdiradm
        kill -p <pid obtained by previous command>
        ```
      - For Windows-based systems, use **Control panel ->Services**, select **IBM Directory Admin Daemon**, click **Stop**.
   c. Start the administration daemon.
      - For Unix-based systems issue the command:
        ```
        ibmdiradm
        ```
      - For Windows-based systems, use **Control panel ->Services**, select **IBM Directory Admin Daemon**, click **Start**.
   d. Start the server.

Using the command line:

To use the command line to configure SSL communications, issue the command:

```
ldapmodify -D <AdminDN> -w <Adminpassword> -i <filename>
```

where <*filename*> contains:

```
dn: cn=SSL,cn=Configuration
changetype: modify
replace: ibm-slapdSslAuth
ibm-slapdSslAuth: serverAuth | serverClientAuth
-
replace: ibm-slapdSecurity
ibm-slapdSecurity: none | SSL | SSlOnly
```

You must restart the server and the administration daemon for the changes to take effect.

Configuring for SSL certificate revocation verification

If you have selected to use server and client authentication in your SSL settings, you might want to configure your server to check for revoked or expired certificates.

When a client sends an authenticated request to a server, the server reads the certificate and sends a query to an LDAP server with a list that contains revoked certificates. If the client certificate is not found in the list, communications between the client and server are allowed over SSL. If the certificate is found, communications are not allowed.

To configure SSL certificate revocation verification, expand the **Manage security properties** category in the navigation area of the Web Administration Tool, select the **Certificate revocation** tab.

1. Enter the name of the server that contains certificates that have been revoked . This server is designated by the certificate granting authority (CA) that you use, for example Verisign. The format of the host name is hostName.domainName, for example, myserver.myorg.com.

2. Enter the port used to communicate with the server, for example 389.
3. Enter the DN used to bind to the verifying server, for example cn=root. This is optional if the verifying server allows anonymous searches for certificate revocation lists (CRLs).
4. Enter the password associated with the bind DN. This is required if you specified a DN.
5. Type the bind password again to confirm that there are no typographical errors.
6. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.

Note: Expired certificates are not included in the list because the expiration date is contained in the certificate itself.

Using the command line:

To use the command line to configure for SSL certificate revocation verification, issue the command:

```
ldapmodify -D <AdminDN> -w <Adminpassword> -i <filename>
```

where *<filename>* contains:

```
dn: cn=CRL,cn=SSL,cn=Configuration
changetype: modify
replace: ibm-slapdCrlHost
ibm-slapdCrlHost: <newhostname>
-
replace: ibm-slapdCrlPassword
ibm-slapdCrlPassword: <password>
-
replace: ibm-slapdCrlPort
ibm-slapdCrlPort: <portnumber>
-
replace: ibm-slapdCrlUser
ibm-slapdCrlUser: <username>
```

You must restart the server and the administration daemon for the changes to take effect.

Setting the SSL key database

Expand the **Manage security properties** category in the navigation area of the Web Administration Tool, select the **SSL key database** tab.

1. Specify the **Key label**. This administrator-defined key label indicates what part of the key database to use.
2. Specify the **Key database path and file name**. This is the fully-qualified file specification of the key database file. If a password stash file is defined, it is assumed to have the same file specification, with an extension of **.sth**.
3. Specify the **Key password**. If a password stash file is not being used, the password for the key database file must be specified here. Then specify the password again in the **Confirm password** field.
4. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.

Note: In order for the server to use this file, it must be readable by the userid **ldap**. See “File permissions” on page 239.

Using the command line:

To use the command line to set the SSL key database, issue the command:

```
ldapmodify -D <AdminDN> -w <Adminpassword> -i <filename>
```

where <*filename*> contains:

```
dn: cn=SSL,cn=Configuration
changetype: modify
replace: ibm-slapdSSLKeyDatabase
ibm-slapdSSLKeyDatabase: <databasename>
-
replace: ibm-slapdSSLKeyDatabasePW
ibm-slapdSSLKeyDatabasePW: <password>
-
replace: ibm-slapdSslKeyRingFile
ibm-slapdSslKeyRingFile: <filename>
-
replace: ibm-slapdSslKeyRingFilePW
ibm-slapdSslKeyRingFilePW: <password>
```

You must restart the server and the administration daemon for the changes to take effect.

Setting the SSL level of encryption

By default the SSL version of IBM Directory Server uses the following list of ciphers when performing cipher negotiation with the client (during the SSL handshake).

From the navigation area in the Web Administration tool, select **SSL encryption**.

1. Select the method of SSL encryption that you want to use based on the clients accessing the server. If you select multiple encryption methods, the highest level of encryption is used by default, however clients using the selected lower encryption levels still have access to the server.

   Note: The IBM Directory Server Version 5.1 now supports the Advanced Encryption Standard (AES) level of encryption. For information on AES, see the NIST Web page at http://csrc.nist.gov/encryption/aes/.

Table 4.

Encryption level	**Attribute**
Triple DES encryption with a 168-bit key and a SHA-1 MAC	ibm-slapdSslCipherSpec: TripleDES-168
DES encryption with a 56-bit key and a SHA-1 MAC	ibm-slapdSslCipherSpec: DES-56
RC4 encryption with a 128-bit key and a SHA-1 MAC	ibm-slapdSslCipherSpec: RC4-128-SHA
RC4 encryption with a 128-bit key and a MD5 MAC	ibm-slapdSslCipherSpec: RC4-128-MD5
RC2 encryption with a 40-bit key and a MD5 MAC	ibm-slapdSslCipherSpec: RC2-40-MD5
RC4 encryption with a 40-bit key and a MD5 MAC	ibm-slapdSslCipherSpec: RC4-40-MD5
AES encryption	ibm-slapdSslCipherSpec: AES

The selected ciphers are stored in the configuration file using the **ibm-slapdsslCipherSpec** keyword and the attribute defined from the preceding table. For example, to use only Triple DES, select **Triple DES encryption with a 168-bit key and an SHA-1 MAC**. The attribute **ibm-slapdSslCipherSpec: TripleDES-168** is added to the **ibmslapd.conf** file. In this case, only clients that also support Triple DES are able to establish an SSL connection with the server.

2. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.

Using the command line:

To use the command line to set the SSL level of encryption, in this example to AES, issue the command:

```
ldapmodify -D <AdminDN> -w <Adminpassword> -i <filename>
```

where *<filename>* contains:

```
dn: cn=SSL,cn=Configuration
changetype: modify
replace: ibm-slapdSslCipherSpec
ibm-slapdSslCipherSpec: AES
```

See Table 4 on page 46 for other encryption values. You must restart the server and the administration daemon for the changes to take effect.

Secure Sockets Layer

The IBM Directory Server has the ability to protect LDAP access by encrypting data with Secure Sockets Layer (SSL) security. When using SSL to secure LDAP communications with the IBM Directory, both server authentication and client authentication are supported.

With server authentication, the IBM Directory Server must have a digital certificate (based on the X.509 standard). This digital certificate is used to authenticate the IBM Directory Server to the client application (such as the Directory Management Tool or **ldapsearch**) or an application built from the application development package, for LDAP access over SSL.

For server authentication the IBM Directory Server supplies the client with the IBM Directory 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 IBM Directory Server and the client application.

For server authentication to work, the IBM Directory Server must have a private key and associated server certificate in the server's key database file.

Client authentication provides for two-way authentication between the LDAP client and the LDAP server.

With client authentication, the LDAP client must have a digital certificate (based on the X.509 standard). This digital certificate is used to authenticate the LDAP client to the IBM Directory Server. See "Client authentication" on page 51.

To conduct commercial business on the Internet, you might use a widely known Certification Authority (CA), such as VeriSign, to get a high assurance server certificate.

Securing your server with SSL

The following high-level steps are required to enable SSL support for IBM Directory for server authentication. These steps assume you have already installed and configured the IBM Directory Server:

1. Install the IBM Directory GSKit package if it is not installed. See the *IBM Directory Server Version 5.1 Installation and Configuration Guide* for information on installing the GSKit package.

2. Generate the IBM Directory Server private key and server certificate using the gsk6ikm utility (installed with GSKit). The server's certificate can be signed by a commercial CA, such as VeriSign, or it can be self-signed with the gsk6ikm tool. The CA's public certificate (or the self-signed certificate) must also be distributed to the client application's key database file.
3. Store the server's key database file and associated password stash file on the server. The default path for the key database,**...\ldap\etc** directory, is a typical location.
4. Access the Web-based LDAP administrative interface to configure the LDAP server. See "Configuring SSL settings" on page 43 for the procedures.

If you also want to have secure communications between a master IBM Directory Server and one or more replica servers, you must complete the following additional steps:

1. Configure the replica directory server.

   Note: Follow the steps shown above for the master, except perform them for each replica. When configuring a replica for SSL, the replica is like the master with respect to its role when using SSL. The master is an LDAP client (using SSL) when communicating with a replica.
2. Configure the master directory server:
   a. Add the replica's signed server certificate to the master directory server's key database file, as a trusted root. In this situation, the master directory is actually an LDAP client. If using self-signed certificates, you must extract all the self-signed certificates from each replica IBM Directory Server, and add them to the master's key database, and ensure they are marked as trusted-roots. Essentially, you are configuring the master as an SSL client of the replica server.
   b. Configure the master IBM Directory Server to be aware of the replica server. Be sure to set the replicaPort attribute to use the port which the replica IBM Directory Server uses for SSL communication.
3. Restart both the master server and each replica server.

Note: Only one key database is permitted per ldap server.

Setting Server authentication: For server authentication, you can modify the ibmslapd.conf file under the cn=SSL, cn=Configuration entry. To use the Web Administration Tool, see "Configuring SSL settings" on page 43.

To use the command line:

```
ldapmodify -D <AdminDN> -w <Adminpassword> -i <filename>
```

where *<filename>* contains:

```
dn: cn=SSL,cn=Configuration
changetype: modify
replace: ibm-slapdSSLAuth
ibm-slapdSSLAuth: serverAuth
```

You must restart the server and the administration daemon for the changes to take effect.

Server certificate from an external Certificate Authority (CA)

In order to provide a secure connection between IBM Directory and its clients, the server must have an X.509 certificate and a private key.

The steps required to generate a private key, obtain the required server certificate from an external CA, and prepare them for use by the IBM Directory are outlined in the following:

1. Logon as administrator or root.
2. Change to the directory where you wish to create the key database file and where your private key and certificate will be stored.
3. Run gsk6ikm to create a new key database file. You can use any valid value for the key database file name that you want. Whatever file name you use, you need to provide it when configuring the LDAP server to use SSL. Providing a full path name is recommended. The gsk6ikm utility is used to generate a private-public key pair and a certificate request. See“Using gsk6ikm” on page 52 for additional information.

   Note: By default, the new KDB created by GSKit is not readable by the server. You must change the owner to **ldap**.

   ```
   chown ldap:ldap <mykeyring>.*
   ```

   See “Kerberos service name change” on page 239 for a more detailed explanation.
4. If VeriSign is your external CA, obtain a certificate from VeriSign, as follows:
   a. Access the following VeriSign Web site: http://digitalid.verisign.com/server_ids.html
   b. Click on **IBM internet connection servers**.
   c. After reviewing the information at this site, click on **Begin**.
   d. Provide the required information and follow the steps required to request your server certificate. VeriSign is the primary Certification Authority supported for obtaining externally generated, high-assurance server certificates.
5. If you have another CA that you want to use, follow the directions for that CA to submit the contents of the certificate request file to them.

When you receive the resulting certificate from the CA:

1. Logon using your server identity.
2. Change to the directory where you created the key database file.
3. Place the signed certificate from the CA into a file in this directory. The file is used in the next step.
4. From the same directory, run gsk6ikm to receive the certificate into your key database file.
5. Access the LDAP server’s Web administrative interface, and configure the various SSL parameters, including the file specification for the key database file. See “Configuring SSL settings” on page 43.
6. If you have more than one certificate in the key database file, the certificate you wish to use for IBM Directory must be the default.
7. Start the IBM Directory.

Note: If you instruct gsk6ikm to save the password in a password stash file, it is not necessary to change or set the password in the ibmslapd.conf file.

Using a self-signed server certificate

If you are using the IBM Directory in an intranet environment, use gsk6ikm to create your own server certificates. You can also use gsk6ikm to test IBM Directory with SSL without purchasing a VeriSign high-assurance server certificate. These types of certificates are known as self-signed certificates.

Follow these steps to create a key database file using self-signed certificates.

1. On each server:
   a. Change to the directory where you wish to create the key database file and where your private key and certificate is to be stored.
   b. Create a new key database file and the self-sign certificate request that is to be used as your CA certificate.
      - Use the largest key size available.
      - Use a secure server certificate, not a low-assurance certificate.
   c. Obtain the certificate request file. The certificate is put into the key database file automatically by the gsk6ikm tool.
2. If you are using an application created for the client, do the following on each client machine:
   a. Place the CA certificate request file in an accessible location on the client machine.
   b. Receive the CA certificate request file into the client's key database.
   c. Mark the received certificate as a trusted root.

See "Using gsk6ikm" on page 52 for additional information.

Notes:

1. You must always receive the CA certificate into the server's key database file and mark it as a trusted root before receiving the server certificate into the server's key database file.
2. Whenever you use gsk6ikm to manage the IBM Directory Server's key database file, remember to change to the directory in which the key database file exists.
3. Each IBM Directory Server must have its own private key and certificate. Sharing the private key and certificate across multiple IBM Directory Servers increases security risks. By using different certificates and private keys for each server, security exposure is minimized if a key database file for one of the servers is compromised.

Setting up your LDAP client to access IBM Directory

The following steps are required to create a key database file for an LDAP client that contains one or more self-signed server certificates that are marked as trusted by the client. The process can also be used to import CA certificates from other sources, such as VeriSign, into the client's key database file for use as trusted roots. A trusted root is simply an X.509 certificate signed by a trusted entity (for example VeriSign, or the creator of a self-signed server certificate), imported into the client's key database file, and marked as trusted.

1. Copy the server's certificate file (cert.arm) to your client workstation.
2. Run **gsk6ikm** to create a new client key database file or to access an existing one. For a new client key database, choose a file name associated with the client for ease of management. For example, if the LDAP client runs on Fred's machine, you might choose to name the file FRED.KDB.
3. If adding a server's certificate to the existing client key database:
   a. Click **Key database file** and select **Open**.
   b. Enter the path and name of the existing key database file then click **OK**.
   c. Enter the password.
   d. **Ensure signer certificates** is chosen. Click **Add**.

    e. Enter the name and location of the server's certificate file.
    f. Enter a label for the server certificate entry in the client's key database file, for example, Corporate Directory Server then click **OK**.
4. If creating the new Client key database:
    a. Click **Key database file** and select **New**.
    b. Enter the name and location for the new Client Key DataBase file, then click **OK**.
    c. Enter the password.
    d. After the new client key database is created, repeat the previous steps for adding server's certificate to the existing key database file.
5. Exit **gsk6ikm**.

See "Using gsk6ikm" on page 52 for additional information.

When the LDAP client creates a secure SSL connection with the server, it uses the server's self-signed certificate to verify that it is connecting to the proper server.

Repeat the preceding steps for each IBM Directory Server that the LDAP client needs to connect to in a secure fashion.

Migrate the key ring file to key database file

To migrate the old key ring file which was created from MKKF utility, do the following steps:

1. Start gsk6ikm.
2. Click **Key database file** of menu bar and select **Open**.
3. Enter the path and filename of your key ring file then click **OK**.
4. Enter the password of your key ring file. If the key ring file is created without a password, you must use the old MKKF to assign a password for it.
5. After the old Key ring file is opened, click **Key database file** and select **Save ss**.
6. Ensure key database type is set to CMS key database file. Fill out the name and location of the key database file, and then click **OK**.

Client authentication

Client authentication provides for two-way authentication between the LDAP client and the LDAP server.

With client authentication, the LDAP client must have a digital certificate (based on the X.509 standard). This digital certificate is used to authenticate the LDAP client to the IBM Directory Server.

The Simple Authentication and Security Layer (SASL) can be used to add authentication support to connection protocols. A protocol includes a command for identifying and authenticating a user to a server. It can optionally negotiate a security layer for subsequent protocol interactions.

After a server receives the authentication command or any client response, it may issue a challenge or indicate failure or completion. If a client receives a challenge it may issue a response or abort the exchange, depending on the profile of the protocol.

During the authentication protocol exchange, the SASL mechanism performs authentication, transmits an authorization identity (known as userid) from the client to the server, and negotiates the use of a mechanism-specific security layer.

When the LDAP server receives an LDAP bind request from a client, it processes the request in the following order:

1. The server parses the LDAP bind request and retrieves the following information:
   - The DN that the client is attempting to authenticate as.
   - The method of authentication used.
   - Any credentials, such as a password included in the request.
   - If the method of authentication is SASL, the server also retrieves the name of the SASL mechanism used from the LDAP bind request.
2. The server normalizes the DN retrieved from the request.
3. The server retrieves any LDAP control included with the LDAP bind request.
4. If the method of authentication is SASL, the server determines whether or not the SASL mechanism (specified in the request) is supported. If the SASL mechanism is not supported by the server, the server sends an error return code to the client and ends the bind process.
5. If the SASL mechanism is supported (=EXTERNAL) and the SSL authentication type is server and client authentication, the server verifies that the client's certificate is valid, issued by a known CA, and that none of the certificates on the client's certificate chain are invalid or revoked. If the client DN and password, as specified in the **ldap_sasl_bind**, are NULL, then the DN contained within the client's x.509v3 certificate is used as the authenticated identity on subsequent LDAP operations. Otherwise, the client is authenticated anonymously (if DN and password are NULL), or the client is authenticated based on the bind information provided by the client.
6. If the method of authentication is Simple, the server checks to see if the DN is an empty string or if there are no credentials.
7. If the DN is an empty string, or if the DN or no credentials are specified, the server assumes that the client is binding anonymously and returns a good result to the client. The DN and authentication method for the connection are left as NULL and LDAP_AUTH_NONE respectively.
8. If the client has not bound beforehand, and does not present a certificate during the bind operation, the connection is refused.

Setting client authentication: For client authentication, you can modify the ibmslapd.conf file under the cn=SSL, cn=Configuration entry. To use the Web Administration Tool, see "Configuring SSL settings" on page 43.

To use the command line:

```
ldapmodify -D <AdminDN> -w <Adminpassword> -i <filename>
```

where *<filename>* contains:

```
dn: cn=SSL,cn=Configuration
cn: SSL
changetype: modify
replace: ibm-slapdSSLAuth
ibm-slapdSSLAuth: serverClientAuth
```

You must restart the server and the administration daemon for the changes to take effect.

Using gsk6ikm

The following key-management program is provided with IBM's Global Security Kit (GSKit):

- gsk6ikm - A user-friendly GUI for managing key files, implemented as a Java applet.

Note: On the AIX® operating systems, if you are prompted to set JAVA_HOME, you can set it to either the system-installed Java or the Java version included with the IBM Directory Server. If you use the IBM Directory Server version, you also need to set the LIBPATH environment variable as follows:

```
export LIBPATH=/usr/ldap/java/bin:/usr/ldap/java/bin/classic:$LIBPATH
```

Use gsk6ikm to create public-private key pairs and certificate requests, receive certificate requests into a key database file, and manage keys in a key database file.

The tasks you can perform with gsk6ikm include:
- Creating a key pair and requesting a certificate from a certificate authority
- Receiving a certificate into a key database file
- Managing keys and certificates
  - Changing a key database password
  - Showing information about a key
  - Deleting a key
  - Making a key the default key in the key database
  - Creating a key pair and certificate request for self-signing
  - Exporting a key
  - Importing a key into a key database
  - Designating a key as a trusted root
  - Removing trusted root key designation
  - Requesting a certificate for an existing key
- Migrating a keyring file to the key database format

Creating a key pair and requesting a certificate from a Certificate Authority

If your client application is connecting to an LDAP server that requires client and server authentication, then you need to create a public-private key pair and a certificate.

If your client application is connecting to an LDAP server that only requires server authentication, it is not necessary to create a public-private key pair and a certificate. It is sufficient to have a certificate in your client key database file that is marked as a trusted root. If the Certification Authority (CA) that issued the server's certificate is not already defined in your client key database, you need to request the CA's certificate from the CA, receive it into your key database, and mark it as trusted. See "Designating a key as a trusted root" on page 58.

Your client uses its private key to sign messages sent to servers. The server sends its public key to clients so that they can encrypt messages to the server, which the server decrypts with its private key.

To send its public key to a server, the client needs a certificate. The certificate contains the client's public key, the Distinguished Name associated with the client's certificate, the serial number of the certificate, and the expiration date of the certificate. A certificate is issued by a CA, which verifies the identity of the client.

The basic steps to create a certificate that is signed by a CA are:

1. Create a certificate request using **gsk6ikm**.
2. Submit the certificate request to the CA. This can be done using E-mail or an on-line submission from the CA's Web page.
3. Receive the response from the CA to an accessible location on the file system of your server.
4. Receive the certificate into your key database file.

Note: If you are obtaining a signed client certificate from a CA that is not in the default list of trusted CAs, you need to obtain the CA's certificate, receive it into your key database and mark it as trusted. This must be done before receiving your signed client certificate into the key database file.

To create a public-private key pair and request a certificate:

1. Start gsk6ikm Java utility by typing:
   ```
   gsk6ikm
   ```
2. Select **Key database file**.
3. Select **New** (or **Open** if the key database already exists).
4. Specify key database file name and location. Type **OK**.

   Note: A key database is a file that the client or server uses to store one or more key pairs and certificates.
5. When prompted, supply password for the key database file. Click **OK**.
6. Select **Create**.
7. Select **New certificate request**.
8. Supply user-assigned label for key pair. The label identifies the key pair and certificate in the key database file.
9. If you are requesting a low-assurance client certificate, enter the common name. This must be unique and the full name of the user.
10. If you are requesting a high-assurance secure server certificate, then:
    - Enter the X.500 common name of the server. Usually this is the TCP/IP fully qualified host name, for example, www.ibm.com. For a VeriSign server certificate, it must be the fully qualified host name.
    - Enter the organization name. This is the name of your organization. For a VeriSign secure server certificate, if you already have an account with VeriSign, the name in this field must match the name on that account.
    - Enter the organizational unit name. This is an optional field.
    - Enter the locality/city where the server is located. This is an optional field.
    - Enter a three-character abbreviation of the state/province where the server is located.
    - Enter the postal code appropriate for the server's location.
    - Enter the two-character country code where the server is located.
11. Click **OK**.
12. A message identifying the name and location of the certificate request file is displayed. Click **OK**.
13. Send the certificate request to the CA.

    If this is a request for a VeriSign low assurance certificate or secure server certificate, you must E-mail the certificate request to VeriSign.

    You can mail the low assurance certificate request to VeriSign immediately. A secure server certificate request requires more documentation. To find out what VeriSign requires for a secure server certificate request, go to the following URL: http://www.verisign.com/ibm.

14. When you receive the certificate from the CA, use gsk6ikm to receive it into the key database where you stored the key pair. See “Receiving a certificate into a key database”.

Note: Change the key database password frequently. If you specify an expiration date, you need to keep track of when you need to change the password. If the password expires before you change it, the key database is not usable until the password is changed.

Receiving a certificate into a key database

After receiving a response from your CA, you need to receive the certificate into a key database.

To receive a certificate into a key database:

1. Type `gsk6ikm` to start the Java utility.
2. Select **Key database file**.
3. Select **Open**.
4. Specify key database file name and location. Type **OK**.
5. When prompted, supply password for the key database file, click **OK**.
6. Select **Create**.
7. Select **Personal certificates** in the middle display window.
8. Click **Receive**.
9. Enter name and location of the certificate file that contains the signed certificate, as received from the CA. Click **OK**.

Changing a key database password

To change a key database password:

1. Type `gsk6ikm` to start the Java utility.
2. Select **Key database file**.
3. Select **Open**.
4. Specify key database file name and location. Type **OK**.
5. When prompted, supply password for the key database file. Click **OK**.
6. Select **Key database file**.
7. Select **Change password**.
8. Enter <*New password*>.
9. Confirm <*New password*>.
10. Select and set optional password expiration time.
11. Select **Stash the password to a file?** if you want the password to be encrypted and stored on disk.
12. Click **OK**.
13. A message is displayed with the file name and location of the stash password file. Click **OK**.

Note: The password is important because it protects the private key. The private key is the only key that can sign documents or decrypt messages encrypted with the public key.

Showing information about a key

To show information about a key, such as its name, size or whether it is a trusted root:

1. Type `gsk6ikm` to start the Java utility.
2. Select **Key database file**.
3. Select **Open**.
4. Specify key database file name and location. Type **OK**.
5. When prompted, supply password for the key database file. Click **OK**.
6. To see information about keys designated as Personal certificates:
   - Select **Personal certificates** at the top of the **Key database content** window.
   - Select a certificate.
   - Click **View/Edit** to display information about the selected key.
   - Click **OK** to return to the list of Personal Certificates.
7. To see information about keys that are designated as Signer Certificates:
   - Select **Signer certificates** at the top of the **Key database content** window.
   - Select a certificate .
   - Click **View/Edit** to display information about the selected key.
   - Click **OK** to return to the list of Signer Certificates.

Deleting a Key

To delete a key:

1. Type `gsk6ikm` to start the Java utility.
2. Select **Key database file**.
3. Select **Open**.
4. Specify key database file name and location. Type **OK**.
5. When prompted, supply password for the key database file. Click **OK**.
6. Select the type of key you want to delete at the top of the **Key database content** window (Personal certificates, Signer certificates, or Personal certificate requests).
7. Select a certificate.
8. Click **Delete**.
9. Click **Yes** to confirm.

Making a key the default key in the key ring

The default key must be the private key the server uses for its secure communications.

To make a key the default key in the key ring:

1. Type `gsk6ikm` to start the Java utility.
2. Select **Key database file**.
3. Select **Open**.
4. Specify key database file name and location. Type **OK**.
5. When prompted, supply password for the key database file. Click **OK**.
6. Select **Personal certificates** at the top of the **Key database content** window.
7. Select the desired certificate.
8. Click **View/Edit**.
9. Select the **Set the certificates as the default** box. Click **OK**.

Creating a key pair and certificate request for self-signing

By definition, a secure server must have a public-private key pair and a certificate.

The server uses its private key to sign messages to clients. The server sends its public key to clients so they can encrypt messages to the server, which the server decrypts with its private key.

The server needs a certificate to send its public key to clients. The certificate contains the server's public key, the Distinguished Name associated with the server's certificate, the serial number of the certificate, and the expiration date of the certificate. A certificate is issued by a CA, who verifies the identity of the server.

You can request one of the following certificates:

- A low assurance certificate from VeriSign, best for non-commercial purposes, such as a beta test of your secure environment
- A server certificate to do commercial business on the Internet from VeriSign or some other CA
- A self-signed server certificate if you plan to act as your own CA for a private Web network

For information about using a CA such as VeriSign to sign the server certificate, see "Creating a key pair and requesting a certificate from a Certificate Authority" on page 53.

The basic steps to creating a self-signed certificate are:

1. Type `gsk6ikm` to start the Java utility.
2. Select **Key database file**.
3. Select **New**, or **Open** if the key database already exists.
4. Specify key database file name and location. Type **OK**.

   Note: A key database is a file that the client or server uses to store one or more key pairs and certificates.
5. When prompted, supply password for the key database file. Click **OK**.
6. Click **New self-signed**.
7. Supply the following:
   - User-assigned label for key pair. The label identifies the key pair and certificate in the key database file.
   - Select the desired certificate Version.
   - Select the desired Key Size.
   - Enter the X.500 common name of the server. Usually this is the TCP/IP fully qualified host name, for example, www.ibm.com.
   - Enter the organization name. This is the name of your organization.
   - Enter the organizational unit name. This is an optional field.
   - Enter the locality/city where the server is located. This is an optional field.
   - Enter a three-character abbreviation of the state/province where the server is located.
   - Enter the zip code appropriate for the server's location.
   - Enter the two-character country code where the server is located.
   - Enter the Validity Period for the certificate.
8. Click **OK**.

Exporting a key

If you need to transfer a key pair or certificate to another computer, you can export the key pair from its key database to a file. On the other computer, you can import the key pair into a key ring.

To export a key from a key database:

1. Type `gsk6ikm` to start the Java utility.
2. Select **Key database file**.
3. Select **Open**.
4. Specify key database file name and location. Type **OK**.
5. When prompted, supply password for the key database file. Click **OK**.
6. Select **Personal certificates** at the top of the **Key database content** window.
7. Select the desired certificate.
8. Click **Export/Import**.
9. For **Action type**, select **Export key**.
10. Select the Key file type:
    - PKCS12 file
    - CMS Key database file
    - Keyring file (as used by mkkf)
    - SSLight key database class
11. Specify a file name.
12. Specify location.
13. Click **OK**.
14. Enter the required password for the file. Click **OK**.

Importing a key

To import a key into a key ring:

1. Type `gsk6ikm` to start the Java utility.
2. Select **Key database file**.
3. Select **Open**.
4. Specify key database file name and location. Type **OK**.
5. When prompted, supply password for the key database file. Click **OK**.
6. Select **Personal certificates** at the top of the **Key database content** window.
7. Select the desired certificate.
8. Click **Export/Import**.
9. For **Action type**, select **Import key**.
10. Select the desired Key file type.
11. Enter the file name and location.
12. Click **OK**.
13. Enter the required password for the source file. Click **OK**.

Designating a key as a trusted root

A trusted root key is the public key and associated Distinguished Name of a CA. The following trusted roots are automatically defined in each new key database:

- Integrion Certification Authority Root
- IBM World Registry™ Certification Authority
- Thawte Personal Premium CA
- Thawte Personal Freeemail CA

- Thawte Personal Basic CA
- Thawte Premium Server CA
- VeriSign Test CA Root Certificate
- RSA Secure Server Certification Authority
- VeriSign Class 1 Public Primary Certification Authority
- VeriSign Class 2 Public Primary Certification Authority
- VeriSign Class 3 Public Primary Certification Authority
- VeriSign Class 4 Public Primary Certification Authority

Note: Each of these trusted roots are initially set to be trusted roots by default.

To designate a key as a trusted root:
1. Type `gsk6ikm` to start the Java utility.
2. Select **Key database file**.
3. Select **Open**.
4. Specify key database file name and location. Type **OK**.
5. When prompted, supply password for the key database file. Click **OK**.
6. Select **Signer certificates** at the top of the **Key database content** window.
7. Select the desired certificate.
8. Click **View/Edit**.
9. Check the **Set the certificate as a trusted root** box, and click **OK**.
10. Select **Key database file** and then select **Close**.

Removing a key as a trusted root

A trusted root key is the public key and associated Distinguished Name of a CA. The following trusted roots are automatically defined in each new key database:
- Integrion Certification Authority Root
- IBM World Registry Certification Authority
- Thawte Personal Premium CA
- Thawte Personal Freeemail CA
- Thawte Personal Basic CA
- Thawte Premium Server CA
- VeriSign Test CA Root Certificate
- RSA Secure Server Certification Authority
- VeriSign Class 1 Public Primary Certification Authority
- VeriSign Class 2 Public Primary Certification Authority
- VeriSign Class 3 Public Primary Certification Authority
- VeriSign Class 4 Public Primary Certification Authority

Note: Each of these trusted roots are initially set to be trusted roots by default.

To remove the trusted root status of a key:
1. Type `gsk6ikm` to start the Java utility.
2. Select **Key database file**.
3. Select **Open**.
4. Specify key database file name and location. Type **OK**.
5. When prompted, supply password for the key database file. Click **OK**.

6. Select **Signer certificates** at the top of the **Key database content** window.
7. Select the desired certificate.
8. Click **View/Edit**.
9. Uncheck the **Set the certificate as a trusted root** box. Click **OK**.
10. Select **Key database file** and then select **Close**.

Requesting a certificate for an existing key

To create a certificate request for an existing key:

1. Type `gsk6ikm` to start the Java utility.
2. Select **Key database file**.
3. Select **Open**.
4. Specify key database file name and location. Type **OK**.
5. When prompted, supply password for the key database file. Click **OK**.
6. Select **Personal Certificates** at the top of the **Key database content** window.
7. Select the desired certificate.
8. Click **Export/Import**.
9. For **Action type**, select **Export key**.
10. Select the desired Data type:
    - Base 64-encoded ASCII data
    - Binary DER data
    - SSLight Key Database Class
11. Enter the certificate file name and location.
12. Click **OK**.
13. Select **Key database file** and then select **Close**.

Send the certificate request to the CA.

If this is a request for a VeriSign low assurance certificate or secure server certificate, you must E-mail the certificate request to VeriSign.

You can mail the low assurance certificate request to VeriSign immediately. A secure server certificate request requires more documentation. To find out what VeriSign requires for a secure server certificate request, go to the following URL: http://www.verisign.com/ibm.

Migrating a keyring file to the key database format

The gsk6ikm program can be used to migrate an existing keyring file, as created with mkkf, to the format used by gsk6ikm.

To migrate a keyring file:

1. Type `gsk6ikm` to start the Java utility.
2. Select **Key database file**.
3. Select **Open**.
4. Specify key database file name and location. Type **OK**.
5. When prompted, supply password for the keyring file. Click **OK**.
6. Select **Key database file**.
7. Select **Save as...**.
8. Select **CMS key database file** as the Key database type.
9. Specify a file name.

10. Specify location.
11. Click **OK**.

Setting password security

Password policy is a set of rules that controls how passwords are used and administered in the IBM Directory. These rules are made to ensure that users change their passwords periodically, and that the passwords meet the organization's syntactic password requirements. These rules also can restrict the reuse of old passwords and ensure that users are locked out after a defined number of failed attempts.

All users except the directory administrator are forced to comply with this password policy. The password for the administrator never expires and the account is never locked. The directory administrator has sufficient access control privileges to modify users' passwords and the password policy.

Setting password policy

Expand the **Manage security properties** category in the navigation area of the Web Administration Tool, select the **Password policy** tab.

1. Select the type of password encryption from the drop-down list:
   - None
   - imask
   - crypt
   - sha

   See "Password encryption" on page 64 for additional information.
2. Select the **Password policy enabled** checkbox to enable password policy.

   Note: If Password policy is not enabled, none of the other functions on this or the other password panels are available until the checkbox is enabled. By default password policy is disabled.
3. Select the **User can change password** checkbox to specify whether the user can change the password.
4. Select the **User must change password after reset checkbox** to specify whether the user must change the password after logging on with a reset password.
5. Select the **User must send password when changing** checkbox to specify whether the user, after the initial log on, needs to specify the password again before being able to change the password.
6. Set the password expiration limit. Click the **Password Never Expires** radio button to specify that the password does not have to be changed at a specific time interval or click the **Days** radio button and specify the time interval, in days, when the password needs to be reset.
7. Specify whether the system issues a password expiration warning, before the password expires. If you click the **Never warn** radio button, the user is not warned before the previous password expires. The user cannot access the directory until the administrator has created a new password. If you click the **Days before expiration** radio button and specify a number of days (*n*), the user receives a warning prompt to change the password each time the user logs on, starting *n* days before the password expires. The user can still access the directory until the password expires.

8. Specify the number of times, if any, that the user can log on after the password has expired. This selection enables the user to access the directory with an expired password.
9. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.

Using the command line:

To perform the same operations using the command line, issue the following command:

```
ldapmodify -D <AdminDN> -w<Adminpassword> -i<filename>
```

where <*filename*> contains:

```
dn: cn=pwdpolicy
changetype: modify
replace: ibm-pwdpolicy
ibm-pwdpolicy: TRUE|FALSE
#select TRUE to enable, FALSE to disable
-
replace:pwdallowuserchange
pwdallowuserchange: TRUE|FALSE
#select TRUE to enable, FALSE to disable
-
replace:pwdmustchange
pwdmustchange: TRUE|FALSE
#select TRUE to enable, FALSE to disable
-
replace:pwdmaxage
pwdmaxage: 5
-
replace:pwdexpirewarning
pwdexpirewarning: 7
-
replace:pwdgraceloginlimit
pwdgraceloginlimit: 2
```

You must restart the server for the changes to take effect.

Setting password lockout

Expand the **Manage server properties** category in the navigation area of the Web Administration Tool, select the **Password lockout** tab.

Note: If password policy is not enabled on the server, the functions on this panel do not take effect.

1. Specify the number of seconds, minutes, hours or days that must expire before a password can be changed.
2. Specify whether incorrect logins lockout the password.
   - Select the **Passwords are never locked out** radio button if you want to allow unlimited log in attempts. This selection disables the password lockout function.
   - Select the **Attempts** radio button and specify the number of log in attempts that are allowed before locking out the password. This selection enables the password lockout function.
3. Specify the duration of the lockout. Select the **Lockouts never expires** radio button to specify that the system administrator must reset the password or select the **Seconds** radio button and specify the number of seconds before the lockout expires and log in attempts can resume.

4. Specify the expiration time for an incorrect login. Click the **Incorrect logins only cleared with correct password** radio button to specify that incorrect logins are cleared only by a successful login or click the **Seconds** radio button and specify the number of seconds before an unsuccessful login attempt is cleared from memory.

   Note: This option works only if the password is not locked out.
5. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.

Using the command line:

To perform the same operations using the command line, issue the following command:

```
ldapmodify -D <AdminDN> -w<Adminpassword> -i<filename>
```

where <*filename*> contains:

```
dn: cn=pwdpolicy
changetype: modify
replace: pwdlockout
pwdlockout: TRUE|FALSE
#select TRUE to enable, FALSE to disable
-
replace:pwdmaxfailure
pwdmaxfailure: 3
-
replace:pwdlockoutduration
pwdlockoutduration: 15
-
replace:pwdfailurecountinterval
pwdfailurecountinterval: 30
-
replace:pwdexpirewarning
pwdexpirewarning: 7
-
replace:pwdgraceloginlimit
pwdgraceloginlimit: 2
```

Setting password validation

Expand the **Manage server properties** category in the navigation area of the Web Administration Tool, select the **Password validation** tab.

Note: If password policy is not enabled on the server, the functions on this panel do not take effect.

1. Set the number of passwords that must be used before a password can be reused. Enter a number from 0 to 30. If you enter zero, a password may be reused without restriction.
2. From the drop down box, select whether the password is checked for the syntax defined in the following entry fields. You can select:

   Do not check syntax
   No syntax checking is performed.

   Check syntax (except encrypted)
   The syntax checking is performed on all unencrypted passwords.

   Check syntax
   The syntax checking is performed on all passwords.

Setting syntax requirements

1. Specify a number value to set the minimum length of the password. If the value is set to zero, no syntax checking is performed.
   - Specify a number value to set the minimum number of alphabetic characters required for the password.
   - Specify a number value to set the minimum number of numeric and special characters required for the password.

   Note: The sum of the minimum number of alphabetic, and numeric and special characters must be equal to or less than the number specified as the minimum length of the password.
2. Specify the maximum number of characters that can be repeated in the password. This option limits the number of times a specific character can appear in the password. If the value is set to zero, the number of repeated characters is not checked.
3. Specify the minimum number of characters that must be different from the previous password and the number of previous passwords specified in the **Minimum number of passwords before reuse** field. If the value is set to zero, the number of different characters is not checked.
4. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.

Using the command line:

To perform the same operations using the command line, issue the following command:

```
ldapmodify -D <AdminDN> -w<Adminpassword> -i<filename>
```

where <*filename*> contains:

```
dn: cn=pwdpolicy
changetype: modify
replace: pwdinhistory
pwdinhistory: 8
-
replace:pwdchecksyntax
pwdchecksyntax: 0|1|2
# 0=No syntax check
#1=Check syntax (except encrypted)
#2=Check syntax on all-
replace:pwdminlength
pwdminlength: 6
-
replace:passwordminalphachars
passwordminalphachars: 3
-
replace:passwordminotherchars
passwordminotherchars: 3
-
replace:passwordmaxrepeatedchars
passwordmaxrepeatedchars: 2
-
replace:passwordmindiffchars
passwordmindiffchars: 4
```

Password encryption

IBM Directory allows you to prevent unauthorized access to user passwords. User passwords may be encoded and stored in the directory, which prevents clear passwords from being accessed by any users including the system administrators.

The administrator may configure the server to encode userPassword attribute values in either a one-way encoding format or a two-way encoding format.

One-way encoding formats:
- SHA-1
- crypt

After the server is configured, any new passwords (for new users) or modified passwords (for existing users) are encoded before they are stored in the directory database. 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 work.

For applications which require retrieval of clear passwords, such as middle-tier authentication agents, the directory administrator needs to configure the server to perform either a two-way encoding or no encryption on user passwords. In this instance, the clear passwords stored in the directory are protected by the directory ACL mechanism.

Two-way encoding format:
- imask

imask, a 2-way masking option, is provided to allow values of the userPassword attribute to be encoded in the directory 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, however, corporate security policies might prohibit storing clear passwords in a secondary permanent storage. This option satisfies both requirements.

A simple bind will succeed if the password provided in the bind request matches with any of the multiple values of the userPassword attribute.

When you configure the server using Web Administration, you can select one of the following four encryption options:

None No encryption. Passwords are stored in the clear text format.

crypt Passwords are encoded by the UNIX® crypt encoding algorithm before they are stored in the directory.

SHA-1
Passwords are encoded by the SHA-1 encoding algorithm before they are stored in the directory.

imask Passwords are encoded by the imask algorithm before they are stored in the directory, and are retrieved as part of an entry in the original clear format.

The default option is imask. The change is registered in a password encryption directive of the server configuration file:

```
ibm-SlapdPwEncryption: imask
```

The server configuration file is located in:

```
<installation path>\etc\ibmslapd.conf
```

In addition to **userPassword**, values of the **secretKey** attribute are always "imask" encoded in the directory. Unlike **userPassword**, this encoding is enforced for values of **secretKey**. No other option is provided. The **secretKey** attribute is an IBM defined schema. Applications may use this attribute to store sensitive data that need to be always encoded in the directory and retrieve the data in clear via the directory access control.

Consult the *Installation and Configuration Guide* for additional information about the configuration file.

To change the type of encryption using the command line, for example to crypt, issue the following command:

```
ldapmodify -D <adminDN> -w <adminPW> -f <filename>
```

where <*filename*>contains:

```
dn: cn=configuration
changetype: modify
replace: ibm-slapdPWEncryption
ibm-slapdPWEncryption: crypt
```

To update the settings dynamically, issue the following ldapexop command:

```
ldapexop -D <adminDN> -w <adminPW> -op readconfig -scope single
       "cn=configuration" ibm-slapdPWEncryption
```

Notes:

1. When 'imask' is used as the server password encryption method, only the first 46 characters of a password entered are effective. Any characters after the 46th character will be ignored and considered as matched. Similarly, if the UNIX 'crypt' method is used, only the first 8 characters will be effective. Also since the value of SecretKey is encrypted in the database using the 'imask' encryption, the SecretKey values which are longer than 46 characters will not be maintained.
2. A one-way encoded password can be used for password matching but it cannot be decrypted. During user login the login password is encoded and compared with the stored version for matching verification.

Setting Kerberos

The IBM Directory supports Kerberos Version 1.3 servers, such as the IBM Network Authentication Service, on AIX servers and clients, and Windows NT® and Windows 2000 clients only.

Note: You must have a Kerberos client installed to use Kerberos authentication.

Under Network Authentication Service, a client (generally either a user or a service) sends a request for a ticket to the Key Distribution Center (KDC). The KDC creates a ticket-granting ticket (TGT) for the client, encrypts it using the client's password as the key, and sends the encrypted TGT back to the client. The client then attempts to decrypt the TGT, using its password. If the decryption is successful, the client retains the decrypted TGT, indicating proof of the client's identity.

The TGT, which expires at a specified time, permits the client to obtain additional tickets that give permission for specific services. The requesting and granting of these additional tickets does not require user intervention.

Network Authentication Service negotiates authenticated, optionally encrypted communications between two points on the network. It can enable applications to provide a layer of security that is not dependent on which side of a firewall either client is on. Because of this, Network Authentication Service can play a vital role in the security of your network.

You need to create an LDAP server servicename in the key distribution center (KDC) using the principal name ldap/<*hostname*>.<*mylocation*>.<*mycompany*>.com.

Note: An environment variable ″LDAP_KRB_SERVICE_NAME″ is used to determine the case of the LDAP Kerberos service name. If the variable is set to ′LDAP′ then the uppercase LDAP Kerberos service name is used. If the variable is not set, then the lower case ′ldap′ is used. This environment variable is used by both the ldap client and the server. By default this variable is not set. See "Kerberos service name change" on page 239 for more detailed information.

Network Authentication Service provides the following components:

Key distribution center
The KDC is a trusted server that has access to the private keys of all the principals in a realm. The KDC is composed of two parts: the Authentication Server (AS) and the Ticket Granting Server (TGS). The AS handles initial client authentication by issuing a TGT. The TGS issues service tickets that can be used by the client to authenticate to a service.

Administration server
The administration server provides administrative access to the Network Authentication Service database. This database contains the principals, keys, policies and other administrative information for the realm. The administration server allows adding, modifying, deleting and viewing principals and policies.

Password change service
The password change service allows users to change their passwords. The password change service is provided by the administration server.

Client programs
Client programs are provided to manipulate credentials (tickets), manipulate keytab files, change passwords and perform other basic Network Authentication Service operations.

Application programming interfaces (APIs)
Libraries and header files are provided to allow the development of secure distributed applications. The APIs provided are described in the Application Development Reference.

Using Web Administration:

Expand the **Manage server properties** category in the navigation area of the Web Administration Tool, select the **Kerberos** tab.

1. Check the **Enable Kerberos authentication** check box to enable Kerberos authentication.
2. Check the **Map Kerberos IDs to LDAP DNs** check box to enable the directory administrator to use the existing set of ACL data with the Kerberos authentication method. See "Identity mapping for Kerberos" on page 68 for more information.

3. Enter the Kerberos realm using the format hostName.domainName, for example, `TEST.AUSTIN.IBM.COM`.
4. Enter the path and file name of the Kerberos keytab file. This file contains the private key of the LDAP server, as associated with its kerberos account. This file, and the SSL key database file, should be protected.
5. Enter the Alternate administrator ID using the format ibm-kn=value@realm, for example, `ibm-kn=root@TEST.AUSTIN.IBM.COM`.
6. When you are finished, click **OK** to apply your changes or click **Cancel** to exit this panel without making any changes.

Using the command line:

To create a Kerberos entry issue the following command:

```
ldapmodify -D <adminDN> -w <adminPW> -f <filename>
```

where <*filename*>contains:

```
dn: cn=Kerberos, cn=Configuration
cn: Kerberos
ibm-slapdKrbAdminDN: ibm-kn=admin@MYREALM.AUSTIN.IBM.COM
ibm-slapdKrbEnable: true
ibm-slapdKrbIdentityMap: true
ibm-slapdKrbKeyTab: /keytabs/mykeytab.keytab
ibm-slapdKrbRealm: MYREALM.AUSTIN.IBM.COM
objectclass: ibm-slapdKerberos
objectclass: ibm-slapdconfigEntry
objectclass: top
```

To modify a Kerberos entry, for example to change the keytab file, issue the following command:

```
ldapmodify -D <adminDN> -w <adminPW> -f <filename>
```

where <*filename*> contains:

```
dn: cn=Kerberos, cn=Configuration
changetype: modify
replace: ibm-slapdKrbKeyTab
ibm-slapdKrbKeyTab: /keytabs/mynewkeytab.keytab
```

Using Kerberos

Before you can us the command line for Kerberos authentication, you need to do a Kerberos initialization. Issue the following command:

```
kinit <kerberos_principlename>@<realm_name>
```

To use Kerberos authentication you must specify the **-m** option with the GSSAPI parameter on the ldapadd and ldapsearch commands. For example:

```
ldapsearch  -V 3 -m GSSAPI -b <"cn=us"> objectclass=*
```

Identity mapping for Kerberos

Identity mapping enables the directory administrator to use the existing set of ACL data with the Kerberos authentication method. The ACL for the IBM Directory is based on the Distinguished Name (DN) assigned to the client connected to the directory server. The access rights are based on what permissions have been granted for that DN and the permissions for any groups containing that DN as a member. If the bind method for GSSAPI is used (that is, Kerberos is used for authenticating to the server), the DN is something like IBM-KN=your_principal@YOUR_REALM_NAME. This type of DN can be used as members of access groups or access IDs. You can also use the Kerberos Identity Mapping feature to grant access rights for this DN to an entry already in the directory.

For example, if there is an entry in the directory for Reginald Bender:

```
dn: cn=Reginald Bender, ou=internal users, o=SecureWay.com, c=US
objectclass: top
objectclass: person
objectclass: organizationalperson
cn: Reginald Bender
sn: Bender
aclentry: access-id:CN=THIS:critical:rwsc
aclentry: group:CN=ANYBODY:normal:rsc
userpassword: cL1eNt
```

The access rights for this entry allow anyone binding with the DN ″cn=Reginald Bender, ou=internal users, o=SecureWay.com, c=US″ to view critical data such as the password, but no one else.

If Reginald Bender used Kerberos to bind to the server, his DN could be something like IBM-KN=rbender@SW.REALM_1. If identity mapping is not enabled on the server, he is not allowed to view his own entry's password.

If identity mapping is enabled, he can view the password if this entry were changed to include:

```
dn: cn=Reginald Bender, ou=internal users, o=SecureWay.com, c=US...
objectclass: ibm-securityidentities
altsecurityidentities: Kerberos:rbender@SW.REALM_1
```

When Reginald Bender binds to the directory server, the server first searches the whole directory to determine if the directory is a KDC (Key Distribution Center) account registry. If it is not, the server searches the directory for any entry containing an altsecurityidentities attribute with a value matching the Kerberos user principal and realm. In this example, rbender is the user principal and SW.REALM_1 is the realm. This is the default for the Kerberos identity mapping. The bind fails if more than one entry has an attribute with this value. The mapping must be one-to-one. If the mapping is successful, Reginald Bender has all of the access rights for ″cn=Reginald Bender, ou=internal users, o=SecureWay.com, c=US″, including any access groups that has this as a member.

The IBM Directory Server can be used to contain Kerberos account information (krbRealmName-V2 = <*realm_name*> and krbPrincipalName = <*princ_name*>@<*realm_name*>) to serve as the backing store for a KDC.

The server with Kerberos identity mapping enabled first searches the directory for entries with objectclass krbRealm-V2 and krbRealmName-V2 =<*realm_name*>, such as:

```
dn: krbRealmName-V2=SW.REALM_1, o=SecureWay.com, c=US
objectclass:  krbRealm-V2
krbReamlName-V2: SW.REALM_1
```

If no entries are found, the server uses the default Kerberos identity mapping described previously. If more than one entry is found, the bind fails.

However, if the directory contains the single entry:

```
dn: krbRealmName-V2=SW.REALM_1, ou=Group, o=SecureWay.com, c=US
objectclass:  krbRealm-V2
krbRealmName-V2: SW.REALM_1
krbPrincSubtree: ou=internal users,o=SecureWay.com, c=US
krbPrincSubtree: ou=external users,o=SecureWay.com, c=US
```

The server searches each subtree listed as a value of krbPrincSubtree for an entry with an attribute krbPrincipalName.

In this release, for identity mapping to work for Reginald Bender, you need to add two attributes to the ″cn=Reginal Bender, ou=internal users, o=SecureWay.com, c=US″ entry:

```
objectclass: extensibleObject
krbPrincipalName: rbender@SW.REALM_1
```

Depending on whether the directory is a KDC account registry, the final entry is:

```
dn: cn=Reginald Bender, ou=internal users, o=SecureWay.com, c=US...
objectclass: ibm-securityidentities
altsecurityidentities: Kerberos:rbender@SW.REALM_1...
```

or for a KDC account registry:

```
dn: cn=Reginald Bender, ou=internal users, o=SecureWay.com, c=US ...
objectclass: extensibleObject
krbPrincipalName: rbender@SW.REALM_1
```

In either case, the client is mapped to ″cn=Reginald Bender, ou=internal users, o=SecureWay.com, c=US″.

If a DN is not mapped because no entry is found, the mapping fails but the bind is still successful. However, if more than one DN is mapped, the bind fails.

Identity mapping enables the existing ACLs to work with Kerberos authentication. A client using Kerberos with a mapped identity has two distinct identities, both of which are evaluated in granting access.

Identity mapping has some costs. The internal searches at bind time impact performance and identity mapping requires additional setup to add the appropriate attributes to the entries to be mapped.

In this release, if default identity mapping is used, the administrator (either Kerberos or LDAP) must make sure that the data in the KDC and the data in the LDAP server are synchronized. If the data is not synchronized, incorrect results might be returned because of incorrect ACL evaluation.

Note: The object class, such as **KrbPrincipal** and the attributes such as **KrbPrincSubtree**, **KRbAliasedObjectName**, and **KrbHintAliases** are used to define a IBM Directory as a Kerberos KDC. See the Kerberos documentation for more information.

Chapter 8. Managing the IBM Directory schema

A schema is a set of rules that governs the way that data can be stored in the directory. The schema defines the type of entries allowed, their attribute structure and the syntax of the attributes.

Data is stored in the directory using directory entries. A entry consists of an object class, which is required, and its attributes. Attributes can be either required or optional. The object class specifies the kind of information that the entry describes and defines the set of attributes it contains. Each attribute has one or more associated values. See Chapter 11, “Working with directory entries” on page 141 for additional information about entries.

The schema for the IBM Directory Version 5.1 is predefined, however, you can modify the schema, if you have additional requirements.

The IBM Directory Server Version 5.1 includes dynamic schema support. The schema is published as part of the directory information, and is available in the Subschema entry (DN=″cn=schema″). You can query the schema using the ldap_search() API and modify it using ldap_modify(). See the *IBM Directory Client SDK Programming Reference* for more information about these APIs.

The schema has more configuration information than that included in the LDAP Version 3 Request For Comments (RFCs) or standard specifications. For example, for a given attribute, you can state which indexes must be maintained. This additional configuration information is maintained in the subschema entry as appropriate. An additional object class is defined for the subschema entry IBMsubschema , which has ″MAY″ attributes that hold the extended schema information.

IBM Directory Server requires that the schema defined for a naming context be stored in a special directory entry, ″cn=schema″. The entry contains all of the schema defined for the server. To retrieve schema information, you can perform an ldap_search by using the following:

```
DN: "cn=schema", search scope: base, filter: objectclass=subschema
or objectclass=*
```

The schema provides values for the following attribute types:

- objectClasses (See “Working with object classes” on page 73.)
- attributeTypes (See “Working with attributes” on page 79.)
- IBMAttributeTypes (See “The IBMAttributeTypes attribute type” on page 85.)
- matching rules (See “Matching rules” on page 86).
- ldap syntaxes (See “Attribute syntax” on page 88).

The syntax of these schema definitions is based on the LDAP Version 3 RFCs.

A sample schema entry might contain:

```
objectclasses=( 1.3.6.1.4.1.1466.101.120.111
                NAME 'extensibleObject'
                SUP top AUXILIARY )

 objectclasses=(  2.5.20.1
                 NAME 'subschema'
                 AUXILIARY MAY
                     ( dITStructureRules
                    $ nameForms
                    $ ditContentRules
                    $ objectClasses
                    $ attributeTypes
                    $ matchingRules
                    $ matchingRuleUse  ) )
 objectclasses=( 2.5.6.1
                 NAME 'alias'
                 SUP top STRUCTURAL
                 MUST aliasedObjectName )

 attributeTypes {
     ( 2.5.18.10 NAME 'subschemaSubentry' EQUALITY distinguishedNameMatch
       SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 NO-USER-MODIFICATION
       SINGLE-VALUE USAGE directoryOperation )
     ( 2.5.21.5 NAME 'attributeTypes'
         EQUALITY objectIdentifierFirstComponentMatch
         SYNTAX 1.3.6.1.4.1.1466.115.121.1.3 USAGE directoryOperation )
     ( 2.5.21.6 NAME 'objectClasses'
         EQUALITY objectIdentifierFirstComponentMatch
         SYNTAX 1.3.6.1.4.1.1466.115.121.1.37 USAGE directoryOperation )
         SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE directoryOperation )
 }


   ldapSyntaxes {
             ( 1.3.6.1.4.1.1466.115.121.1.5 DESC 'Binary' )
             ( 1.3.6.1.4.1.1466.115.121.1.7 DESC 'Boolean' )
             ( 1.3.6.1.4.1.1466.115.121.1.12 DESC 'DN' )
             ( 1.3.6.1.4.1.1466.115.121.1.15 DESC 'Directory String' )
             ( 1.3.6.1.4.1.1466.115.121.1.24 DESC 'Generalized Time' )
             ( 1.3.6.1.4.1.1466.115.121.1.26 DESC 'IA5 String' )
             ( 1.3.6.1.4.1.1466.115.121.1.27 DESC 'INTEGER' )
             ( 1.3.6.1.4.1.1466.115.121.1.50 DESC 'Telephone Number' )
             ( 1.3.6.1.4.1.1466.115.121.1.53 DESC 'UTC Time' )
   }

   matchingRules {
       ( 2.5.13.2 NAME 'caseIgnoreMatch'
         SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
       ( 2.5.13.0 NAME 'objectIdentifierMatch'
         SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )
       ( 2.5.13.30 NAME 'objectIdentifierFirstComponentMatch'
         SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )
       ( 2.5.13.4 NAME 'caseIgnoreSubstringsMatch'
         SYNTAX 1.3.6.1.4.1.1466.115.121.1.58 )
   }
```

As shown in the preceding example, it is not required that all of the attribute values of a given attribute type be provided in a single production.

The schema information can be modified through the ldap_modify API. Consult the *Client SDK Programming Reference* for additional information. With the DN ″cn=schema″ you can add, delete or replace an attribute type or an object class. To delete a schema entity, provide the oid in parenthesis (oid). You also can provide a full description. You can add or replace a schema entry with the LDAP Version 3 definition or with the IBM attribute extension definition or with both definitions.

Common schema support

The IBM Directory supports standard directory schema as defined in the following:

- The Internet Engineering Task Force (IETF) LDAP Version 3 RFCs, such as RFC 2252 and 2256.
- The Directory Enabled Network (DEN)
- The Common Information Model (CIM) from the Desktop Management Task Force (DMTF)
- The Lightweight Internet Person Schema (LIPS) from the Network Application Consortium

This version of LDAP includes the LDAP Version 3 defined schema in the default schema configuration. It also includes the DEN schema definitions.

IBM also provides a set of extended common schema definitions that other IBM products share when they exploit the LDAP directory. They include:
- Objects for white page applications such as eperson, group, country, organization, organization unit and role, locality, state, and so forth
- Objects for other subsystems such as accounts, services and access points, authorization, authentication, security policy, and so forth.

Working with object classes

An object class specifies a set of attributes used to describe an object. For example, if you created the object class **tempEmployee**, it could contain attributes associated with a temporary employee such as, **idNumber**, **dateOfHire**, or **assignmentLength**. You can add custom object classes to suit the needs of your organization. The IBM Directory Server schema provides some basic types of object classes, including:
- Groups
- Locations
- Organizations
- People

Note: Object classes that are specific to the IBM Directory Server have the prefix 'ibm-'.

Defining object classes

Object classes are defined by the characteristics of type, inheritance, and attributes.

Object class type

An object class can be one of three types:

Structural:
Every entry must belong to one and only one structural object class, which defines the base contents of the entry. This object class represents a real world object. Because all entries must belong to a structural object class, this is the most common type of object class.

Abstract:
This type is used as a superclass or template for other (structural) object classes. It defines a set of attributes that are common to a set of structural object classes. These object classes, if defined as subclasses of the abstract class, inherit the defined attributes. The attributes do not need to be defined for each of the subordinate object classes.

Auxiliary:
This type indicates additional attributes that can be associated with an

entry belonging to a particular structural object class. Although an entry, can belong to only a single structural object class, it may belong to multiple auxiliary object classes.

Object Class Inheritance

This version of the IBM Directory Server supports object inheritance for object class and attribute definitions. A new object class can be defined with parent classes (multiple inheritance) and the additional or changed attributes.

Each entry is assigned to a single structural object class. All object classes inherit from the abstract object class **top**. They can also inherit from other object classes. The object class structure determines the list of required and allowed attributes for a particular entry. Object class inheritance depends on the sequence of object class definitions. An object class can only inherit from object classes that precede it. For example, the object class structure for a person entry might be defined in the LDIF file as:

```
objectClass: top
objectClass: person
objectClass: organizationalPerson
```

In this structure, the organizationalPerson inherits from the person and the top object classes, while person object class only inherits from the top object class. Therefore, when you assign the organizationalPerson object class to an entry, it automatically inherits the required and allowed attributes from the superior object class. In this case, the person object class.

Schema update operations are checked against the schema class hierarchy for consistency before being processed and committed.

Attributes

Every object class includes a number of required attributes and optional attributes. Required attributes are the attributes that must be present in entries using the object class. Optional attributes are the attributes that may be present in entries using the object class.

Viewing object classes

You can view the object classes in the schema using either the Web Administration Tool, the preferred method or using the command line.

Using Web Administration:

Expand **Schema management** in the navigation area and click **Manage object classes**. A read-only panel is displayed that enables you to view the object classes in the schema and their characteristics. The object classes are displayed in alphabetical order. You can move one page backwards or forward by clicking Previous or Next. The field next to these buttons identifies the page that you are on. You can also use the drop down menu of this field to skip to a specific page. The first object class listed on the page is displayed with the page number to help you locate the object class you want to view. For example, if you were looking for the object class **person**, you expand the drop down menu and scroll down until you see **Page 14 of 16 nsLiServer** and **Page 15 of 16 printerLPR**. Because person is alphabetically between nsLiServer and printerLPR, you select Page 14 and click **Go**.

You can also display the object classes sorted by type. Select **Type** and click **Sort**. The object classes are sorted alphabetically within their type, Abstract, Auxiliary, or Structural. Similarly you can reverse the list order by selecting **Descending** and clicking **Sort**.

After you have located the object class that you want, you can view its type, inheritance, required attributes, and optional attributes. Expand the drop down menus for inheritance, required attributes, and optional attributes to see the full listings for each characteristic.

You can choose the object class operations you want to perform from the right-hand tool bar, such as:

- Add
- Edit
- Copy
- Delete

When you are finished click **Close** to return to the IBM Directory Server **Welcome** panel.

Using the command line:

To view the object classes contained in the schema issue the command:

```
ldapsearch -b cn=schema -s base objectclass=* objectclasses
```

Adding an object class

Using Web Administration:

If you have not done so already, expand **Schema management** in the navigation area, then click **Manage object classes**. To create a new object class:

1. Click **Add**.

   Note: You can also access this panel by expanding **Schema management** in the navigation area, then click **Add an object class**.
2. At the **General properties** tab:
   - Enter the **Object class name**. This is a required field, and is descriptive of the function of the object class. For example, **tempEmployee** for an object class used to track temporary employees.
   - Enter a **Description** of the object class, for example, **The object class used for temporary employees**.
   - Enter the **OID** for the object class. This is a required field. See “Object identifier (OID)” on page 89. If you do not have an OID, you can use the **Object class name** appended with **-oid**. For example, if the object class name is **tempEmployee**, then the OID is **tempEmployee-oid**. You can change the value of this field.
   - Select a **Superior object class** from the drop-down list. This determines the object class from which other attributes are inherited. Typically the **Superior object class** is **top**, however, it can be another object class. For example, a superior object class for **tempEmployee** might be **ePerson**.
   - Select an **Object class type**. See “Object class type” on page 73 for additional information about object class types.

- Click the Attributes tab to specify the required and the optional attributes for the object class and view the inherited attributes, or click **OK** to add the new object class or click **Cancel** to return to **Manage object classes** without making any changes.

3. At the **Attributes** tab:
   - Select an attribute from the alphabetical list of **Available attributes** and click **Add to required** to make the attribute required or click **Add to optional** to make the attribute optional for the object class. The attribute is displayed in the appropriate list of selected attributes.
   - Repeat this process for all the attributes you want to select.
   - You can move an attribute from one list to another or delete the attribute from the selected lists by selecting it and clicking the appropriate **Move to** or **Delete** button.
   - You can view the lists of required and optional inherited attributes. Inherited attributes are based on the **Superior object class** selected on the **General** tab. You cannot change the inherited attributes. However, if you change the **Superior object class** on the **General** tab, a different set of inherited attributes is displayed.
4. Click **OK** to add the new object class or click **Cancel** to return to **Manage object classes** without making any changes.

Note: If you clicked **OK** on the **General** tab without adding any attributes, you can add attributes by editing the new object class.

Using the command line:

To add an object class using the command line, issue the following command:

```
ldapmodify -D <adminDN> -w <adminPW> -i <filename>
```

where *<filename>*contains:

```
dn: cn=Schema
changetype: modify
add: objectclasses
objectclasses: ( <myobjectClass-oid> NAME '<myObjectClass>' DESC '<An object class
               I defined for my LDAP application>' SUP '<objectclassinheritance>'
               <objectclasstype> MAY (<attribute1> $ <attribute2>))
```

Editing an object class

Not all schema changes are allowed. See “Disallowed schema changes” on page 90 for change restrictions.

Using Web Administration:

If you have not done so already, expand **Schema management** in the navigation area, then click **Manage object classes**. To edit an object class:

1. Click the radio button next to the object class that you want to edit.
2. Click **Edit** .
3. Select a tab:
   - Use the **General** tab to:
     - Modify the **Description**.
     - Change the **Superior object class**. Select a Superior object class from the drop-down list. This determines the object class from which other attributes are inherited. Typically the **Superior object class** is **top**, however, it can be another object class. For example, a superior object class for **tempEmployee** might be **ePerson**.

  - Change the **Object class type**. Select an object class type. See “Object class type” on page 73 for additional information about object class types.
  - Click the Attributes tab to change the required and the optional attributes for the object class and view the inherited attributes, or click **OK** to apply your changes or click **Cancel** to return to **Manage object classes** without making any changes.
- Use the **Attributes** tab to:

  Select an attribute from the alphabetical list of **Available attributes** and click **Add to required** to make the attribute required or click **Add to optional** to make the attribute optional for the object class. The attribute is displayed in the appropriate list of selected attributes.

  Repeat this process for all the attributes you want to select.

  You can move an attribute from one list to another or delete the attribute from the selected lists by selecting it and clicking the appropriate **Move to** or **Delete** button.

  You can view the lists of required and optional inherited attributes. Inherited attributes are based on the **Superior object class** selected on the **General** tab. You cannot change the inherited attributes. However, if you change the **Superior object class** on the **General** tab, a different set of inherited attributes is displayed.

4. Click **OK** to apply the changes or click **Cancel** to return to **Manage object classes** without making any changes.

Using the command line:

View the object classes contained in the schema issue the command:

```
ldapsearch -b cn=schema -s base objectclass=* objectclasses
```

To edit an object class using the command line, issue the following command:

```
ldapmodify -D <adminDN> -w <adminPW> -i <filename>
```

where *<filename>*contains:

```
dn: cn=schema
changetype: modify
replace: objectclasses
objectclasses: ( <myobjectClass-oid> NAME '<myObjectClass>' DESC '<An object class
                I defined for my LDAP application>' SUP '<newsuperiorclassobject>'
                <newobjectclasstype> MAY (attribute1> $ <attribute2>
                 $ <newattribute3>) )
```

Copying an object class

Using Web Administration:

If you have not done so already, expand **Schema management** in the navigation area, then click **Manage object classes**. To copy an object class:

1. Click the radio button next to the object class that you want to copy.
2. Click **Copy**.
3. Select a tab:
   - Use the **General** tab to:
     - Modify the **object class name**. The default name is the copied object class name appended with the word COPY. For example **tempPerson** is copied as **tempPersonCOPY**.
     - Modify the **Description**.

- Modify the **OID**. The default OID is the copied object class OID appended with the word COPY. For example **tempPerson-oid** is copied as **tempPerson-oidCOPY**.
- Change the **Superior object class**. Select a superior object class from the drop-down list. This determines the object class from which other attributes are inherited. Typically the **Superior object class** is **top**, however, it can be another object class. For example, a superior object class for **tempEmployeeCOPY** might be **ePerson**.
- Change the **Object class type**. Select an object class type. See "Object class type" on page 73 for additional information about object class types.
- Click the **Attributes** tab to change the required and the optional attributes for the object class and view the inherited attributes, or click **OK** to apply your changes or click **Cancel** to return to **Manage object classes** without making any changes.

• Use the **Attributes** tab to:

Select an attribute from the alphabetical list of **Available attributes** and click **Add to required** to make the attribute required or click **Add to optional** to make the attribute optional for the object class. The attribute is displayed in the appropriate list of selected attributes.

Repeat this process for all the attributes you want to select.

You can move an attribute from one list to another or delete the attribute from the selected lists by selecting it and clicking the appropriate **Move to** or **Delete** button.

You can view the lists of required and optional inherited attributes. Inherited attributes are based on the **Superior object class** selected on the **General** tab. You cannot change the inherited attributes. However, if you change the **Superior object class** on the **General** tab, a different set of inherited attributes is displayed.

4. Click **OK** to apply the changes or click **Cancel** to return to **Manage object classes** without making any changes.

Using the command line:

View the object classes contained in the schema issue the command:

```
ldapsearch -b cn=schema -s base objectclass=* objectclasses
```

Select the object class that you want to copy. Use an editor to change the appropriate information and save the changes to <*filename*>. The issue the following command:

```
ldapmodify -D <adminDN> -w <adminPW> -i <filename>
```

where <*filename*>contains:

```
dn: cn=schema
changetype: modify
replace: objectclasses
objectclasses: ( <mynewobjectClass-oid> NAME '<mynewObjectClass>'
                DESC '<A new object class
                 I copied for my LDAP application>'
               SUP '<superiorclassobject>'<objectclasstype> MAY (attribute1>
               $ <attribute2> $ <attribute3>) )
```

Deleting an object class

Not all schema changes are allowed. See "Disallowed schema changes" on page 90 for change restrictions.

Using Web Administration:

If you have not done so already, expand **Schema management** in the navigation area, then click **Manage object classes**. To delete an object class:

1. Click the radio button next to the object class that you want to delete.
2. Click **Delete**.
3. You are prompted to confirm deletion of the object class. Click **OK** to delete the object class or click **Cancel** to return to **Manage object classes** without making any changes.

Using the command line:

View the object classes contained in the schema issue the command:

```
ldapsearch -b cn=schema -s base objectclass=* objectclasses
```

Select the object class you want to delete and issue the following command:

```
ldapmodify -D <adminDN> -w <adminPW> -i <filename>
```

where *<filename>*contains:

```
dn: cn=schema
changetype: modify
delete: objectclasses
objectclasses: ( <myobjectClass-oid> NAME '<myObjectClass>' DESC '<An object class
                I defined for my LDAP application>' SUP '<objectclassinheritance>'
                <objectclasstype> MAY (<attribute1> $ <attribute2>))
```

Working with attributes

Each directory entry has a set of attributes associated with it through it's object class. While the object class describes the type of information that an entry contains, the actual data is contained in attributes. An attribute is represented by one or more name-value-pairs that hold specific data element such as a name, an address, or a telephone number. The IBM Directory Server represents data as name-value-pairs, a descriptive attribute, such as commonName (cn), and a specific piece of information, such as John Doe.

For example, the entry for John Doe might contain several attribute name-value-pairs.

```
dn: uid=jdoe, ou=people, ou=mycompany, c=us,
objectClass: top
objectClass: person
objectClass: organizationalPerson
cn: John Doe
sn: Doe
givenName: Jack
givenName: John
```

While the standard attributes are already defined in the schema file, you can create, edit, copy, or delete attributes to suit the needs of your organization.

Viewing attributes

You can view the attributes in the schema using either the Web Administration Tool, the preferred method or using the command line.

Using Web Administration:

Expand **Schema management** in the navigation area and click **Manage attributes**. A read-only panel is displayed that enables you to view the attributes in the schema and their characteristics. The attributes are displayed in alphabetical order. You can move one page backwards or forward by clicking Previous or Next. The field next to these buttons identifies the page that you are on. You can also use the drop down menu of this field to skip to a specific page. The first object class listed on the page is displayed with the page number to help you locate the object class you want to view. For example, if you were looking for the attribute **authenticationUserID**, you expand the drop down menu and scroll down until you see **Page 3 of 62 applSystemHint** and **Page 4 of 62 authorityRevocatonList**. Because authenticationUserID is alphabetically between applSystemHint and authorityRevocatonList, you select Page 3 and click **Go**.

You can also display the attributes sorted by syntax. Select **Syntax** and click **Sort**. The attributes are sorted alphabetically within their syntax. See"Attribute syntax" on page 88 for a listing or the types of syntax. Similarly you can reverse the list order by selecting **Descending** and clicking **Sort**.

After you have located the attribute that you want, you can view its syntax, whether it is multi-valued, and the object classes that contain it. Expand the drop down menu for object classes to see the list of object classes for the attribute.

When you are finished click **Close** to return to the IBM Directory Server **Welcome** panel.

Using the command line:

To view the attributes contained in the schema issue the command:

```
ldapsearch -b cn=schema -s base objectclass=* attributeTypes IBMAttributeTypes
```

Adding an attribute

Use either of the following methods to create a new attribute. The Web Administration Tool is the preferred method.

Using Web Administration:

If you have not done so already, expand **Schema management** in the navigation area, then click **Manage attributes**. To create a new attribute:

1. Click **Add**.

   Note: You can also access this panel by expanding **Schema management** in the navigation area, then click **Add an attribute**.
2. Enter the **Attribute name**, for example, **tempId**. This is a required field and must begin with an alphabetical character.
3. Enter a **Description** of the attribute, for example, **The ID number assigned to a temporary employee**.
4. Enter the **OID** for the attribute. This is a required field. See "Object identifier (OID)" on page 89. If you do not have an OID, you can use the attribute name appended with -oid. For example, if the attribute name is **tempID**, then the default OID is **tempID-oid**. You can change the value of this field.
5. Select a **Superior attribute** from the drop-down list. The superior attribute determines the attribute from which properties are inherited.
6. Select a **Syntax** from the drop-down list. See "Attribute syntax" on page 88 for additional information about syntax.
7. Enter an **Attribute length** that specifies the maximum length of this attribute. The length is expressed as the number of bytes.
8. Select the **Allow multiple values** checkbox to enable the attribute to have multiple values. See the glossary entry for additional information about multiple values.

9. Select a matching rule from the each of the drop-down menus for equality, ordering, and substring matching rules. See the “Matching rules” on page 86 for a complete listing of matching rules.
10. Click the **IBM extensions** tab to specify additional extensions for the attribute, or click **OK** to add the new attribute or click **Cancel** to return to **Manage attributes** without making any changes.
11. At the **IBM extensions** tab:
    - Modify the **DB2 table name** . The server generates the DB2 table name if this field is left blank. If you enter a DB2 table name, you must also enter a DB2 column name.
    - Modify the **DB2 column name**. The server generates the DB2 column name if this field is left blank. If you enter a DB2 column name, you must also enter a DB2 table name.
    - Set the **Security class** by selecting **normal**, **sensitive**, or **critical** from the drop-down list.
    - Set the **Indexing rules** by selecting one or more indexing rules. See “Indexing rules” on page 87 for additional information about indexing rules.

      Note: At a minimum, it is recommended that you specify Equality indexing on any attributes that are to be used in search filters.
12. Click **OK** to add the new attribute or click **Cancel** to return to **Manage attributes** without making any changes.

Note: If you clicked OK on the General tab without adding any extensions, you can add extensions by the editing the new attribute.

Using the command line:

The following example adds an attribute type definition for an attribute called ″myAttribute″, with Directory String syntax (see “Attribute syntax” on page 88) and Case Ignore Equality matching (see “Matching rules” on page 86). The IBM-specific part of the definition says that the attribute data is stored in a column named ″myAttrColumn″ in a table called ″myAttrTable″. If these names were not specified, both the column and table name would have defaulted to ″myAttribute″. The attribute is assigned to the ″normal″ access class, and values have a maximum length of 200 bytes.

```
ldapmodify -D <admindn> -w <adminpw> -i myschema.ldif
```

where the **myschema.ldif** file contains:

```
dn: cn=schema
changetype: modify
add: attributetypes
attributetypes: ( myAttribute-oid NAME ( 'myAttribute' )
                 DESC 'An attribute I defined for my LDAP application'
                 EQUALITY 2.5.13.2 SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
                 USAGE userApplications )
-
add: ibmattributetypes
ibmattributetypes: ( myAttribute-oid  DBNAME ( 'myAttrTable' 'myAttrColumn' )
                    ACCESS-CLASS normal LENGTH 200 )
```

See “ldapmodify, ldapadd” on page 192 for more information about this command.

Editing an attribute

Not all schema changes are allowed. See “Disallowed schema changes” on page 90 for change restrictions.

Any part of a definition can be changed before you have added entries that use the attribute. Use either of the following methods to edit an attribute. The Web Administration Tool is the preferred method.

Using Web Administration:

If you have not done so already, expand **Schema management** in the navigation area, then click **Manage attributes**. To edit an attribute:

1. Click the radio button next to the attribute that you want to edit.
2. Click **Edit** .
3. Select a tab:
   - Use the **General** tab to:
     - Select a tab, either:
       - **General** to:
         - Modify the **Description**
         - Change the **Syntax**
         - Set the **Attribute length**
         - Change the **Multiple value** settings
         - Select a **Matching rule**
         - Change the **Superior attribute**
       - Click the **IBM extensions** tab to edit the extensions for the attribute, or click **OK** to apply your changes or click **Cancel** to return to **Manage attributes** without making any changes.
       - **IBM extensions**, if you are using the IBM Directory Server, to:
         - Change the **Security class**
         - Change the **Indexing rules**
     - Click **OK** to apply your changes or click **Cancel** to return to **Manage attributes** without making any changes.
4. Click **OK** to apply the changes or click **Cancel** to return to **Manage attributes** without making any changes.

Using the command line:

This example adds indexing to the attribute, so that searching on it is faster. Use the ldapmodify command and the LDIF file to change the definition:

```
ldapmodify -D <admindn> -w <adminpw> -i myschemachange.ldif
```

Where the **myschemachange.ldif** file contains:

```
dn: cn=schema
changetype: modify
replace: attributetypes
attributetypes: ( myAttribute-oid NAME ( 'myAttribute' ) DESC 'An attribute
                 I defined for my LDAP application' EQUALITY 2.5.13.2
                 SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE userApplications )
-
replace: ibmattributetypes
ibmattributetypes: ( myAttribute-oid  DBNAME ( 'myAttrTable' 'myAttrColumn' )
                  ACCESS-CLASS normal LENGTH 200 EQUALITY SUBSTR )
```

Note: Both portions of the definition (**attributetypes** and **ibmattributetypes**) must be included in the replace operation, even though only the **ibmattributetypes** section is changing. The only change is adding ″EQUALITY SUBSTR″ to the end of the definition to request indexes for equality and substring matching.

See “ldapmodify, ldapadd” on page 192 for more information about this command.

Copying an attribute

Use either of the following methods to copy an attribute. The Web Administration Tool is the preferred method.

Using Web Administration:

If you have not done so already, expand **Schema management** in the navigation area, then click **Manage attributes**. To copy an attribute:

1. Click the radio button next to the attribute that you want to copy.
2. Click **Copy**.
3. Modify the **Attribute name**. The default name is the copied attribute name appended with the word COPY. For example **tempID** is copied as **tempIDCOPY**.
4. Modify a **Description** of the attribute, for example, **The ID number assigned to a temporary employee**.
5. Modify the **OID**. The default OID is the copied attribute OID appended with the word COPYOID. For example **tempID-oid** is copied as **tempID-oidCOPYOID**.
6. Select a **Superior attribute** from the drop-down list. The superior attribute determines the attribute from which properties are inherited.
7. Select a **Syntax** from the drop-down list. See “Attribute syntax” on page 88 for additional information about syntax.
8. Enter a **Attribute length** that specifies the maximum length of this attribute. The length is expressed as the number of bytes.
9. Select the **Allow multiple values** checkbox to enable the attribute to have multiple values. See the glossary entry for additional information about multiple values.
10. Select a matching rule from the each of the drop-down menus for equality, ordering, and substring matching rules. See the “Matching rules” on page 86 for a complete listing of matching rules.
11. Click the **IBM extensions** tab to modify additional extensions for the attribute, or click **OK** to apply your changes or click **Cancel** to return to **Manage attributes** without making any changes.
12. At the **IBM extensions** tab:
    - Modify the **DB2 table name** . The server generates the DB2 table name if this field is left blank. If you enter a DB2 table name, you must also enter a DB2 column name.
    - Modify the **DB2 column name**. The server generates the DB2 column name if this field is left blank. If you enter a DB2 column name, you must also enter a DB2 table name.
    - Modify the **Security class** by selecting **normal**, **sensitive**, or **critical** from the drop-down list.
    - Modify the **Indexing rules** by selecting one or more indexing rules. See “Indexing rules” on page 87 for additional information about indexing rules.

      Note: At a minimum, it is recommended that you specify Equal indexing on any attributes that are to be used in search filters.
13. Click **OK** to apply your changes or click **Cancel** to return to **Manage attributes** without making any changes.

Note: If you clicked **OK** on the **General** tab without adding any extensions, you can add or modify extensions by editing the new attribute.

Using the command line:

View the attributes contained in the schema issue the command:

```
ldapsearch -b cn=schema -s base objectclass=* attributeTypes IBMAttributeTypes
```

Select the attribute that you want to copy. Use an editor to change the appropriate information and save the changes to <*filename*>. Then issue the following command:

```
ldapmodify -D <adminDN> -w <adminPW> -i <filename>
```

where <*filename*>contains:

```
dn: cn=schema
changetype: modify
add: attributetypes
attributetypes: ( <mynewAttribute-oid> NAME '<mynewAttribute>' DESC '<A new
                 attribute I copied for my LDAP application> EQUALITY 2.5.13.2
                 SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE userApplications )
-
add: ibmattributetypes
ibmattributetypes: ( myAttribute-oid  DBNAME ( 'myAttrTable' 'myAttrColumn' )
                    ACCESS-CLASS normal LENGTH 200 )
```

Deleting an attribute

Not all schema changes are allowed. See “Disallowed schema changes” on page 90 for change restrictions.

Use either of the following methods to delete an attribute. The Web Administration Tool is the preferred method.

Using Web Administration:

If you have not done so already, expand **Schema management** in the navigation area, then click **Manage attributes**. To delete an attribute:

1. Click the radio button next to the attribute that you want to delete.
2. Click **Delete**.
3. You are prompted to confirm deletion of the attribute. Click **OK** to delete the attribute or click **Cancel** to return to **Manage attributes** without making any changes.

Using the command line:

```
ldapmodify -D <admindn> -w <adminpw> -i myschemadelete.ldif
```

Where the **myschemadelete.ldif** file includes:

```
dn: cn=schema
changetype: modify
delete: attributetypes
attributetypes: ( myAttribute-oid NAME ( 'myAttribute' ) DESC 'An attribute
                 I defined for my LDAP application' EQUALITY 2.5.13.2
                 SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE userApplications )
-
delete: ibmattributetypes
ibmattributetypes: ( myAttribute-oid  DBNAME ( 'myAttrTable' 'myAttrColumn' )
                     ACCESS-CLASS normal LENGTH 200 EQUALITY SUBSTR )
```

See “ldapmodify, ldapadd” on page 192 for more information about this command.

The IBMAttributeTypes attribute type

The IBMAttributeTypes attribute can be used to define schema information not covered by the LDAP Version 3 standard for attributes. Values of IBMAttributeTypes must comply with the following grammar:

```
IBMAttributeTypesDescription = "(" whsp
        numericoid whsp
      [ "DBNAME"   qdescrs ]            ; at most 2 names (table, column)
      [ "ACCESS-CLASS" whsp IBMAccessClass whsp ]
      [ "LENGTH" wlen whsp ]            ; maximum length of attribute
      [ "EQUALITY" [ IBMwlen ] whsp ]   ; create index for matching rule
      [ "ORDERING" [ IBMwlen ] whsp ]   ; create index for matching rule
      [ "APPROX"   [ IBMwlen ] whsp ]   ; create index for matching rule
      [ "SUBSTR"   [ IBMwlen ] whsp ]   ; create index for matching rule
      [ "REVERSE"  [ IBMwlen ] whsp ]   ; reverse index for substring
      whsp ")"

   IBMAccessClass =
      "NORMAL"           / ; this is the default
      "SENSITIVE"        /
      "CRITICAL"         /
      "RESTRICTED"       /
      "SYSTEM"           /
      "OBJECT"

   IBMwlen = whsp len
```

Numericoid
Used to correlate the value in attributetypes with the value in IBMAttributeTypes.

DBNAME
You can provide 2 names at the most, if indeed 2 names are given. The first is the table name used for this attribute. The second is the column name used for the fully normalized value of the attribute in the table. If you provide only one name, it is used as the table name as well as the column name. If you do not provide any DBNAMEs, then the short attribute name is used (from the attributetypes).

ACCESS-CLASS
The access classification for this attribute type. If ACCESS-CLASS is omitted, it defaults to normal.

LENGTH
The maximum length of this attribute. The length is expressed as the number of bytes. IBM Directory Version 5.1 has a provision for specifying the length of an attribute. In the attributetypes value, the string:

```
( attr-oid ... SYNTAX syntax-oid{len} ... )
```

can be used to indicate that the attributetype with oid attr-oid has a maximum length.

EQUALITY, ORDERING, APPROX, SUBSTR, REVERSE
If any of these attributes are used, an index is created for the corresponding matching rule. The optional length specifies the width of the indexed column. For many syntaxes, you can share a single index to implement multiple matching rules. The IBM Directory Server takes advantage of this. It assigns a length when one is not provided by the user. SLAPD can also use a shorter length than what the user requested when it makes sense to do so. For example, when the length of the index exceeds the maximum length of the attribute, the index length is ignored.

Matching rules

A matching rule provides guidelines for string comparison during a search operation. These rules are divided into three categories:

- Equality
- Ordering
- Substring

Table 5.

Equality matching rules		
Matching Rule	**OID**	**Syntax**
caseExactIA5Match	1.3.6.1.4.1.1466.109.114.1	Directory String syntax
caseExactMatch	2.5.13.5 IA5	String syntax
caseIgnoreIA5Match	1.3.6.1.4.1.1466.109.114.2	IA5 String syntax
caseIgnoreMatch	2.5.13.2	Directory String syntax
distinguishedNameMatch	2.5.13.1	DN - distinguished name
generalizedTimeMatch	2.5.13.27	Generalized Time syntax
ibm-entryUuidMatch	1.3.18.0.2.22.2	Directory String syntax
integerFirstComponentMatch	2.5.13.29	Integer syntax - integral number
integerMatch	2.5.13.14	Integer syntax - integral number
objectIdentifierFirstComponentMatch	2.5.13.30	String for containing OIDs. The OID is a string containing digits (0-9) and decimal points (.).
objectIdentifierMatch	2.5.13.0	String for containing OIDs. The OID is a string containing digits (0-9) and decimal points (.)
octetStringMatch	2.5.13.17	Directory String syntax
telephoneNumberMatch	2.5.13.20	Telephone Number syntax
uTCTimeMatch	2.5.13.25	UTC Time syntax

Table 6.

Ordering matching rules		
Matching rule	**OID**	**Syntax**
caseExactOrderingMatch	2.5.13.6	Directory String syntax
caseIgnoreOrderingMatch	2.5.13.3	Directory String syntax
distinguishedNameOrderingMatch	1.3.18.0.2.4.405	DN - distinguished name
generalizedTimeOrderingMatch	2.5.13.28	Generalized Time syntax

Table 7.

Substring matching rules		
Matching rule	**OID**	**Syntax**
caseExactSubstringsMatch	2.5.13.7	Directory String syntax
caseIgnoreSubstringsMatch	2.5.13.4	Directory String syntax
telephoneNumberSubstringsMatch	2.5.13.21	Telephone Number syntax

Note: UTC-Time is time string format defined by ASN.1 standards. See ISO 8601 and X680. Use this syntax for storing time values in UTC-Time format. See "Generalized and UTC time" on page 94.

Indexing rules

Index rules attached to attributes make it possible to retrieve information faster. If only the attribute is given, no indexes are maintained. IBM Directory provides the following indexing rules:

- Equality
- Ordering
- Approximate
- Substring
- Reverse

Indexing rules specifications for attributes:

Specifying an indexing rule for an attribute controls the creation and maintenance of special indexes on the attribute values. This greatly improves the response time to searches with filters which include those attributes. The five possible types of indexing rules are related to the operations applied in the search filter.

Equality

Applies to the following search operations:

- equalityMatch '='

For example:

```
"cn = John Doe"
```

Ordering

Applies to the following search operation:

- greaterOrEqual '>='
- lessOrEqual '<='

For example:

```
"sn >= Doe"
```

Approximate

Applies to the following search operation:

- approxMatch '~='

For example:

```
"sn ~= doe"
```

Substring

Applies to the search operation using the substring syntax:

- substring '*'

For example:

```
"sn = McC*"
"cn = J*Doe"
```

Reverse

Applies to the following search operation:

- '*' substring

For example:

```
"sn = *baugh"
```

At a minimum, it is recommended that you specify equal indexing on any attributes that are to be used in search filters.

Attribute syntax

You can display syntax either alphabetically or by OID. Select **Syntax** or **OID** and click **Sort**. Similarly you can reverse the list order by selecting **Descending** and clicking **Sort**.

Table 8.

Syntax	OID
Attribute Type Description syntax	1.3.6.1.4.1.1466.115.121.1.3
Binary - octet string	1.3.6.1.4.1.1466.115.121.1.5
Boolean - TRUE/FALSE	1.3.6.1.4.1.1466.115.121.1.7
Directory String syntax	1.3.6.1.4.1.1466.115.121.1.15
DIT Content Rule Description syntax	1.3.6.1.4.1.1466.115.121.1.16
DITStructure Rule Description syntax	1.3.6.1.4.1.1466.115.121.1.17
DN - distinguished name	1.3.6.1.4.1.1466.115.121.1.12
Generalized Time syntax	1.3.6.1.4.1.1466.115.121.1.24
IA5 String syntax	1.3.6.1.4.1.1466.115.121.1.26
IBM Attribute Type Description	1.3.18.0.2.8.1
Integer syntax - integral number	1.3.6.1.4.1.1466.115.121.1.27
LDAP Syntax Description syntax	1.3.6.1.4.1.1466.115.121.1.54
Matching Rule Description	1.3.6.1.4.1.1466.115.121.1.30
Matching Rule Use Description	1.3.6.1.4.1.1466.115.121.1.31
Name Form Description	1.3.6.1.4.1.1466.115.121.1.35
Object Class Description syntax	1.3.6.1.4.1.1466.115.121.1.37
String for containing OIDs. The OID is a string containing digits (0-9) and decimal points (.). See "Object identifier (OID)" on page 89.	1.3.6.1.4.1.1466.115.121.1.38
Telephone Number syntax	1.3.6.1.4.1.1466.115.121.1.50
UTC Time syntax. UTC-Time is time string format defined by ASN.1 standards. See ISO 8601 and X680. Use this syntax for storing time values in UTC-Time format. See "Generalized and UTC time" on page 94.	1.3.6.1.4.1.1466.115.121.1.53

Object identifier (OID)

An object identifier (OID) is a string, of decimal numbers, that uniquely identifies an object. These objects are typically an object class or an attribute. These numbers can be obtained from the IANA (Internet Assigned Number Authority). The IANA Website is located at: http://www.iana.org/iana/.

If you do not have an OID, you can specify the object class or attribute name appended with **-oid**. For example, if you create the attribute tempID, you can specify the OID as **tempID-oid**.

The subschema entries

There is one subschema entry per server. All entries in the directory have an implied subschemaSubentry attribute type. The value of the subschemaSubentry attribute type is the DN of the subschema entry that corresponds to the entry. All entries under the same server share the same subschema entry, and their subschemaSubentry attribute type has the same value. The subschema entry has the hardcoded DN ’cn=schema’.

The subschema entry belongs to the object classes ’top’, ’subschema’, and ’IBMsubschema’. The ’IBMsubschema’ object class has no MUST attributes and one MAY attribute type (’IBMattributeTypes’).

The IBMsubschema object class

The IBMsubschema object class is used only in the subschema entry as follows:

```
( <objectClass-oid-TBD> NAME 'IBMsubschema' AUXILIARY
     MAY IBMattributeTypes )
```

Schema queries

The ldap_search() API can be used to query the subschema entry, as shown in the following example:

```
DN           : "cn=schema"
search scope : base
filter       : objectclass=subschema or objectclass=*
```

This example retrieves the full schema. To retrieve all of the values of selected attribute types, use the attrs parameter in ldap_search. You cannot retrieve only a specific value of a specific attribute type.

See the *IBM Directory Version 5.1: Client SDK Programming Reference* for more information about the ldap_search API.

Dynamic schema

To perform a dynamic schema change, use the ldap_modify API with a DN of ″cn=schema″. It is permissible to add, delete, or replace only one schema entity (for example, an attribute type or an object class) at a time.

To delete a schema entity, provide the oid in parentheses:

```
 ( oid )
```

You can also provide a full description. In either case, the matching rule used to find the schema entity to delete is objectIdentifierFirstComponentMatch.

To add or replace a schema entity, you MUST provide a LDAP Version 3 definition and you MAY provide the IBM definition. In all cases, you must provide only the definition or definitions of the schema entity that you want to affect.

For example, to delete the attribute type 'cn' (its OID is 2.5.4.3), use ldap_modify() with:

```
   LDAPMod  attr;
   LDAPMod *attrs[] = { &attr, NULL };
   char    *vals [] = { "( 2.5.4.3 )", NULL };
   attr.mod_op      = LDAP_MOD_DELETE;
   attr.mod_type    = "attributeTypes";
   attr.mod_values  = vals;
   ldap_modify_s(ldap_session_handle, "cn=schema", attrs);
```

To add a new attribute type bar with OID 20.20.20 that has a NAME of length 20 chars:

```
char    *vals1[] = { "( 20.20.20 NAME 'bar' SUP NAME )", NULL };
   char    *vals2[] = { "( 20.20.20 LENGTH 20 )", NULL };
   LDAPMod  attr1;
   LDAPMod  attr2;
   LDAPMod *attrs[] = { &attr1, &attr2, NULL };
   attr1.mod_op = LDAP_MOD_ADD;
   attr1.mod_type = "attributeTypes";
   attr1.mod_values = vals1;
   attr2.mod_op = LDAP_MOD_ADD;
   attr2.mod_type = "IBMattributeTypes";
   attr2.mod_values = vals2;
   ldap_modify_s(ldap_session_handle, "cn=schema", attrs);
```

See "Working with attributes" on page 79 for examples using the Web Administration Tool and the **ldapmodify** command.

See the *IBM Directory Version 5.1: Client SDK Programming Reference* for more information about the ldap_modify API.

Access controls

Dynamic schema changes can be performed only by a replication supplier or the administrator DN.

Replication

When a dynamic schema change is performed, it is replicated just like any other ldap_modify operation.

Disallowed schema changes

Not all schema changes are allowed. Change restrictions include the following:

- Any change to the schema must leave the schema in a consistent state.
- An attribute type that is a supertype of another attribute type may not be deleted. An attribute type that is a ″MAY″ or a ″MUST″ attribute type of an object class may not be deleted.
- An object class that is a superclass of another may not be deleted.
- Attribute types or object classes that refer to nonexisting entities (for example, syntaxes or object classes) cannot be added.
- Attribute types or object classes cannot be modified in such a way that they end up referring to nonexisting entities (for example, syntaxes or object classes).

Changes to the schema that affect the operation of the server are not allowed. The following schema definitions are required by the directory server. They must not be changed.

Object classes:

- accessGroup
- accessRole
- alias
- referral
- replicaObject
- top

Attributes:

- aclEntry
- aclPropagate
- aclSource
- aliasedObjectName, aliasedentryName
- businessCategory
- cn, commonName
- createTimestamp
- creatorsName
- description
- dn, distinguishedName
- entryOwner
- member
- modifiersName
- modifyTimestamp
- name
- o, organizationName, organization
- objectClass
- ou, organizationalUnit, organizationalUnitName
- owner
- ownerPropagate
- ownerSource
- ref
- replicaBindDN
- replicaBindMethod
- replicaCredentials, replicaBindCredentials
- replicaHost
- replicaPort
- replicaUpdateTimeInterval
- replicaUseSSL
- seeAlso

Syntaxes:
All

Matching rules:
All

Schema checking

When the server is initialized, the schema files are read and checked for consistency and correctness. If the checks fail, the server fails to initialize and issues an error message. During any dynamic schema change, the resulting schema is also checked for consistency and correctness. If the checks fail, an error is returned and the change fails. Some checks are part of the grammar (for example, an attribute type can have at most one supertype, or an object class can have any number of superclasses).

The following items are checked for attribute types:

- Two different attribute types cannot have the same name or OID.
- The inheritance hierarchy of attribute types does not have cycles.
- The supertype of an attribute type must also be defined, although its definition might be displayed later, or in a separate file.
- If an attribute type is a subtype of another, they both have the same USAGE.
- All attribute types have a syntax either directly defined or inherited.
- Only operational attributes can be marked as NO-USER-MODIFICATION.

The following items are checked for object classes:

- Two different object classes cannot have the same name or OID.
- The inheritance hierarchy of object classes does not have cycles.
- The superclasses of an object class must also be defined, although its definition might appear later or in a separate file.
- The ″MUST″ and ″MAY″ attribute types of an object class must also be defined, although its definition might appear later or in a separate file.
- Every structural object class is a direct or indirect subclass of top.
- If an abstract object class has superclasses, the superclasses must also be abstract.

Checking an entry against the schema

When an entry is added or modified through an LDAP operation, the entry is checked against the schema. By default, all checks listed in this section are performed. However, you can selectively disable some of them by providing an ibm-slapdSchemaCheck value to the ibmslapd.conf configuration directive. See the *IBM Directory Server Version 5.1 Installation and Configuration Guide* for information about schema configuration attributes.

To comply with the schema an entry is checked for the following conditions:

With respect to object classes:

- Must have at least one value of attribute type ″objectClass″.
- Can have any number of auxiliary object classes including zero. This is not a check, but a clarification. There are no options to disable this.
- Can have any number of abstract object classes, but only as a result of class inheritance. This means that for every abstract object class that the entry has, it also has a structural or auxiliary object class that inherits directly or indirectly from that abstract object class.
- Must have at least one structural object class.
- Must have exactly one immediate or base structural object class. This means that of all the structural object classes provided with the entry, they all must be superclasses of exactly one of them. The most derived

object class is called the ″immediate″ or ″base structural″ object class of the entry, or simply the ″structural″ object class of the entry.
- Cannot change its immediate structural object class (on ldap_modify).
- For each object class provided with the entry, the set of all of its direct and indirect superclasses is calculated; if any of those superclasses is not provided with the entry, then it is automatically added.

The validity of the attribute types for an entry is determined as follows:
- The set of MUST attribute types for the entry is calculated as the union of the sets of MUST attribute types of all of its object classes, including the implied inherited object classes. If the set of MUST attribute types for the entry is not a subset of the set of attribute types contained by the entry, the entry is rejected.
- The set of MAY attribute types for the entry is calculated as the union of the sets of MAY attribute types of all of its object classes, including the implied inherited object classes. If the set of attribute types contained by the entry is not a subset of the union of the sets of MUST and MAY attribute types for the entry, the entry is rejected.
- If any of the attribute types defined for the entry are marked as NO-USER-MODIFICATION, the entry is rejected.

The validity of the attribute type values for an entry is determined as follows:
- For every attribute type contained by the entry, if the attribute type is single-valued and the entry has more than one value, the entry is rejected.
- For every attribute value of every attribute type contained by the entry, if its syntax does not comply with the syntax checking routine for the syntax of that attribute, the entry is rejected.
- For every attribute value of every attribute type contained by the entry, if its length is greater than the maximum length assigned to that attribute type, the entry is rejected.

The validity of the DN is checked as follows:
- The syntax is checked for compliance with the BNF for DistinguishedNames. If it does not comply, the entry is rejected.
- It is verified that the RDN is made up with only attribute types that are valid for that entry.
- It is verified that the values of attribute types used in the RDN appear in the entry.

DEN schema support

The Directory-Enabled Network (DEN) specification defines a standard schema form that stores and describes the relationships among objects that represent users, applications, network elements, and networking services.

To support DEN, the IBM Directory Server provides the following features:
- Subclassing (class inheritance). Class definitions can be created from existing definitions through subclassing. The new class definition inherits the properties from the parent class definition. The SUP option in the object class definition is used to specify the parent (or superior) object classes.
- LDAP syntaxes required by DEN, which include the following:
  - Boolean
  - DN

- – Directory String
- – Generalized Time
- – UTC Time
- – IA5 String
- – Integer

iPlanet compatibility

The parser used by the IBM Directory Server allows the attribute values of schema attribute types (objectClasses and attributeTypes) to be specified using the grammar of iPlanet. For example, descrs and numeric-oids can be specified with surrounding single quotation marks (as if they were qdescrs). However, the schema information is always made available through ldap_search. As soon as a single dynamic change (using ldap_modify) is performed on an attribute value in a file, the whole file is replaced by one where all attribute values follow the IBM Directory Version 5.1 specifications. Because the parser used on the files and on ldap_modify requests is the same, an ldap_modify that uses the iPlanet grammar for attribute values is also handled correctly.

When a query is made on the subschema entry of a iPlanet server, the resulting entry can have more than one value for a given OID. For example, if a certain attribute type has two names (such as ’cn’ and ’commonName’), then the description of that attribute type is provided twice, once for each name. The IBM Directory Server can parse a schema where the description of a single attribute type or object class appears multiple times with the same description (except for NAME and DESCR). However, when the IBM Directory Server publishes the schema it provides a single description of such an attribute type with all of the names listed (the short name comes first). For example, here is how iPlanet describes the common name attribute:

```
( 2.5.4.3 NAME 'cn'
  DESC 'Standard Attribute'
   SYNTAX '1.3.6.1.4.1.1466.115.121.1.15' )

( 2.5.4.3 NAME 'commonName'
  DESC 'Standard Attribute, alias for cn'
  SYNTAX '1.3.6.1.4.1.1466.115.121.1.15' )
```

This is how the IBM Directory Server describes it:

```
( 2.5.4.3 NAME ( 'cn' 'commonName' ) SUP name )
```

The IBM Directory Server supports subtypes. If you do not want ’cn’ to be a subtype of name (which deviates from the standard), you can declare the following:

```
( 2.5.4.3 NAME ( 'cn' 'commonName' )
     DESC 'Standard Attribute'
     SYNTAX '1.3.6.1.4.1.1466.115.121.1.15' )
```

The first name (’cn’) is taken as the preferred or short name and all other names after ’cn’ as alternate names. From this point on, the strings ’2.3.4.3’, ’cn’ and ’commonName’ (as well as their case-insensitive equivalents) can be used interchangeably within the schema or for entries added to the directory.

Generalized and UTC time

There are different notations used to designate date and time-related information. For example, the fourth day of February in the year 1999 can be written as:

```
2/4/99
4/2/99
99/2/4
4.2.1999
04-FEB-1999
```

as well as many other notations.

IBM Directory Server standardizes the timestamp representation by requiring the LDAP servers to support two syntaxes:

- The Generalized Time syntax, which takes the form:

  ```
  YYYYMMDDHHMMSS[.|,fraction][(+|-HHMM)|Z]
  ```

  There are 4 digits for the year, 2 digits each for the month, day, hour, minute, and second, and an optional fraction of a second. Without any further additions, a date and time is assumed to be in a local time zone. To indicate that a time is measured in Coordinated Universal Time, append a capital letter Z to a time or a local time differential. For example:

  ```
   "19991106210627.3"
  ```

  which in local time is 6 minutes, 27.3 seconds after 9 p.m. on 6 November 1999.

  ```
  "19991106210627.3Z"
  ```

  which is the coordinated universal time.

  ```
  "19991106210627.3-0500"
  ```

  which is local time as in the first example, with a 5 hour difference in relation to the coordinated universal time.

  If you designate an optional fraction of a second, a period or a comma is required. For local time differential, a ’+’ or a ’-’ must precede the hour-minute value
- The Universal time syntax, which takes the form:

  ```
  YYMMDDHHMM[SS][(+ | -)HHMM)|Z]
  ```

  There are 2 digits each for the year, month, day, hour, minute, and optional second fields. As in GeneralizedTime, an optional time differential can be specified. For example, if local time is a.m. on 2 January 1999 and the coordinated universal time is 12 noon on 2 January 1999, the value of UTCTime is either:

  ```
  "9901021200Z"
       or
  "9901020700-0500"
  ```

  If the local time is a.m. on 2 January 2001 and the coordinated universal time is 12 noon on 2 January 2001, the value of UTCTime is either:

  ```
  "0101021200Z"
       or
  "0101020700-0500"
  ```

  UTCTime allows only 2 digits for the year value, therefore the usage is not recommended.

The supported matching rules are generalizedTimeMatch for equality and generalizedTimeOrderingMatch for inequality. Substring search is not allowed. For example, the following filters are valid:

```
 generalized-timestamp-attribute=199910061030
 utc-timestamp-attribute>=991006
 generalized-timestamp-attribute=*
```

The following filters are not valid:

```
generalized-timestamp-attribute=1999*
utc-timestamp-attribute>=*1010
```

Chapter 9. Replication

Replication is a technique used by directory servers to improve performance and reliability. The replication process keeps the data in multiple directories synchronized.

Replication provides two main benefits:
- Redundancy of information - replicas back up the content of their supplier servers.
- Faster searches - search requests can be spread among several different servers, all having the same content, instead of a single server. This improves the response time for the request completion.

Replication topology

Replication topology is managed in a new way in the IBM Directory Server Version 5.1. Some terminology used in describing replication:

Cascading replication
A replication topology in which there are multiple tiers of servers. A peer/master server replicates to a set of read-only (forwarding) servers which in turn replicate to other servers. Such a topology off-loads replication work from the master servers.

Consumer server
A server which receives changes through replication from another (supplier) server.

Credentials
Identify the method and required information that the supplier uses in binding to the consumer. For simple binds, this includes the DN and password. The credentials are stored in an entry the DN of which is specified in the replica agreement.

Forwarding server
A read-only server that replicates all changes sent to it. This contrasts to a peer/master server in that a peer/master server does not replicate changes sent to it from another peer/master server; it only replicates changes that are originally made on the peer/master server.

Master server
A server which is writable (can be updated) for a given subtree.

Nested subtree
A subtree within a replicated subtree of the directory.

Peer server
The term used for a master server when there are multiple masters for a given subtree.

Replication agreement
Information contained in the directory that defines the ’connection’ or ’replication path’ between two servers. One server is called the supplier (the one that sends the changes) and the other is the consumer (the one

that receives the changes). The agreement contains all the information needed for making a connection from the supplier to the consumer and scheduling replication.

Replication context
Identifies the root of a replicated subtree. The ibm-replicationContext auxiliary object class may be added to an entry to mark it as the root of a replicated area. The configuration information related to replication is maintained in a set of entries created below a replication context.

Replica group
The first entry created under a replication context has objectclass ibm-replicaGroup and represents a collection of servers participating in replication. It provides a convenient location to set ACL's to protect the replication topology information. The administration tools currently support one replica group under each replication context, named **ibm-replicagroup=default**.

Replica subentry
Below a replica group entry, one or more entries with objectclass ibm-replicaSubentry may be created; one for each server participating in replication as a supplier. The replica subentry identifies the role the server plays in replication: master or read-only. A read-only server might, in turn, have replication agreements to support cascading replication.

Replicated subtree
A portion of the DIT that is replicated from one server to another. Under this design, a given subtree can be replicated to some servers and not to others. A subtree can be writable on a given server, while other subtrees may be read-only.

Schedule
Replication can be scheduled to occur at particular times, with changes on the supplier accumulated and sent in a batch. The replica agreement contains the DN for the entry that supplies the schedule.

Supplier server
A server which sends changes to another (consumer) server.

Specific entries in the directory are identified as the roots of replicated subtrees, by adding the ibm-replicationContext objectclass to them. Each subtree is replicated independently. The subtree continues down through the directory information tree (DIT) until reaching the leaf entries or other replicated subtrees. Entries are added below the root of the replicated subtree to contain the replication configuration information. These entries are one or more replica group entries, under which are created replica subentries. Associated with each replica subentry are replication agreements that identify the servers that are supplied (replicated to) by each server, as well as defining the credentials and schedule information.

Through replication, a change made to one directory is propagated to one or more additional directories. In effect, a change to one directory shows up on multiple different directories. The IBM Directory supports an expanded master-slave replication model. Replication topologies are expanded to include:

- Replication of subtrees of the Directory Information Tree (DIT) to specific servers
- A multi-tier topology referred to as cascading replication
- Assignment of server role (master or replica) by subtree.
- Multiple master servers, referred to as peer to peer replication.

The advantage of replicating by subtrees is that a replica does not need to replicate the entire directory. It can be a replica of a part, or subtree, of the directory.

The expanded model changes the concept of master and replica. These terms no longer apply to servers, but rather to the roles that a server has regarding a particular replicated subtree. A server can act as a master for some subtrees and as a replica for others. The term, master, is used for a server that accepts client updates for a replicated subtree. The term, replica, is used for a server that only accepts updates from other servers designated as a supplier for the replicated subtree.

There are three types of directories as defined by function: *master/peer*, *cascading*, and *read-only*.

Table 9. Server roles

Master/peer	The master/peer server contains the master directory information from where updates are propagated to the replicas. All changes are made and occur on the master server, and the master is responsible for propagating these changes to the replicas. There can be several servers acting as masters for directory information, with each master responsible for updating other master servers and replica servers. This is referred to as peer replication. Peer replication can improve performance and reliability. Performance is improved by providing a local server to handle updates in a widely distributed network. Reliability is improved by providing a backup master server ready to take over immediately if the primary master fails. **Notes:** 1. Master servers replicate all client updates, but do not replicate updates received from other masters. 2. Updates to the same entry made by multiple servers might cause inconsistencies in directory data because there is no conflict resolution. See "ldapdiff" on page 229 for information on resynchronizing servers.
Cascading (forwarding)	A cascading server is a replica server that replicates all changes sent to it. This contrasts to a master/peer server in that a master/peer server only replicates changes that are made by clients connected to that server. A cascading server can relieve the replication workload from the master servers in a network which contains many widely dispersed replicas.
Replica (read-only)	An additional server that contains a copy of directory information. The replicas are copies of the master (or the subtree that it is a replica of). The replica provides a backup of the replicated subtree.

You can request updates on a replica server, but the update is actually forwarded to the master server by returning a referral to the client. If the update is successful, the master server then sends the update to the replicas. Until the master has completed replication of the update, the change is not reflected on the replica server where it was originally requested. If the replication fails, it is repeated even if the master is restarted. Changes are replicated in the order in which they are made on the master.

If you are no longer using a replica, you must remove the replica agreement from the supplier. Leaving the definition causes the server to queue up all updates and use unnecessary directory space. Also, the supplier continues trying to contact the missing consumer to retry sending the data.

Replication agreements

A replication agreement is an entry in the directory with the object class **ibm-replicationAgreement** created beneath a replica subentry to define replication from the server represented by the subentry to another server. These objects are similar to the replicaObject entries used by prior versions of the Directory Server. The replication agreement consists of the following items:

- A user friendly name, used as the naming attribute for the agreement.
- An LDAP URL specifying the server, port number, and whether SSL should be used.
- The consumer server id, if known -- ’unknown’ for a server whose server id is not known (as in a server running a previous release).
- The DN of an object containing the credentials used by the supplier to bind to the consumer.
- An optional DN pointer to an object containing the schedule information for replication. If the attribute is not present, changes are replicated immediately.

The user friendly name might be the consumer server name or some other descriptive string.

The consumer server id is used by the administrative GUI to traverse the topology. Given the consumer’s server ID, the GUI can find the corresponding subentry and its agreements. To aid in enforcing the accuracy of the data, when the supplier binds to the consumer, it retrieves the server IDfrom the root DSE and compares it to the value in the agreement. A warning is logged if the server IDs do not match.

Because the replication agreement can be replicated, a DN to a credentials object is used. This allows the credentials to be stored in a nonreplicated area of the directory. Replicating the credentials objects (from which ’clear text’ credentials must be obtainable) represents a potential security exposure. The cn=localhost suffix is an appropriate default location for creating credentials objects. Use of a separate object also makes it easier to support various authentication methods; new object classes can be created rather than trying to make sense of numerous optional attributes.

Object classes are defined for each of the supported authentication methods:

- Simple bind
- SASL EXTERNAL mechanism with SSL
- Kerberos authentication

You can designate that part of a replicated subtree not be replicated by adding the ibm-replicationContext auxiliary class to the root of the subtree, without defining any replica subentries.

Note: The Web Administration Tool also refers to agreements as ’queues’ when referring to the set of changes that are waiting to be replicated under a given agreement.

Defining replication topology

To define a replication topology, you must

1. Create a master server and define what it contains. Select the subtree that you want to be replicated and specify the server as the master.
2. Create credentials to be used by the supplier.

3. Create a replica server.
4. Export data to the replica.

Creating a replicated subtree (master server)

Note: The server must be running to perform this task.

This task designates an entry as the root of an independently replicated subtree and creates a **ibm-replicasubentry** representing this server as the single master for the subtree. To create a replicated subtree, you must designate the subtree that you want the server to replicate. Expand the Replication management category in the navigation area and click **Manage topology**.

If the entry that you trying to add is not a suffix in the server, before you can use the **Add subtree** function, you must ensure that its ACLs defined as follows:

For non-filtered ACLs:

```
ownersource: <same as the entry DN>
ownerpropagate: TRUE

aclsource: <same as the entry DN>
aclpropagate: TRUE
```

For filtered ACLs:

```
ibm-filteraclinherit: FALSE
```

To satisfy the ACL requirements, if the entry is not a suffix in the server, edit the ACL for that entry in the **Manage entries** panel. Select the entry and click **Edit ACL**. If you want to add Non-fltered ACLs, select that tab and add an entry **cn=this** with the role **access-id** for both ACLs and owners. Ensure that **Propagate ACLs** and **Propagate owner** are checked. If you want to add Fltered ACLs select that tab and add an entry **cn=this** with the role **access-id** for both ACLs and owners. Ensure that **Accumulate filtered ACLs** is unchecked and that **Propagate owner** is checked. See "Working with Access Control Lists" on page 161 for more detailed information.

- Click **Add subtree**.
- Enter the DN of the subtree that you want to replicate or click **Browse** to expand the entries to select the entry that is to be the root of the subtree.
- Enter the master server referral URL. This must be in the form of an LDAP URL, for example:

  ```
  ldap://<myservername>.<mylocation>.<mycompany>.com
  ```

  Note: The master server referral URL is optional. It is used only:
  - If server contains (or will contain) any read-only subtrees.
  - To define a referral URL that is returned for updates to any read-only subtree on the server.
- Click **OK**.
- The new server is displayed on the **Manage topology** panel under the heading **Replicated subtrees**.

To add additional masters (peers), you first add the server as a read-only replica of the existing masters (see "Creating a replica server" on page 104), initialize the directory data, and then promote the server to be a master (see "Moving or promoting a server" on page 108).

Initially, the **ibm-replicagroup** object created by this process inherits the ACL of the root entry for the replicated subtree. These ACLs might be inappropriate for controlling access to the replication information in the directory.

For the Add subtree operation to be successful, the entry DN which you are adding must have correct ACLs, if it is not a suffix in the server.

For Non-filtered ACLs :

- ownersource : <*the entry DN*>
- ownerpropagate : TRUE
- aclsource : <*the entry DN*>
- aclpropagate: TRUE

Filtered ACLs :

- ownersource : <*the entry DN*>
- ownerpropagate : TRUE
- ibm-filteraclinherit : FALSE
- ibm-filteraclentry : <*any value*>

Use the **Edit ACLs** function to set ACLs for the replication information associated with the newly created replicated subtree (see "Editing access control lists" on page 110).

Note: On the Linux, Solaris, and HP-UX platforms, if a referral fails because the server being referred to is not running, ensure that the environment variable LDAP_LOCK_REC has been set in your system environment. No specific value is required.

```
set LDAP_LOCK_REC=anyvalue
```

Creating credentials

Expand the Replication management category in the navigation area of the Web Administration Tool and click **Manage credentials**

1. Select the location that you want to use to store the credentials from the list of subtrees. The Web Administration Tool allows you to define credentials in two places:
   - **cn=replication,cn=localhost**, which keeps the credentials only on the current server.
   - Within the replicated subtree, in which case the credentials are replicated with the rest of the subtree.

   Placing credentials in cn=replication,cn=localhost is considered more secure. Credentials placed in the replicated subtree are created beneath the **ibm-replicagroup=default** entry for that subtree.

   Note: If no subtrees are displayed, go to "Creating a replicated subtree (master server)" on page 101 for instructions about creating the subtree that you want to replicate.
2. Click **Add**.
3. Enter the name for the credentials you are creating, for example, **mycreds**.
4. Select the type of authentication method you want to use and click **Next**.
   - If you selected simple bind authentication:
     a. Enter the DN that the server uses to bind to the replica, for example, cn=any

b. Enter the password uses when it binds to the replica, for example, secret.

c. Enter the password again to confirm that there are no typographical errors.

d. If you want, enter a brief description of the credentials.

e. Click **Finish**.

Note: You might want to record the credential's bind DN and password for future reference. You will need this password when you create the replica agreement.

- If you selected Kerberos authentication:

  a. If you want, enter a brief description of the credentials. No other information is necessary. See "Setting Kerberos" on page 66 for additional information.

  b. Click **Finish**.

  By default, the supplier uses its own service principal to bind with the consumer. For example, if the supplier is named master.our.org.com and the realm is SOME.REALM, the DN is **ibm-Kn=ldap/master1.our.org.com@SOME.REALM**. If there is more than one supplier, you must specify the principal and password to be used by all of the suppliers.

  On the server where you created the credentials:

  a. Expand **Directory management** and click on **Manage entries**.

  b. Select the subtree where you stored the credentials, for example **cn=localhost** and click **Expand**.

  c. Select **cn=replication** and click **Expand**.

  d. Select the kerberos credentials (ibm-replicationCredentialsKerberos) and click **Edit attributes**.

  e. Click the **Other attributes** tab.

  f. Enter the **replicaBindDN**, for example, **ibm-kn=myprincipal@SOME.REALM**.

  g. Enter the **replicaCredentials**. This is the KDC password used for **myprincipal**.

  Note: This principal and password should be the same as the ones you use to run **kinit** from the command line.

  On the replica

  a. Click **Manage replication properties** in the navigation area.

  b. Select a supplier from the **Supplier information** drop-down menu or enter the name of the replicated subtree for which you want to configure supplier credentials.

  c. Click **Edit**.

  d. Enter the replication bindDN. In this example, **ibm-kn=myprincipal@SOME.REALM**.

  e. Enter and confirm the **Replication bind password**. This is the KDC password used for **myprincipal**.

- If you selected SSL with certificate authentication you do not need to provide any additional information, if you are using the server's certificate. If you choose to use a certificate other than the server's:

  a. Enter the key file name.

b. Enter the key file password.
c. Enter the key label.
d. If you want, enter a brief description.
e. Click **Finish**.

See "Secure Sockets Layer" on page 47 for additional information.

Creating a replica server

Note: The server must be running to perform this task.

Expand the **Replication management** category in the navigation area and click **Manage topology**.

1. Select the subtree that you want to replicate and click **Show topology**.
2. Click the arrow next to the **Replication topology** selection to expand the list of supplier servers.
3. Select the supplier server and click **Add replica**.

On the **Server** tab of the **Add replica** window:

- Enter the host name and port number for the replica you are creating. The default port is 389 for non-SSL and 636 for SSL. These are required fields.
- Select whether to enable SSL communications.
- Enter the replica name or leave this field blank to use the host name.
- Enter the replica ID. If the server on which you are creating the replica is running, click **Get replica ID** to automatically prefill this field. This is a required field, if the server you are adding is going to be a peer or forwarding server. It is recommended for all IBM Directory Server Version 5.1 replica servers.
- Enter a description of the replica server.

On the **Additional** tab:

1. Specify the credentials that the replica uses to communicate with the master.

   Note: The Web Administration Tool allows you to define credentials in two places:

   - **cn=replication,cn=localhost**, which keeps the credentials only on the server that uses them
   - Within the replicated subtree, in which case the credentials are replicated with the rest of the subtree.

   Placing credentials in cn=replication,cn=localhost is considered more secure. Credentials placed in the replicated subtree are created beneath the **ibm-replicagroup=default** entry for that subtree.

   a. Click **Select**.
   b. Select the location for the credentials you want to use. Preferably this is cn=replication,cn=localhost.
   c. Click **Show credentials**.
   d. Expand the list of credentials and select the one you want to use.
   e. Click **OK**.

   See "Creating credentials" on page 102 for additional information on agreement credentials.

2. Specify a replication schedule from the drop-down list or click **Add** to create one. See "Creating replication schedules" on page 112
3. From the list of supplier capabilities, you can deselect any capabilities that you do not want replicated to the consumer.

   If your network has a mix of servers at different releases, capabilities are available on later releases that are not available on earlier releases. Some capabilities, like filter ACLs and password policy, make use of operational attributes that are replicated with other changes. In most cases, if these features are used, you want all servers to support them. If all of the servers do not support the capability, you do not want to use it. For example, you would not want different ACLs in effect on each server. However, there might be cases where you might want to use a capability on the servers that support it, and not have changes related to the capability replicated to servers that do not support the capability. In such cases, you can use the capabilities list to mark certain capabilities to not be replicated.
4. Click **OK** to create the replica.

Copying data to the replica

After creating the replica, you must now export the topology from the master to the replica. This is a manual procedure.

On the master server create an LDIF file for the data. To copy all the data contained on the master server, issue the command:

```
db2ldif -o <masterfile.ldif>
```

If you want to copy just the data from a single subtree the command is:

```
db2ldif -o <masterfile.ldif> -s <subtreeDN>
```

On the machine where you are creating the replica:

1. Ensure that the suffixes used by the master are defined in the **ibmslapd.conf** file.
2. Stop the replica server.
3. Copy the <*masterfile.ldif*> to the replica and issue the command:

   ```
   ldif2db -r no -i <masterfile.ldif>
   ```

   The replication agreements, schedules, credentials (if stored in the replicated subtree) and entry data are loaded on the replica.
4. Start the server.

Adding the supplier information to the replica

You need to change the replica's configuration to identify who is authorized to replicate changes to it, and add a referral to a master.

On the machine where you are creating the replica:

1. Click **Manage replication properties** in the navigation area.
2. Click **Add**.
3. Select a supplier from the **Replicated subtree** drop-down menu or enter the name of the replicated subtree for which you want to configure supplier credentials. If you are editing supplier credentials, this field is not editable.
4. Enter the replication bindDN. In this example, cn=any.

   Note: You can use either of these two options, depending on your situation.

    - Set the replication bind DN (and password) and a default referral for all subtrees replicated to a server using the 'default credentials and referral'. This might be used when all subtrees are replicated from the same supplier.
    - Set the replication bind DN and password independently for each replicated subtree by adding supplier information for each subtree. This might be used when each subtree has a different supplier (that is, a different master server for each subtree).
5. Depending on the type of credential, enter and confirm the credential password. (You previously recorded this for future use.)
   - **Simple Bind** - Specify the DN and password
   - **Kerberos** - If the credentials on the supplier do not identify the principal and password, that is, the server's own service principal is to be used, then the bind DN is ibmkn=ldap/<*yourservername@yourrealm*>. If the credentials has a principal name such as <*myprincipal@myrealm*>, use that as the DN. In either case a password in not needed.
   - **SSL w/ EXTERNAL bind** - Specify the subject DN for the certificate and no password

   See "Creating credentials" on page 102.
6. Click **OK**.
7. You must restart the replica for the changes to take effect.

See "Modifying replication properties" on page 110 for additional information.

The replica is in a suspended state and no replication is occurring. After you have finished setting up your replication topology, you must click **Manage queues**, select the replica and click **Suspend/resume** to start replication. See "Managing queues" on page 113 for more detailed information. The replica now receives updates from the master.

Setting up a forwarding server

If you have set up a replication topology (see "Creating a replicated subtree (master server)" on page 101) with a master (server1) and a replica (server2), you can change the role of server2 to that of a forwarding server. To do this you need to create a new replica (server3) under server2.

1. Connect Web Administration to the master (server1)
2. Expand the Replication management category in the navigation area and click **Manage topology**.
3. Select the subtree that you want to replicate and click **Show topology**.
4. Click the arrow next to the **Replication topology** selection to expand the list of supplier servers.
5. Click the arrow next to the **server1** selection to expand the list of servers.
6. Select server2 and click **Add replica**.
7.

   On the **Server** tab of the **Add replica** window:
   - Enter the host name and port number for the replica (server3) you are creating. The default port is 389 for non-SSL and 636 for SSL. These are required fields.
   - Select whether to enable SSL communications.
   - Enter the replica name or leave this field blank to use the host name.

- Enter the replica ID. If the server on which you are creating the replica is running, click **Get replica ID** to automatically prefill this field. This is a required field, if the server you are adding is going to be a peer or forwarding server. It is recommended for all IBM Directory Server Version 5.1 replica servers.
- Enter a description of the replica server.

On the **Additional** tab:

a. Specify the credentials that the replica uses to communicate with the master.

   Note: The Web Administration Tool allows you to define credentials in two places:
   - **cn=replication,cn=localhost**, which keeps the credentials only on the server that uses them.
   - Within the replicated subtree, in which case the credentials are replicated with the rest of the subtree.

   Placing credentials in cn=replication,cn=localhost is considered more secure. Credentials placed in the replicated subtree are created beneath the **ibm-replicagroup=default** entry for that subtree.

   1) Click **Select**.
   2) Select the location for the credentials you want to use. Preferably this is cn=replication,cn=localhost.
   3) Click **Show credentials**.
   4) Expand the list of credentials and select the one you want to use.
   5) Click **OK**.

   See "Creating credentials" on page 102 for additional information on agreement credentials.

b. Specify a replication schedule from the drop-down list or click **Add** to create one. See "Creating replication schedules" on page 112.

c. From the list of supplier capabilities, you can deselect any capabilities that you do not want replicated to the consumer.

   If your network has a mix of servers at different releases, capabilities are available on later releases that are not available on earlier releases. Some capabilities, like filter ACLs and password policy, make use of operational attributes that are replicated with other changes. In most cases, if these features are used, you want all servers to support them. If all of the servers do not support the capability, you do not want to use it. For example, you would not want different ACLs in effect on each server. However, there might be cases where you might want to use a capability on the servers that support it, and not have changes related to the capability replicated to servers that do not support the capability. In such cases, you can use the capabilities list to mark certain capabilities to not be replicated.

d. Click **OK** to create the replica.

8. Copy data from server2 to the new replica server3. See "Copying data to the replica" on page 105 for information on how to do that.
9. Add the supplier agreement to server3 that makes server2 a supplier to server 3 and server 3 a consumer to server2. See "Adding the supplier information to the replica" on page 105 for information on how to do this.

Your topology in now:

- server1 (master)

– server2 (forwarder)
  - server3 (replica)

Managing topologies

Topologies are specific to the replicated subtrees.

Viewing the topology

Note: The server must be running to perform this task.

Expand the **Replication management** category in the navigation area and click **Manage topology**.

1. Select the subtree that you want to view and click **Show topology**.

The topology is displayed in the Replication topology list. Expand the topologies by clicking the blue triangles. From this list you can:

- Add a replica.
- Edit information on an existing replica.
- Change to a different supplier server for the replica or promote the replica to a master server
- Delete a replica.

Adding a replica

See “Creating a replica server” on page 104.

Editing a replica

You can change the following information for the replica:

On the **Server** tab you can only change

- Hostname
- Port
- Enable SSL
- Description

On the **Additional** tab you can change:

- Credentials - see “Creating credentials” on page 102.
- Replication schedules - see “Creating replication schedules” on page 112.
- Change the capabilities replicated to the consumer replica. From the list of supplier capabilities, you can deselect any capabilities that you do not want replicated to the consumer.
- When you are finished, click **OK**.

Moving or promoting a server

1. Select the subtree you want to promote or move and Click **Move**.
2. Select the server that you want to move the replica to, or select **Replication topology** to promote the replica to a master. Click **Move**.
3. In some cases the **Select credentials** panel will pop up asking for a credential which is located in a place other than cn=replication,cn=localhost. In such situations you must provide a credential object which is located in a place other than cn=replication,cn=localhost. Select the credentials the subtree is going to use form the existing sets of credentials or create new credentials. See “Creating credentials” on page 102.

4. Click **OK**.

The change in the topology tree reflects the moving of the server.

See "Peer to peer replication" on page 114 for more information.

Replicating a subtree

Note: The server must be running to perform this task.

Expand the **Replication management** category in the navigation area and click **Manage topology**.

- Click **Add subtree**.
- Enter the DN of the subtree that you want to replicate or click **Browse** to expand the entries to select the entry that is to be the root of the subtree.
- Enter the master server referral URL. This must be in the form of an LDAP URL, for example:
  ```
  ldap://<myservername>.<mylocation>.<mycompany>.com
  ```
- Click **OK**.
- The new server is displayed on the **Manage topology** panel under the heading **Replicated subtrees**.

Note: On the Linux, Solaris, and HP-UX platforms, if a referral fails, ensure that the environment variable LDAP_LOCK_REC has been set in your system environment. No specific value is required.

```
set LDAP_LOCK_REC=anyvalue
```

Editing a subtree

Use this option to change the URL of the master server that this subtree and its replicas send updates to. You need do this if you change the port number or host name of the master server, change the master to a different server

1. Select the subtree you want to edit.
2. Click **Edit subtree**.
3. Enter the master server referral URL. This must be in the form of an LDAP URL, for example:
   ```
   ldap://<mynewservername>.<mylocation>.<mycompany>.com
   ```

Depending on the role being played by the server on this subtree (whether it is a master, replica or forwarding), different labels and buttons appear on the panel.

- When the subtree's role is replica, a label indicating that the server acts as a replica or forwarder is displayed along with the button **Make server a master**. If this button is clicked then the server which the Web Administration Tool is connected to becomes a master.
- When the subtree is configured for replication only by adding the auxiliary class (no default group and subentry present), then the label **This subtree is not replicated** is displayed along with the button **Replicate subtree**. If this button is clicked, the default group and the subentry is added so that the server with which the Web Administration Tool is connected to becomes a master.
- If no subentries for the master servers are found,, the label **No master server is defined for this subtree** is displayed along with the button titled **Make server a master**. If this button is clicked, the missing subentry is added so that the server with which the Web Administration Tool is connected to becomes a master.

Removing a subtree

1. Select the subtree you want to remove
2. Click **Delete subtree**.
3. When asked to confirm the deletion, click **OK**.

The subtree is removed from the **Replicated subtree** list.

Note: This operation succeeds only if the ibm-replicaGroup=default is entry is empty.

Quiescing the subtree

This function is useful when you want to perform maintenance on or make changes to the topology. It minimizes the number of updates that can be made to the server. A quiesced server does not accept client requests. It accepts requests only from an administrator using the Server Administration control.

This function is Boolean.

1. Click **Quiesce/Unquiesce** to quiesce the subtree.
2. When asked to confirm the action, click **OK**.
3. Click **Quiesce/Unquiesce** to unquiesce the subtree.
4. When asked to confirm the action, click **OK**.

Editing access control lists

Replication information (replica subentries, replication agreements, schedules, possibly credentials) are stored under a special object, **ibm-replicagroup=default**. The ibm-replicagroup object is located immediately beneath the root entry of the replicated subtree. By default, this subtree inherits ACL from the root entry of the replicated subtree. This ACL might not be appropriate for controlling access to replication information.

Required authorities:

- Control replication - You must have write access to the ibm-replicagroup=default object (or be the owner/administrator).
- Cascading control replication - You must have write access to the ibm-replicagroup=default object (or be the owner/administrator).
- Control queue - You must have write access to the replication agreement.

To view ACL properties using the Web Administration Tool utility and to work with ACLs, see “Working with Access Control Lists” on page 161.

See Chapter 12, “Access Control Lists” on page 149 for additional information.

Modifying replication properties

Expand the **Replication management** category in the navigation area and click **Manage replication properties**.

On this panel you can:

- Change the maximum number of pending changes to return from replication status queries. The default is 200.
- Add, edit, or delete supplier information.

Adding supplier information

1. Click **Add**.
2. Select a supplier from the drop-down menu or enter the name of the replicated subtree that you want to add as a supplier .
3. Enter the replication bind DN for the credentials.

   Note: You can use either of these two options, depending on your situation.
   - Set the replication bind DN (and password) and a default referral for all subtrees replicated to a server using the 'default credentials and referral'. This might be used when all subtrees are replicated from the same supplier.
   - Set the replication bind DN and password independently for each replicated subtree by adding supplier information for each subtree. This might be used when each subtree has a different supplier (that is, a different master server for each subtree).
4. Depending on the type of credential, enter and confirm the credential password. (You previously recorded this for future use.)
   - **Simple Bind** - specify the DN and password
   - **Kerberos** - specify a pseudo DN of the form 'ibm-kn=LDAP-service-name@realm' without a password
   - **SSL w/ EXTERNAL bind** - specify the subject DN for the certificate and no password

   See "Creating credentials" on page 102.
5. Click **OK**.

The subtree of the supplier is added to the Supplier information list.

Editing supplier information

1. Select the supplier subtree that you want to edit.
2. Click **Edit**.
3. If you are editing **Default credentials and referral**, which is used to create the cn=Master Server entry under cn=configuration, enter the URL of the server from which the client wants to receive replica updates in the Default supplier's LDAP URL field. This needs to be a valid LDAP URL (ldap://). Otherwise, skip to step 4.
4. Enter the replication bind DN for the new credentials you want to use.
5. Enter and confirm the credential password.
6. Click **OK**.

Removing supplier information

1. Select the supplier subtree that you want to remove.
2. Click **Delete**.
3. When asked to confirm the deletion, click **OK**.

The subtree is removed from the Supplier information list.

Creating replication schedules

You can optionally define replication schedules to schedule replication for particular times, or to not replicate during certain times. If you do not use a schedule, the server schedules replication whenever a change is made. This is equivalent to specifying a schedule with immediate replication starting at 12:00 AM on all days.

Expand the **Replication management** category in the navigation area and click **Manage schedules**.

On the **Weekly schedule** tab, select the subtree for which you want to create the schedule and click **Show schedules**. If any schedules exist, they are displayed in the **Weekly schedules** box. To create or add a new schedule:

1. Click **Add**.
2. Enter a name for the schedule. For example **schedule1**.
3. For each day, Sunday through Saturday, the daily schedule is specified as **None**. This means that no replication update events are scheduled. The last replication event, if any, is still in effect. Because this is a new replica, there are no prior replication events, therefore, the schedule defaults to immediate replication.
4. You can select a day and click **Add a daily schedule** to create a daily replication schedule for it. If you create a daily schedule it becomes the default schedule for each day of the week. You can:
   - Keep the daily schedule as the default for each day or select a specific day and change the schedule back to none. Remember that the last replication event that occurred is still in effect for a day that has no replication events scheduled.
   - Modify the daily schedule by selecting a day and clicking **Edit a daily schedule**. Remember changes to a daily schedule affect all days using that schedule, not just the day you selected.
   - Create a different daily schedule by selecting a day and clicking **Add a daily schedule**. After you have created this schedule it is added to the **Daily schedule** drop-down menu. You must select this schedule for each day that you want the schedule to be used.

   See “Creating a daily schedule” for more information on setting up daily schedules.
5. When you are finished, click **OK**.

Creating a daily schedule

Expand the **Replication management** category in the navigation area and click **Manage schedules**.

On the **Daily schedule** tab, select the subtree for which you want to create the schedule and click **Show schedules**. If any schedules exist, they are displayed in the **Daily schedules** box. To create or add a new schedule:

1. Click **Add**.
2. Enter a name for the schedule. For example **monday1**.
3. Select the time zone setting, either UTC or local.
4. Select a replication type from the drop-down menu:

Immediate
Performs any pending entry updates since the last replication event and then updates entries continuously until the next scheduled update event is reached.

Once Performs all pending updates prior to the starting time. Any updates made after the start time wait until the next scheduled replication event.

5. Select a start time for the replication event.
6. Click **Add**. The replication event type and time are displayed.
7. Add or remove events to complete your schedule. The list of events is refreshed in chronological order.
8. When you are finished, click **OK**.

For example:

Table 10.

Replication type	**Start time**
Immediate	12:00 AM
Once	10:00 AM
Once	2:00 PM
Immediate	4:00 PM
Once	8:00 PM

In this schedule, the first replication event occurs at midnight and updates any pending changes prior to that time. Replication updates continue to be made as they occur until 10:00 AM. Updates made between 10:00 AM and 2:00 PM wait until 2:00 PM to be replicated. Any updates made between 2:00 PM and 4:00 PM wait the replication event scheduled at 4:00 PM, afterwards replication updates continue until the next scheduled replication event at 8:00 PM. Any updates made after 8:00 PM wait until the next scheduled replication event.

Note: If replication events are scheduled too closely together, a replication event might be missed if the updates from the previous event are still in progress when the next event is scheduled.

Managing queues

This task allows you to monitor status of replication for each replication agreement (queue) used by this server.

Expand the **Replication management** category in the navigation area and click **Manage queues**.

Select the replica for which you want to manage the queue.

- Depending on the status of the replica, you can click **Suspend/resume** to stop or start replication.
- Click **Force replication** to replicate all the pending changes regardless of when the next replication is scheduled.
- Click **Queue details**, for more complete information about the replica's queue. You can also manage the queue from this selection.
- Click **Refresh** to update the queues and clear server messages.

Queue details

If you clicked **Queue details**, three tabs are displayed:

- Status
- Last attempted details
- Pending changes

The **Status** tab displays the replica name, its subtree, its status, and a record of replication times. From this panel you can suspend or resume replication by clicking **Resume**. Click **Refresh** to update the queue information.

The **Last attempted details** tab gives information about the last update attempt. If an entry is not able to be loaded press **Skip blocking entry** to continue replication with the next pending entry. Click **Refresh** to update the queue information.

The **Pending changes** tab shows all the pending changes to the replica. If replication is blocked you can delete all the pending changes by clicking **Skip all**. Click **Refresh** to update the list of pending changes to reflect any new update or updates that have been processed.

Note: If you choose to skip blocking changes, you must ensure that the consumer server is eventually updated. See “ldapdiff” on page 229 for more information.

Peer to peer replication

Peer replication is a replication topology in which multiple servers are masters. However, unlike a multi-master environment, no conflict resolution is done among peer servers. LDAP servers accept the updates provided by peer servers, and update their own copies of the data. No consideration is given for the order the updates are received, or whether multiple updates conflict.

Use peer replication only in environments where the update vectors are well known. Updates to particular objects within the directory must be done only by one peer server. This is intended to prevent the scenario of one server deleting an object, followed by another server modifying the object. This scenario creates the possibility of a peer server receiving a delete command followed by a modify command; which creates a conflict.

Creating a complex replication topology

Use this high level overview as a guide for setting up a complex replication topology.

1. Start or place all of the potential peer masters and replicas-to-be in Configuration only mode.
2. Start the ’first’ master and configure it as a master for the context.
3. Load the data for the subtree to be replicated on the ’first’ master, if the data is not already loaded.
4. Select the subtree to be replicated.
5. Add all of the potential peer masters as replicas of the ’first’ master.
6. Add all of the other replicas.
7. Move the other peer masters to promote them.
8. Add replica agreements for the replicas to each of the peer masters.

Note: If the credentials are to be created in **cn=replication,cn=localhost**, the credentials must be created on each server after they are restarted. Replication by the peers fails until the credential objects are created.

9. Add replica agreements for the other masters to each of the peer masters. The ’first’ master already has that information.
10. Quiesce the replicated subtree.
11. Use Queue management to skip all for each queue.
12. Export the data for the replicated subtree from the ’first’ master.
13. Unquiesce the subtree.
14. Import the data for the replicated subtree on to each replica and peer master.
15. Manage the replication properties on each replica and peer master to set the credentials to be used by suppliers.
16. Restart the replicas and peer masters as soon as each is ready.

Creating a peer server

Using the forwarding topology created in “Setting up a forwarding server” on page 106, you can promote a server to be a peer. In this example you are going to promote the replica (server3) to be a peer to the master server (server1).

1. Connect Web Administration to the master (server1).
2. Expand the Replication management category in the navigation area and click **Manage topology**.
3. Select the subtree that you want to replicate and click **Show topology**.
4. Click the arrow next to the **Replication topology** selection to expand the list of supplier servers.
5. Click the arrow next to the **server1** selection to expand the list of servers.
6. Click the arrow next to the **server2** selection to expand the list of servers.

   Note: The server you want to move must be a leaf replica with no subordinate replicas.
7. Select server3 and click **Move**.
8. Select **Replication topology** to promote the replica to a master. Click **Move**.
9. In some cases the **Select credentials** panel will pop up asking for a credential which is located in a place other than cn=replication,cn=localhost. In such situations you must provide a credential object which is located in a place other than cn=replication,cn=localhost. Select the credentials the subtree is going to use form the existing sets of credentials or create new credentials. See “Creating credentials” on page 102.
10. Click **OK**. The topology is now:
    - server1 (master-peer)
      – server2 (replica)
      – server3 (replica)
    - server3 (master-peer)

    As it can be seen from the topology, an agreement between server2 and server3 which was exisitng previously was deleted. An agreement between server1 and server3 is created (required for a peer master) A subentry corresponding to the machine server3 is also created. To complete the setting up of a peer master, an agreement from server3 to server1 must be added.
11. Select server3 and click **Add replica**.

12. Make server1 a replica for server3. See "Creating a replica server" on page 104. On the **Additional** tab, use the credential object which is located inside the default group entry of the replicated subtree.

    Note: For peer replication to work correctly ensure that the credentials you uses for each agreement have the same DN and password. This DN cannot be the same as the AdminDN.
13. Because this is a peer environment you must also make server2 a replica for server3. See "Creating a replica server" on page 104. Now server2 can be updated by either server1 or server3.

The peer topology is now:
- server1 (master-peer)
  - server2 (replica)
  - server3 (replica)
- server3 (master-peer)
  - server1 (replica)
  - server2 (replica)

If you wanted to promote server2 so that the topology contains three master-peer servers, use the same procedure that was used to promote server3. Agreements must be made among all peer servers. The three-peer topology is now:
- server1 (master-peer)
  - server2 (replica)
  - server3 (replica)
- server3 (master-peer)
  - server1 (replica)
  - server2 (replica)
- server2 (master-peer)
  - server1 (replica)
  - server3 (replica)

Demoting a master

To change the role of a server from a master to a replica do the following:
1. Connect the Web Administration Tool to the server that you want to demote.
2. Click **Manage topology**.
3. Select the subtree and click **Show topology**.
4. Delete all the agreements for the server you want to demote.
5. Select the server you are demoting and click **Move**.
6. Select the server under which you are going to place the demoted server and click **Move**.
7. Just as you would for a new replica, create new supplier agreements between the demoted server and its supplier. See "Creating a replica server" on page 104 for instructions.

Command line replication

The following are examples of setting up replication using the command line utilities and an LDIF file. The scenarios are of increasing complexity:

- One master and one replica
- One master, one forwarder, and one replica
- Two peer/masters, two forwarders, and four replicas.

Each of these scenarios assume that you are creating new replicated subtrees.

Note:

```
dn: o=ibm,c=us
objectclass: organization
objectclass: ibm-replicationContext
```

is the subtree you want to create. If this entry already exists, then modify it to add **objclass=ibm-replicationContext** instead of adding the entire entry.

Setting up a single master and replica

To create a replica for a subtree, you need to create a replica agreement between the master and the replica, see "Replication agreements" on page 100. This agreement needs to be loaded on both the master and the replica.

The relationship between the two servers is that the master is a supplier to the replica and the replica is a consumer of the master.

To create the master (master) and replica (replica1) for the subtree **o=ibm,c=us**:

1. At the machine where the master is located, create a file to contain the agreement information, for example, ***myreplicainfofile***, where ***myreplicainfofile*** contains:

   Note: Replace all occurrences of <*master-uuid*> in the following files with the value of the **ibm-slapdServerId** attribute from the master server's **cn=Configuration** entry. This value is generated by the server the first time it is started. You can find it either by performing an **ldapsearch** of the **cn=Configuration** entry or using the **grep** command on the **ibmslapd.conf** file, if you have a UNIX-based system. Similarly, all occurrences of the <*replica1-uuid*> must be replaced with the value of the **ibm-slapdServerId** attribute from the replica server's **cn=Configuration** entry.

```
###Replication Context  - needs to be on all suppliers and consumers
dn: cn=replication,cn=localhost
objectclass: container

dn: o=ibm,c=us
objectclass: organization
objectclass: ibm-replicationContext


###Copy the following to servers at v5.1 or later.

###Replica Group
dn: ibm-replicaGroup=default, o=IBM, c=US
objectclass: top
objectclass: ibm-replicaGroup
ibm-replicaGroup: default

###Bind Credentials/method to replica server - replication agreement
```

```
###points to this.
dn: cn=replica1 BindCredentials,cn=replication,cn=localhost
objectclass: ibm-replicationCredentialsSimple
cn: replica1 BindCredentials
replicaBindDN: cn=master
replicaCredentials: master
description: Bindmethod of master to replica1

###Replica SubEntry
dn: ibm-replicaServerId=<master-uuid>,ibm-replicaGroup=default,o=IBM, c=US
objectclass: top
objectclass: ibm-replicaSubentry
ibm-replicaServerId: <master-uuid>
ibm-replicationServerIsMaster: true
cn: master
description: master server

###Replication Agreement to Replica Server
dn: cn=replica1,ibm-replicaServerId=<master-uuid>,
    ibm-replicaGroup=default,o=IBM,c=US
objectclass: top
objectclass: ibm-replicationAgreement
cn: replica1
ibm-replicaConsumerId: <replica1-uuid>
ibm-replicaUrl: ldap://<replicahostname:replicaport>
ibm-replicaCredentialsDN: cn=replica1 BindCredentials,cn=replication,
                          cn=localhost
description: replica server number one
```

2. Stop the master, if it is not already stopped.
3. Issue the command:

   ```
   ldif2db -r no -i <myreplicainfofile>
   ```
4. Issue the command:

   ```
   db2ldif -o <masterfile.ldif>
   ```

   See "db2ldif utility" on page 228 for more information.
5. Copy *<masterfile.ldif>* to the machine where replica1 is located.
6. Stop the replica if it is running.
7. You must configure replica1 to be a replica server. Use an editor to add the following stanza to the **ibmslapd.conf** file on replica1:

   ```
   dn: cn=Master Server, cn=configuration
   objectclass: ibm-slapdReplication
   cn: Master Server
   ibm-slapdMasterDN: <cn=masterbndn>
   ibm-slapdMasterPW: <masterbnpw>
   ibm-slapdMasterReferral: ldap://<masterhostname>:<masterport>/
   ```
8. Save the **ibmslapd.conf** file.
9. Issue the following command:

   ```
   ldif2db -r no -i <masterfile.ldif>
   ```
10. Start master and replica1.

Note: If you are copying a subtree to a v4.1 or earlier server, you must not copy the **ibm-replicagroup=default** subtree and you must remove the **ibm-replicationcontext** auxiliary class, because neither of these are supported by the 4.1 schema.

Setting up a single master, a forwarder, and a replica

This procedure is similar to the one for a single master and replica, except that the entire topology must be added to each of the servers and the content of the agreement information file is more complex. The file now includes information for the forwarding server and supplier-consumer information.

The supplier-consumer relationship for this scenario is:
- The master is a supplier to the forwarder.
- The forwarder has two roles:
  1. A consumer of the master
  2. A supplier to the replica
- The replica is a consumer of the forwarder.

To create the master (master), a forwarder (forwarder1), and replica (replica1) for the subtree **o=ibm,c=us**:

1. At the machine where the master server is located, create a file to contain the agreement information, for example ***myreplicainfofile*** where ***myreplicainfofile*** contains:

   Note: Replace all occurrences of <*master-uuid*> in the following files with the value of the **ibm-slapdServerId** attribute from the master server's **cn=Configuration** entry. This value is generated by the server the first time it is started. You can find it either by performing an **ldapsearch** of the **cn=Configuration** entry or using the **grep** command on the **ibmslapd.conf** file, if you have a UNIX-based system. Similarly, all occurrences of the <*forwarder1-uuid*> and the <*replica1-uuid*> must be replaced with the value of the **ibm-slapdServerId** attribute from the respective server's **cn=Configuration** entry.

```
dn: cn=replication,cn=localhost
objectclass: container

dn: o=ibm,c=us
objectclass: organization
objectclass: ibm-replicationContext

dn: ibm-replicaGroup=default, o=ibm,c=us
objectclass: top
objectclass: ibm-replicaGroup
ibm-replicaGroup: default

dn: cn=forwarder1 BindCredentials,cn=replication,cn=localhost
objectclass: ibm-replicationCredentialsSimple
       #or ibm-replicationCredentialsExternal or
       #ibm-replicationCredentialsKerberos
cn: forwarder1 BindCredentials
replicaBindDN: <cn=forw1bnddn>
replicaCredentials: <forw1bndpw>
cn:forwarder1 BindCredentials
description: Bindmethod of master to forwarder1

dn: cn=replica1 BindCredentials,cn=replication,cn=localhost
objectclass: ibm-replicationCredentialsSimple
cn: replica1 BindCredentials
replicaBindDN: <cn=rep1bnddn>
replicaCredentials: <rep1bndpw>
description: Bindmethod of forwarder1 to replica1

dn: ibm-replicaServerId=<master-uuid>,ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicaSubentry
ibm-replicaServerId: <master-uuid>        #whatever the id is in the config
ibm-replicationServerIsMaster: true  #true if master, false if forwarder
cn: master
```

```
description: master ibm-replicaSubentry

dn: ibm-replicaServerId=<forwarder1-uuid>,ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicaSubentry
ibm-replicaServerId: <forwarder1-uuid>ibm-replicationServerIsMaster: false
cn: forwarder1
description: forwarder1 ibm-replicaSubentry

dn: cn=forwarder1,ibm-replicaServerId=<master-uuid>,
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicationAgreement
cn: forwarder1
ibm-replicaConsumerId: <forwarder1-uuid>
ibm-replicaUrl: ldap://<forwarder1hostname:forwarder1port>
ibm-replicaCredentialsDN: cn=forwarder1 BindCredentials,cn=replication,
                          cn=localhost
description: master1 to forwarder1 agreement

dn: cn=replica1,ibm-replicaServerId=<forwarder1-uuid>,
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicationAgreement
cn: replica1
ibm-replicaConsumerId: <replica1-uuid>-uuid
ibm-replicaUrl: ldap://<replica1hostname:replica1port>
ibm-replicaCredentialsDN: cn=replica1 BindCredentials,cn=replication,
                          cn=localhost
description: forwarder1 to replica1 agreement
```

2. Stop the master, if it is not already stopped.
3. Issue the command:

   ```
   ldif2db -r no -i  <myreplicainfofile>
   ```
4. Issue the command:

   ```
   db2ldif -o <masterfile.ldif>
   ```

   See "db2ldif utility" on page 228 for more information.
5. Copy *<masterfile.ldif>* to the machine where forwarder1 is located.
6. Stop forwarder1 if it is running.
7. You must configure forwarder1 to be a forwarding server. Use an editor to add the following stanza to the **ibmslapd.conf** file on forwarder1:

   ```
   dn: cn=Master Server, cn=configuration
   objectclass: ibm-slapdReplication
   cn: Master Server
   ibm-slapdMasterDN: <cn=masterbnddn>
   ibm-slapdMasterPW: <masterbndp>w
   ibm-slapdMasterReferral: ldap://masterhostname:masterport/
            #referral to master when trying to add to consumer.
            #Referral can also be added to replicaContext, which would be
            #checked first for a valid server.
   ```
8. Save the **ibmslapd.conf** file.
9. Copy *<masterfile.ldif>* to the machine where replica1 is located.
10. Stop replica1 if it is running.
11. You must configure replica1 to be a replica server. Use an editor to add the following stanza to the **ibmslapd.conf** file on replica1:

    ```
    dn: cn=Master Server, cn=configuration
    objectclass: ibm-slapdReplication
    cn: Master Server
    ```

```
ibm-slapdMasterDN: <cn=forw1bndn>
ibm-slapdMasterPW: <forw1bnpw>
ibm-slapdMasterReferral: ldap://forw1hostname:forw1port/
```

12. Save the **ibmslapd.conf** file.
13. At the machines where forwarder1 and replica1 are located, issue the following command:

    ```
    ldif2db -r no -i <masterfile.ldif>
    ```

14. Start master, forwarder1 and replica1.

Setting up a complex topology

In this example the topology is more complex. It includes two peer-masters (peer1 and peer2), two forwarders (forwarder1 and forwarder2) and four replicas (replica1, replica2, replica3, and replica4). The relationship among the servers is as follows:

- peer1 and peer2 are peer-master servers. That means that while they receive updates from each other, they only replicate entries received from clients. While both masters have the same entry content, only the server that has received the client request replicates the entry. Both masters are suppliers and consumers to each other and suppliers to the forwarding servers.
- forwarder1 and forwarder 2 have two roles. They are both consumers of peer1 and peer2 and suppliers to their respective replicas. They do not perform any client updates. They pass replicated updates to their consumers. In this scenario
  - forwarder1 is a supplier to replica1 and replica2
  - forwarder2 is a supplier to replica3 and replica4

  There is no interaction between forwarder1 and forwarder2.
- replica 1 and replica 2 are consumers of forwarder1 and replica3 and replica4 are consumers of forwarder2.

To create the peer-masters (peer1 and peer2), the forwarders (forwarder1 and forwarder2), and the replicas (replica1, replica2, replica3, and replica4) for the subtree **o=ibm,c=us**:

1. Stop peer1 and peer2.
2. You must configure peer1 and peer2 to be peer servers. Use an editor to add the following stanza to the **ibmslapd.conf** files on peer1 and peer2:

   ```
   dn: cn=MasterServer, cn=configuration
   objectclass: ibm-slapdReplication
   cn: Master Server
   ibm-slapdMasterDN: cn=master
   ibm-slapdMasterPW: master
   ```

   Note: It is critical that these stanzas be exactly the same on both servers because this example uses a credentials object that is shared on all the servers. However, there is no requirement that each server be configured with the same replication bind DN.
3. Save the **ibmslapd.conf** files.
4. At the machine where the master server, peer1, is located, create a file to contain the agreement information, for example *mycredentialsfile* where *mycredentialsfile* contains:

   ```
   dn: cn=simple,cn=replication,cn=localhost
   objectclass: ibm-replicationCredentialsSimple
   cn: peer2 BindCredentials
   replicaBindDN: cn=master
   replicaCredentials: master
   ```

```
description: Bindmethod for topology
```

5. Issue the command:

   ```
   ldif2db -r no -i  <mycredentialsfile>
   ```
6. Copy *<mycredentialsfile>* to the machines where peer2, forwarder1, and forwarder2 are located and issue the command:

   ```
   ldif2db -r no -i  <mycredentialsfile>
   ```

   on each machine.
7. At the machine where peer1 is located create a file *<mytopologyfile>* where *<mytopologyfile>* includes:

   Note: Replace all occurrences of *<master-uuid>* in the following files with the value of the **ibm-slapdServerId** attribute from the master server's **cn=Configuration** entry. This value is generated by the server the first time it is started. You can find it either by performing an **ldapsearch** of the **cn=Configuration** entry or using the **grep** command on the **ibmslapd.conf** file, if you have a UNIX-based system. Similarly, all occurrences of the *<peerx-uuid>*, *<forwarderx-uuid>* and the *<replicax-uuid>* (where x represents a number) must be replaced with the value of the **ibm-slapdServerId** attribute from the respective server's **cn=Configuration** entry.

   ```
   dn: cn=replication,cn=localhost
   objectclass: container

   dn: o=ibm,c=us
   o: ibm
   objectclass: top
   objectclass: organization
   objectclass: ibm-replicationContext

   objectclass: ibm-replicaGroup
   ibm-replicaGroup: default

   dn: ibm-replicaGroup=default, o=ibm,c=us
   objectclass: top
   objectclass: ibm-replicaGroup
   ibm-replicaGroup: default

   dn: ibm-replicaServerId=<peer1-uuid>,ibm-replicaGroup=default,o=ibm,c=us
   objectclass: top
   objectclass: ibm-replicaSubentry
   ibm-replicaServerId: <peer1-uuid>
   ibm-replicationServerIsMaster: true
   cn: peer1
   description: peer1 server

   dn: ibm-replicaServerId=<peer2-uuid>,ibm-replicaGroup=default,o=ibm,c=us
   objectclass: top
   objectclass: ibm-replicaSubentry
   ibm-replicaServerId: <peer2-uuid>
   ibm-replicationServerIsMaster: true
   cn: peer2
   description: peer2 server

   dn: ibm-replicaServerId=<forwarder1-uuid>,
       ibm-replicaGroup=default,o=ibm,c=us
   objectclass: top
   objectclass: ibm-replicaSubentry
   ibm-replicaServerId: <forwarder1-uuid>ibm-replicationServerIsMaster: false
   ```

```
cn: forwarder1
description: forwarder server number one

dn: ibm-replicaServerId=<forwarder2-uuid>,
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicaSubentry
ibm-replicaServerId: <forwarder2-uuid>
ibm-replicationServerIsMaster: false
cn: forwarder2
description: forwarder server number two

#peer1 to peer2 agreement
dn: cn=peer2,ibm-replicaServerId=<peer1-uuid>,
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicationAgreement
cn: peer2
ibm-replicaConsumerId: <peer2-uuid>
ibm-replicaUrl: ldap://<peer2hostname:peer2port>
ibm-replicaCredentialsDN: cn=simple,cn=replication,cn=localhost
description: peer2 server

#peer1 to forwarder1 agreement
dn: cn=forwarder1,ibm-replicaServerId=<peer1-uuid>,
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicationAgreement
cn: forwarder1
ibm-replicaConsumerId: <forwarder1-uuid>
ibm-replicaUrl: ldap://<forwarder1hostname:forwarder1port>
ibm-replicaCredentialsDN: cn=simple,cn=replication,cn=localhost
description: forwarder server one

#peer1 to forwarder2 agreement
dn: cn=forwarder2,ibm-replicaServerId=<peer1-uuid>
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicationAgreement
cn: forwarder2
ibm-replicaConsumerId: <forwarder2-uuid>
ibm-replicaUrl: ldap://<forwarder2hostname:forwarder2port>
ibm-replicaCredentialsDN: cn=simple,cn=replication,cn=localhost
description: forwarder server two

#peer2 to peer1 agreement
dn: cn=peer1,ibm-replicaServerId=<peer2-uuid>,
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicationAgreement
cn: forwarder1
ibm-replicaConsumerId: <peer1-uuid>
ibm-replicaUrl: ldap://<peer1hostname:peer1port>
ibm-replicaCredentialsDN: cn=simple,cn=replication,cn=localhost
description: peer server one

#peer2 to forwarder1 agreement
dn: cn=forwarder1,ibm-replicaServerId=<peer2-uuid>
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicationAgreement
cn: forwarder1
ibm-replicaConsumerId: forwarder1-uid
ibm-replicaUrl: ldap://<forwarder1hostname:forwarder1port>
ibm-replicaCredentialsDN: cn=simple,cn=replication,cn=localhost
description: forwarder server one
```

```
#peer2 to forwarder2 agreement
dn: cn=forwarder2,ibm-replicaServerId=<peer2-uuid>,
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicationAgreement
cn: forwarder2
ibm-replicaConsumerId: <forwarder2-uuid>
ibm-replicaUrl: ldap://$<forwarder2hostname:forwarder2port>
ibm-replicaCredentialsDN: cn=simple,cn=replication,cn=localhost
description: forwarder server two

#forwarder1 to replica1 agreement
dn: cn=replica1,ibm-replicaServerId=<forwarder1-uuid>,
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicationAgreement
cn: replica1
ibm-replicaConsumerId: <replica1-uuid>
ibm-replicaUrl: ldap://<replica1hostname:replica1port>
ibm-replicaCredentialsDN: cn=simple,cn=replication,cn=localhost
description: replica server number one

#forwarder1 to replica2 agreement
dn: cn=replica2,ibm-replicaServerId=<forwarder1-uuid>,
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicationAgreement
cn: replica2
ibm-replicaConsumerId: <replica2-uuid>
ibm-replicaUrl: ldap://<replica2hostname:replica2port>
ibm-replicaCredentialsDN: cn=simple,cn=replication,cn=localhost
description: replica server number two

#forwarder2 to replica3 agreement
dn: cn=replica3,ibm-replicaServerId=<forwarder2-uuid>,
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicationAgreement
cn: replica3
ibm-replicaConsumerId: <replica3-uuid>
ibm-replicaUrl: ldap://<replica3hostname:replica3port>
ibm-replicaCredentialsDN: cn=simple,cn=replication,cn=localhost
description: replica server number three

#forwarder2 to replica4 agreement
dn: cn=replica4,ibm-replicaServerId=<forwarder2-uuid>,
    ibm-replicaGroup=default,o=ibm,c=us
objectclass: top
objectclass: ibm-replicationAgreement
cn: replica4
ibm-replicaConsumerId: <replica4-uuid>
ibm-replicaUrl: ldap://<replica4hostname:replica4port>
ibm-replicaCredentialsDN: cn=simple,cn=replication,cn=localhost
description: replica server number four
```

8. Issue the command:
   ```
   ldif2db -r no -i  <mytopologyfile>
   ```
9. At this point you might want to load additional data for you subtree.
10. When you have finished loading the data, issue the command:
    ```
    db2ldif  -s"o=ibm,c=us" -o <mymasterfile.ldif>
    ```
    See "db2ldif utility" on page 228 for more information.
11. Copy <*masterfile.ldif*> to the machine where peer2 is located.
12. At the machine where peer2 is located, issue the following command:

```
ldif2db -r no -i <masterfile.ldif>
```

13. Ensure that forwarder1 and forwarder2 are stopped.
14. You must configure forwarder1 and forwarder2 to be forwarding servers. Use an editor to add the following stanza to the **ibmslapd.conf** files on forwarder1 and forwarder2:
    ```
    dn: cn=MasterServer, cn=configuration
    cn: Master Server
    ibm-slapdMasterDN: cn=master
    ibm-slapdMasterPW: master
    ibm-slapdMasterReferral: ldap://peer1hostname:peer1port/
    ```
    Note: This ensures that all updates from the clients are referred to peer1.
15. Copy <*masterfile.ldif*> to the machines where forwarder1 and forwarder2 are located.
16. At each of these machines, issue the following command:
    ```
    ldif2db -r no -i <masterfile.ldif>
    ```
17. Ensure that replica1, replica2, replica3, and replica4 are stopped.
18. You must configure replica1, replica2, replica3, and replica4 to be replica servers. Use an editor to add the following stanza to the **ibmslapd.conf** files on each of the replicas:
    ```
    dn: cn=MasterServer, cn=configuration
    cn: Master Server
    ibm-slapdMasterDN: cn=master
    ibm-slapdMasterPW: master
    ibm-slapdMasterReferral: ldap://peer1hostname:peer1port/
    ```
19. Save the **ibmslapd.conf** files.
20. Copy <*masterfile.ldif*> to the machines where replica1, replica2, replica3, and replica4 are located.
21. At each of these machines, issue the following command:
    ```
    ldif2db -r no -i <masterfile.ldif>
    ```
22. Start peer1, peer2, forwarder1, forwarder2, replica1, replica2, replica3, and replica4.

Specifying a supplier DN and password for a subtree

You can specify a supplier DN and PW for a particular subtree. To do this the following information is needed on the replica and master.

To create a replica for a subtree, you need to create a replica agreement between the master and the replica, see “Replication agreements” on page 100. This agreement needs to be loaded on both the master and the replica. The relationship between the two servers is that the master is a supplier to the replica and the replica is a consumer of the master.

1. At the machine where the master is located, create a file to contain the agreement information, for example, ***mysupplierinfofile***, where ***mysupplierinfofile*** contains:
   ```
   #Replication data on the master:

   dn: o=IBM,c=US
   objectclass: organization

   dn: ou=Test,o=IBM,c=US
   objectclass: organizationalunit
   objectclass: ibm-replicationContext
   aclentry: access-id:CN=this:object:a:normal:rwsc:sensitive:rwsc:critical:rwsc
   entryowner: access-id:CN=this
   ```

```
dn: ibm-replicaGroup=default, ou=Test,o=IBM,c=US
objectclass: top
objectclass: ibm-replicaGroup
ibm-replicaGroup: default

dn: cn=replica1 BindCredentials, cn=localhost
objectclass: ibm-replicationCredentialsSimple
cn: replica1 BindCredentials
replicaBindDN: cn=s1
replicaCredentials: s1
description: Bindmethod of master to replica1

dn: ibm-replicaServerId=<master-uuid>,ibm-replicaGroup=default,
    ou=Test,o=IBM,c=US
#master uuid is whatever the server ID is set to in your ibmslapd.conf
#on the master.
objectclass: top
objectclass: ibm-replicaSubentry
ibm-replicaServerId: <master-uuid>
ibm-replicationServerIsMaster: true
cn: master
description: master server

dn: cn=replica1,ibm-replicaServerId=<master-uuid>,ibm-replicaGroup=default,
   ou=Test,o=IBM,c=US
objectclass: top
objectclass: ibm-replicationAgreement
cn: replica1
ibm-replicaConsumerId: <replica1-uuid>
#<replica1-uuid> is whatever the server ID is set to in your
#replica ibmslapd.conf file.
ibm-replicaUrl: ldap://<replica1hostname:replica1port>
ibm-replicaCredentialsDN: cn=replica1 BindCredentials, cn=localhost
description: replica server number one
```

2. Stop the master, if it is not already stopped.
3. Issue the command:

   ```
   ldif2db -r no -i <mysupplierinfofile>
   ```
4. Issue the command:

   ```
   db2ldif -o <masterfile.ldif>
   ```

   See "db2ldif utility" on page 228 for more information.
5. Copy *<masterfile.ldif>* to the machine where replica1 is located.
6. Stop the replica if it is running.
7. You must configure replica1 to be a replica server. Use an editor to add the following stanza to the **ibmslapd.conf** file on replica1:

   ```
   dn: cn=MasterServer, cn=configuration
   cn: Master Server
   ibm-slapdMasterDN: cn=master
   ibm-slapdMasterPW: <masterserverpassword>
   ibm-slapdMasterReferral: ldap://<masterhostname:masterport>
   objectclass: ibm-slapdReplication

   dn: cn=Supplier s1, cn=configuration
   cn: Supplier s1
   ibm-slapdMasterDN: cn=s1
   ibm-slapdMasterPW: s1
   ibm-slapdReplicaSubtree: ou=Test, o=IBM, c=US
   objectclass: ibm-slapdSupplier
   ```
8. Save the **ibmslapd.conf** file.

9. Issue the following command:

   ```
   ldif2db -r no -i <masterfile.ldif>
   ```
10. Start master and replica1.

Viewing replication configuration information

A great deal of information related to replication activity is available using searches. To see the replication topology information related to a particular replicated subtree, you can do a one-level search with the base set to the DN of the subtree and the filter set as (**objectclass=ibm-replicaGroup**) to find the subentry that is the base of the topology information. If this replication context was created through the web admin interface, the name of the entry will be **ibm-replicaGroup=default**.

```
ldapsearch -D <adminDN> -w <adminPW> -b <suffixentryDN> (objectclass=*)
```

The objects returned will include the replica group itself, plus the following:

- An object with **objectclass=ibm-replicaSubentry** for each server that replicates data within this context. Replica subentries contain a server id attribute and an indication of the role the server plays (**ibm-replicationServerIsMaster**).
- For each replica subentry, there is a replication agreement object for each consumer server that receives replication updates from the server described by the replica subentry. Each replication agreement contains the following information:
  - **ibm-replicaConsumerId**: The server id of the consumer server.
  - **ibm-replicaURL**: The LDAP URL of the consumer server.
  - **ibm-replicaCredentialsDN**: The DN of the entry containing the credentials used to bind to the consumer.

  Agreements may also contain the following:
  - **ibm-replicaScheduleDN**: The DN of a schedule entry that determines when replication updates are sent to this consumer. If no schedule is specified, replication defaults to ″immediate″ mode.
  - **ibm-replicationOnHold**: A boolean indicating that replication to this consumer is suspended (or not).
  - **ibm-replicationExcludedCapability**: The values of this attribute list OIDs of features that the consumer does not support. Operations related to these capabilities are then excluded from the updates sent to this consumer.

Monitoring Replication Status

In addition, there are many operational attributes that provide replication status information when explicitly requested on a search. One of these attributes is associated with the entry that is the base of the replicated subtree, that is, the entry that the **ibm-replicationContext** objectclass was added to. If you do a base search of that entry, and request that the **ibm-replicationIsQuiesced** attribute is returned. This attribute is a boolean that indicates if the subtree has been quiesced. If the subtree is quiesced, no client updates are allowed (only updates from replication suppliers are accepted). There is an extended operation that can be used to quiesce a subtree, see “ldapexop” on page 204.

The remainder of the status-related operational attributes are all associated with a replication agreement object. These attributes are only returned when explicitly requested on the search. The attributes available are:

- **ibm-replicationLastActivationTime**: The time that the last replication session started between this supplier and consumer.

- **ibm-replicationLastFinishTime**: The time that the last replication session finished between this supplier and consumer.
- **ibm-replicationLastChangeId**: The change ID of the last update sent to this consumer.
- **ibm-replicationLastGlobalChangeId**: The change ID of the last update to a global entry sent to this consumer. Global entries are things like cn=schema or cn=pwdpolicy that apply to the entire contents of a DIT.
- **ibm-replicationState**: The current state of replication with this consumer. Possible values are:

  Active Actively sending updates to consumer (possibly retrying due to errors).

  Ready In immediate replication mode, ready to send updates as they occur.

  Waiting
  Waiting for next scheduled replication time.

  Binding
  In the process of binding to the consumer.

  Connecting
  In the process of connecting to the consumer.

  OnHold
  This replication agreement has been suspended or ″held″.
- ibm-replicationLastResult The results of the last attempted update to this consumer, in the form:

  <time stamp> <change id> <result code> <operation> <entry DN>
- **ibm-replicationLastResultAdditional**: Any additional error information returned from the consumer for the last update.
- **ibm-replicationPendingChangeCount**: The number of updates queued to be replicated to this consumer.
- **ibm-replicationPendingChanges**: Each value of this attribute gives information about one of the pending changes in the form:

  <change id> <operation> <entry DN>

  Requesting this attribute might return many values. Check the change count before requesting this attribute.
- **ibm-replicationChangeLDIF**: Gives the full details of the last failing update in LDIF.

Chapter 10. Logging Utilities

The IBM Directory Server Version 5.1 provides several logging utilities that can be viewed either through the Web Administration Tool or the system command line.

Notes:

1. In the Web Administration Tool the **Logfiles** field in the task title bar accesses the Web Administration console log files. The IBM Directory Server log files are accessible by using the procedures specified in the following sections.
2. On Windows-based systems, if a path begins with the drive letter and a colon, it is assumed to be the full path. A path without the drive letter, starts in the installation tree. As examples: `c:\tmp\mylog` is a full path, while `\tmp\mylog` is interpreted as `c:\program files\ibm\ldap\tmp\mylog`.

Modifying error logging

The error log, **ibmslapd.log**, is enabled by default, to modify the error log settings:

1. Expand Server administration in the navigation area, click **Logs**, click **Modify error log settings**.
2. Enter the path and file name for the error log. Ensure that the path is valid. If the file does not exist, it is created. The error log can also be directed to something other than a file, for example, a line printer.

   Note: If you specify a file that is not an acceptable file name (for example, invalid syntax or if the server does not have the rights to create and/or modify the file), the attempt fails with the following error: `LDAP Server is unwilling to perform the operation.`
3. Select either Low, Medium, or High for the level of error logging.
   - Low logs the least amount of error information, for example,

     ```
     Mar 29 11:03:23 2002  IBM Directory, Version 5.1
     slapd started.
     ```
   - Medium logs a medium amount of error information, for example,

     ```
     Mar 29 11:07:51 2002  Configuration read securePort 636.
     Mar 29 11:07:51 2002  Plugin of type PREOPERATION is successfully
                           loaded from libDSP.dll.
     Mar 29 11:07:51 2002  Plugin of type DATABASE is successfully loaded from
                           C:\Program Files\IBM\LDAP/bin/libback-rdbm.dll.
     Mar 29 11:08:11 2002  Non-SSL port initialized to 389.
     Mar 29 11:08:12 2002  IBM Directory, Version 5.1
     slapd started.
     ```
   - High logs the most amount of error information, for example

     ```
     Mar 29 11:04:05 2002  Configuration read securePort 636.
     Mar 29 11:04:05 2002  Configuration read cipher specifications
                           mask to be 12288.
     Mar 29 11:04:05 2002  Plugin of type PREOPERATION is successfully
                           loaded from libDSP.dll.
     Mar 29 11:04:05 2002  Plugin of type DATABASE is successfully loaded from
                           C:\Program Files\IBM\LDAP/bin/libback-rdbm.dll
     Mar 29 11:04:24 2002  Configuration file successfully read.
     Mar 29 11:04:24 2002  Non-SSL port initialized to 389.
     Mar 29 11:04:25 2002  IBM Directory, Version 5.1
     slapd started.
     ```

4. Click **OK** to apply your changes or click **Cancel** to return to the IBM Directory Server Web Administration **Welcome** panel without making any changes.
5. Click **OK** to return to the IBM Directory Server Web Administration **Welcome** panel.

Using the command line:

Issue the command:

```
ldapmodify -D <AdminDN <-w >AdminPW> -i <filename>
```

where *<filename>* contains:

```
dn: cn=Configuration
changetype: modify
replace: ibm-slapdErrorLog
ibm-slapdErrorLog: <newpathname>
-
replace: ibm-slapdSysLogLevel
ibm-slapdSysLogLevel: {l | m | h}
```

To update the settings dynamically issue the following **ldapexop** command:

```
ldapexop -D cn=root -w root -op readconfig -scope entire
```

The **ldapexop** command only updates those attributes that are dynamic. For other changes to take effect you must stop and restart the server. See "Dynamically-changed attributes" on page 278 for a list of the attributes that can be updated dynamically.

Viewing the error log

Use the following procedures to view the error log.

Using Web Administration:

1. Expand **Logs** in the navigation area, then click **View error log**.
2. You can:
   - The panel displays the first page of the error log and the navigation arrows at the bottom of the panel allow you to go to the **Next** page or to the **Previous** page.
   - From the drop-down menu, you can select a specific page, for example Page 6 of 16, and click **Go** to display that page of the error log.
   - Click **Clear log** to delete all entries in the error log.

Using the command line:

To view the error log issue the following command:

```
more /var/ldap/ibmslapd.log
```

where `var/ldap/ibmslapd.log` is your error log.

Note: `var/ldap/ibmslapd.log` is the default error log for UNIX systems and *`installpath`*`\var\ibmslapd.log` is the default error log for Windows systems.

To view and clear the error log dynamically:

```
ldapexop -D cn=root -w root -op readlog -log  slapd  -lines all
ldapexop -D cn=root -w root -op clearlog -log  slapd
```

Note: Only the administrator is allowed to access the log files.

Audit Logging

Audit logging is used to improve the security of the directory server. A default audit plug-in is provided with the server. Depending on the audit configuration parameters, this plug-in might log an audit entry in the default or specified audit log for each LDAP operation the server processed. The system administrator can use the activities stored in the audit log to check for suspicious patterns of activity in an attempt to detect security violations. If security is violated, the audit log can be used to determine how and when the problem occurred and perhaps the amount of damage done. This information is very useful, both for recovery from the violation and, possibly, in the development of better security measures to prevent future problems. You can also write your own audit plug-ins to either replace, or add more processing to, the default audit plug-in.

By default the audit log is disabled.

Enabling the audit log and modifying audit log settings

To enable audit logging:

1. Expand **Logs** in the navigation area, click **Modify audit log settings**.
2. Select **Enable audit logging** to use the audit log utility.
3. Select to either log **Only failed attempts** of the selected operations or to log **All attempts** of the selected operations.
4. Enter the **Path and file name** for the audit log. The audit log can also be directed to something other than a file, for example, a line printer.
5. Select the operations you wish to log. Consult the field help for additional information about the various operations you can log.
   - **Bind** - records connections to the server
   - **Unbind** - records disconnections from the server
   - **Search** - records LDAP search operations performed by any client
   - **Add** - records additions to LDAP
   - **Modify** - records modifications to LDAP
   - **Delete** - records deletions from LDAP
   - **Modify RDN** - records modifications made to RDN’s
   - **Event notification** - records event notifications
6. Click **OK** to apply your changes or click **Cancel** to return to the IBM Directory Server Web Administration **Welcome** panel without making any changes.
7. If you click **OK**, a message is displayed to remind you that you need to restart the server.

   Note: If you have enabled audit logging, you must restart the server for the changes to take effect. If you were modifying only the settings, the server does not need to be restarted.

   Click **OK** to return to the IBM Directory Server Web Administration **Welcome** panel.

Using the command line:

Issue the command:

```
ldapmodify -D <AdminDN> -w <AdminPW> -i <filename>
```

where <*filename*> contains:

```
dn: cn=audit, cn=localhost
changetype: modify
replace: ibm-audit
ibm-audit: true

-
replace: ibm-auditadd
ibm-auditadd: TRUE|FALSE
#select TRUE to enable, FALSE to disable
-
replace: ibm-auditbind
ibm-auditbind: TRUE|FALSE
#select TRUE to enable, FALSE to disable
-
replace: ibm-auditdelete
ibm-auditdelete: TRUE|FALSE
-
replace: ibm-auditextopevent
ibm-auditextopevent: TRUE|FALSE
#select TRUE to enable, FALSE to disable
-
replace: ibm-auditfailedoponly
ibm-auditfailedoponly: TRUE|FALSE
#select TRUE to enable, FALSE to disable
-
replace: ibm-auditlog
ibm-auditlog: <newpathname>
-
replace: ibm-auditmodify
ibm-auditmodify: TRUE|FALSE
#select TRUE to enable, FALSE to disable
-
replace: ibm-auditmodifydn
ibm-auditmodifydn: TRUE|FALSE
#select TRUE to enable, FALSE to disable
-
replace: ibm-auditsearch
ibm-auditsearch: TRUE|FALSE
#select TRUE to enable, FALSE to disable
-
replace: ibm-auditunbind
ibm-auditunbind: TRUE|FALSE
#select TRUE to enable, FALSE to disable
```

Note: If you are using audit logging in **Configuration only mode**, the DN specified is `dn: cn=audit, cn=configuration`. Any changes made to this DN are overwritten with the `dn: cn=audit, cn=localhost` values when the server is started in normal mode.

Disabling the audit log

To disable audit logging:

1. Expand **Logs** in the navigation area, click **Modify audit log settings**.
2. Deselect **Enable audit logging**.
3. Click **OK** to apply your changes or click **Cancel** to return to the IBM Directory Server Web Administration **Welcome** panel without making any changes.

Using the command line:

Issue the command:

```
ldapmodify -D <AdminDN> -w <AdminPW> -i <filename>
```

where *<filename>* contains:

```
dn: cn=audit, cn=localhost
changetype: modify
replace: ibm-audit
ibm-audit: flase
```

Note: If you are using audit logging in **Configuration only mode**, the DN specified is `dn: cn=audit, cn=configuration`. Any changes made to this DN are overwritten with the `dn: cn=audit, cn=localhost` values when the server is started in normal mode.

Viewing the audit log

The audit log displays log entries chronologically. Each non-message entry contains a general information header followed by operation-specific data. For example,

```
2000-03-23-16:01:01.345-06:00--V3 Bind--bindDN:Y249bWFuYWdlcg0K
   --client:9.1.2.3:12345--
ConnectionID:12--received:2000-03-23-16:01:01.330-06:00
   --success name: Y249bWFuYWdlcg0K
authenticationChoice: simple
```

The header is in the following format:

Timestamp 1 ″--″
The local time the entry is logged, that is, the time the request was processed. The timestamp is in the format YYYY-MM-DD-HH:MM:SS.mmm=(or-)HH:MM. The =(or=)HH:MM is UTC offset. mmm is milliseconds.

Version number+[SSL]+[unauthenticated or anonymous] Operation ″--″
Shows the LDAP request that was received and processed. Version number is either V2 or V3. SSL displays only when SSL was used for the connection. unauthenticated or anonymous displays to indicate whether the request was from an unauthenticated or anonymous client. Neither unauthenticated or anonymous display if the request is from an authenticated client.

bindDN: server encoded DN string ″--″
Shows the Bind DN. The encoding done by the server is to hide the DN from trivial viewing. The server provides a decoding function through an LDAP search of special audit related attributes. For V3 unauthenticated or anonymous requests, this field is not shown.

client:Client IP address:Port number ″--″
Shows the client IP address and port number.

ConnectionID: xxxx ″--″
Is used to group all the entries received in the same connection, meaning between the bind and unbind, together.

received: Timestamp 2 ″--″
Is the local time when the request was received, or to be more specific, the beginning time when the request was processed. Its format is the same as Timestamp 1.

Result or Status string
Shows the result or status of the LDAP operation. For the result string, the textual form of the LDAP resultCode is logged, for example, success or operationsError, instead of 0 or 1.

Operation-specific data follows the header and displays operation-specific data, for example,

- Bind operations

  ```
  name: Y249bWFuYWdlcg0K
  authenticationChoice: simple
  ```
- Add operations

  ```
  entry: cn=Jim Brown, ou=sales,o=ibm_us,c=us
  attributes: objectclass, cn, sn, telphonenumber
  ```
- Delete operations

  ```
  entry:  cn=Jim Brown, ou=sales,o=ibm_us,c=us
  ```
- Modify operations

  ```
  object: cn=Jim Brown, ou=sales,o=ibm_us,c=us
  add: mail
  delete: telephonenumber
  ```

Use the following procedures to view the audit log:

Using Web Administration:

To view the audit log :

1. Expand **Logs** in the navigation area, click **View audit log**.
2. You can:
   - The panel displays the first page of the audit log and the navigation arrows at the bottom of the panel allow you to go to the **Next** page or to the **Previous** page.
   - From the drop-down menu, you can select a specific page, for example Page 6 of 16, and click **Go** to display that page of the audit log.
   - Click **Clear log** to delete all entries in the audit log.

Using the command line:

To view the error log issue the following command:

```
more /var/ldap/audit.log
```

where `/var/ldap/audit.log` is your error log.

Note: `/var/ldap/audit.log` is the default audit log for UNIX systems and *`installpath`*`\var\audit.log` is the default audit log for Windows systems.

To view and clear the audit log dynamically:

```
ldapexop -D cn=root -w root -op readlog -log audit -lines all
ldapexop -D cn=root -w root -op clearlog -log audit
```

Note: Only the administrator is allowed to access the log files.

DB2 error logging

Modifying DB2 error log settings

1. Expand **Logs** in the navigation area, click **Modify DB2 log settings**.
2. Enter the path and file name for the error log. Typically this is the db2cli.log file located in the **/var/ldap** directory. Ensure that the path is valid. If the file does not exist, it is created.

   Note: `var/ldap/db2cli.log` is the default DB2 error log for UNIX systems and *`installpath`*`\var\db2cli.log` is the default DB2 error log for Windows systems.

3. Click **OK** to apply your changes or click **Cancel** to return to the IBM Directory Server Web Administration **Welcome** panel without making any changes.
4. Click **OK** to return to the IBM Directory Server Web Administration **Welcome** panel.

Using the command line:

Issue the command:

```
ldapmodify -D <AdminDN> -w <AdminPW> -i <filename>
```

where *<filename>* contains:

```
dn: cn=Directory, cn=RDBM Backends, cn=IBM Directory, cn=Schemas, cn=Configuration
changetype: modify
replace: ibm-slapdCLIErrors
ibm-slapdCLIErrors: <newpathname>
```

To update the settings dynamically issue the following **ldapexop** command:

```
ldapexop -D cn=root -w root -op readconfig -scope single
      "cn=Directory,cn=RDBM Backends,cn=IBM Directory,cn=Schemas,cn=Configuration"
      ibm-slapdCLIErrors
```

The **ldapexop** command only updates those attributes that are dynamic. For other changes to take effect you must stop and restart the server. See "Dynamically-changed attributes" on page 278 for a list of the attributes that can be updated dynamically.

Viewing the DB2 error log

Use the following procedures to view the DB2 error log.

Using Web Administration:

1. Expand **Logs** in the navigation area, then click **View DB2 log**.
2. You can:
   - The panel displays the first page of the DB2 error log and the navigation arrows at the bottom of the panel allow you to go to the **Next** page or to the **Previous** page.
   - From the drop-down menu, you can select a specific page, for example Page 6 of 16, and click **Go** to display that page of the DB2 error log.
   - Click **Clear log** to delete all entries in the DB2 error log.

Using the command line:

To view the DB2 error log issue the following command:

```
more /var/ldap/db2cli.log
```

where `var/ldap/db2cli.log` is your DB2 error log.

Note: `var/ldap/db2cli.log` is the default DB2 error log for UNIX systems and `*installpath*\var\db2cli.log` is the default DB2 error log for Windows systems.

To view and clear the DB2 error log dynamically:

```
ldapexop -D cn=root -w root -op readlog -log  cli  -lines all
ldapexop -D cn=root -w root -op clearlog -log  cli
```

Note: Only the administrator is allowed to access the log files.

bulkload error logging

Modifying bulkload error log settings

1. Expand **Logs** in the navigation area, click **Modify bulkload log settings**.
2. Enter the path and file name for the error log. Typically this is the bulkload.log file located in the **var/ldap** directory. Ensure that the file exists on the ldap server and that the path is valid.

   Note: `var/ldap/bulkload.log` is the default bulkload error log for UNIX systems and *`installpath`*`\var\bulkload.log` is the default bulkload error log for Windows systems.
3. Click **OK** to apply your changes or click **Cancel** to return to the IBM Directory Server Web Administration **Welcome** panel without making any changes.
4. If you click **OK**, a message is displayed to remind you that you need to restart the server. Click **OK** to return to the IBM Directory Server Web Administration **Welcome** panel.

Using the command line:

Issue the command:

```
ldapmodify -D <AdminDN> -w <AdminPW> -i <filename>
```

where <*filename*> contains:

```
dn: cn=Directory, cn=RDBM Backends, cn=IBM Directory, cn=Schemas, cn=Configuration
changetype: modify
replace: ibm-slapdBulkloadErrors
ibm-slapdBulkloadErrors: <newpathname>
```

To update the settings dynamically issue the following **ldapexop** command:

```
ldapexop -D cn=root -w root -op readconfig -scope single
      "cn=Directory,cn=RDBM Backends,cn=IBM Directory,cn=Schemas,cn=Configuration"
      ibm-slapdBulkloadErrors
```

The **ldapexop** command only updates those attributes that are dynamic. For other changes to take effect you must stop and restart the server. See "Dynamically-changed attributes" on page 278 for a list of the attributes that can be updated dynamically.

Viewing the bulkload error log

Use the following procedures to view the bulkload error log.

Using Web Administration:

1. Expand **Logs** in the navigation area, then click **View bulkload log**.
2. You can:
   - The panel displays the first page of the bulkload error log and the navigation arrows at the bottom of the panel allow you to go to the **Next** page or to the **Previous** page.
   - From the drop-down menu, you can select a specific page, for example Page 6 of 16, and click **Go** to display that page of the bulkload error log.
   - Click **Clear log** to delete all entries in the bulkload error log.

Using the command line:

To view the bulkload error log issue the following command:

```
more /var/ldap/bulkload.log
```

where `var/ldap/bulkload.log` is your bulkload error log.

Note: `var/ldap/bulkload.log` is the default error log for UNIX systems and `*installpath*\var\bulkload.log` is the default bulkload error log for Windows systems.

To view and clear the bulkload error log dynamically:

```
ldapexop -D cn=root -w root -op readlog -log bulkload -lines all
ldapexop -D cn=root -w root -op clearlog -log bulkload
```

Note: Only the administrator is allowed to access the log files.

Administration daemon error logging

Modifying administration daemon error log settings

1. Expand **Logs** in the navigation area, click **Modify admin daemon log settings**.
2. Enter the path and file name for the Web Administration error log. Typically this is the ibmdiradm.log file located in the **var/ldap/** directory. Ensure that the file exists on the ldap server and that the path is valid.

   Note: `var/ldap/ibmdiradm.log` is the default Web Administration error log for UNIX systems and `*installpath*\var\ibmdiradm.log` is the default Web Administration error log for Windows systems.
3. Click **OK** to apply your changes or click **Cancel** to return to the IBM Directory Server Web Administration **Welcome** panel without making any changes.
4. If you click **OK**, a message is displayed to remind you that you need to restart the server. Click **OK** to return to the IBM Directory Server Web Administration **Welcome** panel.
5. You must restart the administration daemon for the changes to take effect.

Using the command line:

Issue the command:

```
ldapmodify -D <AdminDN <-w >AdminPW> -i <filename>
```

where *<filename>* contains:

```
dn: cn=Admin, cn=Configuration
changetype: modify
replace: ibm-slapdErrorLog
ibm-slapdErrorLog: <newpathname>
```

Viewing the administration daemon error log

Use the following procedures to view the administration daemon error log.

Using Web Administration:

1. Expand **Logs** in the navigation area, then click **View admin deamon log**.
2. You can:
   - The panel displays the first page of the Web Administration error log and the navigation arrows at the bottom of the panel allow you to go to the **Next** page or to the **Previous** page.
   - From the drop-down menu, you can select a specific page, for example Page 6 of 16, and click **Go** to display that page of the Web Administration error log.
   - Click **Clear log** to delete all entries in the administration daemon error log.

Using the command line:

To view the administration daemon error log issue the following command:

```
more /var/ldap/ibmdiradm.log
```

where `var/ldap/ibmdiradm.log` is your Web Administration error log.

Note: `var/ldap/ibmdiradm.log` is the default Web Administration error log for UNIX systems and `*installpath*\var\ibmdiradm.log` is the default Web Administration error log for Windows systems.

To view and clear the Web Administration error log dynamically:

```
ldapexop -D cn=root -w root -op readlog -log  ibmdiradm  -lines all
ldapexop -D cn=root -w root -op clearlog -log  ibmdiradm
```

Note: Only the administrator is allowed to access the log files.

Part 3. Directory Management

Chapter 11. Working with directory entries

Expand the **Directory management** category in the navigation area of the Web Administration Tool. All the directory entry tasks that you want to perform can be accessed by selecting **Manage entries**. Two short cuts have been added to the navigation area for the specific tasks of adding an entry and finding (searching for) entries

You can perform the following operations with directory entries:
- Browse the directory tree
- Add or create an entry
- Add or delete an auxiliary object class to an entry
- Edit the attributes of an entry
- Copy an entry
- Edit ACLs
- Search for entries

Browsing the tree

If you have not done so already, expand the **Directory management** category in the navigation area, then click **Manage entries**. You can expand the various subtrees and select the entry that you want to work on. You can choose the operation you want to perform from the right-side tool bar.

Adding an entry

If you have not done so already, expand the **Directory management** category in the navigation area.

1. Click **Add an entry**.
2. Select one **Structural object class** from the drop-down list .
3. Click **Next**.
4. Select any **Auxiliary object classes** you wish to use from the Available box and click **Add**. Repeat this process for each auxiliary object class you want to add. You can also delete an auxiliary object class from the Selected box by selecting it and clicking **Remove**.
5. Click **Next**.
6. In the **Relative DN** field, enter the relative distinguished name (RDN) of the entry that you are adding, for example, cn=John Doe.
7. In the **Parent DN** field, enter the distinguished name of the tree entry you selected, for example, ou=Austin, o=IBM. You can also click **Browse** to select the Parent DN from the list. You can also expand the selection to view other choices lower in the subtree. Specify the your choice and click **Select** to specify the Parent DN that you want. The **Parent DN** defaults to the entry selected in the tree.

   Note: If you started this task from the **Manage entries** panel, this field is prefilled for you. You selected the **Parent DN** before clicking **Add** to start the add entry process.

8. At the **Required attributes** tab enter the values for the required attributes. If you want to add more than one value for a particular attribute, click **Multiple values** and then add the values one at a time.
9. Click **Optional attributes**.
10. At the **Optional attributes** tab enter the values as appropriate for the optional attributes. See "Binary attributes" on page 143 for information on adding binary values. If you want to add more than one value for a particular attribute, click **Multiple values** and then add the values one at a time.
11. Click OK to create the entry.
12. Click the **ACL** button to modify the access control list for this entry. See "Working with Access Control Lists" on page 161 for information on ACLs.
13. After completing at least the required fields, click **Add** to add the new entry or click **Cancel** to return to **Browse tree** without making changes to the directory.

Deleting an entry

If you have not done so already, expand the **Directory management** category in the navigation area, then click **Manage entries**. You can expand the various subtrees and select the subtree, the suffix, or the entry that you want to work on. Click **Delete** from the right-side tool bar.

- You are requested to confirm the deletion. Click **OK**.
- The entry is deleted from the entry and you are returned to the list of entries.

Editing an entry

If you have not done so already, expand the **Directory management** category in the navigation area, then click **Manage entries**. You can expand the various subtrees and select the entry that you want to work on. Click **Edit attributes** from the right-side tool bar.

1. At the **Required attributes** tab enter the values for the required attributes. See "Binary attributes" on page 143 for information on adding binary values. If you want to add more than one value for a particular attribute, click **Multiple values** and then add the values one at a time.
2. Click **Optional attributes**.
3. At the **Optional attributes** tab enter the values as appropriate for the optional attributes. If you want to add more than one value for a particular attribute, click **Multiple values** and then add the values one at a time.
4. Click **Memberships**.
5. If you have created any groups, at the **Memberships** tab:
   - Select a group from **Available groups** and click **Add** to make the entry a member of the selected **Static group membership**.
   - Select a group from **Static group memberships** and click **Remove** to remove the entry from the selected group.
6. If the entry is a group entry, a **Members** tab is available. The **Members** tab displays the members of the selected group. You can add and remove members from the group.
   - To add a member to the group:
     a. Either click **Multiple values** by the **Members** tab or at the **Members** tab, click **Members**.
     b. In the Member field, enter the DN of the entry you want to add.

   c. Click **Add**.
   d. Click **OK**.
- To remove a member from the group:
   a. Either click **Multiple values** by the **Members** tab or click the **Members** tab and click **Members**.
   b. Select the entry you want to remove.
   c. Click **Remove**.
   d. Click **OK**.
- To refresh the members list, click the **Update**.

7. Click OK to modify the entry.

Binary attributes

If an attribute requires binary data, a **Binary data** button is displayed next to the attribute field. If the attribute has no data the field is blank. Because binary attributes cannot be displayed, if an attribute contains binary data, the field displays **Binary Data - 1**. If the attribute contains multiple values, the field displays as a drop-down list.

Click the **Binary data** button to work with binary attributes.

You can import, export or delete binary data.

To add binary data to the attribute:
1. Click the **Binary data** button.
2. Click **Import**.
3. You can either enter the path name for the file you want or click **Browse** to locate and select the binary file.
4. Click **Submit file**. A `File uploaded` message is displayed.
5. Click **Close**. **Binary Data - 1** is now displayed under **Binary data entries**.
6. Repeat the import process for as many binary files as you want to add. The subsequent entries are listed as **Binary Data - 2**, **Binary Data -3**, and so on.
7. When you are finished adding binary data, click **OK**.

To export binary data:
1. Click the **Binary data** button.
2. Click **Export**.
3. Click on the link **Binary data to download**.
4. Follow the directions of your wizard to either display the binary file or to save it to a new location.
5. Click **Close**.
6. Repeat the import process for as many binary files as you want to export.
7. When you are finished exporting data, click **OK**.

To delete binary data:
1. Click the **Binary data** button.
2. Check the binary data file you want to delete. Multiple files can be selected.
3. Click **Delete**.

4. When you are prompted to confirm the deletion, click **OK**. The binary data marked for deletion are removed from the list.
5. When you are finished deleting data, click **OK**.

Note: Binary attributes are not searchable.

Copying an entry

This function is useful if you are creating similar entries. The copy inherits all the attributes of the original. You need to make some modifications to name the new entry.

If you have not done so already, expand the **Directory management** category in the navigation area, then click **Manage entries**. You can expand the various subtrees and select the entry, such as John Doe, that you want to work on. Click **Copy** from the right-side tool bar.
- Change the RDN entry in the DN field. For example change cn=John Doe to cn=Jim Smith.
- On the required attributes tab, change the cn entry to the new RDN. In this example Jim Smith.
- Change the other required attributes as appropriate. In this example change the sn attribute from Doe to Smith.
- When you have finished making the necessary changes click **OK** to create the new entry.
- The new entry Jim Smith is added to the bottom of the entry list.

Note: This procedure copies only the attributes of the entry. The group memberships of the original entry are not copied to the new entry. Use the Edit attributes function to add memberships.

Editing access control lists

To view ACL properties using the Web Administration Tool utility and to work with ACLs, see “Working with Access Control Lists” on page 161.

See Chapter 12, “Access Control Lists” on page 149 for additional information.

Adding an auxiliary object class

Use the **Add auxiliary class** button on the toolbar to add an auxiliary object class to an existing entry in the directory tree. An auxiliary object class provides additional attributes to the entry to which it is added.

If you have not done so already, expand the **Directory management** category in the navigation area, then click **Manage entries**. You can expand the various subtrees and select the entry, such as John Doe, that you want to work on. Click **Add auxiliary class** from the right-side tool bar.
1. Select any **Auxiliary object classes** you wish to use from the Available box and click **Add**. Repeat this process for each auxiliary object class you want to add. You can also delete an auxiliary object class from the Selected box by selecting it and clicking **Remove**.
2. At the **Required attributes** tab enter the values for the required attributes. If you want to add more than one value for a particular attribute, click **Multiple values** and then add the values one at a time.

3. Click **Optional attributes**.
4. At the **Optional attributes** tab enter the values as appropriate for the optional attributes. If you want to add more than one value for a particular attribute, click **Multiple values** and then add the values one at a time.
5. Click **Memberships**.
6. If you have created any groups, at the **Memberships** tab:
   - Select a group from **Available groups** and click **Add** to make the entry a member of the selected **Static group membership**.
   - Select a group from **Static group memberships** and click **Remove** to remove the entry from the selected group.
7. Click **OK** to modify the entry.

Deleting an auxiliary class

Although you can delete an auxiliary class during the add auxiliary class procedure, it is easier to use the delete auxiliary class function if you are going to delete a single auxiliary class from an entry. However, it might be more convenient to use the add auxiliary class procedure if you are going to delete multiple auxiliary classes from and entry.

1. If you have not done so already, expand the **Directory management** category in the navigation area, then click **Manage entries**. You can expand the various subtrees and select the entry, such as John Doe, that you want to work on. Click **Delete auxiliary class** from the right-side tool bar.
2. From the list of auxiliary classes select the one you want to delete and press **OK**.
3. You are asked to confirm the deletion, click **OK**.
4. The auxiliary class is deleted from the entry and you are returned to the list of entries.

Repeat these steps for each auxiliary class that you want to delete.

Changing group membership

If you have not done so already, expand the **Directory management** category in the navigation area.

1. Click **Manage entries**.
2. Select a user from the directory tree and click the **Edit attributes** icon on the toolbar. tree.
3. Click the **Memberships** tab.
4. To modify the memberships for the user. The **Change memberships** panel displays the **Available groups** to which the user can be added, as well as the entry's **Static Group Memberships**.
   - Select a group from **Available groups** and click **Add** to make the entry a member of the selected group.
   - Select a group from **Static Group Memberships** and click **Remove** to remove the entry from the selected group.
5. Click **OK** to save your changes or click **Cancel** to return to the previous panel without saving your changes.

Searching the directory entries

There are three options for searching the directory tree:

- A Simple search using a predefined set of search criteria
- An Advanced search using a user-defined set of search criteria
- A Manual search

The search options are accessible by expanding the **Directory management** category in the navigation area, click **Find entries**. Select one of the following tabs:

Note: Binary entries, for example passwords, are not searchable.

Search filters

Select on of the following types of searches:

Simple search

A simple search uses a default search criteria:

- Base DN is **All suffixes**
- Search scope is **Subtree**
- Search size is **Unlimited**
- Time limit is **Unlimited**
- Alias dereferencing is **never**
- Chase referrals is deselected (off)

To perform a simple search:

1. On the **Search filter** tab, click **Simple search**.
2. Select an object classes from the drop-down list.
3. Select a specific attribute for the selected entry type. If you select to search on a specific attribute, select an attribute from the drop-down list and enter the attribute value in the **Is equal to** box. If you do not specify an attribute, the search returns all the directory entries of the selected entry type.

Advanced search

An advanced search enables you to specify search constraints and enable search filters. Use the Simple search to use default search criteria.

- To perform an advanced search:
  1. On the **Search filter** tab, click **Advanced search**.
  2. Select an **Attribute** from the drop-down list.
  3. Select a **Comparison** operator
     – =The attribute is equal to the value.
     – ! The attribute is not equal to the value.
     – < The attribute is less than or equal to the value.
     – > The attribute is greater than or equal to the value.
     – ~ The attribute approximately equal to the value.
  4. Enter the **Value** for comparison.
  5. Use the search operator buttons for complex queries.
     – If you already added at least one search filter, specify the additional criteria and click **AND**. The **AND** command returns entries that match both sets of search criteria.

– If you already added at least one search filter, specify the additional criteria and click **OR**. The **OR** command returns entries that match either set of search criteria.

6.

– Click **Add** to add the search filter criteria to the advanced search

– Click **Delete** to remove the search filter criteria from the advanced search

– Click **Reset** to clear all search filters.

Manual search

Use this method to create a search filter. For example to search on surnames enter `sn=*` in the field. If you are searching on multiple attributes, you must use search filter syntax. For example to search for the surnames of a particular department you enter:

```
(&(sn=*)(dept=<departmentname>))
```

Options

At the **Options tab**:

- **Search base DN** - Select suffix from the drop-down list to search only within that suffix.

  Note: If you started this task from the **Manage entries** panel, this field is prefilled for you. You selected the **Parent DN** before clicking **Add** to start the add entry process.

  You can also select **All suffixes** to search the entire tree.
- **Search scope**
  - Select **Object** to search only within the selected object.
  - Select **Single level** to search only within the immediate children of the selected object.
  - Select **Subtree** to search all descendants of the selected entry.
- **Search size limit** - Enter the maximum number of entries to search or select **Unlimited**.
- **Search time limit** - Enter the maximum number of seconds for the search or select **Unlimited**.
- Select a type of **Alias dereferencing** from the drop-down list.
  - **Never** - If the selected entry is an alias, it is not dereferenced for the search, that is, the search ignores the reference to the alias.
  - **Finding** - If the selected entry is an alias, the search dereferences the alias and search from the location of the alias.
  - **Searching** - The selected entry is not dereferenced, but any entries found in the search are dereferenced.
  - **Always** - All aliases encountered in the search are dereferenced.
- Select the **Chase referrals** check box to follow referrals to another server if a referral is returned in the search. When a referral directs the search to another server, the connection to the server uses the current credentials. If you are logged in as Anonymous you might need to log in to the server using an authenticated DN. See “Logging on to the Web Administration Tool” on page 11 for information about logging in. See “Referrals” on page 38 for more information about referrals.

See “Setting Searches” on page 27 for additional information about searches.

Chapter 12. Access Control Lists

Access Control Lists (ACL) provide a means to protect information stored in a 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 distinguished name of the object as well as a set of attributes and their corresponding values.

The object attributes associated with access control, such as owner, ownerSource, ownerPropagate, aclEntry, aclSource and aclPropagate are unusual in that they are logically associated with each object, but can have values that depend upon other objects higher in the tree. Depending upon how they are established, these attribute values can be explicit to an object or inherited from an ancestor.

The access control model defines two sets of attributes: The Access Control Information (ACI) and the entryOwner Information. The ACI specifically defines a subject's permission to perform a given operation against certain LDAP objects. The entryOwner information controls which subjects can define the ACIs. The entryOwnership also acquires full access rights to the target object.

Using ACL, administrators can restrict access to different portions of the directory, specific directory entries and, based on the attribute name or attribute access class, the attributes contained in the entries. Each entry within the LDAP directory has a set of associated ACI. In conformance with the LDAP model, the ACI and entryOwner information is represented as attribute-value pairs. Furthermore, the LDIF syntax is used to administer these values. The attributes are:
- aclEntry
- aclPropagate
- entryOwner
- ownerPropagate

Filtered ACLs

In the current access control model, ACLs apply explicitly to the directory entry that contains them, but may be propagated to none, or all of its descendant entries. Filter-based ACLs differ in that they employ a filter-based comparison, using a specified object filter, to match target objects with the effective access that applies to them.

Although they perform the same function, the behavior of the two types of ACLs is significantly different. Filter-based ACLs do not propagate in the same way that non-filter-based ACLs currently do. By nature, they inherently propagate to any comparison matched objects in the associated subtree. For this reason, the aclPropagate attribute, which is used to stop propagation of non-filter ACLs, does not apply to the new filter-based ACLs.

The default behavior of filter-based ACLs to accumulate from the lowest containing entry, upward along the ancestor entry chain, to the highest containing entry in the DIT. The effective access is calculated as the union of the access rights granted, or denied, by the constituent ancestor entries. There is an exception to this behavior. For compatibility with the subtree replication feature, and to allow greater administrative control, a ceiling attribute is used as a means to stop accumulation at the entry in which it is contained.

A new set of access control attributes are used specifically for filter-based ACL support, rather than merging filter-based characteristics into the existing non-filter based ACLs. The attributes are:
- ibm-filterAclEntry
- ibm-filterAclInherit

The ibm-filterAclEntry attribute has the same format as aclEntry, with the addition of an object filter component. The associated ceiling attribute is ibm-filterAclInherit. By default it is set to true. When set to false, it terminates the accumulation.

The access control attribute syntax

Each of these attributes can be managed using LDIF notation. The syntax for the new filter-based ACL attributes are modified versions of the current non-filter-based ACL attributes.The following defines the syntax for the ACI and entryOwner attributes using baccus naur form (BNF).

```
<aclEntry> ::=  <subject> [ ":"  <rights> ]

<aclPropagate> ::=  "true" | "false"
<ibm-filterAclEntry> ::=  <subject>  ":" <object filter>  [ ":"  <rights> ]

<ibm-filterAclInherit> ::=  "true" | "false"
<entryOwner> ::=  <subject>

<ownerPropagate> ::= "true" | "false"

<subject> ::= <subjectDnType> ':' <subjectDn> |
                       <pseudoDn>

<subjectDnType> ::= "role" | "group" | "access-id"

<subjectDn> ::= <DN>

<DN> ::= distinguished name as described in RFC 2251, section 4.1.3.

<pseudoDn> ::= "group:cn=anybody" | "group:cn=authenticated" |
                        "access-id:cn=this"
<object filter> ::= string search filter as defined in RFC 2254, section 4
                       (extensible matching is not supported).
<rights> ::= <accessList> [":" <rights> ]

<accessList> ::=  <objectAccess> | <attributeAccess> |
                           <attributeClassAccess>

<objectAccess> ::= "object:" [<action> ":"]  <objectPermissions>

<action> ::= "grant" | "deny"

<objectPermisssions> ::=  <objectPermission> [ <objectPermissions> ]

<objectPermission> ::= "a" | "d" |  ""

<attributeAccess> ::= "at." <attributeName> ":" [<action> ":"]
                              <attributePermissions>

<attributeName> ::= attributeType name as described in RFC 2251, section 4.1.4.
                       (OID or alpha-numeric string with leading
                         alphabet, "-" and ";" allowed)

<attributePermissions> ::=  <attributePermission>
                               [<attributePermissions>]

<attributePermission> ::=  "r" | "w" | "s" | "c" |  ""

<attributeClassAccess> ::= <class> ":" [<action> ":"]
                                <attributePermissions>

<class> ::= "normal" | "sensitive" | "critical"
```

AclEntry and ibm-filterAclEntry

Subject

A subject (the entity requesting access to operate on an object) consists of the combination of a DN (Distinguished Name) type and a DN. The valid DN types are: access Id, Group and Role.

The DN identifies a particular access-id, role or group. For example, a subject might be access-id: cn=personA, o=IBM or group: cn=deptXYZ, o=IBM.

Because the field delimiter is the colon (:), a DN containing colons must be surrounded by double-quotation marks (“”). If a DN already contains characters with double-quotation marks, these characters must be escaped with a backslash (\).

All directory groups can be used in access control.

Note: Any group of **AccessGroup**, **GroupOfNames**, **GroupofUniqueNames**, or **groupOfURLs** structural objectclasses or the **ibm-dynamicGroup**, **ibm-staticGroup** auxiliary objectclasses can be used for access control.

Another DN type used within the access control model is role. While roles and groups are similar in implementation, conceptually they are different. When a user is assigned to a role, there is an implicit expectation that the necessary authority has already been set up to perform the job associated with that role. With group membership, there is no built in assumption about what permissions are gained (or denied) by being a member of that group.

Roles are similar to groups in that they are represented in the directory by an object. Additionally, roles contain a group of DNs. Roles that are used in access control must have an objectclass of **AccessRole**.

Pseudo DN

The LDAP directory contains several pseudo DNs. These are used to refer to large numbers of DNs which at bind time share a common characteristic, in relation to either the operation being performed, or the target object on which the operation is being performed.

Currently, three pseudo DNs are defined:

group:cn=anybody
Refers to all subjects, including those that are unauthenticated. All users belong to this group automatically.

group:cn=authenticated
Refers to any DN which has been authenticated to the directory. The method of authentication is not considered.

access-id:cn=this
Refers to the bind Dn which matches the target object's DN on which the operation is performed.

Object filter

This parameter applies to filtered ACLs only. The string search filter as defined in RFC 2254, is used as the object filter format. Because the target object is already known, the string is not used to perform an actual search. Instead, a filter-based compare on the target object in question is performed to determine if a given set of ibm-filterAclEntry values apply to it.

Rights

Access rights can apply to an entire object or to attributes of the object. The LDAP access rights are discrete. One right does not imply another right. The rights may be combined together to provide the desired rights list following a set of rules discussed later. Rights can be of an unspecified value, which indicates that no access rights are granted to the subject on the target object. The rights consist of three parts:

Action:
Defined values are **grant** or **deny**. If this field is not present, the default is set to **grant**.

Permission:
There are six basic operations that may be performed on a directory object. From these operations, the base set of ACI permissions are taken. These are: add an entry, delete an entry, read an attribute value, write an attribute value, search for an attribute, and compare an attribute value.

The possible attribute permissions are: read (r), write (w), search (s), and compare (c). Additionally, object permissions apply to the entry as a whole. These permissions are add child entries (a) and delete this entry (d).

The following table summarizes the permissions needed to perform each of the LDAP operations.

Table 11.

Operation	Permission Needed
ldapadd	add (on parent)
ldapdelete	delete (on object)
ldapmodify	write (on attributes being modified)
ldapsearch	• search, read (on attributes in RDN) • search (on attributes specified in the search filter) • search (on attributes returned with just names) • search, read (on attributes returned with values)
ldapmodrdn	write (on RDN attributes)
ldapcompare	compare (on compared attribute)

Note: For search operations, the subject is required to have search (s) access to all the attributes in the search filter or no entries are returned. For returned entries from a search, the subject is required to have search (s) and read (r) access to all the attributes in the RDN of the returned entries or these entries are not returned.

Access Target:
These permissions can be applied to the entire object (add child entry, delete entry), to an individual attribute within the entry, or can be applied to groups of attributes (Attribute Access Classes) as described in the following.

Attributes requiring similar permissions for access are grouped together in classes. Attributes are mapped to their attribute classes in the directory schema file. These classes are discrete; access to one class does not imply access to another class. Permissions are set with regard to the attribute access class as a whole. The permissions set on a particular attribute class apply to all attributes within that access class unless the individual attribute access permissions are specified.

IBM defines three attribute classes that are used in evaluation of access to user attributes: **normal**, **sensitive**, and **critical**. For example, attribute **commonName** falls into the normal class, and attribute userpassword belongs to the critical class. User defined attributes belong to the normal access class unless otherwise specified.

Two other access classes are also defined: system and restricted. The system class attributes are:
- **creatorsName**
- **modifiersName**
- **createTimestamp**
- **modifyTimestamp**
- **ownerSource**
- **aclSource**

These are attributes maintained by the LDAP server and are read-only to the directory users. **OwnerSource** and **aclSource** are described in the Propagation section.

The restricted class of attributes that define the access control are:
- **aclEntry**
- **aclPropagate**
- **entryOwner**
- **ownerPropagate**
- **ibm-filterAclEntry**
- **ibm-filterAclInherit**
- **ibm-effectiveAcl**

All users have read access to the restricted attributes but only **entryOwners** can create, modify, and delete these attributes.

Note: The attribute, **ibm-effectiveAcl**, is read-only.

EntryOwner

The entry owners have complete permissions to perform any operation on the object regardless of the aclEntry. Additionally, the entry owners are the only ones who are permitted to administer the aclEntries for that object. EntryOwner is an access control subject, it can be defined as individuals, groups or roles.

Note: The directory administrator is one of the entryOwners for all objects in the directory by default, and the directory administrator's entryOwnership can not be removed from any object.

Propagation

Entries on which an aclEntry has been placed are considered to have an explicit **aclEntry**. Similarly, if the **entryOwner** has been set on a particular entry, that entry has an explicit owner. The two are not intertwined, an entry with an explicit owner may or may not have an explicit **aclEntry**, and an entry with an explicit **aclEntry** might have an explicit owner. If either of these values is not explicitly present on an entry, the missing value is inherited from an ancestor node in the directory tree.

Each explicit **aclEntry** or **entryOwner** applies to the entry on which it is set. Additionally, the value might apply to all descendants that do not have an explicitly set value. These values are considered propagated; their values propagate through the directory tree. Propagation of a particular value continues until another propagating value is reached.

Note: Filter-based ACLs do not propagate in the same way that non-filter-based ACLs do. They propagate to any comparison matched objects in the associated subtree. See "Filtered ACLs" on page 149 for more information on the differences.

AclEntry and **entryOwner** can be set to apply to just a particular entry with the propagation value set to ″false″, or an entry and its subtree with the propagation value set to ″true″. Although both **aclEntry** and **entryOwner** can propagate, their propagation is not linked in anyway.

The **aclEntry** and **entryOwner** attributes allow multi-values, however, the propagation attributes (**aclPropagate** and **ownerPropagate**) can only have a single value for all **aclEntry** or **entryOwner** attribute values within the same entry.

The system attributes **aclSource** and **ownerSource** contain the DN of the effective node from which the **aclEntry** or **entryOwner** are evaluated, respectively. If no such node exists, the value **default** is assigned.

An object's effective access control definitions can be derived by the following logic:

- If there is a set of explicit access control attributes at the object, then that is the object's access control definition.
- If there is no explicitly defined access control attributes, then traverse the directory tree upwards until an ancestor node is reached with a set of propagating access control attributes.
- If no such ancestor node is found, the default access described below is granted to the subject.

Access evaluation

Access for a particular operation is granted or denied based on the subject's bind DN for that operation on the target object. Processing stops as soon as access can be determined.

The checks for access are done by first finding the effective **entryOwnership** and **ACI** definition, checking for entry ownership, and then by evaluating the object's ACI values.

Filter-based ACLs accumulate from the lowest containing entry, upward along the ancestor entry chain, to the highest containing entry in the DIT. The effective access is calculated as the union of the access rights granted, or denied, by the constituent ancestor entries. The existing set of specificity and combinatory rules are used to evaluate effective access for filter based ACLs.

Filter-based and non-filter-based attributes are mutually exclusive within a single containing directory entry. Placing both types of attributes into the same entry is not allowed, and is a constraint violation. Operations associated with the creation of, or updates to, a directory entry fail if this condition is detected.

When calculating effective access, the first ACL type to be detected in the ancestor chain of the target object entry sets the mode of calculation. In filter-based mode, non-filter-based ACLs are ignored in effective access calculation. Likewise, in non-filter-based mode, filter-based ACLs are ignored in effective access calculation.

To limit the accumulation of filter-based ACLs in the calculation of effective access, an **ibm-filterAclInherit** attribute set to a value of ″false″ may be placed in any entry between the highest and lowest occurrence of **ibm-filterAclEntry** in a given subtree. This causes the subset of **ibm-filterAclEntry** attributes above it in the target object's ancestor chain to be ignored.

To exclude the accumulation of filter-based ACLs in the calculation of effective access, an **ibm-filterAclInherit** attribute set to a value of ″false″ may be placed in any entry below the lowest occurrence of **ibm-filterAclEntry** in a given subtree. This causes all **ibm-filterAclEntry** attributes above it in the target object's ancestor chain to be ignored. The resulting access resolves to the default filter ACL value.

By default, the directory administrator and the master server or the peer server (for replication) get full access rights to all objects in the directory except write access to system attributes. Other **entryOwners** get full access rights to the objects under their ownership except write access to system attributes. All users have read access rights to system and restricted attributes. These predefined rights cannot be altered. If the requesting subject has **entryOwnership**, access is determined by the above default settings and access processing stops.

If the requesting subject is not an entryOwner, then the ACI values for the object entries are checked. The access rights as defined in the ACIs for the target object are calculated by the specificity and combinatory rules.

Specificity rule

The most specific aclEntry definitions are the ones used in the evaluation of permissions granted/denied to a user. The levels of specificity are:

- Access-id is more specific than group or role. Groups and roles are on the same level.

- Within the same **dnType** level, individual attribute level permissions are more specific than attribute class level permissions.
- Within the same attribute or attribute class level, **deny** is more specific than **grant**.

Combinatory rule

Permissions granted to subjects of equal specificity are combined. If the access cannot be determined within the same specificity level, the access definitions of lesser specific level are used. If the access is not determined after all defined ACIs are applied, the access is denied.

Note: After a matching access-id level **aclEntry** is found in access evaluation, the group level aclEntries are not included in access calculation. The exception is that if the matching access-id level **aclEntries** are all defined under cn=this, then all matching group level **aclEntries** are also combined in the evaluation.

In other words, within the object entry, if a defined ACI entry contains an access-id subject DN that matches the bind DN, then the permissions are first evaluated based on that aclEntry. Under the same subject DN, if matching attribute level permissions are defined, they supersede any permissions defined under the attribute classes. Under the same attribute or attribute class level definition, if conflicting permissions are present, denied permissions override granted permissions.

Note: A defined null value permission prevents the inclusion of less specific permission definitions.

If access still can not be determined and all found matching aclEntries are defined under ″cn=this″, then group membership is evaluated. If a user belongs to more than one groups, the user receives the combined permissions from these groups. Additionally, the user automatically belongs to the cn=Anybody group and possibly the cn=Authenticated group if the user did an authenticated bind. If permissions are defined for those groups, the user receives the specified permissions.

Note: Group and Role membership is determined at bind time and last until either another bind takes place, or until an unbind request is received. Nested groups and roles, that is a group or role defined as a member of another group or role, are not resolved in membership determination nor in access evaluation.

For example, assume attribute1 is in the sensitive attribute class, and user cn=Person A, o=IBM belongs to both group1 and group2 with the following aclEntries defined:

1. aclEntry: access-id: cn=Person A, o=IBM: at.attributel:grant:rsc:sensitive:deny:rsc
2. aclEntry: group: cn=group1,o=IBM:critical:deny:rwsc
3. aclEntry: group: cn=group2,o=IBM:critical:grant:r:normal:grant:rsc

This user gets:

- Access of ′rsc′ to attribute1, (from 1. Attribute level definition supersedes attribute class level definition).
- No access to other sensitive class attributes in the target object, (from 1).
- No other rights are granted (2 and 3 are NOT included in access evaluation).

For another example, with the following aclEntries:

1. aclEntry: access-id: cn=this: sensitive
2. aclEntry: group: cn=group1,o=IBM:sensitive:grant:rsc:normal:grant:rsc

The user has:

- no access to sensitive class attributes, (from 1. Null value defined under access-id prevents the inclusion of permissions to sensitive class attributes from group1).
- and access of 'rsc' to normal class attributes (from 2).

Defining the ACIs and entry owners

The following two examples show an administrative subdomain being established. The first example shows a single user being assigned as the entryOwner for the entire domain. The second example shows a group assigned as the entryOwner.

```
entryOwner: access-id:cn=Person A,o=IBM
ownerPropagate: true

entryOwner: group:cn=System Owners, o=IBM
ownerPropagate: true
```

The next example shows how an access id "cn=Person 1, o=IBM" is being given permissions to read, search, and compare attribute1. The permission applies to any node in the entire subtree, at or below the node containing this ACI, that matches the "(objectclass=groupOfNames)" comparison filter. The accumulation of matching ibm-filteraclentry attributes in any ancestor nodes has been terminated at this entry by setting the ibm-filterAclInherit attribute to "false".

```
ibm-filterAclEntry: access-id:cn=Person 1,o=IBM:(objectclass=groupOfNames):
                    at.attribute1:grant:rsc

ibm-filterAclInherit: false
```

The next example shows how a group "cn=Dept XYZ, o=IBM" is being given permissions to read, search and compare attribute1. The permission applies to the entire subtree below the node containing this ACI.

```
aclEntry: group:cn=Dept XYZ,o=IBM:at.attribute1:grant:rsc
aclPropagate: true
```

The next example shows how a role "cn=System Admins,o=IBM" is being given permissions to add objects below this node, and read, search and compare attribute2 and the critical attribute class. The permission applies only to the node containing this ACI.

```
aclEntry: role:cn=System Admins,o=IBM:object:grant:a:at.
        attribute2:grant:rsc:critical:grant:rsc
aclPropagate: false
```

Modifying the ACI and entry owner values

Modify-replace

Modify-replace works the same way as all other attributes. If the attribute value does not exist, create the value. If the attribute value exists, replace the value.

Given the following ACIs for an entry:

```
aclEntry: group:cn=Dept ABC,o=IBM:normal:grant:rsc
aclPropagate: true
```

perform the following change:

```
dn: cn=some entry
changetype: modify
replace: aclEntry
aclEntry: group:cn=Dept XYZ,o=IBM:normal:grant:rsc
```

The resulting ACI is:

```
aclEntry: group:cn=Dept XYZ,o=IBM:normal:grant:rsc
aclPropagate: true
```

ACI values for Dept ABC are lost through the replace.

Given the following ACIs for an entry:

```
ibm-filterAclEntry: group:cn=Dept ABC,o=IBM:(cn=Manager ABC):normal
                  :grant:rsc
ibm-filterAclInherit: true
```

perform the following changes:

```
dn: cn=some entry
changetype: modify
replace: ibm-filterAclEntry
ibm-filterAclEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):normal
                  :grant:rsc

dn: cn=some entry
changetype: modify
replace: ibm-filterAclInherit
ibm-filterAclInherit: false
```

The resulting ACI is:

```
ibm-filterAclEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):normal
                   :grant:rsc
ibm-filterAclInherit: false
```

ACI values for Dept ABC are lost through the replace.

Modify-add

During an ldapmodify-add, if the ACI or entryOwner does not exist, the ACI or entryOwner with the specific values is created. If the ACI or entryOwner exists, then add the specified values to the given ACI or entryOwner. For example, given the ACI:

```
 aclEntry: group:cn=Dept XYZ,o=IBM:normal:grant:rsc
```

with a modification:

```
dn: cn=some entry
changetype: modify
add: aclEntry
aclEntry: group:cn=Dept ABC,o=IBM:at.attribute1:grant:rsc
```

would yield an multi-valued aclEntry of:

```
aclEntry: group:cn=Dept XYZ,o=IBM:normal:grant:rsc
aclEntry: group:cn=Dept ABC,o=IBM:at.attribute1:grant:rsc
```

For example, given the ACI:

```
Ibm-filterAclEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):normal
                  :grant:rsc
```

with a modification:

```
dn: cn=some entry
changetype: modify
add: ibm-filterAclEntry
ibm-filterAclEntry: group:cn=Dept ABC,o=IBM:(cn=Manager ABC)
                  :at.attribute1:grant:rsc
```

would yield an multi-valued aclEntry of:

```
Ibm-filterAclEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):normal
                  :grant:rsc
ibm-filterAclEntry: group:cn=Dept ABC,o=IBM:(cn=Manager ABC):at.attribute1
                  :grant:rsc
```

The permissions under the same attribute or attribute class are considered as the basic building blocks and the actions are considered as the qualifiers. If the same permission value is being added more than once, only one value is stored. If the same permission value is being added more than once with different action values, the last action value is used. If the resulting permission field is empty (""), this permission value is set to null and the action value is set to **grant**.

For example, given the following ACI:

```
 aclEntry: group:cn=Dept XYZ,O=IBM:normal:grant:rsc
```

with a modification:

```
dn: cn=some entry
changetype: modify
add: aclEntry
aclEntry: group:cn=Dept XYZ,o=IBM:normal:deny:r:critical:deny::sensitive
        :grant:r
```

yields an aclEntry of:

```
aclEntry: group:cn=Dept XYZ,O=IBM:normal:grant:sc:normal:deny:r:critical
        :grant::sensitive:grant:r
```

For example, given the following ACI:

```
Ibm-filterAclEntry: group:cn=Dept XYZ,O=IBM:(cn=Manager XYZ):normal
                  :grant:rsc
```

with a modification:

```
dn: cn=some entry
changetype: modify
add: ibm-filterAclEntry
ibm-filterAclEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):normal
                  :deny:r:critical:deny::sensitive:grant:r
```

yields an aclEntry of:

```
ibm-filterAclEntry: group:cn=Dept XYZ,O=IBM:(cn=Manager XYZ):normal
                  :grant:sc:normal:deny:r:critical:grant::sensitive
                  :grant:r
```

Modify-delete

To delete a particular ACI value, use the regular ldapmodify-delete syntax.

Given an ACI of:

```
aclEntry: group:cn=Dept XYZ,o=IBM:object:grant:ad
aclEntry: group:cn=Dept XYZ,o=IBM:normal:grant:rwsc

dn: cn = some entry
```

```
changetype: modify
delete: aclEntry
aclEntry: group:cn=Dept XYZ,o=IBM:object:grant:ad
```

yields a remaining ACI on the server of :

```
aclEntry: group:cn=Dept XYZ,o=IBM:normal:grant:rwsc
```

Given an ACI of:

```
ibm-filterAclEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):object
                  :grant:ad
ibm-filterAclEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):normal
                  :grant:rwsc

dn: cn = some entry
changetype: modify
delete: ibm-filterAclEntry
ibm-filterAclEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):object
                  :grant:ad
```

yields a remaining ACI on the server of:

```
ibm-filterAclEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):normal
                  :grant:rwsc
```

Deleting an ACI or entryOwner value that does not exist results in an unchanged ACI or entryOwner and a return code specifying that the attribute value does not exist.

Deleting the ACI/entry owner values

With the ldapmodify-delete operation, the entryOwner can be deleted by specifying

```
dn: cn = some entry
changetype: modify
delete: entryOwner
```

In this case, the entry would then have no explicit entryOwner. The ownerPropagate is also removed automatically. This entry would inherit its entryOwner from the ancestor node in the directory tree following the propagation rule.

The same can be done to delete aclEntry completely:

```
dn: cn = some entry
changetype: modify
delete: aclEntry
```

Deleting the last ACI or entryOwner value from an entry is not the same as deleting the ACI or entryOwner. It is possible for an entry to contain an ACI or entryOwner with no values. In this case, nothing is returned to the client when querying the ACI or entryOwner and the setting propagates to the descendent nodes until it is overridden. To prevent dangling entries that nobody can access, the directory administrator always has full access to an entry even if the entry has a null ACI or entryOwner value.

Retrieving the ACI/entry owner values

The effective ACI or entryOwner values can be retrieved by simply specifying the desired ACL or entryOwner attributes in a search, for example,

```
ldapsearch -b "cn=object A, o=ibm" -s base "objectclass=*"
    aclentry aclpropagate aclsource entryowner ownerpropagate ownersource
    ibm-filterAclEntry ibm-filterAclInherit ibm-effectiveAcl
```

returns all ACL or entryOwner information that is used in access evaluation on object A. Note that the returned values might not look exactly the same as they are first defined. The values are the equivalent of the original form.

Searching on the ibm-filterAclEntry attribute alone only returns the values specific to the containing entry.

A read-only operational attribute, ibm-effectiveAcl, is used to show the accumulated effective access. A search request for ibm-effectiveAcl returns the effective access that applies to the target object based on: non-filter ACLs, or filter ACLs, depending on how they have been distributed in the DIT.

Because filter-based ACLs might come from several ancestor sources, a search on the aclSource attribute produces a list of the associated sources.

Working with Access Control Lists

To view ACL properties using the Web Administration Tool utility and to work with ACLs.

1. Select a directory entry. For example, cn=John Doe,ou=Advertising,o=ibm,c=US.
2. Click **Edit ACL**. The Edit Acl panel is displayed with the **Effective ACLs** tab preselected.

This panel has five tabs:

- Effective ACLs
- Effective owners
- Non-filtered ACLs
- Filtered ACLs
- Owners

The **Effective ACLs** and **Effective owners** tabs contain read-only information about the ACLs.

Effective ACLs

Effective ACLs are the explicit and inherited ACLs of the selected entry. You can view the access rights for a specific effective ACL by selecting it and clicking the **View** button. The **View access rights** panel opens.

Viewing access rights

- The **Rights** section displays the addition and deletion rights of the subject.
  - **Add child** grants or denies the subject the right to add a directory entry beneath the selected entry.
  - **Delete entry** grants or denies the subject the right to delete the selected entry. In the previous example , it grants or denies cn=Marketing Group the ability to delete cn=John Doe.
- The **Security** class section defines permissions for security classes. Attributes are grouped into security classes:
  - **Normal** - Normal attribute classes require the least security, for example, the attribute commonName.

  - **Sensitive** - Sensitive attribute classes require a moderate amount of security, for example homePhone.
  - **Critical** - Critical attribute classes require the most security, for example, the attribute userpassword.

  Each security class has permissions associated with it.
  - **Read** - the subject can read attributes.
  - **Write** - the subject can modify the attributes.
  - **Search** - the subject can search attributes.
  - **Compare** - the subject can compare attributes.

Click **OK** to return to the Effective ACLs tab.

Click **Cancel** to return to the Edit ACL panel.

Effective owners

Effective owners are the explicit and inherited owners of the selected entry.

Non-filtered ACLs

You can add new non-filtered ACLs to an entry, or edit existing non-filtered ACLs.

Non-filtered ACLs can be propagated. This means that access control information defined for one entry can be applied to all of its subordinate entries. The ACL source is the source of current ACL for the selected entry. If the entry does not have an ACL, it inherits an ACL from parent objects based on the ACL settings of the parent objects.

Enter the following information on the **Non-filtered** ACLs tab:
- Propagate ACLs - Select the **Propagate** check box to allow descendants without an explicitly defined ACL to inherit from this entry. If the check box is selected, the descendent inherits ACLs from this entry and if the ACL is explicitly defined for the child entry, then the acl which was inherited from parent is replaced with the new ACL that was added. If the check box is not selected, descendant entries without an explicitly defined ACL will inherit ACLs from a parent of this entry that has this option enabled.
- DN (Distinguished Name) - Enter the **(DN) Distinguished name** of the entity requesting access to perform operations on the selected entry, for example, cn=Marketing Group.
- Type - Enter the **Type** of DN. For example, select access-id if the DN is a user.

Adding and editing access rights

Click the either the **Add** button to add the DN in the DN (Distinguished Name) field to the ACL list or the Edit button to modify the ACLs of an existing DN.

The **Add access rights** and **Edit access rights** panels allow you to set the access rights for a new or existing Access Control List (ACLs). The **Type** field defaults to the type you selected on the **Edit ACL** panel. If you are adding an ACL, all other fields default to blank. If you are editing an ACL, the fields contain the values set last time the ACL was modified.

You can:
- Change the ACL type

- Set addition and deletion rights
- Set permissions for security classes

To set access rights:

1. Select the **Type** of entry for the ACL. For example, select access-id if the DN is a user.
2. The **Rights** section displays the addition and deletion rights of the subject.
   - **Add child** grants or denies the subject the right to add a directory entry beneath the selected entry.
   - **Delete entry** grants or denies the subject the right to delete the selected entry.
3. The **Security class** section defines permissions for attribute classes. Attributes are grouped into security classes:
   - Normal - Normal attribute classes require the least security, for example, the attribute commonName.
   - Sensitive - Sensitive attribute classes require a moderate amount of security, for example homePhone.
   - Critical - Critical attribute classes require the most security, for example, the attribute userpassword.

   Each security class has permissions associated with it.
   - Read - the subject can read attributes.
   - Write - the subject can modify the attributes.
   - Search - the subject can search attributes.
   - Compare - the subject can compare attributes.

   Additionally, you may specify permissions based on the attribute instead of the security class to which the attribute belongs. The attribute section is listed below the **Critical security class**.
   - Select an attribute from the **Define an attribute** drop-down list.
   - Click **Define**. The attribute is displayed with a permissions table.
   - Specify whether to grant or deny each of the four security class permissions associated with the attribute.
   - You can repeat this procedure for multiple attributes.
   - To remove an attribute, simply select the attribute and click **Delete**.
   - When you are finished click **OK**.

Removing ACLs

You can remove ACLs in either of two ways:

- Select the radio button next to the ACL you want to delete. Click **Remove**.
- Click **Remove all** to delete all DNs from the list.

Filtered ACLs

You can add new filtered ACLs to an entry, or edit existing filtered ACLs.

Filter-based ACLs employ a filter-based comparison, using a specified object filter, to match target objects with the effective access that applies to them.

The default behavior of filter-based ACLs to accumulate from the lowest containing entry, upward along the ancestor entry chain, to the highest containing entry in the DIT. The effective access is calculated as the union of the access rights granted, or denied, by the constituent ancestor entries. There is an exception to this behavior. For compatibility with the subtree replication feature, and to allow greater administrative control, a ceiling attribute is used as a means to stop accumulation at the entry in which it is contained.

Enter the following information on the Filtered ACLs tab:

- Accumulate filtered ACLs -
  - Select the **Not specified** radio button to remove the ibm-filterACLInherit attribute from the selected entry.
  - Select the **True** radio button to allow the ACLs for the selected entry to accumulate from that entry, upward along the ancestor entry chain, to the highest filter ACL containing entry in the DIT.
  - Select the **False** radio button to stop the accumulation of filter ACLs at the selected entry.
- DN (Distinguished Name) - Enter the **(DN) Distinguished name** of the entity requesting access to perform operations on the selected entry, for example, cn=Marketing Group.
- Type - Enter the **Type** of DN. For example, select access-id if the DN is a user.

Adding and editing access rights

Click the either the **Add** button to add the DN in the DN (Distinguished Name) field to the ACL list or the Edit button to modify the ACLs of an existing DN.

The **Add access rights** and **Edit access rights** panels allow you to set the access rights for a new or existing Access Control List (ACLs). The Type field defaults to the type you selected on the Edit ACL panel. If you are adding an ACL, all other fields default to blank. If you are editing an ACL, the fields contain the values set last time the ACL was modified.

You can:

- Change the ACL type
- Set addition and deletion rights
- Set the object filter for filtered ACLs
- Set permissions for security classes

To set access rights:

1. Select the **Type** of entry for the ACL. For example, select access-id if the DN is a user.
2. The **Rights** section displays the addition and deletion rights of the subject.
   - **Add child** grants or denies the subject the right to add a directory entry beneath the selected entry.
   - **Delete entry** grants or denies the subject the right to delete the selected entry.
3. Set the object filter for a filter based comparison. In the **Object filte**r field, enter the desired object filter for the selected ACL. Click the **Edit filter** button for assistance in composing the search filter string. The current filtered ACL propagates to any descendant object in the associated subtree that matches the filter in this field.
4. The **Security class** section defines permissions for attribute classes. Attributes are grouped into security classes:
   - Normal - Normal attribute classes require the least security, for example, the attribute commonName.

- Sensitive - Sensitive attribute classes require a moderate amount of security, for example homePhone.
- Critical - Critical attribute classes require the most security, for example, the attribute userpassword.

Each security class has permissions associated with it.

- Read - the subject can read attributes.
- Write - the subject can modify the attributes.
- Search - the subject can search attributes.
- Compare - the subject can compare attributes.

Additionally, you may specify permissions based on the attribute instead of the security class to which the attribute belongs. The attribute section is listed below the **Critical security class**.

- Select an attribute from the **Define an attribute** drop-down list.
- Click **Define**. The attribute is displayed with a permissions table.
- Specify whether to grant or deny each of the four security class permissions associated with the attribute.
- You can repeat this procedure for multiple attributes.
- To remove an attribute, simply select the attribute and click **Delete**.
- When you are finished click **OK**.

Removing ACLs

You can remove ACLs in either of two ways:

- Select the radio button next to the ACL you want to delete. Click **Remove**.
- Click **Remove all** to delete all DNs from the list.

Owners

Entry owners have complete permissions to perform any operation on an object. Entry owners can be explicit or propagated (inherited).

Enter the following information on the **Owners** tab:

- Select the **Propagate owners** check box to allow descendants without an explicitly defined owner to inherit from this entry. If the check box is not selected, descendant entries without an explicitly defined owner will inherit owner from a parent of this entry that has this option enabled.
- DN (Distinguished Name) - Enter the **(DN) Distinguished name** of the entity requesting access to perform operations on the selected entry, for example, cn=Marketing Group.
- Type - Enter the **Type** of DN. For example, select access-id if the DN is a user.

Adding an owner

Click **Add** to add the DN in the **DN (Distinguished Name)** field to the list.

Removing an owner

You can remove an owner in either of two ways:

- Select the radio button next to the owner's DN that you want to delete. Click **Remove**.
- Click **Remove all** to delete all owner DNs from the list.

Subtree replication considerations

For filter-based access to be included in subtree replication, any ibm-filterAclEntry attributes must reside at, or below, the associated ibm-replicationContext entry.

Because effective access cannot be accumulated from an ancestor entry above a replicated subtree, the ibm-filterAclInherit attribute must be set to a value of **false**, and reside at the associated ibm-replicationContext entry.

Chapter 13. Groups and roles

Groups

A group is a list, a collection of names. A groups can be used in **aclentry**, **ibm-fliterAclEntry**, and **entryowner** attributes to control access or in application-specific uses such as a mailing list; see Chapter 12, "Access Control Lists" on page 149. Groups can be defined as either static, dynamic, or nested.

Static groups

A static group defines each member individually using the structural objectclass **groupOfNames**, **groupOfUniqueNames**, **accessGroup**, or **accessRole**; or the auxiliary objectclass **ibm-staticgroup**. These objectclasses require the attribute **member** (or uniqueMember in the case of groupOfUniqueNames). A static group using these structural objectclasses must have at least one member; it cannot be empty. A static group may also be defined using the auxiliary objectclass: **ibm-staticGroup**, which does not require the **member** attribute, and therefore may be empty.

A typical group entry is:

```
DN: cn=Dev.Staff,ou=Austin,c=US
 objectclass: accessGroup
 cn: Dev.Staff
 member: cn=John Doe,o=IBM,c=US
 member: cn=Jane Smith,o=IBM,c=US
 member: cn=James Smith,o=IBM,c=US
```

Each group object contains a multivalued attribute consisting of member DNs.

Upon deletion of an access group, the access group is also deleted from all ACLs to which it has been applied.

Dynamic groups

A dynamic group defines its members differently than a static group. Instead of listing them individually, the dynamic group defines its members using an LDAP search. The dynamic group uses the structural objectclass **groupOfURLs** (or auxiliary objectclass **ibm-dynamicGroup**) and the attribute, **memberURL** to define the search using a simplified LDAP URL syntax.

```
ldap:///<base DN of search> ? ? <scope of search> ? <searchfilter>
```

Note: As the example illustrates, the host name must not be present in the syntax. The remaining parameters are just like normal ldap URL syntax. Each parameter field must be separated by a ?, even if no parameter is specified. Normally, a list of attributes to return would be included between the base DN and scope of the search. This parameter is also not used by the server when determining dynamic membership, and so may be omitted, however, the separator **?** must still be present.

where:

base DN of search
: Is the point from which the search begins in the directory. It can be the suffix or root of the directory such as **ou=Austin**. This parameter is required.

scope of search
Specifies the extent of the search. The default scope is base.

base	Returns information only about the base DN specified in the URL
one	Returns information about entries one level below the base DN specified in the URL. It does not include the base entry.
sub	Returns information about entries at all levels below and includes the base DN.

searchfilter
Is the filter that you want to apply to the entries within the scope of the search. See "the ldapsearch filter option" on page 218 for information about the syntax of the searchfilter. The default is objectclass=*

The search for dynamic members is always internal to the server, so unlike a full ldap URL, a host name and port number is never specified, and the protocol is always **ldap** (never **ldaps**). The **memberURL** attribute may contain any kind of URL, but the server only uses **memberURL**s beginning with **ldap:///** to determine dynamic membership.

Examples

A single entry in which the scope defaults to base and the filter defaults to objectclass=*:

```
ldap:///cn=John Doe, cn=Employees, o=Acme, c=US
```

All entries that are 1-level below cn=Employees, and the filter defaults to objectclass=*:

```
ldap:///cn=Employees, o=Acme, c=US??one
```

All entries that are under o-Acme with the objectclass=person:

```
ldap:///o=Acme, c=US??sub?objectclass=person
```

Depending on the object classes you use to define user entries, those entries might not contain attributes which are appropriate for determining group membership. You can use the auxiliary object class, **ibm-dynamicMember**, to extend your user entries to include the **ibm-group** attribute. This attribute allows you to add arbitrary values to your user entries to serve as targets for the filters of your dynamic groups. For example:

The members of this dynamic group are entries directly under the cn=users,ou=Austin entry that have an ibm-group attribute of GROUP1:

```
dn: cn=GROUP1,ou=Austin
 objectclass: groupOfURLs
 cn: GROUP1
 memberURL: ldap:///cn=users,ou=Austin??one?(ibm-group=GROUP1)
```

Here is an example member of cn=GROUP1,ou=Austin:

```
dn: cn=Group 1 member, cn=users, ou=austin
 objectclass: person
 objectclass: ibm-dynamicMember
 sn: member
 userpassword: memberpassword
 ibm-group: GROUP1
```

Nested groups

The nesting of groups enables the creation of hierarchical relationships that can be used to define inherited group membership. A nested group is defined as a child group entry whose DN is referenced by an attribute contained within a parent group entry. A parent group is created by extending one of the structural group object classes (**groupOfNames**, **groupOfUniqueNames**, **accessGroup**, **accessRole**, or **groupOfURLs**) with the addition of the **ibm-nestedGroup** auxiliary object class. After nested group extension, zero or more **ibm-memberGroup** attributes may be added, with their values set to the DNs of nested child groups. For example:

```
dn: cn=Group 2, cn=Groups, o=IBM, c=US
 objectclass: groupOfNames
 objectclass: ibm-nestedGroup
 objectclass: top
 cn: Group 2
 description: Group composed of static, and nested members.
 member: cn=Person 2.1, cn=Dept 2, cn=Employees, o=IBM, c=US
 member: cn=Person 2.2, cn=Dept 2, cn=Employees, o=IBM, c=US
 ibm-memberGroup: cn=Group 8, cn=Nested Static, cn=Groups, o=IBM, c=US
```

The introduction of cycles into the nested group hierarchy is not allowed. If it is determined that a nested group operation results in a cyclical reference, either directly or through inheritance, it is considered a constraint violation and therefore, the update to the entry fails.

Hybrid groups

Any of the structural group object classes mentioned can be extended such that group membership is described by a combination of static, dynamic, and nested member types. For example:

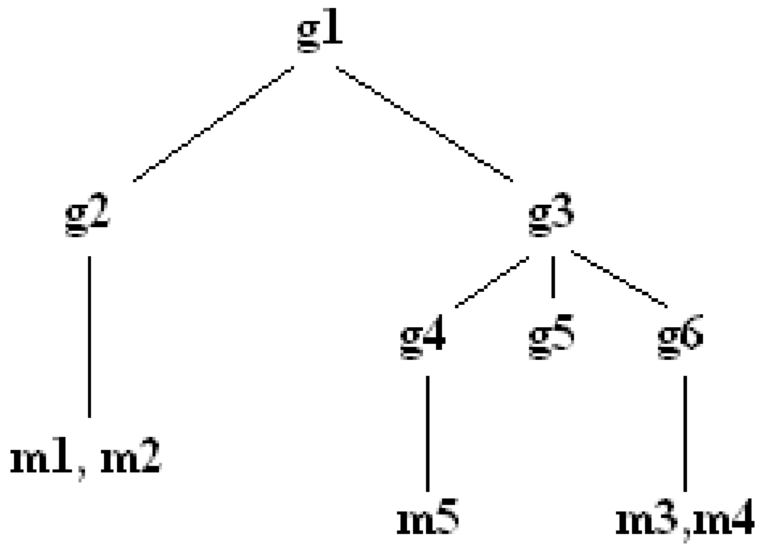
```
dn: cn=Group 10, cn=Groups, o=IBM, c=US
 objectclass: groupOfURLs
 objectclass: ibm-nestedGroup
 objectclass: ibm-staticGroup
 objectclass: top
 cn: Group 10
 description: Group composed of static, dynamic, and nested members.
 memberURL: ldap:///cn=Austin, cn=Employees, o=IBM, c=US??one?objectClass=person
 ibm-memberGroup: cn=Group 9, cn=Nested Dynamic, cn=Groups, o=IBM, c=US
 member: cn=Person 10.1, cn=Dept 2, cn=Employees, o=IBM, c=US
 member: cn=Person 10.2, cn=Dept 2, cn=Employees, o=IBM, c=US
```

Determining group membership

Two operational attributes can be used to query aggregate group membership. For a given group entry, the **ibm-allMembers** operational attribute enumerates the aggregate set of group membership, including static, dynamic, and nested members, as described by the nested group hierarchy. For a given user entry, the **ibm-allGroups** operational attribute enumerates the aggregate set of groups, including ancestor groups, to which that user has membership.

A requester may only receive a subset of the total data requested, depending on how the ACLs have been set on the data. Anyone can request the **ibm-allMembers** and **ibm-allGroups** operational attributes, but the data set returned only contains data for the LDAP entries and attributes that the requester has access rights to. The user requesting the **ibm-allMembers** or **ibm-allGroups** attribute must have access to the **member** or **uniquemember** attribute values for the group and nested groups in order to see static members, and must be able to perform the searches specified in the **memberURL** attribute values in order to see dynamic members. For examples:

Hierarchy examples



For this example, **m1** and **m2** are in the member attribute of **g2**. The ACL for **g2** allows **user1** to read the member attribute, but **user 2** does not have access to the member attribute. The entry LDIF for the **g2** entry is as follows:

```
dn: cn=g2,cn=groups,o=ibm,c=us
objectclass: accessGroup
cn: g2
member: cn=m1,cn=users,o=ibm,c=us
member: cn=m2,cn=users,o=ibm,c=us
aclentry: access-id:cn=user1,cn=users,o=ibm,c=us:normal:rsc
aclentry: access-id:cn=user2,cn=users,o=ibm,c=us:normal:rsc:at.member:deny:rsc
```

The **g4** entry uses the default aclentry, which allows both **user1** and **user2** to read its member attribute. The LDIF for the **g4** entry is as follows:

```
dn: cn=g4, cn=groups,o=ibm,c=us
objectclass: accessGroup
cn: g4
member: cn=m5, cn=users,o=ibm,c=us
```

The **g5** entry is a dynamic group, which gets its two members from the memberURL attribute. The LDIF for the **g5** entry is as follows:

```
dn: cn=g5, cn=groups,o=ibm,c=us
objectclass: container
objectclass: ibm-dynamicGroup
cn: g5
memberURL: ldap:///cn=users,o=ibm,c=us??sub?(|(cn=m3)(cn=m4))
```

The entries **m3** and **m4** are members of group **g5** because they match the **memberURL** The ACL for the **m3** entry allows both **user1** and **user2** to search for it. The ACL for the **m4** entries doesn't allow **user2** to search for it. The LDIF for **m4** is as follows:

```
dn: cn=m4, cn=users,o=ibm,c=us
objectclass:person
cn: m4
```

```
sn: four
aclentry: access-id:cn=user1,cn=users,o=ibm,c=us:normal:rsc
aclentry: access-id:cn=user2,cn=users,o=ibm,c=us
```

Example 1:

User 1 does a search to get all the members of group **g1**. User 1 has access to all members, so they are all returned.

```
ldapsearch -D cn=user1,cn=users,o=ibm,c=us -w user1pwd -s base -b cn=g1,
           cn=groups,o=ibm,c=us objectclass=* ibm-allmembers


cn=g1,cn=groups,o=ibm,c=us
ibm-allmembers: CN=M1,CN=USERS,O=IBM,C=US
ibm-allmembers: CN=M2,CN=USERS,O=IBM,C=US
ibm-allmembers: CN=M3,CN=USERS,O=IBM,C=US
ibm-allmembers: CN=M4,CN=USERS,O=IBM,C=US
ibm-allmembers: CN=M5,CN=USERS,O=IBM,C=US
```

Example 2:

User 2 does a search to get all the members of group **g1**. User 2 does not have access to members **m1** or **m2** because they do not have access to the member attribute for group **g2**. User 2 has access to the member attribute for **g4** and therefore has access to member **m5**. User 2 can perform the search in the group **g5** memberURL for entry **m3**, so that member are listed, but cannot perform the search for **m4**.

```
ldapsearch -D cn=user2,cn=users,o=ibm,c=us -w user2pwd -s base -b cn=g1,
           cn=groups,o=ibm,c=us objectclass=* ibm-allmembers


cn=g1,cn=groups,o=ibm,c=us
ibm-allmembers: CN=M3,CN=USERS,O=IBM,C=US
ibm-allmembers: CN=M5,CN=USERS,O=IBM,C=US
```

Example 3:

User 2 does a search to see if **m3** is a member of group **g1**. User 2 has access to do this search, so the search shows that **m3** is a member of group **g1**.

```
ldapsearch -D cn=user2,cn=users,o=ibm,c=us -w user2pwd -s base -b cn=m3,
           cn=users,o=ibm,c=us objectclass=* ibm-allgroups


cn=m3,cn=users,o=ibm,c=us
ibm-allgroups: CN=G1,CN=GROUPS,O=IBM,C=US
```

Example 4:

User 2 does a search to see if **m1** is a member of group **g1**. User 2 does not have access to the member attribute, so the search does not show that **m1** is a member of group **g1**.

```
ldapsearch -D cn=user2,cn=users,o=ibm,c=us -w user2pwd -s base -b
           cn=m1,cn=users,o=ibm,c=us objectclass=* ibm-allgroups


cn=m1,cn=users,o=ibm,c=us
```

Group object classes

ibm-dynamicGroup

This auxiliary class allows the optional **memberURL** attribute. Use it with a structural class such as **groupOfNames** to create a hybrid group with both static and dynamic members.

ibm-dynamicMember
This auxiliary class allows the optional **ibm-group** attribute. Use it as a filter attribute for dynamic groups.

ibm-nestedGroup
This auxiliary class allows the optional **ibm-memberGroup** attribute. Use it with a structural class such as **groupOfNames** to enable sub-groups to be nested within the parent group.

ibm-staticGroup
This auxiliary class allows the optional **member** attribute. Use it with a structural class such as **groupOfURLs** to create a hybrid group with both static and dynamic members.

Note: The **ibm-staticGroup** is the only class for which **member** is *optional*, all other classes taking **member** require at least 1 member.

Group attribute types

ibm-allGroups
Shows all groups to which an entry belongs. An entry can be a member directly by the **member**, **uniqueMember**, or **memberURL** attributes, or indirectly by the **ibm-memberGroup** attribute. This **Read-only** operational attribute is not allowed in a search filter.

ibm-allMembers
Shows all members of a group. An entry can be a member directly by the **member**, **uniqueMember**, or **memberURL** attributes, or indirectly by the **ibm-memberGroup** attribute. This **Read-only** operational attribute is not allowed in a search filter.

ibm-group
Is an attribute taken by the auxiliary class **ibm-dynamicMember**. Use it to define arbitrary values to control membership of the entry in dynamic groups. For example, add the value ″Bowling Team″ to include the entry in any **memberURL** that has the filter ″ibm-group=Bowling Team″.

ibm-memberGroup
Is an attribute taken by the auxiliary class **ibm-nestedGroup**. It identifies sub-groups of a parent group entry. Members of all such sub-groups are considered members of the parent group when processing ACLs or the **ibm-allMembers** and **ibm-allGroups** operational attributes. The sub-group entries themselves are *not* members. Nested membership is recursive.

Roles

Role-based authorization is a conceptual complement to the group-based authorization, and is useful in some cases. As a member of a role, you have the authority to do what is needed for the role in order to accomplish a job. Unlike a group, a role comes with an implicit set of permissions. There is not a built-in assumption about what permissions are gained (or lost) by being a member of a group.

Roles are similar to groups in that they are represented in the directory by an object. Additionally, roles contain a group of DNs. Roles which are to be used in access control must have an objectclass of ′AccessRole′. The ′Accessrole′ objectclass is a subclass of the ′GroupOfNames′ objectclass.

For example, if there are a collection of DNs such as ’sys admin’, your first reaction may be to think of them as the ’sys admin group’ (since groups and users are the most familiar types of privilege attributes). However, since there are a set of permissions that you would expect to receive as a member of ’sys admin’ the collection of DNs may be more accurately defined as the ’sys admin role’.

Part 4. User-related tasks

Chapter 14. Realms, templates, users, and groups

A realm is a collection of users and the groups to which they belong. For example a company, a bowling team, or a club could all be realms.

Realms are defined by creating entries of object class ″ibm-realm″ anywhere in a user naming context (not under cn=localhost, cn=schema or cn=configuration). The ibm-realm object defines the realm's name (cn), a group of realm administrators (ibm-realmAdminGroup), a user-template object (ibm-realmUserTemplate) specifying the object classes and attributes for users in the realm, and the location of container entries under which user and group entries are stored (ibm-realmUserContainer and ibm-realmGroupContainer). The directory administrator is responsible for managing user-templates, realms and realm administrator groups. After a realm is created, members of that realm's administrator group (realm administrators) are responsible for managing the users and groups within that realm.

Creating a realm

Expand the **Realms and templates** category in the navigation area of the Web Administration Tool.

1. Click **Add realm**.
   - Enter the name for the realm. For example **realm1**.
   - Enter the Parent DN that identifies the location of the realm. This entry is in the form of a suffix, for example o=ibm,c=us. You can also click **Browse** to select the location of the subtree that you want.
2. Click **Next** to continue or click **Finish**.
3. If you clicked **Next**, review the information. At this point you haven't actually created the realm, so **User template** and **User search filter** can be ignored.
4. Click **Finish** to create the realm.

Creating a realm administrator

To create a realm administrator, you must first create an administration group for the realm.

Creating the realm administration group

Expand the **Directory management** category in the navigation area of the Web Administration Tool.

1. Click **Manage entries**.
2. Expand the tree and select the realm you just created, **cn=realm1,o=ibm,c=us**.
3. Click **Edit ACL**.
4. Click the **Owners** tab.
5. Ensure that **Propagate owner** is checked.
6. Enter the DN for the realm, **cn=realm1,o=ibm,c=us**.
7. Change the **Type** to group.
8. Click **Add**.

Creating the administrator entry

If you do not already have a user entry for the administrator, you must create one.

Expand the **Directory management** category in the navigation area of the Web Administration Tool.

1. Click **Manage entries**.
2. Expand the tree to the location where you want the administrator entry to reside.

   Note: Locating the administrator entry outside of the realm avoids giving the administrator the ability to accidently delete him or herself. In this example the location might be **o=ibm,c=us**.
3. Click **Add**.
4. Select the **Structural object class**, for example **inetOrgPerson**.
5. Click **Next**.
6. Select any auxiliary object class you want to add.
7. Click **Next**.
8. Enter the required attributes for the entry. For example,
   - **RDN** cn=JohnDoe
   - **DN** o=ibm,c=us
   - **cn** John Doe
   - **sn** Doe
9. On the **Other attributes** tab ensure that you have assigned a password.
10. When you are done, click **Finish**.

Adding the administrator to the administration group.

Expand the **Directory management** category in the navigation area of the Web Administration Tool.

1. Click **Manage entries**.
2. Expand the tree and select the realm you just created, **cn=realm1,o=ibm,c=us**.
3. Click **Edit attributes**.
4. Click the **Members** tab.
5. Click **Members**.
6. In the **Members** field enter the DN of the administrator, in this example **cn=John Doe,o=ibm,c=us**.
7. Click **Add**. The DN is displayed in the **Members** list.
8. Click **OK**.
9. Click **Update**. The DN is displayed in the **Current members** list.
10. Click **OK**.

You have created an administrator that can manage entries within the realm.

Creating a template

After you have created a realm, your next step is to create a user template. A template helps you to organize the information you want to enter. Expand the **Realms and templates** category in the navigation area of the Web Administration Tool.

1. Click **Add user template**.

- Enter the name for the template, for example, **template1**.
- Enter the location where the template is going to reside. For replication purposes, locate the template in the subtree of the realm that is going to use this template. For example, the realm created in the previous operations **cn=realm1,o=ibm,c=us**. You can also click **Browse** to select a different subtree for the location of the template.

2. Click **Next**. You can click **Finish** to create an empty template. You can later add information to the template, see "Editing a template" on page 184.
3. If you clicked **Next**, choose the structural object class for the template, for example **inetOrgPerson**. You can also add any auxiliary object classes that you want.
4. Click **Next**.
5. A **Required** tab has been created on the template. You can modify the information contained on this tab.
   a. Select **Required** in the tab menu and click **Edit**. The **Edit tab** panel is displayed. You see the name of the tab **Required** and the selected attributes that are required by the object class, **inetOrgPerson**:
      - *sn - surname
      - *cn - common name

      Note: The * denotes required information.

   b. If you want to add additional information to this tab, select the attribute from the **Attributes** menu. For example, select **departmentNumber** and click **Add**. Select **employeeNumber** and click **Add**. Select **title** and click **Add**. The **Selected attributes** menu now reads:
      - title
      - employeeNumber
      - departmentNumber
      - *sn
      - *cn

   c. You can rearrange the way that these fields appear on the template by highlighting the selected attribute and clicking **Move up** or **Move down**. This changes the position of the attribute by one position. Repeat this procedure until you have the attributes arranged in the order you want them. For example,
      - *sn
      - *cn
      - title
      - employeeNumber
      - departmentNumber

   d. You can also modify each selected attribute.
      1) Highlight the attribute in the **Selected attributes** box and click **Edit**.
      2) You can change the display name of the field used on the template. For example, if you want **departmentNumber** to be displayed as **Department number** enter that into the **Display name** field.
      3) You can also supply a default value to prefill the attribute field in the template. For example, if most of the users that are going to be entered are members of Department 789, you can enter 789 as the default value. The field on the template is prefilled with 789. The value can be changed when you add the actual user information.

4) Click **OK**.

e. Click **OK**.

6. To create another tab category for additional information, click **Add**.
   - Enter the name for the new tab. For example, Address information.
   - For this tab, select the attributes from the **Attributes** menu. For example, select **homePostalAddress** and click **Add**. Select **postOfficeBox** and click **Add**. Select **telephoneNumber** and click **Add**. Select **homePhone** and click **Add**. Select **facsimileTelephoneNumber** and click **Add**. The **Selected attributes** menu reads:
     – homePostalAddress
     – postOfficeBox
     – telephoneNumber
     – homePhone
     – facsimileTelephoneNumber
   - You can rearrange the way that these fields appear on the template by highlighting the selected attribute and clicking **Move up** or **Move down**. This changes the position of the attribute by one position. Repeat this procedure until you have the attributes arranged in the order you want them. For example,
     – homePostalAddress
     – postOfficeBox
     – telephoneNumber
     – facsimileTelephoneNumber
     – homePhone
   - Click **OK**.
7. Repeat this process for as many tabs as you want to create. When you are finished click **Finish** to create the template.

Adding the template to a realm

After you have created a realm and a template, you need to add the template to the realm. Expand the **Realms and templates** category in the navigation area of the Web Administration Tool.

1. Click **Manage realms**.
2. Select the realm you want to add the template to, in this example, **cn=realm1,o=ibm,c=us** and click **Edit**.
3. Scroll down to **User template** and expand the drop down menu.
4. Select the template, in this example, **cn=template1,cn=realm1,o=ibm,c=us**.
5. Click **OK**.
6. Click **Close**.

Creating groups

Expand the **Users and groups** category in the navigation area of the Web Administration Tool.

1. Click **Add group**.
2. Enter the name of the group that you want to create. For example **group1**.
3. Select the realm that you want to add the user to from the drop-down menu. In this case **realm1**.

4. Click **Finish** to create the group. If you already have users in the realm you can click **Next** and select users to add to group1. Then click **Finish**.

See “Groups” on page 167 for additional information.

Adding a user to the realm

Expand the **Users and groups** category in the navigation area of the Web Administration Tool.

1. Click **Add user**.
2. Select the realm that you want to add the user to from the drop-down menu. In this case **realm1**.
3. Click **Next**. The template that you just created, template1, is displayed. Fill in the required fields, denoted by an asterisk (*) and any of the other fields on the tabs. If you have already created groups within the realm, you can also add the user to one or more groups.
4. When you are done, click **Finish**.

Managing realms

After you have set up and populated your initial realm, you can add more realms or modify existing realms.

Expand the **Realms and templates** category in the navigation area and click **Manage realms**. A list of existing realms is displayed. From this panel you can add a realm, edit a realm, remove a realm or edit the access control list (acls) of the realm.

Adding a realm

Expand the **Realms and templates** category in the navigation area of the Web Administration Tool.

1. Click **Add realm**.
   - Enter the name for the realm. For example **realm2**.
   - If you have preexisting realms, for example **realm1**, you can select a realm to have its settings copied to the realm you are creating.
   - Enter the Parent DN that identifies the location of the realm. This entry is in the form of a suffix, for example **o=ibm,c=us**. You can also click **Browse** to select the location of the subtree that you want.
2. Click **Next** to continue or click **Finish**.
3. If you clicked **Next**, review the information.
4. Select a **User template** from the drop-down menu. If you copied the settings from a preexisting realm, its template is prefilled in this field.
5. Enter a **User search filter**.
6. Click **Finish** to create the realm.

Editing a realm

Expand the **Realms and templates** category in the navigation area of the Web Administration Tool.

- Click **Manage realms**.
- Select the realm that you want to edit from the list of realms.
- Click **Edit**.

– You can use the **Browse** buttons to change the
  - Administrator group
  - Group container
  - User container
– You can select a different template from the drop-down menu.
– Click **Edit** to modify the **User search filter**.

- Click **OK** when you are finished.

Removing a realm

Expand the **Realms and templates** category in the navigation area of the Web Administration Tool.

1. Click **Manage realms**.
2. Select the realm you want to remove.
3. Click **Delete**.
4. When prompted to confirm the deletion, click **OK**.
5. The realm is removed from the list of realms.

Editing ACLs on the realm

To view ACL properties using the Web Administration Tool utility and to work with ACLs, see "Working with Access Control Lists" on page 161.

See Chapter 12, "Access Control Lists" on page 149 for additional information.

Managing templates

After you have created your initial template, you can add more templates or modify existing templates.

Expand the **Realms and templates** category in the navigation area and click **Manage user templates**. A list of existing templates is displayed. From this panel you can add a template, edit a template, remove a template or edit the access control list (ACLs) of the template.

Adding a user template

Expand the **Realms and templates** category in the navigation area of the Web Administration Tool.

1. Click **Add user template** or click **Manage user templates** and click **Add**.
   - Enter the name for the new template. For example **template2**.
   - If you have preexisting templates, for example **template1**, you can select a template to have its settings copied to the template you are creating.
   - Enter the Parent DN that identifies the location of the template. This entry is in the form of a DN, for example **cn=realm1,o=ibm,c=us**. You can also click **Browse** to select the location of the subtree that you want.
2. Click **Next**. You can click **Finish** to create an empty template. You can later add information to the template see "Editing a template" on page 184.
3. If you clicked **Next**, choose the structural object class for the template, for example **inetOrgPerson**. You can also add any auxiliary object classes that you want.
4. Click **Next**.

5. A **Required** tab has been created on the template. You can modify the information contained on this tab.
   a. Select **Required** in the tab menu and click **Edit**. The **Edit tab** panel is displayed. You see the name of the tab **Required** and the selected attributes that are required by the object class, **inetOrgPerson**:
      - *sn - surname
      - *cn - common name

      Note: The * denotes required information.
   b. If you want to add additional information to this tab, select the attribute from the **Attributes** menu. For example, select **departmentNumber** and click **Add**. Select **employeeNumber** and click **Add**. Select **title** and click **Add**. The **Selected attributes** menu now reads:
      - title
      - employeeNumber
      - departmentNumber
      - *sn
      - *cn
   c. You can rearrange the way that these fields appear on the template by highlighting the selected attribute and clicking **Move up** or **Move down**. This changes the position of the attribute by one position. Repeat this procedure until you have the attributes arranged in the order you want them. For example,
      - *sn
      - *cn
      - title
      - employeeNumber
      - departmentNumber
   d. You can also modify each selected attribute.
      1) Highlight the attribute in the **Selected attributes** box and click **Edit**.
      2) You can change the display name of the field used on the template. For example, if you want **departmentNumber** to be displayed as **Department number** enter that into the **Display name** field.
      3) You can also supply a default value to prefill the attribute field in the template. For example, if most of the users that are going to be entered are members of Department 789, you can enter 789 as the default value. The field on the template is prefilled with 789. The value can be changed when you add the actual user information.
      4) Click **OK**.
   e. Click **OK**.
6. To create another tab category for additional, click **Add**.
   - Enter the name for the new tab. For example, Address information.
   - To this tab, select the attribute from the **Attributes** menu. For example, select **homePostalAddress** and click **Add**. Select **postOfficeBox** and click **Add**. Select **telephoneNumber** and click **Add**. Select **homePhone** and click **Add**. Select **facsimileTelephoneNumber** and click **Add**. The **Selected attributes** menu reads:
     – homePostalAddress
     – postOfficeBox

    - telephoneNumber
    - homePhone
    - facsimileTelephoneNumber
  - You can rearrange the way that these fields appear on the template by highlighting the selected attribute and clicking **Move up** or **Move down**. This changes the position of the attribute by one position. Repeat this procedure until you have the attributes arranged in the order you want them. For example,
    - homePostalAddress
    - postOfficeBox
    - telephoneNumber
    - facsimileTelephoneNumber
    - homePhone
  - Click **OK**.
7. Repeat this process for as many tabs as you want to create. When you are finished click **Finish** to create the template.

Editing a template

Expand the **Realms and templates** category in the navigation area of the Web Administration Tool.

- Click **Manage user templates**.
- Select the realm that you want to edit from the list of realms.
- Click **Edit**.
- If you have preexisting templates, for example template1, you can select a template to have its settings copied to the template you are editing.
- Click **Next**.
  - You can use the drop-down menu to change the structural object class of the template
  - You can add or remove auxiliary object classes.
- Click **Next**.
- You can modify the tabs and attributes contained in the template. See 5 on page 183 for information on how to modify the tabs.
- When you are done, click **Finish**.

Removing a template

Expand the **Realms and templates** category in the navigation area of the Web Administration Tool.

1. Click **Manage user templates**.
2. Select the template that you want to remove.
3. Click **Delete**.
4. When prompted to confirm the deletion, click **OK**.
5. The template is removed from the list of templates.

Editing ACLs on the template

Expand the **Realms and template** category in the navigation area of the Web Administration Tool.

1. Click **Manage user templates**.
2. Select the template for which you want to edit the ACLs.

3. Click **Edit ACL**.

To view ACL properties using the Web Administration Tool utility and to work with ACLs, see "Working with Access Control Lists" on page 161.

See Chapter 12, "Access Control Lists" on page 149 for additional information.

Managing users

After you have set up your realms and templates, you can populate them with users.

Adding users

Expand the **Users and groups** category in the navigation area of the Web Administration Tool.

1. Click **Add user** or click **Managing users** and click **Add**.
2. Select the realm that you want to add the user to from the drop-down menu.
3. Click **Next**. The template that is associated with that realm, is displayed. Fill in the required fields, denoted by an asterisk (*) and any of the other fields on the tabs. If you have already created groups within the realm, you can also add the user to one or more groups.
4. When you are done, click **Finish**.

Finding users within the realm

Expand the **Users and groups** category in the navigation area of the Web Administration Tool.

1. Click **Find user** or click **Manage users** and click **Find**.
2. Select the realm that you want to search to from the **Select realm** field.
3. Enter the search string in the **Naming attribute** field. Wildcards are supported, for example if you entered ***smith**, the result is all entries that have the naming attribute of smith.
4. You can perform the following operations on a selected user:
   - **Edi**t - See "Editing a user's information".
   - **Copy** - See "Copying a user" on page 186.
   - **Delete** - See "Removing a user" on page 186.
5. When you are done, click **OK**.

Editing a user's information

Expand the **Users and groups** category in the navigation area of the Web Administration Tool.

1. Click **Manage users**.
2. Select a realm from the drop-down menu. Click **View users**, if the users are not already displayed in the **Users** box.
3. Select the user you want to edit and click **Edit**.
4. Modify the information on the tabs, modify group membership.
5. When you are done click, **OK**.

Copying a user

If you need to create a number of users that have mostly identical information, you can create the additional users by copying the initial user and modifying the information.

Expand the **Users and groups** category in the navigation area of the Web Administration Tool.

1. Click **Manage users**.
2. Select a realm from the drop-down menu. Click **View users**, if the users are not already displayed in the **Users** box.
3. Select the user you want to copy and click **Copy**.
4. Modify the appropriate information for the new user, for example the required information that identifies a specific user, such as sn or cn. Information that is common to both users need not be changed.
5. When you are done click, **OK**.

Removing a user

Expand the **Users and groups** category in the navigation area of the Web Administration Tool.

1. Click **Manage users**.
2. Select a realm from the drop-down menu. Click **View users**, if the users are not already displayed in the **Users** box.
3. Select the user you want to remove and click **Delete**.
4. When prompted to confirm the deletion, click **OK**.
5. The user is removed from the list of users.

Managing groups

After you have set up your realms and templates, you can create groups.

Adding groups

Expand the **Users and groups** category in the navigation area of the Web Administration Tool.

1. Click **Add group** or click **Manage groups** and click **Add**.
2. Enter the name of the group that you want to create.
3. Select the realm that you want to add the user to from the drop-down menu.
4. Click **Finish** to create the group. If you already have users in the realm you can click **Next** and select users to add to the group. Then click **Finish**.

See “Groups” on page 167 for additional information.

Finding groups within the realm

Expand the **Users and groups** category in the navigation area of the Web Administration Tool.

1. Click **Find group** or click **Manage groups** and click **Find**.
2. Select the realm that you want to search to from the **Select realm** field.
3. Enter the search string in the **Naming attribute** field. Wildcards are supported, for example if you entered ***club**, the result is all groups that have the naming attribute of club, for example, book club, chess club, garden club and so forth.
4. You can perform the following operations on a selected group:

- **Edit** - See “Editing a group’s information”.
- **Copy** - See “Copying a group”.
- **Delete** - See “Removing a group”.

5. When you are done, click **Close**.

Editing a group’s information

Expand the **Users and groups** category in the navigation area of the Web Administration Tool.

1. Click **Manage groups**.
2. Select a realm from the drop-down menu. Click **View groups**, if the groups are not already displayed in the **Groups** box.
3. Select the group you want to edit and click **Edit**.
4. You can click **Filter** to limit the number of **Available users**. For example entering *smith in the Last name field, limits the available users to those whose name ends in smith such as Ann Smith, Bob Smith, Joe Goldsmith, and so forth.
5. You can add or remove users from the group.
6. When you are done click, **OK**.

Copying a group

If you need to create a number of groups that have mostly the same members, you can create the additional groups by copying the initial group and modifying the information.

Expand the **Users and groups** category in the navigation area of the Web Administration Tool.

1. Click **Manage groups**.
2. Select a realm from the drop-down menu. Click **View groups**, if the users are not already displayed in the **Groups** box.
3. Select the group you want to copy and click **Copy**.
4. Change the group name in the **Group name** field. The new group has the same members as the original group.
5. You can modify the group members.
6. When you are done click, **OK**. The new group is created and contains the same members as the original group with any addition or removal modifications you made during the copy procedure.

Removing a group

Expand the **Users and groups** category in the navigation area of the Web Administration Tool.

1. Click **Manage groups**.
2. Select a realm from the drop-down menu. Click **View groups**, if the groups are not already displayed in the **Groups** box.
3. Select the group you want to remove and click **Delete**.
4. When prompted to confirm the deletion, click **OK**.
5. The group is removed from the list of groups.

Part 5. Command line utilities

Use these utilities as an alternative method of administering your IBM Directory Server.

Chapter 15. Command line utilities

This section describes the utilities that can be run from a command prompt.

Client Utilities

- “ldapmodify, ldapadd” on page 192
- “ldapchangepwd” on page 197
- “ldapdelete” on page 201
- “ldapexop” on page 204
- “ldapmodrdn” on page 210
- “ldapsearch” on page 214
- “ibmdirctl” on page 222

Server Utilities

- “ldif utility” on page 224
- “ldif2db utility” on page 224
- “db2ldif utility” on page 228
- “bulkload utility” on page 225
- “ldapdiff” on page 229
- “dbback” on page 236
- “dbrestore” on page 236
- “runstats” on page 236

The client utilities all use the ldap_sasl_bind API. When bind is invoked, several results can be returned. Following are bind results using various combinations of user IDs and passwords.

- If specifying the admin DN, the password must be correctly specified or the bind is not successful.
- If a null DN is specified, or a 0 length DN is specified, you receive unauthenticated access unless you are using an external bind (SASL) such as Kerberos.
- If a DN is specified, and is non-null, a password must also be specified or an error is returned.
- If a DN and password are specified but do not fall under any suffix in the directory, a referral is returned.
- If a DN and password are specified and are correct, the user is bound with that identity.
- If a DN and password are specified but the DN does not exist, unauthenticated access is given.
- If a DN and password are specified and the DN exists but the object does not have user password, an error message is returned.

Client utilities

This section provides a description of the client utilities. They are also documented in "Chapter 2. Ldap Utilities" in the *IBM Directory Server Version 5.1: Client SDK Programming Reference*.

ldapmodify, ldapadd

The LDAP modify-entry and LDAP add-entry tools

Synopsis

```
ldapmodify [-a] [-b] [-c] [-C charset] [-d debuglevel][-D binddn][-i file]
[-h ldaphost] [-k] [-K keyfile] [-m mechanism] [-M] [-N certificatename]
[-O maxhops] [-p ldapport] [-P keyfilepw] [-r] [-R] [-v] [-V]
[-w passwd | ?] [-Z]
```

```
ldapadd [-a] [-b] [-c] [-C charset] [-d debuglevel][-D binddn][-i file]
[-h ldaphost] [-k] [-K keyfile] [-m mechanism] [-M] [-N certificatename]
[-O maxhops] [-p ldapport] [-P keyfilepw] [-r] [-R] [-v] [-V] [-w passwd | ?]
[-Z]
```

Description

ldapmodify is a command-line interface to the ldap_modify and ldap_add library calls. **ldapadd** is implemented as a renamed version of ldapmodify. When invoked as ldapadd, the **-a** (add new entry) flag is turned on automatically.

ldapmodify opens a connection to an LDAP server, and binds to the server. You can use **ldapmodify** to modify or add entries. The entry information is read from standard input or from file through the use of the **-i** option.

To display syntax help for **ldapmodify** or **ldapadd**, type

```
ldapmodify -?
```

or

```
ldapadd -?
```

Options

-a Add new entries. The default action 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 \`/' are binary values and that the actual value is in a file whose path is specified in place of the valuer.

-c Continuous operation mode. Errors are reported, but **ldapmodify** continues with modifications. Otherwise he default action is to exit after reporting an error.

-C *charset*
Specifies that strings supplied as input to the **ldapmodify** and **ldapadd** utilities are represented in a local character set as specified by charset, and must be converted to UTF-8. When the **ldapmodify** and **ldapadd** records are received from standard input, the specified charset value is used to convert the attribute values that are designated as strings that is, the attribute types are followed by a single colon. If the records are received from an LDIF file that contains a charset tag, the charset tag in the LDIF file overrides the charset value specified on the command-line. See "IANA character sets supported by platform" on page 255 for the specific charset values that are supported for each operating system platform. Note that the supported values for charset are the same values supported for the charset tag that is optionally defined in Version 1 LDIF files.

-d *debuglevel*
Set the LDAP debugging level to debuglevel. See "Server debug mode" on page 242 for information about debug levels.

-D *binddn*
Use ***binddn*** to bind to the LDAP directory. ***binddn*** is a string-represented DN.

Note: **-D** *binddn* **-w** *passwd* does not call bind functions on superuser DNs.

-h *ldaphost*
Specify an alternate host on which the ldap server is running.

-i *file* Read the entry modification information from an LDIF file instead of from standard input. If an LDIF file is not specified, you must use standard input to specify the update records in LDIF format.

-k Specifies to use server administration control.

-K *keyfile*
Specify the name of the SSL key database file with default extension of **kdb**. If the key database file is not in the current directory, specify the fully-qualified key database filename. If a key database filename is not specified, this utility will first look for the presence of the SSL_KEYRING environment variable with an associated filename. If the SSL_KEYRING environment variable is not defined, the default keyring file will be used, if present.

A default keyring file that is, ldapkey.kdb, and the associated password stash file that is, ldapkey.sth, are installed in the /lib directory under LDAPHOME, where LDAPHOME is the path to the installed LDAP support. LDAPHOME varies by operating system platform:

- AIX, Linux operating systems- /usr/ldap
- Solaris operating systems - /usr/IBMldapc
- HP-UX operating systems- /usr/IBMldap
- Windows operating systems - c:\Program Files\IBM\LDAP

Note: This is the default install location. The actual LDAPHOME is determined during installation.

See the *IBM Directory C-Client SDK Programming Reference* for more information about default key database files, and default Certificate Authorities.

If a keyring database file cannot be located, a ″hard-coded″ set of default trusted certificate authority roots is used. The key database file typically contains one or more certificates of certificate 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, see “Using gsk6ikm” on page 52. Also see the “SSL notes” on page 196 and “Secure Sockets Layer” on page 47 for more information about SSL and certificates.

This parameter effectively enables the **-Z** switch.

-m *mechanism*
Use ***mechanism*** to specify the SASL mechanism to be used to bind to the server. The ldap_sasl_bind_s() API is used. The **-m** parameter is ignored if **-V 2** is set. If **-m** is not specified, simple authentication is used.

-M Manage referral objects as regular entries.

-N *certificatename*
: Specify the label associated with the client certificate in the key database file. If the LDAP server is configured to perform server authentication only, a client certificate is not required. If the LDAP server is configured to perform client and server Authentication, a client certificate might be required. ***certificatename*** is not required if a default certificate/private key pair has been designated as the default. Similarly, ***certificatename*** is not required if there is a single certificate/private key pair in the designated key database file. This parameter is ignored if neither **-Z** nor **-K** is specified.

-O *maxhops*
: Specify ***maxhops*** to set the maximum number of hops that the client library takes when chasing referrals. The default hopcount is 10.

-p *ldapport*
: Specify an alternate TCP port where the ldap server is listening. The default LDAP port is 389. If **-p** is not specified and **-Z** is specified, the default LDAP SSL port 636 is used.

-P *keyfilepw*
: Specify the key database password. This password is required to access the encrypted information in the key database file, which might include one or more private keys. If a password stash file is associated with the key database file, the password is obtained from the password stash file, and the **-P** parameter is not required. This parameter is ignored if neither **-Z** nor **-K** is specified.

-r Replace existing values by default.

-R Specifies that referrals are not to be automatically followed.

-v Use verbose mode, with many diagnostics written to standard output.

-V Specifies the LDAP version to be used by **ldapmodify** when it binds to the LDAP server. By default, an LDAP V3 connection is established. To explicitly select LDAP V3, specify **-V 3**. Specify **-V 2** to run as an LDAP V2 application. An application, like **ldapmodify**, selects LDAP V3 as the preferred protocol by using ldap_init instead of ldap_open.

-w *passwd* **|** **?**
: Use ***passwd*** as the password for authentication. Use the ? to generate a password prompt. Using this prompt prevents your password from being visible through the **ps** command.

-Z Use a secure SSL connection to communicate with the LDAP server. The **-Z** option is only supported when the SSL component entry, as provided by IBM's GSKit, is installed.

Input format

The contents of file (or standard input if no **-i** flag is given on the command line) should conform to the LDIF format.

Alternative 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 the following:

```
Distinguished Name (DN)

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. 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 `\\' to continue values across lines and preserve new lines in the value itself.

`attr` should be preceded by a - to remove a value. The = and `value` should be omitted to remove an entire attribute.

`attr` should be preceded by a + to add a value in the presence of the **-r** flag.

Examples

Assuming that the file /tmp/entrymods exists and has the following contents:

```
dn: cn=Modify Me, o=University of Higher Learning, c=US

      changetype: modify

      replace: mail

      mail: modme@student.of.life.edu

      -

      add: title

      title: Grand Poobah

      -

      add: jpegPhoto

      jpegPhoto: /tmp/modme.jpeg

      -

      delete: description

      -
```

the command:

```
ldapmodify -b -r -i /tmp/entrymods
```

will replace the contents of the Modify Me entry's mail attribute with the value modme@student.of.life.edu, add a title of Grand Poobah, and the contents of the file /tmp/modme.jpeg as a jpegPhoto, and completely remove the description attribute. These same modifications can be performed using the older ldapmodify input format:

```
cn=Modify Me, o=University of Higher Learning, c=US

      mail=modme@student.of.life.edu

      +title=Grand Poobah
      +jpegPhoto=/tmp/modme.jpeg

      -description
```

and the command:

```
ldapmodify -b -r -i /tmp/entrymods
```

Assuming that the file /tmp/newentry exists and has the following contents:

```
dn: cn=John Doe, o=University of Higher Learning, c=US

      objectClass: person

      cn: John Doe

      cn: Johnny

      sn: Doe

      title: the world's most famous mythical person

      mail: johndoe@student.of.life.edu

      uid: jdoe
```

the command:

```
 ldapadd -i /tmp/entrymods
```

adds a new entry for John Doe, using the values from the file /tmp/newentry.

Assuming that the file /tmp/newentry exists and has the contents:

```
dn: cn=John Doe, o=University of Higher Learning, c=US

changetype: delete
```

the command:

```
 ldapmodify -i /tmp/entrymods
```

removes John Doe's entry.

Notes

If entry information is not supplied from file through the use of the **-i** option, the **ldapmodify** command will wait to read entries from standard input. To break out of the wait, use Ctrl+C or Ctrl+D.

SSL notes

To use the SSL-related functions associated with this utility, the SSL libraries and tools must be installed. The SSL libraries and tools are provided with IBM's Global Security Kit (GSKit), which includes security software developed by RSA Security Inc.

Note: For information regarding the use of 128-bit and triple DES encryption by LDAP applications, including the LDAP sample programs, see "LDAP_SSL" in the *IBM Directory C-Client SDK Programming Reference*. This section describes the steps required to build the sample programs and your applications so they can use SSL with the strongest encryption algorithms available.

See the makefile associated with the sample programs for more information on linking an LDAP application so that it has access to 128-bit and triple-DES encryption algorithms.

The content of a client's key database file is managed with the gsk6ikm utility. For more information on this Java utility, see "Using gsk6ikm" on page 52. The gsk6ikm 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 'trusted' certificates issued by one of the trusted CAs. The gsk6ikm utility can also be used to obtain a client certificate, so that client and server authentication can be performed.

If the LDAP servers accessed by the client use server authentication only, 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 or ldap_simple_bind_s. For example, if the LDAP server is using a high-assurance VeriSign certificate, you should obtain a CA certificate from VeriSign, import it into your key database file, and mark it as trusted. If the LDAP server is using a self-signed server certificate, the administrator of the LDAP server can supply you with a copy of the server's certificate request file. Import the certificate request file into your key database file and mark it as trusted.

If the LDAP servers accessed by the client use client and server authentication, it is necessary to:

- Define one or more trusted root certificates in the key database file. This allows the client to 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 or ldap_simple_bind_s.
- Create a key pair using gsk6ikm and request a client certificate from a CA. After receiving the signed certificate from the CA, store the certificate in the client key database file.

Diagnostics

Exit status is 0 if no errors occur. Errors result in a non-zero exit status and a diagnostic message being written to standard error.

See also

ldapchangepwd, ldapdelete, ldapexop, ldapmodrdn, ldapsearch

ldapchangepwd

The LDAP modify password tool.

Synopsis

```
ldapchangepwd -D binddn -w passwd | ? -n newpassword | ?
[-C charset] [-d debuglevel][-h ldaphost] [-K keyfile]
[-m mechanism] [-M]  [-N certificatename] [-O maxhops]
[-p ldapport] [-P keyfilepw]  [-R] [-v] [-V version]
[-Z] [-?]
```

Description

Sends modify password requests to an LDAP server.

Options

-C *charset*
: Specifies that the DNs supplied as input to the **ldapdelete** utility are represented in a local character set, as specified by charset. Use **-C** *charset* to override the default, where strings must be supplied in UTF-8. See "IANA character sets supported by platform" on page 255 for the specific charset values that are supported for each operating system platform. Note that the supported values for charset are the same values supported for the charset tag that is optionally defined in Version 1 LDIF files.

-d *debuglevel*
: Set the LDAP debugging level to debuglevel. See "Server debug mode" on page 242 for information about debug levels.

-D *binddn*
: Use ***binddn*** to bind to the LDAP directory. ***binddn*** is a string-represented DN.

-h *ldaphost*
: Specify an alternate host on which the ldap server is running.

-K *keyfile*
: Specify the name of the SSL key database file with default extension of **kdb**. If the key database file is not in the current directory, specify the fully-qualified key database filename. If a key database filename is not specified, this utility will first look for the presence of the SSL_KEYRING environment variable with an associated filename. If the SSL_KEYRING environment variable is not defined, the default keyring file will be used, if present.

A default keyring file that is, ldapkey.kdb, and the associated password stash file that is, ldapkey.sth, are installed in the /lib directory under LDAPHOME, where LDAPHOME is the path to the installed LDAP support. LDAPHOME varies by operating system platform:

- AIX operating systems - /usr/ldap
- HP-UX operating systems - /usr/IBMldap
- Linux operating systems - /usr/ldap
- Solaris operating systems - /usr/IBMldapc
- Windows operating systems - c:\Program Files\IBM\LDAP

Note: This is the default install location. The actual LDAPHOME is determined during installation.

See *IBM Directory C-Client SDK Programming Reference* for more information about default key database files, and default Certificate Authorities.

If a keyring database file cannot be located, a "hard-coded" set of default trusted certificate authority roots is used. The key database file typically contains one or more certificates of certificate 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, see "Using gsk6ikm" on page 52. Also see the "SSL notes" on page 200 and "Secure Sockets Layer" on page 47 for more information about SSL and certificates.

This parameter effectively enables the **-Z** switch.

-m *mechanism*
: Use ***mechanism*** to specify the SASL mechanism to be used to bind to the server. The ldap_sasl_bind_s() API will be used. The **-m** parameter is ignored if **-V 2** is set. If **-m** is not specified, simple authentication is used.

-M
: Manage referral objects as regular entries.

-n *newpassword* **| ?**
: Specifies the new password. Use the ? to generate a password prompt. Using this prompt prevents your password from being visible through the **ps** command.

-N *certificatename*
: Specify the label associated with the client certificate in the key database file. If the LDAP server is configured to perform server authentication only, a client certificate is not required. If the LDAP server is configured to perform client and server Authentication, a client certificate might be required. ***certificatename*** is not required if a default certificate/private key pair has been designated as the default. Similarly, ***certificatename*** is not required if there is a single certificate/private key pair in the designated key database file. This parameter is ignored if neither **-Z** nor **-K** is specified.

-O *maxhops*
: Specify ***maxhops*** to set the maximum number of hops that the client library takes when chasing referrals. The default hopcount is 10.

-p *ldapport*
: Specify an alternate TCP port where the ldap server is listening. The default LDAP port is 389. If **-p** is not specified and **-Z** is specified, the default LDAP SSL port 636 is used.

-P *keyfilepw*
: Specify the key database password. This password is required to access the encrypted information in the key database file, which may include one or more private keys. If a password stash file is associated with the key database file, the password is obtained from the password stash file, and the **-P** parameter is not required. This parameter is ignored if neither **-Z** nor **-K** is specified.

-R
: Specifies that referrals are not to be automatically followed.

-v
: Use verbose mode, with many diagnostics written to standard output.

-V *version*
: Specifies the LDAP version to be used by **ldapdchangepwd** when it binds to the LDAP server. By default, an LDAP V3 connection is established. To explicitly select LDAP V3, specify **-V 3**. Specify **-V 2** to run as an LDAP V2 application. An application, like **ldapdchangepwd**, selects LDAP V3 as the preferred protocol by using ldap_init instead of ldap_open.

-w *passwd* **| ?**
: Use ***passwd*** as the password for authentication. Use the ? to generate a password prompt. Using this prompt prevents your password from being visible through the **ps** command.

-Z
: Use a secure SSL connection to communicate with the LDAP server. The **-Z** option is only supported when the SSL component entry, as provided by IBM's GSKit, is installed.

-?
: Displays the syntax help for ldapchangepwd.

Examples

The following command,

```
ldapchangepwd -D cn=John Doe -w a1b2c3d4 -n wxyz9876
```

changes the password for the entry named with commonName ″John Doe″ from a1b2c3d4 to wxyz9876

SSL notes

To use the SSL-related functions associated with this utility, the SSL libraries and tools must be installed. The SSL libraries and tools are provided with IBM's Global Security Kit (GSKit), which includes security software developed by RSA Security Inc.

Note: For information regarding the use of 128-bit and triple DES encryption by LDAP applications, including the LDAP sample programs, see ″LDAP_SSL″ in the *IBM Directory C-Client SDK Programming Reference*. This section describes the steps required to build the sample programs and your applications so they can use SSL with the strongest encryption algorithms available.

See the makefile associated with the sample programs for more information on linking an LDAP application so that it has access to 128-bit and triple-DES encryption algorithms.

The content of a client's key database file is managed with the gsk6ikm utility. For more information on this Java utility, see "Using gsk6ikm" on page 52. The gsk6ikm 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 'trusted' certificates issued by one of the trusted CAs. The gsk6ikm utility can also be used to obtain a client certificate, so that client and server authentication can be performed.

If the LDAP servers accessed by the client use server authentication only, 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 or ldap_simple_bind_s. For example, if the LDAP server is using a high-assurance VeriSign certificate, you should obtain a CA certificate from VeriSign, import it into your key database file, and mark it as trusted. If the LDAP server is using a self-signed server certificate, the administrator of the LDAP server can supply you with a copy of the server's certificate request file. Import the certificate request file into your key database file and mark it as trusted.

If the LDAP servers accessed by the client use client and server authentication, it is necessary to:

- Define one or more trusted root certificates in the key database file. This allows the client to 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 or ldap_simple_bind_s.
- Create a key pair using gsk6ikm and request a client certificate from a CA. After receiving the signed certificate from the CA, store the certificate in the client key database file.

Diagnostics

Exit status is 0 if no errors occur. Errors result in a non-zero exit status and a diagnostic message being written to standard error.

See also

ldapadd, ldapdelete, ldapexop, ldapmodify, ldapmodrdn, ldapsearch

ldapdelete

The LDAP delete-entry tool

Synopsis

```
ldapdelete [-c] [-C charset] [-d debuglevel][-D binddn][-i file]
[-h ldaphost] [-k] [-K keyfile] [-m mechanism] [-M] [-n] [-N certificatename]
[-O maxops] [-p ldapport] [-P keyfilepw] [-R] [-s][-v] [-V version]
[-w passwd | ?] [-Z] [dn]...
```

Description

ldapdelete is a command-line interface to the ldap_delete library call.

ldapdelete opens a connection to an LDAP server, binds, and deletes one or more entries. If one or more Distinguished Name (DN) arguments are provided, entries with those DNs are deleted. Each DN is a string-represented DN. If no DN arguments are provided, a list of DNs is read from standard input, or from file if the **-i** flag is used.

To display syntax help for **ldapdelete**, type:

```
ldapdelete -?
```

.

Options

-c Continuous operation mode. Errors are reported, but **ldapdelete** continues with modifications. Otherwise the default action is to exit after reporting an error.

-C *charset*
Specifies that the DNs supplied as input to the **ldapdelete** utility are represented in a local character set, as specified by charset. Use **-C** *charset* to override the default, where strings must be supplied in UTF-8. See "IANA character sets supported by platform" on page 255 for the specific charset values that are supported for each operating system platform. Note that the supported values for charset are the same values supported for the charset tag that is optionally defined in Version 1 LDIF files.

-d *debuglevel*
Set the LDAP debugging level to debuglevel. See "Server debug mode" on page 242 for information about debug levels.

-D *binddn*
Use ***binddn*** to bind to the LDAP directory. ***binddn*** is a string-represented DN.

-h *ldaphost*
Specify an alternate host on which the ldap server is running.

-i *file* Read a series of lines from file, performing one LDAP delete for each line in the file. Each line in the file should contain a single distinguished name.

-k Specifies to use server administration control.

-K *keyfile*

Specify the name of the SSL key database file with default extension of **kdb**. If the key database file is not in the current directory, specify the fully-qualified key database filename. If a key database filename is not specified, this utility will first look for the presence of the SSL_KEYRING environment variable with an associated filename. If the SSL_KEYRING environment variable is not defined, the default keyring file will be used, if present.

A default keyring file that is, ldapkey.kdb, and the associated password stash file that is, ldapkey.sth, are installed in the /lib directory under LDAPHOME, where LDAPHOME is the path to the installed LDAP support. LDAPHOME varies by operating system platform:

- AIX operating systems - /usr/ldap
- HP-UX operating systems - /usr/IBMldap
- Linux operating systems - /usr/ldap
- Solaris operating systems - /usr/IBMldapc
- Windows operating systems - c:\Program Files\IBM\LDAP

Note: This is the default install location. The actual LDAPHOME is determined during installation.

See *IBM Directory C-Client SDK Programming Reference* for more information about default key database files, and default Certificate Authorities.

If a keyring database file cannot be located, a "hard-coded" set of default trusted certificate authority roots is used. The key database file typically contains one or more certificates of certificate 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, see "Using gsk6ikm" on page 52. Also see the "SSL notes" on page 203 and "Secure Sockets Layer" on page 47 for more information about SSL and certificates.

This parameter effectively enables the **-Z** switch.

-m *mechanism*

Use ***mechanism*** to specify the SASL mechanism to be used to bind to the server. The ldap_sasl_bind_s() API will be used. The **-m** parameter is ignored if **-V 2** is set. If **-m** is not specified, simple authentication is used.

-M Manage referral objects as regular entries.

-n Show what would be done, but don't actually modify entries. Useful for debugging in conjunction with **-v**.

-N *certificatename*

Specify the label associated with the client certificate in the key database file. If the LDAP server is configured to perform server authentication only, a client certificate is not required. If the LDAP server is configured to perform client and server Authentication, a client certificate might be required. ***certificatename*** is not required if a default certificate/private key pair has been designated as the default. Similarly, ***certificatename*** is not required if there is a single certificate/private key pair in the designated key database file. This parameter is ignored if neither **-Z** nor **-K** is specified.

-O *maxhops*
Specify ***maxhops*** to set the maximum number of hops that the client library takes when chasing referrals. The default hopcount is 10.

-p *ldapport*
Specify an alternate TCP port where the ldap server is listening. The default LDAP port is 389. If **-p** is not specified and **-Z** is specified, the default LDAP SSL port 636 is used.

-P *keyfilepw*
Specify the key database password. This password is required to access the encrypted information in the key database file, which may include one or more private keys. If a password stash file is associated with the key database file, the password is obtained from the password stash file, and the **-P** parameter is not required. This parameter is ignored if neither **-Z** nor **-K** is specified.

-R Specifies that referrals are not to be automatically followed.

-s Use this option to delete the subtree rooted at the specified entry.

-v Use verbose mode, with many diagnostics written to standard output.

-V Specifies the LDAP version to be used by **ldapdelete** when it binds to the LDAP server. By default, an LDAP V3 connection is established. To explicitly select LDAP V3, specify **-V 3**. Specify **-V 2** to run as an LDAP V2 application. An application, like **ldapdelete**, selects LDAP V3 as the preferred protocol by using ldap_init instead of ldap_open.

-w *passwd* **| ?**
Use ***passwd*** as the password for authentication. Use the ? to generate a password prompt. Using this prompt prevents your password from being visible through the **ps** command.

-Z Use a secure SSL connection to communicate with the LDAP server. The **-Z** option is only supported when the SSL component entry, as provided by IBM's GSKit, is installed.

-dn Specifies one or more DN arguments. Each DN should be a string-represented DN.

Examples

The following command,

```
ldapdelete "cn=Delete Me, o=University of Life, c=US"
```

attempts to delete the entry named with commonName ″Delete Me″ directly below the University of Life organizational entry. It might 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 waits to read a list of DNs from standard input. To break out of the wait, use Ctrl+C or Ctrl+D.

SSL notes

To use the SSL-related functions associated with this utility, the SSL libraries and tools must be installed. The SSL libraries and tools are provided with IBM's Global Security Kit (GSKit), which includes security software developed by RSA Security Inc.

Note: For information regarding the use of 128-bit and triple DES encryption by LDAP applications, including the LDAP sample programs, see ″LDAP_SSL″

in the *IBM Directory C-Client SDK Programming Reference*. This section describes the steps required to build the sample programs and your applications so they can use SSL with the strongest encryption algorithms available.

See the makefile associated with the sample programs for more information on linking an LDAP application so that it has access to 128-bit and triple-DES encryption algorithms.

The content of a client's key database file is managed with the gsk6ikm utility. For more information on this Java utility, see "Using gsk6ikm" on page 52. The gsk6ikm 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 'trusted' certificates issued by one of the trusted CAs. The gsk6ikm utility can also be used to obtain a client certificate, so that client and server authentication can be performed.

If the LDAP servers accessed by the client use server authentication only, 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 or ldap_simple_bind_s. For example, if the LDAP server is using a high-assurance VeriSign certificate, you should obtain a CA certificate from VeriSign, import it into your key database file, and mark it as trusted. If the LDAP server is using a self-signed server certificate, the administrator of the LDAP server can supply you with a copy of the server's certificate request file. Import the certificate request file into your key database file and mark it as trusted.

If the LDAP servers accessed by the client use client and server authentication, it is necessary to:

- Define one or more trusted root certificates in the key database file. This allows the client to 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 or ldap_simple_bind_s.
- Create a key pair using gsk6ikm and request a client certificate from a CA. After receiving the signed certificate from the CA, store the certificate in the client key database file.

Diagnostics

Exit status is 0 if no errors occur. Errors result in a non-zero exit status and a diagnostic message being written to standard error.

See also

ldapadd, ldapchangepwd, ldapexop, ldapmodify, ldapmodrdn, ldapsearch

ldapexop

The LDAP extended operation tool

Synopsis

```
ldapexop [-C charset] [-d debuglevel][-D binddn][-e] [-h ldaphost]
[-help][-K keyfile] [-m mechanism] [-N certificatename]
[-p ldapport] [-P keyfilepw] [-?] [-v] [-w passwd | ?] [-Z]
-op {cascrepl | clearlog | controlqueue | controlrepl | getlogsize |
quiesce | readconfig | readlog}
```

Description

The **ldapexop** utility is a command-line interface that provides the capability to bind to a directory and issue a single extended operation along with any data that makes up the extended operation value.

The **ldapexop** utility supports the standard host, port, SSL, and authentication options used by all of the LDAP client utilities. In addition, a set of options is defined to specify the operation to be performed, and the arguments for each extended operation

To display syntax help for **ldapexop**, type:

```
ldapexop -?
```

or

```
ldapexop -help
```

Options

The options for the ldapexop command are divided into two categories:

1. General options that specify how to connect to the directory server. These options must be specified before operation specific options.
2. Extended operation option that identifies the extended operation to be performed.

General options: These options specify the methods of connecting to the server and must be specified before the **-op** option.

-C *charset*
: Specifies that the DNs supplied as input to the **ldapexop** utility are represented in a local character set, as specified by charset. Use **-C** *charset* to override the default, where strings must be supplied in UTF-8. See "IANA character sets supported by platform" on page 255 for the specific charset values that are supported for each operating system platform. Note that the supported values for charset are the same values supported for the charset tag that is optionally defined in Version 1 LDIF files.

-d *debuglevel*
: Set the LDAP debugging level to debuglevel. See "Server debug mode" on page 242 for information about debug levels.

-D *binddn*
: Use ***binddn*** to bind to the LDAP directory. ***binddn*** is a string-represented DN.

-e Displays the ldap library version information and then exits.

-h *ldaphost*
: Specify an alternate host on which the ldap server is running.

-help Displays the usage

-K *keyfile*
: Specify the name of the SSL key database file with default extension of **kdb**. If the key database file is not in the current directory, specify the

fully-qualified key database filename. If a key database filename is not specified, this utility will first look for the presence of the SSL_KEYRING environment variable with an associated filename. If the SSL_KEYRING environment variable is not defined, the default keyring file will be used, if present.

A default keyring file that is, ldapkey.kdb, and the associated password stash file that is, ldapkey.sth, are installed in the /lib directory under LDAPHOME, where LDAPHOME is the path to the installed LDAP support. LDAPHOME varies by operating system platform:

- AIX operating systems - /usr/ldap
- HP-UX operating systems - /usr/IBMldap
- Linux operating systems - /usr/ldap
- Solaris operating systems - /usr/IBMldapc
- Windows operating systems - c:\Program Files\IBM\LDAP

Note: This is the default install location. The actual LDAPHOME is determined during installation.

See *IBM Directory C-Client SDK Programming Reference* for more information about default key database files, and default Certificate Authorities.

If a keyring database file cannot be located, a "hard-coded" set of default trusted certificate authority roots is used. The key database file typically contains one or more certificates of certificate 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, see "Using gsk6ikm" on page 52. Also see the "SSL notes" on page 209 and "Secure Sockets Layer" on page 47 for more information about SSL and certificates.

This parameter effectively enables the **-Z** switch.

-m *mechanism*
Use ***mechanism*** to specify the SASL mechanism to be used to bind to the server. The ldap_sasl_bind_s() API will be used. The **-m** parameter is ignored if **-V 2** is set. If **-m** is not specified, simple authentication is used.

-N *certificatename*
Specify the label associated with the client certificate in the key database file. If the LDAP server is configured to perform server authentication only, a client certificate is not required. If the LDAP server is configured to perform client and server Authentication, a client certificate might be required. ***certificatename*** is not required if a default certificate/private key pair has been designated as the default. Similarly, ***certificatename*** is not required if there is a single certificate/private key pair in the designated key database file. This parameter is ignored if neither **-Z** nor **-K** is specified.

-p *ldapport*
Specify an alternate TCP port where the ldap server is listening. The default LDAP port is 389. If **-p** is not specified and **-Z** is specified, the default LDAP SSL port 636 is used.

-P *keyfilepw*
Specify the key database password. This password is required to access the encrypted information in the key database file, which may include one or more private keys. If a password stash file is associated with the key

database file, the password is obtained from the password stash file, and the **-P** parameter is not required. This parameter is ignored if neither **-Z** nor **-K** is specified.

-? Displays the usage.

-v Use verbose mode, with many diagnostics written to standard output.

-w *passwd* **|** **?**
Use ***passwd*** as the password for authentication. Use the ? to generate a password prompt. Using this prompt prevents your password from being visible through the **ps** command.

-Z Use a secure SSL connection to communicate with the LDAP server. The **-Z** option is only supported when the SSL component entry, as provided by IBM's GSKit, is installed.

Extended operations option: The **-op** extended-op option identifies the extended operation to be performed. The extended operation can be one of the following values:

- **cascrepl**: cascading control replication extended operation. The requested action is applied to the specified server and also passed along to all replicas of the given subtree. If any of these are forwarding replicas, they pass the extended operation along to their replicas. The operation cascades over the entire replication topology.

  -action quiesce | unquiesce | replnow | wait
  This is a required attribute that specifies the action to be performed.

  quiesce
  No further updates are allowed, except by replication.

  unquiesce
  Resume normal operation, client updates are accepted.

  replnow
  Replicate all queued changes to all replica servers as soon as possible, regardless of schedule.

  wait Wait for all updates to be replicated to all replicas.

  -rc *contextDn*
  This is a required attribute that specifies the root of the subtree.

  -timeout *secs*
  This is an optional attribute that if present, specifies the timeout period in seconds. If not present, or 0, the operation waits indefinitely.

  Example:

  ```
  ldapexop -op cascrepl -action -quiesce -rc "o=acme,c=us" -timeout 60
  ```

- **clearlog**: clear log file extended operation

  -log audit | bulkload | cli | slapd | ibmdiradm
  This is a required attribute that specifies the log file to be cleared.

  Example:

  ```
  ldapexop -op clearlog -log audit
  ```

- **controlqueue**: control queue extended operation

  -skip all | change-id
  This is a required attribute.

  - **all** indicates to skip all pending changes for this agreement.
  - **change-id** identifies the single change to be skipped. If the server is not currently replicating this change, the request fails.

-ra *agreementDn*
: This is a required attribute that specifies the DN of the replication agreement.

Examples:

```
ldapexop -op controlqueue -skip all -ra "cn=server3,
            ibm-replicaSubentry=master1-id,ibm-replicaGroup=default,
            o=acme,c=us"

ldapexop -op controlqueue -skip 2185 -ra "cn=server3,
            ibm-replicaSubentry=master1-id,ibm-replicaGroup=default,
            o=acme,c=us"
```

- **controlrepl**: control replication extended operation

  -action suspend | resume | replnow
  : This is a required attribute that specifies the action to be performed.

  -rc *contextDn* **| -ra** *agreementDn*
  : The **-rc** *contextDn* is the DN of the replication context. The action is performed for all agreements for this context. The **-ra** *agreementDn* is the DN of the replication agreement. The action is performed for the specified replication agreement.

  Example:

```
ldapexop -op controlrepl -action suspend -ra "cn=server3,
            ibm-replicaSubentry=master1-id,ibm-replicaGroup=default,
            o=acme,c=us"
```

- **getlogsize**: request log file size extended operation

  -log audit | bulkload | cli | slapd | ibmdiradm
  : This is a required attribute that specifies the log file to be queried The size of the log file, in lines, is written to standard output.

  Example:

```
ldapexop -op getlogsize -log slapd
2000 lines
```

- **quiesce**: quiesce or unquiesce subtree extended operation

  -rc *contextDn*
  : This is a required attribute that specifies the DN of the replication context (subtree) to be quiesced or unquiesced.

  -end
  : This is an optional attribute that if present, specifies to unquiesce the subtree. If not specified the default is to quiesce the subtree.

  Examples:

```
ldapexop -op quiesce -rc "o=acme,c=us"

ldapexop -op quiesce -end -rc "o=ibm,c=us"
```

- **readconfig**: reread configuration file extended operation

  -scope entire | single<*entry DN***><***attribute***>**
  : This is a required attribute.
    - **entire** indicates to reread the entire configuration file.
    - **single** means to read the single entry and attribute specified.

Examples:

```
ldapexop -op readconfig -scope entire

ldapexop -op readconfig -scope single "cn=configuration" ibm-slapdAdminPW
```

- **readlog**: request lines from log file extended operation

  -log audit | bulkload | cli | slapd | ibmdiradm
  This is a required attribute that specifies the log file to be queried.

  -lines <*first*><*last*> **| all**
  This is a required attribute that specifies either the first and last lines to be read from the file or all lines. Lines are numbered starting at 0. The specified lines are written to standard output.

  Examples:

```
ldapexop -op readlog -log audit -lines 10 20

ldapexop -op readlog -log slapd -lines all
```

Notes

If no DN arguments are provided, the **ldapdexop** command waits to read a list of DNs from standard input. To break out of the wait, use Ctrl+C or Ctrl+D.

SSL notes

To use the SSL-related functions associated with this utility, the SSL libraries and tools must be installed. The SSL libraries and tools are provided with IBM's Global Security Kit (GSKit), which includes security software developed by RSA Security Inc.

Note: For information regarding the use of 128-bit and triple DES encryption by LDAP applications, including the LDAP sample programs, see ″LDAP_SSL″ in the *IBM Directory C-Client SDK Programming Reference*. This section describes the steps required to build the sample programs and your applications so they can use SSL with the strongest encryption algorithms available.

See the makefile associated with the sample programs for more information on linking an LDAP application so that it has access to 128-bit and triple-DES encryption algorithms.

The content of a client's key database file is managed with the gsk6ikm utility. For more information on this Java utility, see "Using gsk6ikm" on page 52. The gsk6ikm 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 'trusted' certificates issued by one of the trusted CAs. The gsk6ikm utility can also be used to obtain a client certificate, so that client and server authentication can be performed.

If the LDAP servers accessed by the client use server authentication only, 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 or ldap_simple_bind_s. For example, if the LDAP server is using a high-assurance VeriSign certificate, you should obtain a CA certificate from VeriSign, import it into your key database file, and mark it as trusted. If the LDAP server is using a self-signed server certificate, the administrator of the LDAP server can supply you with a copy of the server's certificate request file. Import the certificate request file into your key database file and mark it as trusted.

If the LDAP servers accessed by the client use client and server authentication, it is necessary to:
- Define one or more trusted root certificates in the key database file. This allows the client to 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 or ldap_simple_bind_s.
- Create a key pair using gsk6ikm and request a client certificate from a CA. After receiving the signed certificate from the CA, store the certificate in the client key database file.

Diagnostics
Exit status is 0 if no errors occur. Errors result in a non-zero exit status and a diagnostic message being written to standard error.

See also
ldapadd, ldapchangepwd, ldapdelete, ldapmodify, ldapmodrdn, ldapsearch

ldapmodrdn

The LDAP modify-entry RDN tool

Synopsis
```
ldapmodrdn [-c] [-C charset] [-d debuglevel][-D binddn] [-h ldaphost]
[-i file] [-k] [-K keyfile] [-m mechanism] [-M] [-n]
[-N certificatename] [-O hopcount] [-p ldapport] [-P keyfilepw]
[-r] [-R] [-v] [-V] [-w passwd | ?] [-Z] [dn newrdn | [-i file]]
```

Description
ldapmodrdn is a command-line interface to the ldap_modrdn library call.

ldapmodrdn opens a connection to an LDAP server, binds, and modifies the RDN of entries. The entry information is read from standard input, from file through the use of the **- f** option, or from the command-line pair dn and rdn.

See LDAP Distinguished Names for information about RDNs (Relative Distinguished Names) and DNs (Distinguished Names).

To display syntax help for **ldapmodrdn**, type:
```
ldapmodrdn -?
```

Options

-c
Continuous operation mode. Errors are reported, but **ldapmodrdn** continues with modifications. Otherwise the default action is to exit after reporting an error.

-C *charset*
Specifies that the strings supplied as input to the **ldapmodrdn** utility are represented in a local character set, as specified by charset. Use **-C** *charset* to override the default, where strings must be supplied in UTF-8. See "IANA character sets supported by platform" on page 255 for the specific charset values that are supported for each operating system platform. Note that the supported values for charset are the same values supported for the charset tag that is optionally defined in Version 1 LDIF files.

-d *debuglevel*
: Set the LDAP debugging level to debuglevel. See "Server debug mode" on page 242 for information about debug levels.

-D *binddn*
: Use ***binddn*** to bind to the LDAP directory. binddn should be a string-represented DN.

-h *ldaphost*
: Specify an alternate host on which the ldap server is running.

-i *file*
: Read the entry modification information from file instead of from standard input or the command-line (by specifying rdn and newrdn). Standard input can be supplied from a file, as well ("< file").

-k
: Specifies to use server administration control.

-K *keyfile*
: Specify the name of the SSL key database file (with default extension of "kdb"). If the key database file is not in the current directory, specify the fully-qualified key database filename. If a key database filename is not specified, this utility will first look for the presence of the SSL_KEYRING environment variable with an associated filename. If the SSL_KEYRING environment variable is not defined, the default keyring file will be used, if present.

A default keyring file (that is, ldapkey.kdb) and the associated password stash file (that is, ldapkey.sth) are installed in the /lib directory under LDAPHOME, where LDAPHOME is the path to the installed LDAP support. LDAPHOME varies by operating system platform:

- AIX, Linux operating systems - /usr/ldap
- Solaris operating systems - /usr/IBMldapc
- HP-UX operating systems - /usr/IBMldap
- Windows operating systems - c:\Program Files\IBM\LDAP (Note: This is the default install location. The actual LDAPHOME is determined during installation.)

See *IBM Directory C-Client SDK Programming Reference* for more information about default key database files, and default Certificate Authorities.

If a keyring database file cannot be located, a "hard-coded" set of default trusted certificate authority roots is used. The key database file typically contains one or more certificates of certificate 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, see "Using gsk6ikm" on page 52. Also see the "SSL notes" on page 213 and "Secure Sockets Layer" on page 47 for more information about SSL and certificates.

This parameter effectively enables the **-Z** switch.

-m *mechanism*
: Use ***mechanism*** to specify the SASL mechanism to be used to bind to the server. The ldap_sasl_bind_s() API is used. The **-m** parameter is ignored if **-V 2** is set. If **-m** is not specified, simple authentication is used.

-M
: Manage referral objects as regular entries.

-n
: Show what would be done, but don't actually modify entries. Useful for debugging in conjunction with **-v**.

-N *certificatename*
Specify the label associated with the client certificate in the key database file. Note that if the LDAP server is configured to perform server authentication only, a client certificate is not required. If the LDAP server is configured to perform client and server Authentication, a client certificate might be required. ***certificatename*** is not required if a default certificate/private key pair has been designated as the default. Similarly, ***certificatename*** is not required if there is a single certificate/private key pair in the designated key database file. This parameter is ignored if neither **-Z** nor **-K** is specified.

-O *hopcount*
Specify ***hopcount*** to set the maximum number of hops that the client library takes when chasing referrals. The default hopcount is 10.

-p *ldapport*
Specify an alternate 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.

-P *keyfilepw*
Specify the key database password. This password is required to access the encrypted information in the key database file (which may include one or more private keys). If a password stash file is associated with the key database file, the password is obtained from the password stash file, and the **-P** parameter is not required. This parameter is ignored if neither **-Z** nor **-K** is specified.

-r Remove old RDN values from the entry. Default action is to keep old values.

-R Specifies that referrals are not to be automatically followed.

-v Use verbose mode, with many diagnostics written to standard output.

-V Specifies the LDAP version to be used by **ldapmodrdn** when it binds to the LDAP server. By default, an LDAP V3 connection is established. To explicitly select LDAP V3, specify **-V 3**. Specify **-V 2** to run as an LDAP V2 application. An application, like **ldapmodrdn**, selects LDAP V3 as the preferred protocol by using ldap_init instead of ldap_open.

-w *passwd* **|** **?**
Use ***passwd*** as the password for authentication. Use the ? to generate a password prompt. Using this prompt prevents your password from being visible through the **ps** command.

-Z Use a secure SSL connection to communicate with the LDAP server. The **-Z** option is only supported when the SSL component entry, as provided by IBM's GSKit, is installed.

dn newrdn
See the following section, "Input format for dn newrdn" for more information.

Input format for dn newrdn

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 **- i** flag is given) 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 and RDN pair.

Examples

Assuming that the file /tmp/entrymods exists and has the contents:

```
cn=Modify Me, o=University of Life, c=US
cn=The New Me
```

the command:

```
ldapmodrdn -r -i /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.

Notes

If entry information is not supplied from file through the use of the **-i** option (or from the command-line pair *dn* and *rdn*), the **ldapmodrdn** command waits to read entries from standard input. To break out of the wait, use Ctrl+C or Ctrl+D.

SSL notes

To use the SSL-related functions associated with this utility, the SSL libraries and tools must be installed. The SSL libraries and tools are provided with IBM's Global Security Kit (GSKit), which includes security software developed by RSA Security Inc.

Note: For information regarding the use of 128-bit and triple DES encryption by LDAP applications, including the LDAP sample programs, see ″LDAP_SSL″ in the *IBM Directory C-Client SDK Programming Reference*. This section describes the steps required to build the sample programs and your applications so they can use SSL with the strongest encryption algorithms available.

See the make file associated with the sample programs for more information on linking an LDAP application so that it has access to 128-bit and triple-DES encryption algorithms.

The content of a client's key database file is managed with the gsk6ikm utility. For more information on this Java utility, see "Using gsk6ikm" on page 52. The gsk6ikm 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 trusted CAs. The gsk6ikm utility can also be used to obtain a client certificate, so that client and server authentication can be performed.

If the LDAP servers accessed by the client use server authentication only, 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 or ldap_simple_bind_s. For example, if the LDAP server is using a high-assurance VeriSign certificate, you should obtain a CA certificate from VeriSign, import it into your key database file, and mark it as trusted. If the LDAP server is using a self-signed server certificate, the administrator of the LDAP server can supply you with a copy of the server's certificate request file. Import the certificate request file into your key database file and mark it as trusted.

If the LDAP servers accessed by the client use client and server authentication, it is necessary to:

- Define one or more trusted root certificates in the key database file. This allows the client to 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 or ldap_simple_bind_s.
- Create a key pair using gsk6ikm and request a client certificate from a CA. After receiving the signed certificate from the CA, receive the certificate into the client key database file.

Diagnostics

Exit status is 0 if no errors occur. Errors result in a non-zero exit status and a diagnostic message being written to standard error.

See also

ldapadd, ldapchangepwd, ldapdelete, ldapexop, ldapmodify, ldapsearch

ldapsearch

The LDAP search tool and sample program

Synopsis

```
ldapsearch [-a deref] [-A] [-b searchbase] [-B] [-C charset] [-d debuglevel]
[-D binddn] [-F sep] [-h ldaphost] [-i file] [-K keyfile] [-l timelimit] [-L]
[-m mechanism] [-M] [-n] [-N certificatename][-o attr_type] [-O maxhops]
[-p ldapport] [-P keyfilepw] [-q pagesize]  [-R] [-s scope ] [-t] [-T seconds]
[-v] [-V version] [-w passwd | ?] [-z sizelimit] [-Z] filter [-9 p] [-9 s]
[attrs...]
```

Description

ldapsearch is a command-line interface to the ldap_search library call.

ldapsearch opens a connection to an LDAP server, binds, and performs a search using the filter. The filter should conform to the string representation for LDAP filters (see ldap_search in the *IBM Directory Server Version 5.1 C-Client SDK Programming Reference* for more information on filters).

If **ldapsearch** finds one or more entries, the attributes specified by attrs are retrieved and the entries and values are printed to standard output. If no attrs are listed, all attributes are returned.

To display syntax help for **ldapsearch**, type `ldapsearch -?`.

Options

-a deref
: Specify how aliases dereferencing is done. 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.

-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 searchbase
: Use searchbase as the starting point for the search instead of the default. If

-b is not specified, this utility will examine the LDAP_BASEDN environment variable for a searchbase definition. If neither is set, the default base is set to ″″.

-B Do not suppress display of non-ASCII values. This is useful when dealing with values that appear in alternate characters sets such as ISO-8859.1. This option is implied by the **-L** option.

-C charset
Specifies that strings supplied as input to the ldapsearch utility are represented in a local character set (as specified by charset). String input includes the filter, the bind DN and the base DN. Similarly, when displaying data, **ldapsearch** converts data received from the LDAP server to the specified character set. Use ″-C charset″ to override the default, where strings must be supplied in UTF-8. Also, if the **-C** option and the **-L** option are both specified, input is assumed to be in the specified character set, but output from **ldapsearch** is always preserved in its UTF-8 representation, or a base-64 encoded representation of the data when non-printable characters are detected. This is the case because standard LDIF files only contain UTF-8 (or base-64 encoded UTF-8) representations of string data. See "IANA character sets supported by platform" on page 255 for the specific charset values that are supported for each operating system platform. Note that the supported values for charset are the same values supported for the charset tag that is optionally defined in Version 1 LDIF files.

-d debuglevel
Set the LDAP debugging level to debuglevel. See "Server debug mode" on page 242 for information about debug levels.

-D binddn
Use binddn to bind to the LDAP directory. binddn should be a string-represented DN (see LDAP Distinguished Names).

-e Display the LDAP library version information and exits.

-F sep Use sep as the field separator between attribute names and values. The default separator is `=', unless the **-L** flag has been specified, in which case this option is ignored.

-h ldaphost
Specify an alternate host on which the ldap server is running.

-i 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 ″-″ character, then the lines are read from standard input.

-K keyfile
Specify the name of the SSL key database file (with default extension of ″kdb″). If the key database file is not in the current directory, specify the fully-qualified key database filename. If a key database filename is not specified, this utility will first look for the presence of the SSL_KEYRING environment variable with an associated filename. If the SSL_KEYRING environment variable is not defined, the default keyring file will be used, if present.

A default keyring file (that is, ldapkey.kdb) and the associated password stash file (that is, ldapkey.sth) are installed in the /lib directory under LDAPHOME, where LDAPHOME is the path to the installed LDAP support. LDAPHOME varies by operating system platform:

- AIX, Linux operating systems - /usr/ldap
- Solaris operating systems - /usr/IBMldapc
- HP-UX operating systems - /usr/IBMldap
- Windows operating systems - c:\Program Files\IBM\LDAP (Note: This is the default install location. The actual LDAPHOME is determined during installation.)

See the ″Default Keyring and Password″ section of the LDAP_SSL API in the *IBM C-Client SDK Programming Reference* for more information about default key database files, and default Certificate Authorities.

If a keyring database file cannot be located, a ″hard-coded″ set of default trusted certificate authority roots is used. The key database file typically contains one or more certificates of certificate 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, see "Using gsk6ikm" on page 52. Also see the "SSL notes" on page 221 below and LDAP SSL APIs for more information about SSL and certificates.

This parameter effectively enables the **-Z** switch.

-l timelimit
Wait at most timelimit seconds for a search to complete.

-L Display search results in LDIF format. This option also turns on the **-B** option, and causes the **-F** option to be ignored.

-m mechanism
Use mechanism to specify the SASL mechanism to be used to bind to the server. The ldap_sasl_bind_s() API will be used. The **-m** parameter is ignored if **-V 2** is set. If **-m** is not specified, simple authentication is used.

-M Manage referral objects as regular entries.

-n Show what would be done, but don't actually modify entries. Useful for debugging in conjunction with **-v**.

-N certificatename
Specify the label associated with the client certificate in the key database file.

Note: If the LDAP server is configured to perform server authentication only, a client certificate is not required. If the LDAP server is configured to perform client and server Authentication, a client certificate might be required. *certificatename* is not required if a default certificate/private key pair has been designated as the default. Similarly, *certificatename* is not required if there is a single certificate/private key pair in the designated key database file. This parameter is ignored if neither **-Z** nor **-K** is specified.

-o *attr_type*
To specify an attribute to use for sort criteria of search results, you can use the -o (order) parameter. You can use multiple -o parameters to further define the sort order. In the following example, the search results are sorted first by surname (sn), then by given name, with the given name (givenname) being sorted in reverse (descending) order as specified by the prefixed minus sign (-):

```
-o sn -o -givenname
```

Thus, the syntax of the sort parameter is as follows:

```
[-]<attribute name>[:<matching rule OID>]
```

where

- `attribute name` is the name of the attribute you want to sort by.
- `matching rule OID` is the optional OID of a matching rule that you want to use for sorting.
- The minus sign (-) indicates that the results must be sorted in reverse order.
- The criticality is always critical.

The default ldapsearch operation is not to sort the returned results.

-O maxhops
Specify maxhops to set the maximum number of hops that the client library takes when chasing referrals. The default hopcount is 10.

-p ldapport
Specify an alternate 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.

-P keyfilepw
Specify the key database password. This password is required to access the encrypted information in the key database file (which may include one or more private keys). If a password stash file is associated with the key database file, the password is obtained from the password stash file, and the **-P** parameter is not required. This parameter is ignored if neither **-Z** nor **-K** is specified.

-q *pagesize*
To specify paging of search results, two new parameters can be used: -q (query page size), and -T (time between searches in seconds). In the following example, the search results return a page (25 entries) at a time, every 15 seconds, until all the results for that search are returned. The ldapsearch client handles all connection continuation for each paged results request for the life of the search operation.

```
 -q 25  -T 15
```

If the -v (verbose) parameter is specified, ldapsearch lists how many entries have been returned so far, after each page of entries returned from the server, for example, **30 total entries have been returned.**

Multiple -q parameters are enabled such that you can specify different page sizes throughout the life of a single search operation. In the following example, the first page is 15 entries, the second page is 20 entries, and the third parameter ends the paged result/search operation:

```
-q 15 -q 20 -q 0
```

In the following example, the first page is 15 entries, and all the rest of the pages are 20 entries, continuing with the last specified **-q** value until the search operation completes:

```
-q 15 -q 20
```

The default ldapsearch operation is to return all entries in a single request. No paging is done for the default ldapsearch operation.

-R Specifies that referrals are not to be automatically followed.

-s scope
: Specify the scope of the search. scope should be one of base, one, or sub to specify a base object, one-level, or subtree search. The default is sub.

-t Write retrieved values to a set of temporary files. This is useful for dealing with non-ASCII values such as jpegPhoto or audio.

-T *seconds*
: Time between searches (in seconds). The **-T** option is only supported when the **-q** option is specified.

-v Use verbose mode, with many diagnostics written to standard output.

-V Specifies the LDAP version to be used by ldapmodify when it binds to the LDAP server. By default, an LDAP V3 connection is established. To explicitly select LDAP V3, specify ″-V 3″. Specify ″-V 2″ to run as an LDAP V2 application. An application, like ldapmodify, selects LDAP V3 as the preferred protocol by using ldap_init instead of ldap_open.

-w *passwd* **| ?**
: Use ***passwd*** as the password for authentication. Use the ? to generate a password prompt. Using this prompt prevents your password from being visible through the **ps** command.

-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.

-Z Use a secure SSL connection to communicate with the LDAP server. The **-Z** option is only supported when the SSL component entry, as provided by IBM's GSKit, is installed.

-9 p Sets criticality for paging to false. The search is handled without paging.

-9 s Sets criticality for sorting to false. The search is handled without sorting.

filter Specifies a string representation of the filter to apply in the search. Simple filters can be specified as attributetype=attributevalue. More complex filters are specified using a prefix notation according to the following Backus Naur Form (BNF):

```
<filter> ::='('<filtercomp>')'
<filtercomp> ::= <and>|<or>|<not>|<simple>
<and> ::= '&' <filterlist>
<or> ::= '|' <filterlist>
<not> ::= '!' <filter>
<filterlist> ::= <filter>|<filter><filtertype>
<simple> ::= <attributetype><filtertype>
<attributevalue>
<filtertype> ::= '='|'~='|'<='|'>='
```

The ′~=′ construct is used to specify approximate matching. The representation for <*attributetype*> and <*attributevalue*> are as described in ″RFC 2252, LDAP V3 Attribute Syntax Definitions″. In addition, <*attributevalue*> can be a single * to achieve an attribute existence test, or can contain text and asterisks (*) interspersed to achieve substring matching.

For example, the filter ″mail=*″ finds any entries that have a mail attribute. The filter ″mail=*@student.of.life.edu″ finds any entries that have a mail attribute ending in the specified string. To put parentheses in a filter, escape them with a backslash (\) character.

Note: A filter like "`cn=Bob *`", where there is a space between Bob and the asterisk (*), matches ″Bob Carter″ but not ″Bobby Carter″ in IBM Directory. The space between ″Bob″ and the wildcard character (*) affects the outcome of a search using filters.

See ″RFC 2254, A String Representation of LDAP Search Filters″ for a more complete description of allowable filters.

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 `=′ character. 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

The following command:

```
 ldapsearch "cn=john doe" cn telephoneNumber
```

performs a subtree search (using the default search base) for entries with a commonName of ″john doe″. The commonName and telephoneNumber values is retrieved and printed to standard output. The output might look something like this if two entries are found:

```
  cn=John E Doe, ou="College of Literature, Science, and the Arts",
ou=Students, ou=People, o=University of Higher Learning, c=US

  cn=John Doe

  cn=John Edward Doe

  cn=John E Doe 1

  cn=John E Doe

  telephoneNumber=+1 313 555-5432



  cn=John B Doe, ou=Information Technology Division,
ou=Faculty and Staff, ou=People, o=University of Higher Learning, c=US

  cn=John Doe

  cn=John B Doe 1
  cn=John B Doe

  telephoneNumber=+1 313 555-1111
```

The command:

```
ldapsearch -t "uid=jed" jpegPhoto audio
```

performs a subtree search using the default search base for entries with user id of ″jed″. 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=John E Doe, ou=Information Technology Division,

  ou=Faculty and Staff,

  ou=People, o=University of Higher Learning, c=US

  audio=/tmp/ldapsearch-audio-a19924

  jpegPhoto=/tmp/ldapsearch-jpegPhoto-a19924
```

This command:

```
ldapsearch -L -s one -b "c=US" "o=university*" o description
```

will perform a one-level search at the c=US level for all organizations whose organizationName begins with university. Search results will be displayed in the LDIF format (see LDAP Data Interchange Format). The organizationName and description attribute values will be 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


 ...
```

This command:

```
ldapsearch -b "c=US" -o ibm-slapdDN "objectclass=person" ibm-slapdDN
```

performs a subtree level search at the c=US level for all persons. This special attribute is new in 5.1 and when used for sorted searches, the search results are sorted by the string representation of the Distinguished Name (DN). The output might look something like this:

```
cn=Al Edwards,ou=Widget Division,ou=Austin,o=IBM,c=US

cn=Al Garcia,ou=Home Entertainment,ou=Austin,o=IBM,c=US

cn=Amy Nguyen,ou=In Flight Systems,ou=Austin,o=IBM,c=US

cn=Arthur Edwards,ou=Widget Division,ou=Austin,o=IBM,c=US

cn=Becky Garcia,ou=In Flight Systems,ou=Austin,o=IBM,c=US

cn=Ben Catu,ou=In Flight Systems,ou=Austin,o=IBM,c=US

cn=Ben Garcia Jr,ou=Home Entertainment,ou=Austin,o=IBM,c=US

cn=Bill Keller Jr.,ou=In Flight Systems,ou=Austin,o=IBM,c=US

cn=Bob Campbell,ou=In Flight Systems,ou=Austin,o=IBM,c=US
```

SSL notes

To use the SSL-related functions associated with this utility, the SSL libraries and tools must be installed. The SSL libraries and tools are provided with IBM's Global Security Kit (GSKit), which includes security software developed by RSA Security Inc.

Note: For information regarding the use of 128-bit and triple DES encryption by LDAP applications, including the LDAP sample programs, see SSL Usage Notes. This section describes the steps required to build the sample programs (and your applications) so they can use SSL with the strongest available encryption algorithms.

See the make file associated with the sample programs for more information on linking an LDAP application so that it has access to 128-bit and triple-DES encryption algorithms.

The contents of a client's key database file is managed with the gsk6ikm utility. For more information on this Java utility, see "Using gsk6ikm" on page 52. The gsk6ikm 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. The gsk6ikm utility can also be used to obtain a client certificate, so that client and server authentication can be performed.

If the LDAP servers accessed by the client use server authentication only, 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 or ldap_simple_bind_s (see the LDAP_Bind APIs).

For example, if the LDAP server is using a high-assurance VeriSign certificate, you should obtain a CA certificate from VeriSign, import it into your key database file, and mark it as trusted. If the LDAP server is using a self-signed server certificate, the administrator of the LDAP server can supply you with a copy of the server's certificate request file. Import the certificate request file into your key database file and mark it as trusted.

If the LDAP servers accessed by the client use client and server authentication, it is necessary to:

- Define one or more trusted root certificates in the key database file. This allows the client to 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 or ldap_simple_bind_s (see the LDAP_Bind APIs).
- Create a key pair using gsk6ikm and request a client certificate from a CA. After receiving the signed certificate from the CA, receive the certificate into the client key database file.

Diagnostics

Exit status is 0 if no errors occur. Errors result in a non-zero exit status and a diagnostic message being written to standard error.

See also

ldapadd, ldapchangepwd, ldapdelete, ldapexop, ldapmodify, ldapmodrdn

ibmdirctl

The administration daemon control program.

Note: Only the administrator may use this utility.

Synopsis

```
ibmdirctl [-D adminDN] [-h hostname] [-K keyfile] [ -N key_name ]
          [-p port] [-v] [-w adminPW | ?] [-Z] [-?]
          command -- [ibmslapd options]
```

where *command* is {start|stop|restart|status}

Description

The administration daemon control program, **ibmdirctl**, is used to start, stop, restart or query the status of the IBM Directory Server. If ibmslapd options are requested, they must be preceded by the **--**.

To display syntax help for **ibmdirctl**, type `ibmdirctl -?`.

Options

-D adminDN
: Use adminDNto bind to the LDAP directory. The adminDN should be a string-represented DN (see LDAP Distinguished Names).

-h hostname
: Specify an alternate host on which the ldap server and the admin daemon are running.

-K keyfile
: Specifies the file to use for keys.

-N key_name
: Specifies the private key name to use in keyfile.

-p port
: Specify an alternate TCP port where the admin daemon is listening. The default LDAP port is 3538.

-v Specifies to run in verbose mode.

-w *adminPW* **|** **?**
: Use ***adminPW*** as the password for authentication. Use the ? to generate a password prompt. Using this prompt prevents your password from being visible through the **ps** command.

-? Displays the help screen.

command

- **start** - Start the server.
- **stop** - Stop the server.
- **restart** - Stop then start the server.
- **status** - Query the status the server.

Note: The **stop** command may be issued directly to the ldap server.

-- ibmslapd options
: The **ibmslapd** options are any options the **ibmslapd** process takes at startup time, typically:

- **-a** | **-A** - Start the server in configuration only mode.
- **-n** | **-N** - Do not start the server, if the server is unable to start with the database backends (no configuration only mode).

Notes:

1. If **ibmslapd** options are requested, they must be preceded by the **--**.
2. The **ibmslapd** options are ignored if the **stop** command is issued.

Example

To start the server in configuration only mode issue the command:

```
ibmdirctl -h mymachine -D myDN -w mypassword -p 3538 start -- -a
```

To stop the server issue the command:

```
ibmdirctl -h mymachine -D myDN -w mypassword -p 3538 stop
```

Server utilities

This sections describes the server utilities.

Notes:

1. Except for **ldif** and **db2ldif**, the server must be stopped before using the server utilities.
2. Ensure that no applications are attached to the directory database. If there are applications attached, none of the server utilities will run.

ldif utility

The LDAP Data Interchange Format (LDIF) tool **ldif** is a shell-accessible utility that converts arbitrary data values to LDIF. It reads input values from standard input and produces records appropriate for use in an LDIF file.

Synopsis:

ldif [-b]<*attrname*>

Command Line Option:

All options are case sensitive.

-b
Input is a single raw binary value. Output is a base64 encoded value.

<*attrname*>
The name of the attribute for which values are to be converted. Without the **-b** option, **ldif** considers each line of standard input to be a separate value of the attribute.

See Appendix D, "LDAP data interchange format (LDIF)" on page 253 for more information about LDIF.

Examples

To find the LDIF format for the attribute sn (surname) with a value of smith at a command line enter:

1. Enter `ldif sn`
2. Enter `smith`
3. This returns `sn: smith`
4. Press **Ctrl C** to exit.

Using the -b option:

1. Enter `ldif -b sn`
2. Enter `smith`
3. Press **Ctrl C** to process.
4. This returns `sn:: c21pdGgNCg==`

ldif2db utility

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

Notes:

1. The server must be stopped before using the server import utilities.
2. Ensure that no applications are attached to the directory database. If there are applications attached, none of the server utilities will run.
3. If you have installed a 5.1 server over a 4.1 server, you must initially start the server before using the **ldif2db** utility so that one-time migration processing can be completed.

Synopsis:

ldif2db -i <*inputfile*> [-f <*configurationfile*>] [-r yes|no] | -?

Command line options:

All options are case insensitive.

-i <*inputfile*>
Specify the name of the LDIF input file, containing directory entries in LDIF format. This option is required. If the file is not in the current directory, a full path and file name must be specified.

-f <*configurationfile*>
Use this option to specify the slapd configuration file.

-r [yes|no]
Specifies whether to replicate. The default is **yes** which means entries are put into the Change table and are replicated when the server restarts.

-? Displays the usage of the command.

All other command line inputs result in a syntax error message, after which the correct syntax is displayed.

Note: When records are added using **ldif2db**, the master server should be stopped and then restarted immediately.

bulkload utility

The **bulkload** utility is used to load the directory data from an LDIF file. It is a faster alternative to **ldif2db** and is available for bulk-loading large amounts of data in LDIF format.

Notes:

1. The server must be stopped before using the server import utilities.
2. Ensure that no applications are attached to the directory database. If there are applications attached, none of the server utilities will run.
3. All bulkload environment variables are deprecated in IBM Directory Server Version 5.1. The ACLCHECK, ACTION, LDAPIMPORT, SCHEMACHECK, and STRING_DELIMITER environment variables are replaced with the command line options -A, -a, -L, -S, -s respectively. The command line switches are now **case sensitive**.
4. To run the bulkload utility you must have dbadm or sysadm privilege. If you use a Windows system, you must also run the bulkload utility within the DB2 command line interpreter (CLI). To start the DB2 CLI, click **Start->Run** and type **db2cmd**.
5. If archival logging is enabled in DB2, the bulkload utility will fail. Make sure archival logging is disabled before using the bulkload utility.
   ```
   update database configuration for ldapdb2 using LOGRETAIN OFF USEREXIT OFF
   ```

Synopsis:

bulkload -i <*ldiffile*>[-a <parse_and_load|parseonly|loadonly>] [-A <yes|no>] [-c | -C<yes|no>] [-d <*number*>] [-E <*number*>] [-f <*configurationfile*>] [-I <yes|no>] [-L <*path*>] [-n | -N] [-?][-p | -P <yes|no>] [-s <*character*>] [-R <yes|no>] [-S <yes|no|only>] [-v] [-x|-X <yes|no>]

Command line options:

-i <*ldiffile*>
: Specifies the name of the input file containing the LDIF data to be loaded into the directory. It might include a path. The file /usr/ldap/examples/sample.ldif contains some sample data in the correct format.

-a <parse_and_load|parseonly|loadonly>
: Specifies the load action mode.

-A <yes|no>
: Specifies whether to process the ACL information contained in the LDIF file. The default is **yes**. The **no** parameter loads the default acls.

-c | -C <yes|no>
: Allows you to skip index recreation. For example, if you are running successive bulkloads and you want to skip recreation between loads, you can postpone index creation until the last bulkload. Issue the final bulkload with **-c yes**.

-d <*number*>
: Use the **-d** to set the level of the debug mask and to turn debug on. Use this option to find out the data records that might have a problem and cause parsing errors. See "Server debug mode" on page 242 for information about debug levels.

  Note: Ensure that the **ldtrc** utility is on before using the **-d** option, otherwise no messages are displayed. Issue the command `ldtrc on`.

-E <*number*>
: Specifies the number limit for parsing errors reported. When the limit is reached the **bulkload** command exits. The default is infinity.

-f <*configurationfile*>
: Use this option to specify the slapd configuration file.

-I <yes|no>
: Specifies whether to drop the indexes before the load. The default is **no**.

-L <*path*>
: Specifies the directory used for storing temporary data. The default path for this temporary storage is:
  - AIX operating systems /tmp/ldapimport
  - Windows operating systems c:\tmp\ldapimport
  - Linux, Solaris, and HP operating systems /var/ldap/ldapimport

-n | -N
: Specifies that the load is nonrecoverable.

-? Requests the bulkload syntax help message.

-p | -P <yes|no>
: Specifies whether to generate password policy attributes for entries containing the attribute userpassword.

-R <yes|no>
: Specifies whether to remove the directory which was used for temporary data. Default is **yes**.

Note: This directory is the default directory or the one specified by the **-L** parameter.

-s <*character*>
Specifies the string delimiting character used for importing

Note: **Bulkload** might fail to load LDIF files that contain certain UTF-8 characters. This is because of a problem with the DB2 LOAD tool when parsing the default **bulkload** string delimiter, vertical bar (|) in multi-byte character sets. Reassign the string delimiter to $.

```
bulkload -i <ldiffile> -s $
```

-S <yes|no|only>
Verifies that individual directory entries are valid based on the object class definitions and attribute type definitions found in the configuration files.

Schema checking verifies that all object classes and attributes have been defined, that all attributes specified for each entry comply with the list of "required" and "allowed" attributes in the object class definition, and that binary attribute values are in the correct 64-bit encoded form.

yes Schema checking is done on the data, before adding it to the directory.

no No schema checking is done on the data before adding it to the directory. This provides much faster performance. This option assumes that the data in the input file is valid. This is the default option.

only Schema checking is done on the data, but it is not added to the directory. This option provides the most feedback and error reporting.

The recommended approach is to use the **-S only** option first to validate the data, and thereafter to use the default **-S no** whenever loading the data into the directory.

-v Specifies verbose mode. This option gives you the greatest amount of detail.

-x|-X <yes|no>
Specifies whether to translate entry data to database code page. Default is **no**.

Note: This parameter is only necessary when using a non-UTF-8 database.

For better performance the **bulkload** tool assumes that the data in the input file is correct or that the data has been checked in an earlier loading. The **bulkload** tool can, however, perform some basic checks on the input data.

The **bulkload** utility cannot run while the directory server (**slapd**) is running.

In addition to the disk space required for data storage in the local database directory, the **bulkload** tool requires temporary storage for data manipulation before inserting the data into the database. The default path for this temporary storage is platform specific. See the **-L** option description for the path names. You can change the path using the **-L** option:

```
bulkload -i <ldiffile> -L /newpath
```

You must have write permission to this directory. You need temporary storage at least 2.5 times the size of the LDIF file that is available in the ldapimport directory. You still might need additional temporary storage depending on your data.

If you receive an error like the following:

```
SQL3508N Error in accessing a file of type "SORTDIRECTORY" during load
or load query.  Reason code: "2".  Path: "/u/ldapdb2/sqllib/tmp/".
```

you need to set the environment variable DB2SORTTMP to a directory (or directories) in a file system with more space to be utilized during the bulkload. Multiple directories can be specified separated by a comma (,) as in:

```
export DB2SORTTMP=/sortdir1,/sortdir2
```

When running **bulkload**, inspect the output messages carefully. If errors occur during execution, the directory might be incompletely populated. You might need to drop all the LDAP tables, or drop the database (recreate an empty database), and start over. If this happens, no data was added to the directory, and the **bulkload** must be attempted again. In addition, you might lose data when you drop all the LDAP tables.

The file /usr/ldap/examples/sample.ldif includes sample data. You can use the data in the file to experiment with populating a directory using the bulkload tool, or you can use the **ldif2db** command line utility. However, the **ldif2db** utility might be considerably slower than the **bulkload** utility for large amounts of data.

For performance reasons, the **bulkload** tool does not check for duplicate entries. Ensure that your input LDIF file does not contain duplicate entries. If any duplicates exist, remove the duplicate entries.

If **bulkload** fails at the DB2 LOAD phase, see the db2load.log file for the reasons. This log file is located on the Windows operating systems in c:\tmp\ldapimport, on AIX operating systems in /tmp/ldapimport, and on Linux , Solaris, and HP operating systems in /var/ldap/ldapimport. If the **-L** option was specified, find the file in the directory defined by the **-L** option. Correct the problem and rerun **bulkload**. **Bulkload** reloads the files from the last successful load consistency point.

When **bulkload** fails, the recovery information is stored in *<installation directory>*/etc/bulkload_status. This file is not removed until all of the data is loaded successfully. This insures the data integrity of the directory. If you decide to reconfigure the database and start over, the bulkload_status file needs to be removed manually or **bulkload** still tries to recover from the last successful load point.

db2ldif utility

This program is used to dump entries from a directory stored in a relational database into a text file in LDAP Directory Interchange Format (LDIF).

Note: This utility can be run at anytime, the server does not need to be stopped.

Synopsis:

db2ldif -o <*outputfile*> [-f <*configfile*>] [-s <*subtree DN*>[-x]] [-p on|off] [-l] | [-?]

Command line options:

All options are case sensitive.

-f <*configfile*>
: Use this option to specify the slapd configuration file.

-l
: Exports all suffixes, except the cn=pwdpolicy suffix, in addition to the cn=localhost subtree. This option cannot be used with the **-s** option.

-o <*outputfile*>
: Specifies the LDIF output file to contain the directory entries in LDIF. All entries from the specified subtree are written in LDIF to the output file. This option is required. If the file is not in the current directory, a full path and file name must be specified.

-p on|off
: Exports all suffixes, except cn=localhost subtree,in addition to the cn=pwdpolicy suffix . The default setting is **off**. This option cannot be used with the **-s** option.

-?
: Displays the usage of the command.

-s <*subtree DN*> **[-x]**
: 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 out. If this option is not specified, all directory entries stored in the database are written to the output file based on the suffixes specified in the configuration file. When the **-x** option is specified it means to exclude the subtree specified by the **-s** option. This option cannot be used with the **-l** or **-p** options.

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

ldapdiff

The LDAP replica synchronization tool

Synopsis

```
ldapdiff -b baseDN -sh host -ch host [-a] [-C countnumber]
[-cD dn] [-cK keyStore]  [-cw password] -[cN keyStoreType]
[-cp port] [-cP keyStorePwd] [-ct trustStoreType]  [-cT trustStore]
[-cY trustStorePwd] [-cZ] [-F] [-L filename] [-sD dn] [-sK keyStore]
[-sw password] -[sN keyStoreType] [-sp port] [-sP keyStorePwd]
[-st trustStoreType]  [-sT trustStore] [-sY trustStorePwd] [-sZ] [-v]
```

or

```
ldapdiff -S -sh host -ch host [-a] [-C countnumber][-cD dn]
[-cK keyStore] [-cw password] -[cN keyStoreType] [-cp port]
[-cP keyStorePwd] [-ct trustStoreType]  [-cT trustStore]
[-cY trustStorePwd] [-cZ] [-L filename] [-sD dn]
[-sK keyStore] [-sw password] [-sN keyStoreType] [-sp port]
[-sP keyStorePwd] [-st trustStoreType]  [-sT trustStore]
[-sY trustStorePwd] [-sZ] [-v]
```

Description

This tool synchronizes a replica server with its master. To display syntax help for **ldapdiff**, type:

```
ldapdiff -?
```

Options

The following options apply to the **ldapdiff** command. There are two subgroupings that apply specifically to either the supplier server or the consumer server.

-a
Specifies to use server administration control for writes to a read-only replica.

-b *baseDN*
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.

-C *countnumber*
Counts the number of entries to fix. If more than the specified number of mismatches are found, the tool exits.

-F
This is the fix option. If specified, content on the consumer replica is modified to match the content of the supplier server. This cannot be used if the **-S** is also specified.

-L
If the **-F** option is not specified, use this option to generate an LDIF file for output. The LDIF file can be used to update the consumer to eliminate the differences.

-S
Specifies to compare the schema on both of the servers.

-v
Use verbose mode, with many diagnostics written to standard output.

Options for a replication supplier: The following options apply to the consumer server and are denoted by an initial 's' in the option name.

-sD *dn*
Use ***dn*** to bind to the LDAP directory. ***dn*** is a string-represented DN.

-sh *host*
Specifies the host name.

-sK *keyStore*
Specify the name of the SSL key database file with default extension of **kdb**. If the key database file is not in the current directory, specify the fully-qualified key database filename. If a key database filename is not specified, this utility will first look for the presence of the SSL_KEYRING environment variable with an associated filename. If the SSL_KEYRING environment variable is not defined, the default keyring file will be used, if present.

A default keyring file that is, ldapkey.kdb, and the associated password stash file that is, ldapkey.sth, are installed in the /lib directory under LDAPHOME, where LDAPHOME is the path to the installed LDAP support. LDAPHOME varies by operating system platform:

- AIX operating systems - /usr/ldap
- HP-UX operating systems - /usr/IBMldap
- Linux operating systems - /usr/ldap
- Solaris operating systems - /usr/IBMldapc
- Windows operating systems - c:\Program Files\IBM\LDAP

Note: This is the default install location. The actual LDAPHOME is determined during installation.

See *IBM Directory C-Client SDK Programming Reference* for more information about default key database files, and default Certificate Authorities.

If a keyring database file cannot be located, a "hard-coded" set of default trusted certificate authority roots is used. The key database file typically contains one or more certificates of certificate 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, see "Using gsk6ikm" on page 52. Also see the "SSL notes" on page 235 and "Secure Sockets Layer" on page 47 for more information about SSL and certificates.

This parameter effectively enables the **-sZ** switch.

-sN *keyStoreType*
: Specify the label associated with the client certificate in the key database file. If the LDAP server is configured to perform server authentication only, a client certificate is not required. If the LDAP server is configured to perform client and server Authentication, a client certificate might be required. ***keyStoreType*** is not required if a default certificate/private key pair has been designated as the default. Similarly, ***keyStoreType*** is not required if there is a single certificate/private key pair in the designated key database file. This parameter is ignored if neither **-sZ** nor **-sK** is specified.

-sp *ldapport*
: Specify an alternate TCP port where the ldap server is listening. The default LDAP port is 389. If **-sp** is not specified and **-sZ** is specified, the default LDAP SSL port 636 is used.

-sP *keyStorePwd*
: Specify the key database password. This password is required to access the encrypted information in the key database file, which may include one or more private keys. If a password stash file is associated with the key database file, the password is obtained from the password stash file, and the **-sP** parameter is not required. This parameter is ignored if neither **-sZ** nor **-sK** is specified.

-st *trustStoreType*
: Specify the label associated with the client certificate in the trust database file. If the LDAP server is configured to perform server authentication only, a client certificate is not required. If the LDAP server is configured to perform client and server Authentication, a client certificate might be required. ***trustStoreType*** is not required if a default certificate/private key pair has been designated as the default. Similarly, ***trustStoreType*** is not required if there is a single certificate/private key pair in the designated key database file. This parameter is ignored if neither **-sZ** nor **-sT** is specified.

-sT *trustStore*
: Specify the name of the SSL trust database file with default extension of **tdb**. If the trust database file is not in the current directory, specify the fully-qualified trust database filename. If a trust database filename is not specified, this utility will first look for the presence of the SSL_KEYRING

environment variable with an associated filename. If the SSL_KEYRING environment variable is not defined, the default keyring file will be used, if present.

A default keyring file that is, ldapkey.tdb, and the associated password stash file that is, ldapkey.sth, are installed in the /lib directory under LDAPHOME, where LDAPHOME is the path to the installed LDAP support. LDAPHOME varies by operating system platform:

- AIX operating systems - /usr/ldap
- HP-UX operating systems - /usr/IBMldap
- Linux operating systems - /usr/ldap
- Solaris operating systems - /usr/IBMldapc
- Windows operating systems - c:\Program Files\IBM\LDAP

  Note: This is the default install location. The actual LDAPHOME is determined during installation.

See *IBM Directory C-Client SDK Programming Reference* for more information about default key database files, and default Certificate Authorities.

If a keyring database file cannot be located, a ″hard-coded″ set of default trusted certificate authority roots is used. The key database file typically contains one or more certificates of certificate 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, see “Using gsk6ikm” on page 52. Also see the “SSL notes” on page 235 and “Secure Sockets Layer” on page 47 for more information about SSL and certificates.

This parameter effectively enables the **-sZ** switch.

-sw *password* **| ?**
Use ***password*** as the password for authentication. Use the ? to generate a password prompt. Using this prompt prevents your password from being visible through the **ps** command.

-sY The password for the trusted database.

-sZ Use a secure SSL connection to communicate with the LDAP server. The **-Z** option is only supported when the SSL component entry, as provided by IBM’s GSKit, is installed.

Options for a replication consumer: The following options apply to the consumer server and are denoted by an initial ’c’ in the option name.

-cD *dn* Use ***dn*** to bind to the LDAP directory. ***dn*** is a string-represented DN.

-ch *host*
Specifies the host name.

-cK *keyStore*
Specify the name of the SSL key database file with default extension of **kdb**. If the key database file is not in the current directory, specify the fully-qualified key database filename. If a key database filename is not specified, this utility will first look for the presence of the SSL_KEYRING environment variable with an associated filename. If the SSL_KEYRING environment variable is not defined, the default keyring file will be used, if present.

A default keyring file that is, ldapkey.kdb, and the associated password stash file that is, ldapkey.sth, are installed in the /lib directory under LDAPHOME, where LDAPHOME is the path to the installed LDAP support. LDAPHOME varies by operating system platform:

- AIX operating systems - /usr/ldap
- HP-UX operating systems - /usr/IBMldap
- Linux operating systems - /usr/ldap
- Solaris operating systems - /usr/IBMldapc
- Windows operating systems - c:\Program Files\IBM\LDAP

  Note: This is the default install location. The actual LDAPHOME is determined during installation.

See *IBM Directory C-Client SDK Programming Reference* for more information about default key database files, and default Certificate Authorities.

If a keyring database file cannot be located, a "hard-coded" set of default trusted certificate authority roots is used. The key database file typically contains one or more certificates of certificate 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, see "Using gsk6ikm" on page 52. Also see the "SSL notes" on page 235 and "Secure Sockets Layer" on page 47 for more information about SSL and certificates.

This parameter effectively enables the **-cZ** switch.

-cN *keyStoreType*
: Specify the label associated with the client certificate in the key database file. If the LDAP server is configured to perform server authentication only, a client certificate is not required. If the LDAP server is configured to perform client and server Authentication, a client certificate might be required. ***keyStoreType*** is not required if a default certificate/private key pair has been designated as the default. Similarly, ***keyStoreType*** is not required if there is a single certificate/private key pair in the designated key database file. This parameter is ignored if neither **-cZ** nor **-cK** is specified.

-cp *ldapport*
: Specify an alternate TCP port where the ldap server is listening. The default LDAP port is 389. If **-cp** is not specified and **-cZ** is specified, the default LDAP SSL port 636 is used.

-cP *keyStorePwd*
: Specify the key database password. This password is required to access the encrypted information in the key database file, which may include one or more private keys. If a password stash file is associated with the key database file, the password is obtained from the password stash file, and the **-cP** parameter is not required. This parameter is ignored if neither **-cZ** nor **-cK** is specified.

-ct *trustStoreType*
: Specify the label associated with the client certificate in the trust database file. If the LDAP server is configured to perform server authentication only, a client certificate is not required. If the LDAP server is configured to perform client and server Authentication, a client certificate might be required. ***trustStoreType*** is not required if a default certificate/private key

pair has been designated as the default. Similarly, ***trustStoreType*** is not required if there is a single certificate/private key pair in the designated key database file. This parameter is ignored if neither **-cZ** nor **-cT** is specified.

-cT *trustStore*
: Specify the name of the SSL trust database file with default extension of **tdb**. If the trust database file is not in the current directory, specify the fully-qualified trust database filename. If a trust database filename is not specified, this utility will first look for the presence of the SSL_KEYRING environment variable with an associated filename. If the SSL_KEYRING environment variable is not defined, the default keyring file will be used, if present.

  A default keyring file that is, ldapkey.tdb, and the associated password stash file that is, ldapkey.sth, are installed in the /lib directory under LDAPHOME, where LDAPHOME is the path to the installed LDAP support. LDAPHOME varies by operating system platform:

  - AIX operating systems - /usr/ldap
  - HP-UX operating systems - /usr/IBMldap
  - Linux operating systems - /usr/ldap
  - Solaris operating systems - /usr/IBMldapc
  - Windows operating systems - c:\Program Files\IBM\LDAP

    Note: This is the default install location. The actual LDAPHOME is determined during installation.

  See *IBM Directory C-Client SDK Programming Reference* for more information about default key database files, and default Certificate Authorities.

  If a keyring database file cannot be located, a "hard-coded" set of default trusted certificate authority roots is used. The key database file typically contains one or more certificates of certificate 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, see "Using gsk6ikm" on page 52. Also see the "SSL notes" on page 235 and "Secure Sockets Layer" on page 47 for more information about SSL and certificates.

  This parameter effectively enables the **-cZ** switch.

-cw *password* **|** **?**
: Use ***password*** as the password for authentication. Use the ? to generate a password prompt. Using this prompt prevents your password from being visible through the **ps** command.

-cY
: The password for the trusted database.

-cZ
: Use a secure SSL connection to communicate with the LDAP server. The **-cZ** option is only supported when the SSL component entry, as provided by IBM's GSKit, is installed.

Examples

```
ldapdiff -b <baseDN> -sh <supplierhostname> -ch <consumerhostname> [options]
```

or

```
ldapdiff -S > -sh <supplierhostname> -ch <consumerhostname> [options]
```

Notes

If no DN arguments are provided, the **ldapdiff** command waits to read a list of DNs from standard input. To break out of the wait, use Ctrl+C or Ctrl+D.

SSL notes

To use the SSL-related functions associated with this utility, the SSL libraries and tools must be installed. The SSL libraries and tools are provided with IBM's Global Security Kit (GSKit), which includes security software developed by RSA Security Inc.

Note: For information regarding the use of 128-bit and triple DES encryption by LDAP applications, including the LDAP sample programs, see ″LDAP_SSL″ in the *IBM Directory C-Client SDK Programming Reference*. This section describes the steps required to build the sample programs and your applications so they can use SSL with the strongest encryption algorithms available.

See the makefile associated with the sample programs for more information on linking an LDAP application so that it has access to 128-bit and triple-DES encryption algorithms.

The content of a client's key database file is managed with the gsk6ikm utility. For more information on this Java utility, see "Using gsk6ikm" on page 52. The gsk6ikm 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 'trusted' certificates issued by one of the trusted CAs. The gsk6ikm utility can also be used to obtain a client certificate, so that client and server authentication can be performed.

If the LDAP servers accessed by the client use server authentication only, 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 or ldap_simple_bind_s. For example, if the LDAP server is using a high-assurance VeriSign certificate, you should obtain a CA certificate from VeriSign, import it into your key database file, and mark it as trusted. If the LDAP server is using a self-signed server certificate, the administrator of the LDAP server can supply you with a copy of the server's certificate request file. Import the certificate request file into your key database file and mark it as trusted.

If the LDAP servers accessed by the client use client and server authentication, it is necessary to:

- Define one or more trusted root certificates in the key database file. This allows the client to 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 or ldap_simple_bind_s.
- Create a key pair using gsk6ikm and request a client certificate from a CA. After receiving the signed certificate from the CA, store the certificate in the client key database file.

Diagnostics

Exit status is 0 if no errors occur. Errors result in a non-zero exit status and a diagnostic message being written to standard error.

dbback

Synopsis:
dbback [-?] [-d <*backupdir*>] [-w <*filename*>]

Command line options:

-? Displays the syntax format.

-d <*backupdir*
Specifies the directory to use to back up the database.

-w <*filename***>**
Specifies the full path name of the file into which the output is redirected.

dbrestore

Synopsis:
dbrestore [-?] [-d <*backupdir*> [-n]] [-w <*filename*>]

Command line options:

-? Displays the syntax format.

-d <*backupdir*
Specifies the directory from which to restore the database.

-n Specifies not to restore the ibmslapd.conf file. Use this option when you want to resynchronize replica data without overwriting the ibmslapd.conf file of the server.

-w <*filename***>**
Specifies the full path name of the file into which the output is redirected.

runstats

Synopsis:
runstats [-f configfile]

Command line options:

-f configfile
Use this option to specify the slapd configuration file.

Part 6. Appendixes

Appendix A. Troubleshooting

File permissions

On UNIX-based systems file, permissions are frequently altered inadvertently by the actions of copying or editing a key database file. This is because these actions are generally done as the userid **root**, therefore permissions are set for the user root. For the Directory Server to make use of this file, you must change the file permissions so that it is readable by the userid **ldap**. Otherwise the Directory Server fails to start.

```
chown ldap:ldap <mykeyring>.*
```

Kerberos service name change

Before IBM Directory Server 4.1, the LDAP server uses **LDAP** as its Kerberos service name, (**LDAP/ldaphost.austin.ibm.com**, ldaphost is the hostname of the machine where the LDAP server is located) to communicate with its client and the Kerberos KDC. For Version 4.1 and above, a lower case service name is used (**ldap/ldapname.austin.ibm.com**). Because of this change, a 4.1 or 5.1 server might not be able to start after migrating from a 3.x server. This is because the 4.1 or 5.1 server is looking for **ldap** in the keytab file in which an **LDAP** service name was located and used by the previous 3.x server. To correct this situation you can do either of the following:

- Generate a keytab file by adding a lower case LDAP Kerberos service name and start using the new keytab file to communicate.
- Set the environment variable LDAP_KRB_SERVICE_NAME to **LDAP** before starting the server. This environment variable causes the LDAP server to continue using the upper case LDAP server service name in the keytab file and to communicate with its clients. In the latter case, the environment variable needs to be set on the client side as well to continuing using the upper case LDAP service name to communicate with its server.

Websphere Application Server - Express on AIX

Starting the **embedded version of IBM Websphere Application Server - Express** on AIX (**startServer.sh server1**), might not work because port (9090) already being used. See the **BOBCAT_install_path/logs/server1** directory for the actual log files. Usually **SystemErr.log** and **SystemOut.log** are most helpful, although the other logs might have some useful information.

To change the port number for the **embedded version of IBM Websphere Application Server - Express** from 9090 to 9091, which is the port used on AIX machines, edit the **BOBCAT_install_path/config/cells/DefaultNode/virtualhosts.xml** file and change 9090 to 9091. Do the same thing in the

```
BOBCAT_install_path/config/cells/DefaultNode/nodes/DefaultNode/servers/
                              server1/server.xml
```

file.

Note: This path does have two subdirectories called DefaultNode.
Make one change in each file for a total of two updates.

Debugging

Debug output for configuration

During configuration, you might experience some problems with the IBM Directory configuration programs. There are some extra debugging steps that can help you and IBM support teams determine the cause of these problems.

There are three configuration programs in the IBM Directory product. Two are run from a command line, and one is GUI-based. The configuration programs are:

- **ldapcfg** - A command line program to configure an Admin DN, and a database
- **ldapucfg** - A command line program to unconfigure the database
- **ldapxcfg** - A GUI program to configure Admin DN, a database, and perform various other operations.

See the *IBM Directory Server Version 5.1 Installation and Configuration Guide* for information on these utilities.

These configuration programs support two main functions:

- Configuring the Admin DN and password
- Configuring or unconfiguring a database or both for use by the IBM Directory

The configuration of the Admin DN is very straightforward. Generally, the only reasons the configuration of the Admin DN fails are if the IBM Directory configuration file (<*install dir*>/etc/ibmslapd.conf) has had the permissions accidentally changed, or if the user enters an invalid DN.

Database configuration

This leaves the most problematic option, the database configuration/unconfiguration. Because there are so many variables at play during configuration, errors can occur. Some of the factors that can affect this option are:

- Which platform, and which version of the operating system, the user is using.
- Which version of DB2, and which fix packs have been installed for it.

  Note: DB2 comes in a wide variety of packages: Personal Edition, Enterprise Edition, Extended Enterprise Edition, and others. Many of these are supported across several versions of DB2 (7.1, 7.2), plus each version can have several available fix packs.
- Amount of disk space available in affected drives and partitions.
- Third party software that alters commonly used environment variables.

If the database configuration fails, the bottom-line question for the user is, ″What failed, and how do I fix it?″ The following sections describe sources of output that can be used to debug configuration problems.

Standard sources of output

There are several ″standard″ sources of information available to the user:

- The output on the screen.

  All of the configuration programs are either started from a console command line prompt (ldapcfg, ldapucfg) or open a background console (ldapxcfg). As the database configuration progresses, status messages (and limited error messages) are displayed in the associated console window. If a problem occurs, the user should copy these messages to the system clipboard and then save them in a file for the support teams.

- DB2 log files.

  If the error is a direct error from DB2, then DB2 often creates message/error files in the /tmp directory (on UNIX platforms). If the user has a database configuration problem on UNIX systems , they need to examine all of the files in the /tmp directory that were created around the time of the attempted configuration. On Windows systems , examine any DB2 error logs in your DB2 install directory under the directory named for the instance you were trying to configure. For example, if your were trying to create the default ldapdb2 instance and database, and if your DB2 was installed in D:\sqllib, then you need to examine the files in the D:\sqllib\ldadb2 directory if it exists. Especially look for and examine the 'db2diag.log' file in that directory.
- IBM Directory logs:

  IBM Directory logs most configuration errors in the file 'ldacfg.out'. On UNIX platforms, this file can be found in the /tmp directory. On Windows platforms, this file is created in the root directory of the drive you ran configuration from.

Creating advanced debug output

There are two more sources of output that can be used to debug configuration problems. Both of them are initiated by setting environment variables before running the configuration. For both of these debug options, it is strongly recommended that your console window be set to scrollable so that any messages that scroll off the screen can be seen later.

JAVA_DEBUG

Set this environment variable to any non-empty value Example:

```
JAVA_DEBUG=1
```

On UNIX platforms, use `export JAVA_DEBUG=1`. This causes certain Java debug information built into the code to be displayed on stdout (the console).

LDAP_DBG

Set this environment variable to any non-empty value. Example:

```
LDAP_DBG =1
```

On UNIX platforms, use `export LDAP_DBG=1`. This causes a debug file to be created for the IBM support and development teams. The file name that is created is **dbg.log**.

On the Windows NT and Windows 2000 platforms, **dbg.log** is created in the <*ldapinstalldir*> **/var** directory. On UNIX platforms, **dbg.log** is created in the **/var/ldap** directory.

Note: This debug log file contains code-specific information intended for the IBM development team and is not intended for use by the customer. Send this file along with any other debug information to the IBM support team.

ibmslapd command parameters

The **ibmslapd** command has two parameters on UNIX systems and an additional two parameters on Windows systems.

-h <*debug_mask*>

Causes **ibmslapd** to generate debug output to **stdout**. The *debug_mask* is a bit mask that controls which output is generated with values up to 65535. This parameter is for use by IBM service personnel.

-f <*path_to_configuration_file*>
: Specifies the location of the configuration file used when starting the server. This parameter is used if you want to use a customized configuration file. If not specified, **ibmslapd** defaults to the platform dependent location where the configuration file was installed.

Additional parameters for Windows systems:

-i <*servicename*>
: Installs IBM Directory as service on the server.

-u <*servicename*>
: Removes IBM Directory as service from the server.

Server debug mode

If the error logs do not provide enough information to resolve a problem, you can run the IBM Directory Server in a special debug mode that generates very detailed information. The server executable ibmslapd must be run from a command prompt to enable debug output. The syntax is as follows:

```
ldtrc on
ibmslapd -h bitmask
```

where the specified bitmask value determines which categories of debug output are generated.

Table 12. Debug categories

Hex	Decimal	Value	Description
0x0001	1	LDAP_DEBUG_TRACE	Entry and exit from routines
0x0002	2	LDAP_DEBUG_PACKETS	Packet activity
0x0004	4	LDAP_DEBUG_ARGS	Data arguments from requests
0x0008	8	LDAP_DEBUG_CONNS	Connection activity
0x0010	16	LDAP_DEBUG_BER	Encoding and decoding of data
0x0020	32	LDAP_DEBUG_FILTER	Search filters
0x0040	64	LDAP_DEBUG_MESSAGE	Messaging subsystem activities and events
0x0080	128	LDAP_DEBUG_ACL	Access Control List activities
0x0100	256	LDAP_DEBUG_STATS	Operational statistics
0x0200	512	LDAP_DEBUG_THREAD	Threading statistics
0x0400	1024	LDAP_DEBUG_REPL	Replication statistics
0x0800	2048	LDAP_DEBUG_PARSE	Parsing activities
0x1000	4096	LDAP_DEBUG_PERFORMANCE	Relational backend performance statistics
0x1000	8192	LDAP_DEBUG_RDBM	Relational backend activities (RDBM)
0x4000	16384	LDAP_DEBUG_REFERRAL	Referral activities
0x8000	32768	LDAP_DEBUG_ERROR	Error conditions
0xffff	65535	ALL	
0x7fffffff	2147483647	LDAP_DEBUG_ANY	All levels of debug

For example, specifying a bitmask value of ″65535″ turns on full debug output and generates the most complete information.

When you are finished, issue the following command at a command prompt:

```
ldtrc off
```

It is recommended that you contact IBM Service for assistance with interpreting of the debug output and resolving of the problem.

Replication command line interface error (Windows platforms only)

If you are using Windows 2000 or Windows NT and have a master server configured to do replication, you might see an error like the following in the ibmslapd error log during updates :

```
[IBM][CLI Driver] CLI0157E Error opening a file.  SQLSTATE=S1507
```

This problem can be resolved by adding the following stanza to the \sqllib\db2cli.ini file:

```
[COMMON]
TempDir=x:\<your directory>
```

where `x:\<your directory>` specifies an existing directory on a drive that has space available. DB2 database writes temporary files to this directory. The amount of space required depends on the size of the directory entries you are adding or updating, but generally does require more space than the size of the largest entry you are updating.

Appendix B. IBM UUID

You can configure DB2 to enforce unique values for an LDAP attribute, such as UID, in the IBM Directory Server. There are some requirements:

All of the values for the attribute must be stored on each LDAP server where values for the attribute can be created or updated. This means that you cannot have referrals in the directory tree that point to other servers that contain values for the attribute. If you have multiple masters (peer servers) where values for the attribute can be updated, be sure that your directory administrators and applications update the attribute on only one of these servers. If applications created, for example, the same value for the UID attribute on two peer servers at exactly the same time, replication conflicts might occur.

The following is a simple procedure you can use to enforce unique values for UID. In this procedure, you issue the SQL statement to set up a DB2 constraint on the table that contains the UID attribute. Then DB2 ensures that the attribute values are unique. To make this work, you must know how to set the parameters of the SQL statement. The first two steps in this procedure determine these SQL statement parameters. The third step creates the SQL constraint.

1. Determine the attribute for which you want to require directory entries to have unique values. Be sure that the attribute does not already have non-unique values. In this example, the UID attribute is used. The database can already have values in it for this UID, but if it does, they must each already be unique. If there are currently any duplicate values for UID in the database, you will not be able to set up the constraint in step 3 until you delete the non-unique values or change them so that they are unique.
2. Determine which DB2 table stores the attribute values and on which column in that table to set a DB2 constraint to enforce unique values. Understand that DB2 needs to create an index to efficiently enforce unique values in a column. It does not create indexes on columns that contain values more than 255 characters in length. So,
   - If the length specified for the attribute in the LDAP schema is 255 characters or less, the column in the DB2 table that contains the values for the attribute that can be indexed for a uniqueness constraint , by default, has exactly the same name as the attribute.
   - If the length of the attribute can be greater than 255 characters, DB2 does not allow the creation of an index on this column. The LDAP server knows this so it creates another column with the same name as the attribute, but with the characters ″_T″ appended to it. This extra truncated column contains the values for the attribute truncated to only 255 characters in length. DB2 can create an index on this column, so this is the column on which you must add a constraint for unique values.

   You can tell what the maximum length of an attribute is by looking at its definition in the LDAP schema, for example with the Web Administration Tool. Note that in the case of the UID attribute, the default length in the schema packaged with the IBM Directory Server is 256. Therefore, the second point is true for this attribute. In this example, we must set a uniqueness constraint on the column ″UID_T″. If we try to set it on the ″UID″ column, the SQL command fails because the required index cannot be created.

3. After determining the table and column on which we want DB2 to enforce unique values, issue a **SQL ALTER TABLE** statement to tell DB2 to allow only unique values for UID.
   a. Go to a DB2 command prompt.
      - On Windows, type **db2cmd** at a Windows command prompt. This brings up a DB2 command window.
      - On UNIX platforms, type **su ldapdb2** while logged on as **root**. This sets the correct DB2 environment.

   b. On Windows, type **set db2instance=ldapdb2** (This step is not needed on UNIX platforms.)
   c. Connect to ldapdb2. This establishes a DB2 connection to the LDAP server's database for the following alter table SQL command.
   d. Type the following command:

      ```
      db2 alter table "ldapdb2.uid" add CONSTRAINT const1 UNIQUE (uid_t)
      ```

      This SQL statement establishes the constraint. Notice that the UNIQUE parameter value is **uid_t** because the UID attribute can be longer than 255 characters. From now on, DB2 does not allow a value to be specified, if the first 255 characters are not unique in the local database. The constraint is named **const1** in this example, but you can name it anything you want. Remember the constraint name in case you want to drop it and allow non-unique values again. To drop the uniqueness constraint issue the following SQL command:

      ```
      alter table "ldapdb2.uid" drop constraint const1
      ```

When an application tries to add an entry to the directory with a value for the UID attribute that duplicates an existing directory entry, an error with result code 20 is returned from the LDAP server; for example:

```
LDAP: error code 20 - Attribute or Value Exists
```

Appendix C. Error codes

The possible values for an LDAP error code are shown in the following tables:

Table 13. General return codes

Dec value	Value	Hex value	Brief description	Detailed description
00	LDAP_SUCCESS	00	Success	The request was successful.
00	LDAP_OPERATIONS_ERROR	01	Operations error	An operations error occurred.
02	LDAP_PROTOCOL_ERROR	02	Protocol error	A protocol violation was detected.
03	LDAP_TIMELIMIT_EXCEEDED	03	Time limit exceeded	An LDAP time limit was exceeded.
04	LDAP_SIZELIMIT_EXCEEDED	04	Size limit exceeded	An LDAP size limit was exceeded.
05	LDAP_COMPARE_FALSE	05	Compare false	A compare operation returned false.
06	LDAP_COMPARE_TRUE	06	Compare true	A compare operation returned true.
07	LDAP_STRONG_AUTH_NOT_SUPPORTED	07	Strong authentication not supported	The LDAP server does not support strong authentication.
08	LDAP_STRONG_AUTH_REQUIRED	08	Strong authentication required	Strong authentication is required for the operation.
09	LDAP_PARTIAL_RESULTS	09	Partial results and referral received	Partial results only returned.
10	LDAP_REFERRAL	0A	Referral returned	Referral returned.
11	LDAP_ADMIN_LIMIT_EXCEEDED	0B	Administration limit exceeded	Administration limit exceeded.
12	LDAP_UNAVAILABLE_CRITICAL_EXTENSION	0C	Critical extension not supported	Critical extension is not supported.
13	LDAP_CONFIDENTIALITY_REQUIRED	0D	Confidentiality is required	Confidentiality is required.
14	LDAP_SASLBIND_IN_PROGRESS	0E	SASL bind in progress	An SASL bind is in progress.
16	LDAP_NO_SUCH_ATTRIBUTE	10	No such attribute	The attribute type specified does not exist in the entry.
17	LDAP_UNDEFINED_TYPE	11	Undefined attribute type	The attribute type specified is not valid.
18	LDAP_INAPPROPRIATE_MATCHING	12	Inappropriate matching	Filter type not supported for the specified attribute.

Table 13. General return codes (continued)

Dec value	Value	Hex value	Brief description	Detailed description
19	LDAP_CONSTRAINT_VIOLATION	13	Constraint violation	An attribute value specified violates some constraint (for example, a postal address has too many lines, or a line that is too long).
20	LDAP_TYPE_OR_VALUE_EXISTS	14	Type or value exists	An attribute type or attribute value specified already exists in the entry.
21	LDAP_INVALID_SYNTAX	15	Invalid syntax	An attribute value that is not valid was specified.
32	LDAP_NO_SUCH_OBJECT	20	No such object	The specified object does not exist in the directory.
33	LDAP_ALIAS_PROBLEM	21	Alias problem	An alias in the directory points to a nonexistent entry.
34	LDAP_INVALID_DN_SYNTAX	22	Invalid DN syntax	A DN that is syntactically not valid was specified.
35	LDAP_IS_LEAF	23	Object is a leaf	The object specified is a leaf.
36	LDAP_ALIAS_DEREF_PROBLEM	24	Alias dereferencing problem	A problem was encountered when dereferencing an alias.
48	LDAP_INAPPROPRIATE_AUTH	30	Inappropriate authentication	Inappropriate authentication was specified (for example, LDAP_AUTH_SIMPLE was specified and the entry does not have a userPassword attribute).
49	LDAP_INVALID_CREDENTIALS	31	Invalid credentials	Invalid credentials were presented (for example, the wrong password).
50	LDAP_INSUFFICIENT_ACCESS	32	Insufficient access	The user has insufficient access to perform the operation.
51	LDAP_BUSY	33	DSA is busy	The DSA is busy.
52	LDAP_UNAVAILABLE	34	DSA is unavailable	The DSA is unavailable.
53	LDAP_UNWILLING_TO_PERFORM	35	DSA is unwilling to perform	The DSA is unwilling to perform the operation.
54	LDAP_LOOP_DETECT	36	Loop detected	A loop was detected.
64	LDAP_NAMING_VIOLATION	40	Naming violation	A naming violation occurred.
65	LDAP_OBJECT_CLASS_VIOLATION	41	Object class violation	An object class violation occurred (for example, a ″required″ attribute was missing from the entry).

Table 13. General return codes (continued)

Dec value	Value	Hex value	Brief description	Detailed description
66	LDAP_NOT_ALLOWED_ON_NONLEAF	42	Operation not allowed on nonleaf	The operation is not allowed on a nonleaf object.
67	LDAP_NOT_ALLOWED_ON_RDN	43	Operation not allowed on RDN	The operation is not allowed on an RDN.
68	LDAP_ALREADY_EXISTS	44	Already exists	The entry already exists.
69	LDAP_NO_OBJECT_CLASS_MODS	45	Cannot modify object class	Object class modifications are not allowed.
70	LDAP_RESULTS_TOO_LARGE	46	Results too large	Results too large.
71	LDAP_AFFECTS_MULTIPLE_DSAS	47	Affects multiple DSAs	Affects multiple DSAs.
80	LDAP_OTHER	50	Unknown error	An unknown error occurred.
81	LDAP_SERVER_DOWN	51	Can’t contact LDAP server	The LDAP library cannot contact the LDAP server.
82	LDAP_LOCAL_ERROR	52	Local error	Some local error occurred. This is usually a failed memory allocation.
83	LDAP_ENCODING_ERROR	53	Encoding error	An error was encountered encoding parameters to send to the LDAP server.
84	LDAP_DECODING_ERROR	54	Decoding error	An error was encountered decoding a result from the LDAP server.
85	LDAP_TIMEOUT	55	Timed out	A time limit was exceeded while waiting for a result.
86	LDAP_AUTH_UNKNOWN	56	Unknown authentication method	The authentication method specified on a bind operation is not known.
87	LDAP_FILTER_ERROR	57	Bad search filter	An invalid filter was supplied to ldap_search (for example, unbalanced parentheses).
88	LDAP_USER_CANCELLED	58	User cancelled operation	The user cancelled the operation.
89	LDAP_PARAM_ERROR	59	Bad parameter to an LDAP routine	An LDAP routine was called with a bad parameter (for example, a NULL ld pointer, etc.).
90	LDAP_NO_MEMORY	5A	Out of memory	A memory allocation (for example malloc) call failed in an LDAP library routine.
91	LDAP_CONNECT_ERROR	5B	Connection error	Connection error.

Table 13. General return codes (continued)

Dec value	Value	Hex value	Brief description	Detailed description
92	LDAP_NOT_SUPPORTED	5C	Not supported	Not supported.
93	LDAP_CONTROL_NOT_FOUND	5D	Control not found	Control not found.
94	LDAP_NO_RESULTS_RETURNED	5E	No results returned	No results returned.
95	LDAP_MORE_RESULTS_TO_RETURN	5F	More results to return	More results to return.
96	LDAP_URL_ERR_NOTLDAP	60	URL doesn't begin with ldap://	The URL does not begin with ldap://.
97	LDAP_URL_ERR_NODN	61	URL has no DN (required)	The URL does not have a DN (required).
98	LDAP_URL_ERR_BADSCOPE	62	URL scope string is invalid	The URL scope string is not valid.
99	LDAP_URL_ERR_MEM	63	Can't allocate memory space	Cannot allocate memory space.
100	LDAP_CLIENT_LOOP	64	Client loop	Client loop.
101	LDAP_REFERRAL_LIMIT_EXCEEDED	65	Referral limit exceeded	Referral limit exceeded.
112	LDAP_SSL_ALREADY_INITIALIZED	70	ldap_ssl_client_init successfully called previously in this process	The ldap_ssl_client_init was successfully called previously in this process.
113	LDAP_SSL_INITIALIZE_FAILED	71	Initialization call failed	SSL Initialization call failed.
114	LDAP_SSL_CLIENT_INIT_NOT_CALLED	72	Must call ldap_ssl_client_init before attempting to use SSL connection	Must call ldap_ssl_client_init before attempting to use SSL connection.
115	LDAP_SSL_PARAM_ERROR	73	Invalid SSL parameter previously specified	An SSL parameter that was not valid was previously specified.
116	LDAP_SSL_HANDSHAKE_FAILED	74	Failed to connect to SSL server	Failed to connect to SSL server.
117	LDAP_SSL_GET_CIPHER_FAILED	75	Not used	Deprecated.
118	LDAP_SSL_NOT_AVAILABLE	76	SSL library cannot be located	Ensure that GSKit has been installed.
128	LDAP_NO_EXPLICIT_OWNER	80	No explicit owner found	No explicit owner was found.
129	LDAP_NO_LOCK	81	Could not obtain lock	Client library was not able to lock a required resource.

In addition, the following DNS-related error codes are defined in the ldap.h file:

Table 14. DNS-related return codes

Dec value	Value	Hex value	Detailed description
133	LDAP_DNS_NO_SERVERS	85	No LDAP servers found
134	LDAP_DNS_TRUNCATED	86	Warning: truncated DNS results
135	LDAP_DNS_INVALID_DATA	87	Invalid DNS Data

Table 14. DNS-related return codes (continued)

Dec value	Value	Hex value	Detailed description
136	LDAP_DNS_RESOLVE_ERROR	88	Can’t resolve system domain or nameserver
137	LDAP_DNS_CONF_FILE_ERROR	89	DNS Configuration file error

The following UTF8-related error codes are defined in the ldap.h file:

Table 15. UTF8-related return codes

Dec value	Value	Hex value	Detailed description
160	LDAP_XLATE_E2BIG	A0	Output buffer overflow
161	LDAP_XLATE_EINVAL	A1	Input buffer truncated
162	LDAP_XLATE_EILSEQ	A2	Unusable input character
163	LDAP_XLATE_NO_ENTRY	A3	No codeset point to map to

Appendix D. LDAP data interchange format (LDIF)

This documentation describes the LDAP Data Interchange Format (LDIF), as used by the ldapmodify, ldapsearch and ldapadd utilities. The LDIF specified here is also supported by the server utilities provided with the IBM Directory.

LDIF is used to represent LDAP entries in text form. The basic form of an LDIF entry is:

```
dn: <distinguished name>
<attrtype> : <attrvalue>
<attrtype> : <attrvalue>
...
```

A line can be continued by starting the next line with a single space or tab character, for example:

```
dn: cn=John E Doe, o=University of Higher
 Learning, c=US
```

Multiple attribute values are specified on separate lines, for example:

```
cn: John E Doe
cn: John Doe
```

If an *<attrvalue>* contains a non-US-ASCII character, or begins with a space or a colon ':', the *<attrtype>* is followed by a double colon and the value is encoded in base-64 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 a blank line. Multiple blank lines are considered a logical end-of-file.

LDIF example

Here is an example of an LDIF file containing three entries.

```
dn: cn=John E Doe, o=University of High
 er Learning, c=US
cn: John E Doe
cn: John Doe
objectclass: person
sn: Doe

dn: cn=Bjorn L Doe, o=University of High
 er Learning, c=US
cn: Bjorn L Doe
cn: Bjorn Doe
objectclass: person
sn: Doe

dn: cn=Jennifer K. Doe, o=University of High
 er Learning, c=US
cn: Jennifer K. Doe
cn: Jennifer Doe
objectclass: person
sn: Doe
```

```
jpegPhoto:: /9j/4AAQSkZJRgABAAAAAQABAAD/2wBDABALD
 A4MChAODQ4SERATGCgaGBYWGDEjJR0oOjM9PDkzODdASFxOQ
 ERXRTc4UG1RV19iZ2hnPk1xeXBkeFx1Z2P/2wBDARESEhgVG
...
```

The jpegPhoto in Jennifer Jensen's entry is encoded using base-64. The textual attribute values can also be specified in base-64 format. However, if this is the case, the base-64 encoding must be in the code page of the wire format for the protocol (that is, for LDAP V2, the IA5 character set and for LDAP V3, the UTF-8 encoding).

Version 1 LDIF support

The client utilities (ldapmodify and ldapadd) have been enhanced to recognize the latest version of LDIF, which is identified by the presence of the ″version: 1″ tag at the head of the file. Unlike the original version of LDIF, the newer version of LDIF supports attribute values represented in UTF-8 (instead of the very limited US-ASCII).

However, manual creation of an LDIF file containing UTF-8 values may be difficult. In order to simplify this process, a charset extension to the LDIF format is supported. This extension allows an IANA character set name to be specified in the header of the LDIF file (along with the version number). A limited set of the IANA character sets are supported. See "IANA character sets supported by platform" on page 255 for the specific charset values that are supported for each operating system platform.

The version 1 LDIF format also supports file URLs. This provides a more flexible way to define a file specification. File URLs take the following form:

```
attribute:< file:///path          (where path syntax depends on platform)
```

For example, the following are valid file Web addresses:

```
jpegphoto:< file:///d:\temp\photos\myphoto.jpg    (DOS/Windows style paths)
jpegphoto:< file:///etc/temp/photos/myphoto.jpg   (Unix style paths)
```

Note: The IBM Directory utilities support both the new file URL specification as well as the older style (e.g. ″jpegphoto: /etc/temp/myphoto″), regardless of the version specification. In other words, the new file URL format can be used without adding the version tag to your LDIF files.

Version 1 LDIF examples

You can use the optional charset tag so that the utilities will automatically convert from the specified character set to UTF-8 as in the following example:

```
version: 1
charset: ISO-8859-1

dn: cn=Juan Griego, o=University of New Mexico, c=US
cn: Juan Griego
sn: Griego
description:: V2hhdCBhIGNhcmVmdWwgcmVhZGVyIH1vd
title: Associate Dean
title: [title in Spanish]
jpegPhoto:> file:///usr/local/photos/jgriego.jpg
```

In this instance, all values following an attribute name and a single colon are translated from the ISO-8859-1 character set to UTF-8. Values following an attribute name and a double colon (such as description:: V2hhdCBhIGNhcm...) must be base-64 encoded, and are expected to be either binary or UTF-8 character strings. Values read from a file, such as the jpegPhoto attribute specified by the Web address in the previous example, are also expected to be either binary or UTF-8. No translation from the specified ″charset″ to UTF-8 is done on those values.

In this example of an LDIF file without the charset tag, content is expected to be in UTF-8, or base-64 encoded UTF-8, or base-64 encoded binary data:

```
# IBM Directorysample LDIF file
 #
 # The suffix "o=IBM, c=US" should be defined before attempting to load
 # this data.

 version: 1

 dn: o=IBM, c=US
 objectclass: top
 objectclass: organization
 o: IBM

 dn: ou=Austin, o=IBM, c=US
 ou: Austin
 objectclass: organizationalUnit
 seealso: cn=Linda Carlesberg, ou=Austin, o=IBM, c=US
```

This same file could be used without the version: 1 header information, as in previous releases of the IBM Directory:

```
 # IBM Directorysample LDIF file
 #
 # The suffix "o=IBM, c=US" should be defined before attempting to load
 # this data.

 dn: o=IBM, c=US
 objectclass: top
 objectclass: organization
 o: IBM

 dn: ou=Austin, o=IBM, c=US
 ou: Austin
 objectclass: organizationalUnit
 seealso: cn=Linda Carlesberg, ou=Austin, o=IBM, c=US
```

Note: The textual attribute values can be specified in base-64 format.

IANA character sets supported by platform

The following table defines the set of IANA-defined character sets that can be defined for the charset tag in a Version 1 LDIF file, on a per-platform basis. The value in the left-most column defines the text string that can be assigned to the charset tag. An ″X″ indicates that conversion from the specified charset to UTF-8 is supported for the associated platform, and that all string content in the LDIF file is assumed to be represented in the specified charset. ″n/a″ indicates that the conversion is not supported for the associated platform.

String content is defined to be all attribute values that follow an attribute name and a single colon.

See IANA Character Sets for more information about IANA-registered character sets.

Table 16.

Character	Locale					DB2 Code Page	
Set Name	HP-UX	Linux, Linux_390,	NT	AIX	Solaris	UNIX	NT
ISO-8859-1	X	X	X	X	X	819	1252
ISO-8859-2	X	X	X	X	X	912	1250
ISO-8859-5	X	X	X	X	X	915	1251
ISO-8859-6	X	X	X	X	X	1089	1256
ISO-8859-7	X	X	X	X	X	813	1253
ISO-8859-8	X	X	X	X	X	916	1255
ISO-8859-9	X	X	X	X	X	920	1254
ISO-8859–15	X	n/a	X	X	X		
IBM437	n/a	n/a	X	n/a	n/a	437	437
IBM850	n/a	n/a	X	X	n/a	850	850
IBM852	n/a	n/a	X	n/a	n/a	852	852
IBM857	n/a	n/a	X	n/a	n/a	857	857
IBM862	n/a	n/a	X	n/a	n/a	862	862
IBM864	n/a	n/a	X	n/a	n/a	864	864
IBM866	n/a	n/a	X	n/a	n/a	866	866
IBM869	n/a	n/a	X	n/a	n/a	869	869
IBM1250	n/a	n/a	X	n/a	n/a		
IBM1251	n/a	n/a	X	n/a	n/a		
IBM1253	n/a	n/a	X	n/a	n/a		
IBM1254	n/a	n/a	X	n/a	n/a		
IBM1255	n/a	n/a	X	n/a	n/a		
IBM1256	n/a	n/a	X	n/a	n/a		
TIS-620	n/a	n/a	X	X	n/a	874	874
EUC-JP	X	X	n/a	X	X	954	n/a
EUC-KR	n/a	n/a	n/a	X	X	970	n/a
EUC-CN	n/a	n/a	n/a	X	X	1383	n/a
EUC-TW	X	n/a	n/a	X	X	964	n/a
Shift-JIS	n/a	X	X	X	X	932	943
KSC	n/a	n/a	X	n/a	n/a	n/a	949
GBK	n/a	n/a	X	X	n/a	1386	1386
Big5	X	n/a	X	X	X	950	950
GB18030	n/a	X	X	X	X		
HP15CN	X (with non-GB18030)						

Appendix E. IBM Directory Server 5.1 schema object classes and attributes

These are the configuration object classes and attributes that are included in the IBM Directory Server Version 5.1. They can be found in the **V3config.oc** and **V3.config.at** files in the **etc** directory. They define the objects that can appear in the ibmslapd.conf file.

Object classes

These are the schema object classes that are shipped with the IBM Directory Server Version 5.1

```
# File generated at 3:51:37 PM on 8/28/02 from IBM LDAP schema version 1.5

objectclasses=( 1.3.18.0.2.6.489
NAME 'ibm-slapdAdmin'
DESC 'Global configuration settings for IBM Admin Daemon'
SUP ( ibm-slapdConfigEntry $ top )
STRUCTURAL
MUST ( cn $ ibm-slapdErrorLog $ ibm-slapdPort )
MAY ( ibm-slapdSecurePort ) )

objectclasses=( 1.3.18.0.2.6.490
NAME 'ibm-slapdConfigBackend'
DESC 'Config backend configuration for IBM Directory'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn $ ibm-slapdPlugin $ ibm-slapdSuffix )
MAY ( ibm-slapdReadOnly ) )

objectclasses=( 1.3.18.0.2.6.486
NAME 'ibm-slapdConfigEntry'
DESC 'ibm slapd config entry'
SUP 'top'
ABSTRACT
MUST ( cn )
MAY ( ibm-slapdInvalidLine ) )

objectclasses=( 1.3.18.0.2.6.493
NAME 'ibm-slapdCRL'
DESC 'Certificate revocation list settings for IBM Directory.'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn $ ibm-slapdLdapCrlHost $ ibm-slapdLdapCrlPort )
MAY ( ibm-slapdLdapCrlPassword $ ibm-slapdLdapCrlUser ) )

objectclasses=( 1.3.18.0.2.6.500
NAME 'ibm-slapdEventNotification'
DESC 'Global event notification settings for IBM Directory.'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn $ ibm-slapdEnableEventNotification )
MAY ( ibm-slapdMaxEventsPerConnection $ ibm-slapdMaxEventsTotal ) )

objectclasses=( 1.3.18.0.2.6.501
NAME 'ibm-slapdFrontEnd'
DESC 'Global front-end settings which the server will load at startup.'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn )
```

```
MAY ( ibm-slapdPlugin $ ibm-slapdSetenv $ ibm-slapdIdleTimeOut $ ibm-slapdACLCache
$ ibm-slapdACLCacheSize $ ibm-slapdFilterCacheSize $ ibm-slapdFilterCacheBypassLimit
$ ibm-slapdEntryCacheSize $ ibm-slapdDB2CP ) )

objectclasses=( 1.3.18.0.2.6.494
NAME 'ibm-slapdKerberos'
DESC 'Global kerberos authentication settings for IBM Directory.'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn $ ibm-slapdKrbAdminDN $ ibm-slapdKrbEnable $ ibm-slapdKrbIdentityMap
$ ibm-slapdKrbKeyTab $ ibm-slapdKrbRealm ) )

objectclasses=( 1.3.18.0.2.6.495
NAME 'ibm-slapdLdcfBackend'
DESC 'LDCF backend configuration for IBM Directory.'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn )
MAY ( ibm-slapdSuffix $ ibm-slapdPlugin ) )

objectclasses=( 1.3.18.0.2.6.526
NAME 'ibm-slapdPendingMigration'
DESC 'Indicates that a server component requires migration.'
SUP 'top'
AUXILIARY
MAY ( ibm-slapdMigrationInfo ) )

objectclasses=( 1.3.18.0.2.6.497
NAME 'ibm-slapdRdbmBackend'
DESC 'DB2 database backend configuration for IBM Directory.'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn $ ibm-slapdDbName $ ibm-slapdDbInstance $ ibm-slapdDbUserID $
ibm-slapdDbUserPW )
MAY ( ibm-slapdPlugin $ ibm-slapdSuffix $ ibm-slapdReadOnly
$ ibm-slapdChangeLogMaxEntries $ ibm-slapdPagedResAllowNonAdmin
$ ibm-slapdPagedResLmt $ ibm-slapdPageSizeLmt $ ibm-slapdSortKeyLimit
$ ibm-slapdSortSrchAllowNonAdmin $ ibm-slapdDbConnections $ ibm-slapdDbLocation
$ ibm-slapdDB2CP $ ibm-slapdReplDbConns $ ibm-slapdCLIErrors
$ ibm-slapdBulkloadErrors $ ibm-slapdDBAlias $ ibm-slapdUseProcessIdPW ) )

objectclasses=( 1.3.18.0.2.6.485
NAME 'ibm-slapdReferral'
DESC 'Global superior referrals for IBM Directory.'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn $ ibm-slapdReferral ) )

objectclasses=( 1.3.18.0.2.6.496
NAME 'ibm-slapdReplication'
DESC 'Contains the default bind credentials and master server referral URL.
This is used when the server contains one or more replication contexts that
are replicated to it by other servers.  This server may be acting as
one of several masters or as a read only replica.  If the MasterDN is
specified without the Master PW attribute, kerberos authentication is used.'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn )
MAY ( ibm-slapdMasterDN $ ibm-slapdMasterPW $ ibm-slapdMasterReferral ) )

objectclasses=( 1.3.18.0.2.6.499
NAME 'ibm-slapdSchema'
DESC 'Global schema settings for IBM Directory.
Multiple schemas are not currently supported, but if they were then there
would be one ibm-slapdSchema entry per schema.'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
```

```
MUST ( cn $ ibm-slapdSchemaCheck $ ibm-slapdIncludeSchema )
MAY ( ibm-slapdSchemaAdditions ) )

objectclasses=( 1.3.18.0.2.6.492
NAME 'ibm-slapdSSL'
DESC 'Global SSL connection settings for IBM Directory.'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn $ ibm-slapdSecurity $ ibm-slapdSecurePort $ ibm-slapdSslAuth )
MAY ( ibm-slapdSslCertificate $ ibm-slapdSslCipherSpec
$ ibm-slapdSslCipherSpecs $ ibm-slapdSSLKeyDatabase
$ ibm-slapdSSLKeyDatabasePW $ ibm-slapdSslKeyRingFilePW ) )

objectclasses=( 1.3.18.0.2.6.488
NAME 'ibm-slapdSupplier'
DESC 'Contains bind credentials used by a replication supplier server to
update the specified subtree on this consumer server. Use of theis object
class overrides the default bind credentials specified in an ibm-slapdReplication
object'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn $ ibm-slapdReplicaSubtree $ ibm-slapdMasterDN )
MAY ( ibm-slapdMasterPW ) )

objectclasses=( 1.3.18.0.2.6.498
NAME 'ibm-slapdTop'
DESC 'Global configuration settings for IBM Directory Server.'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn $ ibm-slapdAdminDN $ ibm-slapdAdminPW $ ibm-slapdErrorLog
$ ibm-slapdPort $ ibm-slapdPwEncryption $ ibm-slapdSizeLimit
$ ibm-slapdSysLogLevel $ ibm-slapdTimeLimit ) MAY ( ibm-slapdServerId
$ ibm-slapdVersion $ ibm-slapdACLAccess $ ibm-slapdMaxPendingChangesDisplayed
$ ibm-slapdSupportedWebAdmVersion ) )

objectclasses=( 1.3.18.0.2.6.491
NAME 'ibm-slapdTransaction'
DESC 'Global transaction support settings for IBM Directory.'
SUP ( top $ ibm-slapdConfigEntry )
STRUCTURAL
MUST ( cn $ ibm-slapdMaxNumOfTransactions $ ibm-slapdMaxOpPerTransaction
$ ibm-slapdMaxTimeLimitOfTransactions $ ibm-slapdTransactionEnable ) )
```

Attributes

These are the configuration attributes that are shipped with the IBM Directory Server Version 5.1. For descriptive names to go with the syntax OIDs, see the **V3.ldapsyntaxes** file in the **etc** directory.

```
# File generated at 3:50:52 PM on 8/28/02 from IBM LDAP schema version 1.5

attributetypes=( 1.3.18.0.2.4.2485
NAME 'ibm-slapdACLAccess'
DESC 'If set to true anyone that can read an entry can also read the entrys ACL
attributes.  If set to false only the entry owner or the administrator can read
the ACL attributes.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
USAGE userApplications )

IBMAttributetypes=( 1.3.18.0.2.4.2485
DBNAME( 'slapdACLAccess'  'slapdACLAccess' )
ACCESS-CLASS normal
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2374
NAME 'ibm-slapdACLCache'
DESC 'Controls whether or not the server caches ACL information'
```

```
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2374
DBNAME( 'ACLCache'  'ACLCache' )
ACCESS-CLASS normal
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2373
NAME 'ibm-slapdACLCacheSize'
DESC 'Maximum number of entries to keep in the ACL Cache'
EQUALITY 2.5.13.14
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2373
DBNAME( 'slapdACLCacheSize'  'slapdACLCacheSize' )
ACCESS-CLASS normal
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2428
NAME 'ibm-slapdAdminDN'
DESC 'Bind DN for the directory administrator, e.g.: cn=root'
EQUALITY 2.5.13.1
ORDERING 1.3.18.0.2.4.405
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2428
DBNAME( 'slapdAdminDN'  'slapdAdminDN' )
ACCESS-CLASS critical
LENGTH 1000
EQUALITY ORDERING )

attributetypes=( 1.3.18.0.2.4.2425
NAME 'ibm-slapdAdminPW'
DESC 'Bind password for the directory administrator'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2425
DBNAME( 'slapdAdminPW'  'slapdAdminPW' )
ACCESS-CLASS critical )

attributetypes=( 1.3.18.0.2.4.2366
NAME 'ibm-slapdAuthIntegration'
DESC 'Specifies integration of LDAP administrator access with
local OS users.  Legal values are : 0 - do not map local OS
users to LDAP administrator, 1 - map local OS users with proper
authority to LDAP administrator.  This is supported
only on OS/400.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2366
DBNAME( 'slapdAuthIntegrat'  'slapdAuthIntegrat' )
ACCESS-CLASS system
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2368
NAME 'ibm-slapdBulkloadErrors'
DESC 'File path or device on ibmslapd host machine to which bulkload
```

```
error messages will be written.   On Windows, forward slashes are
allowed, and a leading slash not preceded by a drive letter is
assumed to be rooted at the install directory
(i.e.: /tmp/bulkload.errors = D:\Program Files\IBM\ldap\tmp\bulkload.errors).'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
USAGE userApplications )

IBMAttributetypes=( 1.3.18.0.2.4.2368
DBNAME( 'slapdBulkloadErro'  'slapdBulkloadErro' )
ACCESS-CLASS normal
LENGTH 1024 )

attributetypes=( 1.3.18.0.2.4.2427
NAME 'ibm-slapdChangeLogMaxEntries'
DESC 'Specifies the maximum number of changelog entries allowed
in the associated backend. Each changelog backend has its own
ibm-slapdChangeLogMaxEntries attribute. If the attribute
is undefined or out of range (negative), it defaults to 0.
Min: 0 (unlimited) Max: 2,147,483,647 (32-bit, signed integer)'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE userApplications )

IBMAttributetypes=( 1.3.18.0.2.4.2427
DBNAME( 'chgLogMaxEntries'  'chgLogMaxEntries' )
ACCESS-CLASS critical
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2432
NAME 'ibm-slapdCLIErrors'
DESC 'File path or device on ibmslapd host machine to which DB2 CLI error
messages will be written. On Windows, forward slashes are allowed,
and a leading slash not preceded by a drive letter is
assumed to be rooted at the install directory
(i.e.: /tmp/cli.errors = D:\Program Files\IBM\ldap\tmp\cli.errors).'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2432
DBNAME( 'slapdCLIErrors'  'slapdCLIErrors' )
ACCESS-CLASS normal
LENGTH 1024 )

attributetypes=( 1.3.18.0.2.4.2369
NAME 'ibm-slapdDB2CP'
DESC 'Specifies the Code Page of the directory database.
1208 is the code page for UTF-8 databases.'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2369
DBNAME( 'slapdDB2CP'  'slapdDB2CP' )
ACCESS-CLASS normal
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2431
NAME 'ibm-slapdDBAlias'
DESC 'The DB2 database alias.'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )
```

```
IBMAttributetypes=( 1.3.18.0.2.4.2431
DBNAME( 'slapdDBAlias'  'slapdDBAlias' )
ACCESS-CLASS normal
LENGTH 8 )

attributetypes=( 1.3.18.0.2.4.2417
NAME 'ibm-slapdDbConnections'
DESC 'Specify the number of DB2 connections the server
will dedicate to the DB2 backend. The value must be
between 5 & 50 (inclusive). The ODBCCONS environment variable
overrides this value. If ibm-slapdDbConnections (or ODBCCONS)
is less than 5 or greater than 50, the server will use 5 or 50
respectively. Additional connections may be created for
replication and change log.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2417
DBNAME( 'DbConnections'  'DbConnections' )
ACCESS-CLASS critical
LENGTH 2 )

attributetypes=( 1.3.18.0.2.4.2418
NAME 'ibm-slapdDbInstance'
DESC 'The DB2 database instance for this backend.'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2418
DBNAME( 'slapdDbInstance'  'slapdDbInstance' )
ACCESS-CLASS critical
LENGTH 8 )

attributetypes=( 1.3.18.0.2.4.2382
NAME 'ibm-slapdDbLocation'
DESC 'The file system path where the backend database is
located. On Unix this is usually the home directory of the
DB2INSTANCE owner (e.g.: /home/ldapdb2). On windows its just
a drive specifier (e.g.: D:)'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2382
DBNAME( 'slapdDbLocation'  'slapdDbLocation' )
ACCESS-CLASS critical
LENGTH 1024 )
attributetypes=( 1.3.18.0.2.4.2426
NAME 'ibm-slapdDbName'
 DESC 'The DB2 database name for this backend.'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2426
DBNAME( 'slapdDbName'  'slapdDbName' )
ACCESS-CLASS critical
LENGTH 8 )

attributetypes=( 1.3.18.0.2.4.2422
NAME 'ibm-slapdDbUserID'
DESC 'The user name with which to connect to the DB2 database
for this backend.'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE

USAGE directoryOperation )
IBMAttributetypes=( 1.3.18.0.2.4.2422
DBNAME( 'slapdDbUserID'  'slapdDbUserID' )
ACCESS-CLASS critical
LENGTH 8 )

attributetypes=( 1.3.18.0.2.4.2423
NAME 'ibm-slapdDbUserPW'
DESC 'The user password with which to connect to the DB2 database
for this backend.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2423
DBNAME( 'slapdDbUserPW'  'slapdDbUserPW' )
ACCESS-CLASS critical )

attributetypes=( 1.3.18.0.2.4.2421
NAME 'ibm-slapdEnableEventNotification'
DESC 'If set to FALSE, the server will reject all extended
operation requests to register for event notification with
the extended result LDAP_UNWILLING_TO_PERFORM.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2421
DBNAME( 'enableEvntNotify'  'enableEvntNotify' )
ACCESS-CLASS critical
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2372
NAME 'ibm-slapdEntryCacheSize'
DESC 'Maximum number of entries to keep in the entry cache'
EQUALITY 2.5.13.14
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2372
DBNAME( 'slapdRDBMCacheSiz'  'slapdRDBMCacheSiz' )
ACCESS-CLASS normal
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2424
NAME 'ibm-slapdErrorLog'
DESC 'File path or device on the ibmslapd host machine to which error
messages will be written.  On Windows, forward slashes are
allowed, and a leading slash not preceded by a drive letter
is assumed to be rooted at the install directory
(i.e.: /tmp/slapd.errors = D:\Program Files\IBM\ldap\tmp\slapd.errors).'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2424
DBNAME( 'slapdErrorLog'  'slapdErrorLog' )
ACCESS-CLASS critical
LENGTH 1024 )
```

```
attributetypes=( 1.3.18.0.2.4.2371
NAME 'ibm-slapdFilterCacheBypassLimit'
DESC 'Search filters that match more than this number of entries
will not be added to the Search Filter cache.  Because the list
of entry ids that matched the filter are included in this cache,
this setting helps to limit memory use.  A value of 0 indicates
no limit.'
EQUALITY 2.5.13.14
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2371
DBNAME( 'slapdRDBMCacheByp'  'slapdRDBMCacheByp' )
ACCESS-CLASS normal
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2370
NAME 'ibm-slapdFilterCacheSize'
DESC 'Specifies the maximum number of entries to keep in
the Search Filter Cache.'
EQUALITY 2.5.13.14
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2370
DBNAME( 'slapdFilterCacheS'  'slapdFilterCacheS' )
ACCESS-CLASS normal
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2378
NAME 'ibm-slapdIdleTimeOut'
DESC 'Reserved for future use.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2378
DBNAME( 'SlapdIdleTimeOut'  'SlapdIdleTimeOut' )
ACCESS-CLASS critical
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2364
NAME 'ibm-slapdIncludeSchema'
DESC 'File path on the ibmslapd host machine containing schema
definitions used by the LDCF backend. Standard values are:
/etc/V3.system.at /etc/V3.system.oc /etc/V3.ibm.at
/etc/V3.ibm.oc /etc/V3.user.at /etc/V3.user.oc
/etc/V3.ldapsyntaxes /etc/V3.matchingrules
/etc/V3.modifiedschema  On Windows, forward slashes are allowed,
and a leading slash not preceded by a drive letter is assumed to
be rooted at the install directory
(i.e.: /etc/V3.system.at = D:\Program Files\IBM\ldap\etc\V3.system.at).'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2364
DBNAME( 'slapdIncldeSchema'  'slapdIncldeSchema' )
ACCESS-CLASS critical
LENGTH 1024 )

attributetypes=( 1.3.18.0.2.4.2430
NAME 'ibm-slapdInvalidLine'
DESC 'This attribute will be prepended to the beginning of any
configuration attribute for which the value is invalid. This allows
invalid configuration settings to be identified with a simple search
for "ibm-slapdInvalidLine=*".'
EQUALITY 2.5.13.2
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
USAGE userApplications )

IBMAttributetypes=( 1.3.18.0.2.4.2430
DBNAME( 'slapdInvalidLine'  'slapdInvalidLine' )
ACCESS-CLASS normal
LENGTH 1024 )

attributetypes=( 1.3.18.0.2.4.2365
NAME 'ibm-slapdIpAddress'
DESC 'Specifies IP addresses the server will listen on.
These can be IPv4 or IPv6 addresses.  If the attribute is
not specified, the server uses all IP addresses assigned
to the host machine. This is supported on OS/400 only.'
EQUALITY 1.3.6.1.4.1.1466.109.114.1
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2365
DBNAME( 'slapdIpAddress'  'slapdIpAddress' )
ACCESS-CLASS system
LENGTH 32 )

attributetypes=( 1.3.18.0.2.4.2420
NAME 'ibm-slapdKrbAdminDN'
DESC 'Specifies the kerberos ID of the LDAP administrator
(e.g. ibm-kn=name@realm). Used when kerberos authentication
is used to authenticate the administrator when logged onto the
Web Admin interface. This is specified instead of adminDN and adminPW.'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2420
DBNAME( 'slapdKrbAdminDN'  'slapdKrbAdminDN' )
ACCESS-CLASS critical
LENGTH 512 )

attributetypes=( 1.3.18.0.2.4.2394
NAME 'ibm-slapdKrbEnable'
DESC 'Must be one of { TRUE | FALSE }.  Specifies whether the
server supports kerberos authentication.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2394
DBNAME( 'slapdKrbEnable'  'slapdKrbEnable' )
ACCESS-CLASS critical
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2419
NAME 'ibm-slapdKrbIdentityMap'
DESC 'If set to TRUE, when a client is authenticated with
a kerberos ID, the server will search for a local user
with matching kerberos credentials, and add that user
DN to the connections bind credentials. This allows ACLs
based on LDAP user DNs to still be usable with kerberos
authentication.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 S
INGLE-VALUE
USAGE directoryOperation )
```

```
IBMAttributetypes=( 1.3.18.0.2.4.2419
DBNAME( 'KrbIdentityMap'  'KrbIdentityMap' )
ACCESS-CLASS critical
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2416
NAME 'ibm-slapdKrbKeyTab'
DESC 'Specifies the LDAP servers keytab file. This file
contains the LDAP servers private key, as associated with
its kerberos account. This file should be protected (like the
servers SSL key database file).  On Windows, forward slashes
are allowed, and a leading slash not preceded by a drive
letter (D:) is assumed to be rooted at the install directory
(i.e.: /tmp/slapd.errors = D:\Program Files\IBM\ldap\tmp\slapd.errors).'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2416
DBNAME( 'slapdKrbKeyTab'  'slapdKrbKeyTab' )
ACCESS-CLASS critical
LENGTH 1024 )

attributetypes=( 1.3.18.0.2.4.2400
NAME 'ibm-slapdKrbRealm'
DESC 'Specifies the LDAP servers kerberos realm. Used to
publish the ldapservicename attribute in the root DSE. Note
that an LDAP server can serve as the repository of
account information for multiple KDCs (and realms), but the
LDAP server, as a kerberos server, can only be a member
of a single realm.'
EQUALITY 2.5.13.2
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2400
DBNAME( 'slapdKrbRealm'  'slapdKrbRealm' )
ACCESS-CLASS critical
LENGTH 256 )

attributetypes=( 1.3.18.0.2.4.2415
NAME 'ibm-slapdLdapCrlHost'
DESC 'Specify the hostname of the LDAP server that
contains the Certificate Revocation Lists (CRLs) for
validating client x.509v3 certificates.  This parameter
is needed when ibm-slapdSslAuth=serverclientauth AND the
client certificates have been issued for CRL validation'
EQUALITY 2.5.13.2
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2415
DBNAME( 'LdapCrlHost'  'LdapCrlHost' )
ACCESS-CLASS critical
LENGTH 256 )

attributetypes=( 1.3.18.0.2.4.2407
NAME 'ibm-slapdLdapCrlPassword'
DESC 'Specify the password that server-side SSL will use
to bind to the LDAP server that contains the Certificate
Revocation Lists (CRLs) for validating client x.509v3
certificates.  This parameter may be needed when
ibm-slapdSslAuth=serverclientauth AND the client certificates
have been issued for CRL validation.  Note:  If the LDAP
```

```
server holding the CRLs permits unauthenticated access to
the CRLs (i.e. anonymous access), then ibm-slapdLdapCrlPassword
is not required.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2407
DBNAME( 'CrlPassword'  'CrlPassword' )
ACCESS-CLASS critical )

attributetypes=( 1.3.18.0.2.4.2404 NAME 'ibm-slapdLdapCrlPort'
DESC 'Specify the LDAP ibm-slapdPort used by the LDAP server
that contains the Certificate Revocation Lists (CRLs) for validating
client x.509v3 certificates.  This parameter is needed when
ibm-slapdSslAuth=serverclientauth AND the client certificates
have been issued for CRL validation. (IP ports are unsigned,
16-bit integers in the range 1 - 65535)'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2404
 DBNAME( 'LdapCrlPort'  'LdapCrlPort' )
ACCESS-CLASS critical
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2403
NAME 'ibm-slapdLdapCrlUser'
DESC 'Specify the bindDN that server-side SSL will use to bind
to the LDAP server that contains the Certificate Revocation Lists (CRLs)
for validating client x.509v3 certificates.  This parameter may be needed
when ibm-slapdSslAuth=serverclientauth AND the client certificates have
been issued for CRL validation.  Note:  If the LDAP server holding the
CRLs permits unauthenticated access to the CRLs (i.e. anonymous
access), then ibm-slapdLdapCrlUser is not required.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2403
DBNAME( 'LdapCrlUser'  'LdapCrlUser' )
ACCESS-CLASS critical
LENGTH 1000 )

attributetypes=( 1.3.18.0.2.4.2409
NAME 'ibm-slapdMasterDN'
DESC 'Bind DN used by a replication supplier server. The value has to match
the replicaBindDN in the credentials object associated with the replication
agreement defined between the servers.
When kerberos is used to authenticate to the replica, ibm-slapdMasterDN
must specify the DN representation of the kerberos ID
(e.g. ibm-kn=freddy@realm1). When kerberos is used, MasterServerPW is ignored.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2409
DBNAME( 'MasterDN'  'MasterDN' )
ACCESS-CLASS critical
LENGTH 1000 )

attributetypes=( 1.3.18.0.2.4.2411
NAME 'ibm-slapdMasterPW'
DESC 'Bind password used by a replication supplier. The value has to
match the replicaBindPW in the credentials object associated with the replication
agreement defined between the servers. When kerberos is used, MasterServerPW
```

```
is ignored.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2411
DBNAME( 'MasterPW'  'MasterPW' )
ACCESS-CLASS critical )

attributetypes=( 1.3.18.0.2.4.2401
NAME 'ibm-slapdMasterReferral'
DESC 'URL of a master replica server (e.g.: ldaps://master.us.ibm.com:636)'
EQUALITY 2.5.13.2
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2401
DBNAME( 'MasterReferral'  'MasterReferral' )
ACCESS-CLASS critical
LENGTH 256 )

attributetypes=( 1.3.18.0.2.4.2412
NAME 'ibm-slapdMaxEventsPerConnection'
DESC 'Maximum number of event notifications which can be registered
per connection. Minimum = 0 (unlimited) Maximum = 2,147,483,647'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2412
DBNAME( 'EventsPerCon'  'EventsPerCon' )
ACCESS-CLASS critical
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2405
NAME 'ibm-slapdMaxEventsTotal'
DESC 'Maximum total number of event notifications which can
be registered for all connections. Minimum = 0 (unlimited)
Maximum = 2,147,483,647'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2405
DBNAME( 'MaxEventsTotal'  'MaxEventsTotal' )
ACCESS-CLASS critical
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2439
NAME 'ibm-slapdMaxNumOfTransactions'
DESC 'Maximum number of transactions active at one time.
0 = unlimited'
EQUALITY 2.5.13.29
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2439
DBNAME( 'MaxNumOfTrans'  'MaxNumOfTrans' )
ACCESS-CLASS critical
LENGTH 11
EQUALITY
ORDERING
SUBSTR
APPROX )
```

```
attributetypes=( 1.3.18.0.2.4.2385
NAME 'ibm-slapdMaxOpPerTransaction'
DESC 'Maximum number of operations per transaction.
0 = unlimited'
EQUALITY 2.5.13.29
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2385
DBNAME( 'MaxOpPerTrans'  'MaxOpPerTrans' )
ACCESS-CLASS critical
LENGTH 11
EQUALITY
ORDERING
APPROX )

attributetypes=( 1.3.18.0.2.4.2486
NAME 'ibm-slapdMaxPendingChangesDisplayed'
DESC 'Maximum number of pending replication updates to be
displayed for any given replication agreement on a supplier
server.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
USAGE userApplications )

IBMAttributetypes=( 1.3.18.0.2.4.2486
DBNAME( 'slapdMaxPendingCh'  'slapdMaxPendingCh' )
ACCESS-CLASS normal
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2386
NAME 'ibm-slapdMaxTimeLimitOfTransactions'
DESC 'The maximum timeout value of a pending transaction in
seconds. 0 = unlimited'
EQUALITY 2.5.13.29
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2386
DBNAME( 'MaxTimeOfTrans'  'MaxTimeOfTrans' )
ACCESS-CLASS critical
LENGTH 11
EQUALITY
ORDERING
 APPROX )

attributetypes=( 1.3.18.0.2.4.2500
NAME 'ibm-slapdMigrationInfo'
DESC 'Information used to control migration of a component.'
EQUALITY 2.5.13.2
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2500
DBNAME( 'slapdMigrationInf'  'slapdMigrationInf' )
ACCESS-CLASS critical
LENGTH 2048 )

attributetypes=( 1.3.18.0.2.4.2376
NAME 'ibm-slapdPagedResAllowNonAdmin'
DESC 'Whether or not the server should allow non-Administrator
bind for paged results requests on a search request.  If the
value read from the ibmslapd.conf file is TRUE, the server will
process any client request, including those submitted by a user
binding anonymously.  If the value read from the ibmslapd.conf
file is FALSE, the server will process only those client requests
```

```
submitted by a user with Administrator authority.  If a client
requests paged results with a criticality of TRUE or FALSE for a
search operation, does not have Administrator authority, and the
value read from the ibmslapd.conf file for this attribute is FALSE,
the server will return to the client with return code
insufficientAccessRights - no searching or paging will be performed.  '
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2376
DBNAME( 'SlapdPagedNonAdmn'  'SlapdPagedNonAdmn' )
ACCESS-CLASS critical
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2380
NAME 'ibm-slapdPagedResLmt'
DESC 'Maximum number of outstanding paged results search requests
allowed active simultaneously.  Range = 0.... If a client requests
a paged results operation, and a maximum number of outstanding paged
results are currently active, then the server will return to the
client with return code of busy - no searching or paging
will be performed.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2380
DBNAME( 'SlapdPagedResLmt'  'SlapdPagedResLmt' )
ACCESS-CLASS critical
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2379
NAME 'ibm-slapdPageSizeLmt'
DESC 'Maximum number of entries to return from search for an
individual page when paged results control is specified, regardless
of any pagesize that may have been specified on the client search
request. Range = 0.... If a client has passed a page size, then
the smaller value of the client value and the value read from
ibmslapd.conf will be used.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2379
DBNAME( 'SlapdPageSizeLmt'  'SlapdPageSizeLmt' )
ACCESS-CLASS critical
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2406
NAME 'ibm-slapdPlugin'
DESC 'A plugin is a dynamically loaded library which extends the
capabilities of the server. An ibm-slapdPlugin attribute specifies
to the server how to load and initialize a plugin library.
The syntax is:   keyword filename init_function [args...]
The syntax will be slightly different for each platform
due to library naming conventions.'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2406
DBNAME( 'slapdPlugin'  'slapdPlugin' )
ACCESS-CLASS critical
LENGTH 2000 )

attributetypes=( 1.3.18.0.2.4.2408
NAME 'ibm-slapdPort'
DESC 'TCP/IP ibm-slapdPort used for non-SSL connections. Can not
have the same value as ibm-slapdSecurePort. (IP ports are unsigned,
16-bit integers in the range 1 - 65535)'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2408
DBNAME( 'slapdPort'  'slapdPort' )
ACCESS-CLASS critical
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2402
NAME 'ibm-slapdPwEncryption'
DESC 'Must be one of { none | imask | crypt | sha }.  Specify the
encoding mechanism for the user passwords before they are stored in
the directory. Defaults to none if unspecified. If the value is set
other than none, SASL cram-md5 bind will fail.'
EQUALITY 2.5.13.2
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2402
DBNAME( 'PwEncryption'  'PwEncryption' )
ACCESS-CLASS critical
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2413
NAME 'ibm-slapdReadOnly'
 DESC 'Must be one of { TRUE | FALSE }.  Specifies whether
the backend can be written to. Defaults to FALSE if unspecified.
If set to TRUE, the server will return LDAP_UNWILLING_TO_PERFORM (0x35)
in response to any client request which would change data in the
readOnly database.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2413
DBNAME( 'ReadOnly'  'ReadOnly' )
ACCESS-CLASS critical
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2487
NAME 'ibm-slapdReferral'
DESC 'Specify the referral LDAP URL to pass back when the
local suffixes do not match the request. Used for superior referral
(i.e. ibm-slapdSuffix is not within the servers naming context).'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2487
DBNAME( 'Referral'  'Referral' )
ACCESS-CLASS critical LENGTH 32700 )

attributetypes=( 1.3.18.0.2.4.2434
NAME 'ibm-slapdReplDbConns'
DESC 'Maximum number of database connections for use by replication'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE userApplications )

IBMAttributetypes=( 1.3.18.0.2.4.2434
DBNAME( 'slapdReplDbConns'  'slapdReplDbConns' )
```

```
ACCESS-CLASS normal
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2367
NAME 'ibm-slapdReplicaSubtree'
DESC 'A DN identifying the top of a replicated subtree.'
EQUALITY 2.5.13.1
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
USAGE userApplications )

IBMAttributetypes=( 1.3.18.0.2.4.2367
DBNAME( 'slapdReplicaSubtr'  'slapdReplicaSubtr' )
ACCESS-CLASS normal
LENGTH 1000 )

attributetypes=( 1.3.18.0.2.4.2437
NAME 'ibm-slapdSchemaAdditions'
DESC 'File path on the ibmslapd host machine containing additional
schema definitions used by the LDCF backend. Standard values
are:  /etc/V3.modifiedschema  On Windows, forward slashes are
allowed, and a leading slash not preceded by a drive letter
is assumed to be rooted at the install directory
(i.e.: /etc/V3.system.at = D:\Program Files\IBM\ldap\etc\V3.system.at).'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2437
DBNAME( 'slapdSchemaAdditi'  'slapdSchemaAdditi' )
ACCESS-CLASS normal
LENGTH 1024 )

attributetypes=( 1.3.18.0.2.4.2363
NAME 'ibm-slapdSchemaCheck'
DESC 'Must be one of { V2 | V3 | V3_lenient }.
Specifies schema checking mechanism for add/modify operation.
V2 = perform LDAP v2 checking.
V3 = perform LDAP v3 checking.
V3_lenient = not ALL parent object classes are required. Only the immediate
object class is needed when adding entries.'
EQUALITY 2.5.13.2
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2363
DBNAME( 'SchemaCheck'  'SchemaCheck' )
ACCESS-CLASS critical
LENGTH 10 )

attributetypes=( 1.3.18.0.2.4.2398
NAME 'ibm-slapdSecurePort'
DESC 'TCP/IP port used for SSL connections. Can not have the same
value as ibm-slapdPort. (IP ports are unsigned, 16-bit integers in
the range 1 - 65535)'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2398
DBNAME( 'SecurePort'  'SecurePort' )
ACCESS-CLASS critical
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2399
NAME 'ibm-slapdSecurity'
DESC 'Must be one of { none | SSL | SSLOnly }.  Specifies types of connections
```

```
accepted by the server.
none - server listens on non-ssl port only.  ssl - server listens on
both ssl and non-ssl ports.  sslonly - server listens on ssl port only.'
EQUALITY 2.5.13.2
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2399
DBNAME( 'Security'  'Security' )
ACCESS-CLASS critical
LENGTH 7 )

attributetypes=( 1.3.18.0.2.4.2433
NAME 'ibm-slapdServerId'
DESC 'Identifies the server for use in replication'
EQUALITY 1.3.6.1.4.1.1466.109.114.1
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
SINGLE-VALUE
USAGE userApplications )

IBMAttributetypes=( 1.3.18.0.2.4.2433
DBNAME( 'slapdServerId'  'slapdServerId' )
ACCESS-CLASS normal
LENGTH 240 )

attributetypes=( 1.3.18.0.2.4.2397
NAME 'ibm-slapdSetenv'
DESC 'Server executes putenv() for all values of ibm-slapdSetenv at startup
to modify its own runtime environment. Shell variables (%PATH% or \24LANG)
will not be expanded. The only current use for this attribute is to set
DB2CODEPAGE=1208, which is required if using UCS-2 (Unicode) databases.'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2397
DBNAME( 'slapdSetenv'  'slapdSetenv' )
ACCESS-CLASS critical
LENGTH 2000 )

attributetypes=( 1.3.18.0.2.4.2396
NAME 'ibm-slapdSizeLimit'
DESC 'Maximum number of entries to return from search, regardless of any
sizelimit that may have been specified on the client search request.
Range = 0.... If a client has passed a limit, then the smaller value of
the client value and the value read from ibmslapd.conf will be used. If a
client has not passed a limit and has bound as admin DN, then the limit
will be considered unlimited. If the client has not passed a limit and
has not bound as admin DN, then the limit will be that which was read
from ibmslapd.conf file. 0 = unlimited.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2396
DBNAME( 'SizeLimit'  'SizeLimit' )
ACCESS-CLASS critical
LENGTH 11 )
attributetypes=( 1.3.18.0.2.4.2381 NAME 'ibm-slapdSortKeyLimit'
DESC 'Maximum number of sort conditions (keys) that can be specified on a
single search request.  Range = 0.... If a client has passed a search request
 with more sort keys than the limit allows, and the sorted search control
criticality is FALSE, then the server will honor the value read from
ibmslapd.conf and ignore any sort keys encountered after the limit has
been reached - searching and sorting will be performed. If a client has
passed a search request with more keys than the limit allows, and the
```

```
sorted search control criticality is TRUE, then the server will return to
the client with return code of adminLimitExceeded - no searching or sorting
will be performed.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2381
DBNAME( 'SlapdSortKeyLimit'  'SlapdSortKeyLimit' )
ACCESS-CLASS critical
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2377
NAME 'ibm-slapdSortSrchAllowNonAdmin'
DESC 'Whether or not the server should allow non-Administrator bind for
sort on a search request.  If the value read from the ibmslapd.conf file
is TRUE, the server will process any client request, including those
submitted by a user binding anonymously.  If the value read from the
ibmslapd.conf file is FALSE, the server will process only those client
requests submitted by a user with Administrator authority.  If a client
requests sort with a criticality of TRUE for a search operation, does not
have Administrator authority, and the value read from the ibmslapd.conf file
for this attribute is FALSE, the server will return to the client with return
code insufficientAccessRights - no searching or sorting will be performed.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2377
DBNAME( 'SlapdSortNonAdmin'  'SlapdSortNonAdmin' )
ACCESS-CLASS critical
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2395
NAME 'ibm-slapdSslAuth'
DESC 'Must be one of { serverauth | serverclientauth }.  Specify authentication
type for ssl connection.  serverauth - supports server authentication at the
client.  serverclientauth - supports both server and client authentication.'
EQUALITY 2.5.13.2
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2395
DBNAME( 'slapdSslAuth'  'slapdSslAuth' )
ACCESS-CLASS critical
LENGTH 16 )

attributetypes=( 1.3.18.0.2.4.2389
NAME 'ibm-slapdSslCertificate'
DESC 'Specify the label that identifies the servers Personal Certificate
in the key database file.  This label is specified when the servers private
key and certificate are created with the ikmgui application. If
ibm-slapdSslCertificate is not defined, the default private key, as defined
in the key database file, is used by the LDAP server for SSL connections.'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2389
DBNAME( 'SslCertificate'  'SslCertificate' )
ACCESS-CLASS critical
LENGTH 128 )

attributetypes=( 1.3.18.0.2.4.2429
NAME 'ibm-slapdSslCipherSpec'
```

```
DESC 'SSL Cipher Spec Value must be set to DES-56, RC2-40-MD5, RC4-128-MD5,
RC4-128-SHA, RC4-40-MD5, TripleDES-168, or AES.  It identifies the
allowable encryption/decryption methods for establishing a SSL connection
between LDAP clients and the server.'
EQUALITY 1.3.6.1.4.1.1466.109.114.1
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2429
DBNAME( 'slapdSslCipherSpe'  'slapdSslCipherSpe' )
ACCESS-CLASS normal
LENGTH 30 )

attributetypes=( 1.3.18.0.2.4.2362
NAME 'ibm-slapdSslCipherSpecs'
DESC 'This attribute is depricated in favor of ibm-slapdSslCipherSpec.
Specifies a decimal number which identifies the allowable
encryption/decryption methods for establishing a SSL connection
between LDAP client(s) and the server.  This number represents the availability
of the encryption/decryption methods supported by the LDAP server.
The pre-defined Cipher values and their descriptions are:
SLAPD_SSL_TRIPLE_DES_SHA_US  0x0A Triple DES encryption with a 168-bit key
                             and a SHA-1 MAC
SLAPD_SSL_DES_SHA_US 0x09DES encryption with a 56-bit key and a SHA-1 MAC
SLAPD_SSL_RC4_SHA_US 0x05 RC4 encryption with a 128-bit key and a SHA-1 MAC
SLAPD_SSL_RC4_MD5_US  0x04 RC4 encryption with a 128-bit key and a MD5 MAC
SLAPD_SSL_RC4_MD5_EXPORT 0x03 RC4 encryption with a 40-bit key and a MD5 MAC
SLAPD_SSL_RC2_MD5_EXPORT 0x06 RC2 encryption with a 40-bit key and a MD5 MAC'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2362
DBNAME( 'SslCipherSpecs'  'SslCipherSpecs' )
ACCESS-CLASS critical
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2375
NAME 'ibm-slapdSSLKeyDatabase'
DESC 'File path to the LDAP servers SSL key database file. This key database
file is used for handling SSL connections from LDAP clients, as well as for
creating secure SSL connections to replica LDAP servers.  On Windows, forward
slashes are allowed, and a leading slash not preceeded by a drive
specifier (D:) is assumed to be rooted at the install directory
(i.e.:  /etc/key.kdb = D:\Program Files\IBM\ldap\etc\key.kdb).'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2375
DBNAME( 'slapdSSLKeyDataba'  'slapdSSLKeyDataba' )
ACCESS-CLASS critical
LENGTH 1024 )

attributetypes=( 1.3.18.0.2.4.2438 NAME 'ibm-slapdSSLKeyDatabasePW'
DESC 'Specify the password associated with the LDAP servers SSL key database file,
as specified on the ibm-slapdSslKeyDatabase parameter.  If the LDAP servers key
database file has an associated password stash file, then the
ibm-slapdSslKeyDatabasePW parameter can be ommitted, or set to
ibm-slapdSslKeyDatabasePW = none.  Note that the password stash file must
be located in the same directory as the key database file and it must have
the same file name as the key database file, but with an extension of .sth,
instead of .kdb'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
SINGLE-VALUE
USAGE directoryOperation )
```

```
IBMAttributetypes=( 1.3.18.0.2.4.2438
DBNAME( 'slapdSSLKeyDPW'  'slapdSSLKeyDPW' )
ACCESS-CLASS normal )

attributetypes=( 1.3.18.0.2.4.2392
NAME 'ibm-slapdSslKeyRingFile'
DESC 'file path to the LDAP servers SSL key database file. This key database
file is used for handling SSL connections from LDAP clients, as well as for
creating secure SSL connections to replica LDAP servers.  On Windows, forward
slashes are allowed, and a leading slash not preceeded by a drive
specifier (D:) is assumed to be rooted at the install directory
(i.e.:  /etc/key.kdb = D:\Program Files\IBM\ldap\etc\key.kdb).'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2392
DBNAME( 'SslKeyRingFile'  'SslKeyRingFile' )
ACCESS-CLASS critical
LENGTH 1024 )

attributetypes=( 1.3.18.0.2.4.2390
NAME 'ibm-slapdSslKeyRingFilePW'
DESC 'Specify the password associated with the LDAP servers SSL key database
file, as specified on the ibm-slapdSslKeyRingFile parameter.  If the LDAP servers
key database file has an associated password stash file, then the
ibm-slapdSslKeyRingFilePW parameter can be ommitted, or set to
ibm-slapdSslKeyRingFilePW = none.  Note that the password stash file must be
located in the same directory as the key database file and it must have the
same file name as the key database file, but with an extension of .sth,
instead of .kdb.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2390
DBNAME( 'SslKeyRingFilePW'  'SslKeyRingFilePW' )
ACCESS-CLASS critical )

attributetypes=( 1.3.18.0.2.4.2388
NAME 'ibm-slapdSuffix'
DESC 'Specifies a naming context to be stored in this backend.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2388
DBNAME( 'slapdSuffix'  'slapdSuffix' )
ACCESS-CLASS critical
LENGTH 1000 )

attributetypes=( 1.3.18.0.2.4.2480
NAME 'ibm-slapdSupportedWebAdmVersion'
DESC 'This attribute defines the earliest version of the web administration
console that supports configuration of this server.'
EQUALITY 2.5.13.2
ORDERING 2.5.13.3
SUBSTR 2.5.13.4
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2480
DBNAME( 'slapdSupWebAdmVer'  'slapdSupWebAdmVer' )
ACCESS-CLASS normal
LENGTH 256 )
```

```
attributetypes=( 1.3.18.0.2.4.2393
NAME 'ibm-slapdSysLogLevel'
DESC 'Must be one of { l | m | h }.  Level at which debugging
and operation statistics are logged in ibmslapd.log file.
h - high (verbose), m - medium, l - low (terse).'
EQUALITY 2.5.13.2
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2393
DBNAME( 'SysLogLevel'  'SysLogLevel' )
ACCESS-CLASS critical
LENGTH 1 )

attributetypes=( 1.3.18.0.2.4.2391
NAME 'ibm-slapdTimeLimit'
DESC 'Maximum number of number of seconds to spend on search
request, regardless of any timelimit that may have been specified
on the client request. Range = 0.... If a client has passed a limit,
then the smaller value of the client value and the value read from
ibmslapd.conf will be used. If a client has not passed a limit and has
bound as admin DN, then the limit will be considered unlimited. If
the client has not passed a limit and has not bound as admin DN,
then the limit will be that which was read from
ibmslapd.conf file. 0 = unlimited.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2391
DBNAME( 'TimeLimit'  'TimeLimit' )
ACCESS-CLASS critical
LENGTH 11 )

attributetypes=( 1.3.18.0.2.4.2384
NAME 'ibm-slapdTransactionEnable'
DESC 'If FALSE, globally disables transaction support; the server
will reject all StartTransaction requests with the response
LDAP_UNWILLING_TO_PERFORM.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2384
DBNAME( 'TransactionEnable'  'TransactionEnable' )
ACCESS-CLASS critical
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2499
NAME 'ibm-slapdUseProcessIdPW'
DESC 'If set to true the server will use the user login id associated
with the ibmslapd process to connect to the database.  If set to false
the server will use the ibm-slapdDbUserID and ibm-slapdDbUserPW
values to connect to the database.'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2499
DBNAME( 'useprocidpw'  'useprocidpw' )
ACCESS-CLASS normal
LENGTH 5 )

attributetypes=( 1.3.18.0.2.4.2436
NAME 'ibm-slapdVersion'
DESC 'IBM Slapd version Number'
EQUALITY 2.5.13.5
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
USAGE directoryOperation )

IBMAttributetypes=( 1.3.18.0.2.4.2436
DBNAME( 'slapdVersion'  'slapdVersion' )
ACCESS-CLASS normal
LENGTH 1024 )
```

Dynamically-changed attributes

The following is a list of attributes that are changed dynamically. You do not have to restart the server for these changes to take effect.

Cn=Configuration

- ibm-slapdadmindn
- ibm-slapdadminpw
- ibm-slapderrorlog
- ibm-slapdpwencryption
- ibm-slapdsizelimit
- ibm-slapdsysloglevel
- ibm-slapdtimelimit

cn=Front End, cn=Configuration

- ibm-slapdaclcache
- ibm-slapdaclcachesize
- ibm-slapdentrycachesize
- ibm-slapdfiltercachebypasslimit
- ibm-slapdfiltercachesize
- ibm-slapdidletimeout

cn=Event Notification, cn=Configuration

- ibm-slapdmaxeventsperconnection
- ibm-slapdmaxeventstotal

cn=Transaction, cn=Configuration

- ibm-slapdmaxnumoftransactions
- ibm-slapdmaxoppertransaction
- ibm-slapdmaxtimelimitoftransactions

cn=ConfigDB, cn=Config Backends, cn=IBM Directory, cn=Schemas, cn=Configuration

- ibm-slapdreadonly

cn=Directory, cn=RDBM Backends, cn=IBM Directory, cn=Schemas, cn=Configuration

- ibm-slapdbulkloaderrors
- ibm-slapdclierrors
- ibm-slapdpagedresallownonadmin
- ibm-slapdpagedreslmt
- ibm-slapdpagesizelmt
- ibm-slapdreadonly
- ibm-slapdsortkeylimit

- ibm-slapdsortsrchallownonadmin
- ibm-slapdsuffix

Appendix F. Notices

This information was developed for products and services offered in the U.S.A. IBM might 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 in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

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

The following paragraph does not apply 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 information. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this information 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.

Trademarks

AIX	DB2	IBM	OS/400	SecureWay®	World Registry	z/OS

Glossary

Use this section to locate definitions of some of the IBM Directory product terms

access control lists (ACLs)
Access Control Lists (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 distinguished name of the object as well as a set of attributes and their corresponding values.

access control groups
Groups to be used for access control. Each group contains a multivalued attribute consisting of member DNs. Access control groups have an object class of ′AccessGroup′.

access permissions
There are two sets of access permissions:

- Permissions that apply to an entire object
- Permissions that apply to attribute access classes or individual attributes.

aclEntry
aclEntry is a multivalue attribute that contains information pertaining to the access allowed to the entry object and each of its attributes. An aclEntry lists the following types of information:

- Who has rights to the entity object (scope of the protection).
- What attributes or classes of attributes the user has access to (attribute access classes).
- What rights the user or group has (permission).

aclPropagate
ACLs can be set on any object in the tree. As is typical in a hierarchical file system, LDAP access control lists can propagate down through the directory hierarchy. These ACLs, called propagating ACLs, have the aclPropagate attribute = true. All children of this object now inherits the ACL set at that point. In order to specify an ACL different from that of its parent, this new ACL must be explicitly set.

aclSource
Each object has an associated aclSource attribute. This contains the DN of the entry in which the ACL is defined. This attribute is kept by the server, but might be retrieved for administrative purposes.

aliases
Aliases can be used within LDAP to reference entries anywhere within the directory tree. An alias is simply a pointer to another directory object.

Aliases Objects are of ′objectclass=aliasObject′. The mandatory attribute in this class, ′aliasedObjectName′, contains the full DN of another directory object (the one to which the alias refers).

On the C API, by default, aliases objects are not dereferenced during a search operation. The client may request dereferencing using a flag on the command line. Aliases may be dereferenced when locating the base entry of the search. If the object specified as the base is an alias object, the object will be derefenced before beginning the search.

For example, an object with dn ″cn=personOfTheWeek, o=Corporation, c=US″ which has the attribute aliasedObjectName: ″cn=personA, o=Corporation,c=US″. With ′deref finding′ set, a search base of ″cn=personOfTheWeek, o=Corporation, c=US″ is dereferenced to ″cn=personA, o=Corporation,c=US″. This now becomes the base for the search.

Another possibility is to dereference aliases during searching. In this case, the dn used as the base is the one given by the client, but alias entries found during the search are dereferenced.

An example of this might be a search for ″cn=*week*″ with a base of o=Corporation, c=US″. While the located Node is ″cn=personOfTheWeek, o=Corporation, c=US″ this object would be dereferenced and the entry ″cn=personA, o=Corporation,c=US″ is returned as the search result.

A dereference of ′all′ might also be used. This means that alias entries are dereferenced both when locating the search base and when objects are found during the search operation.

attribute access classes
Attributes requiring similar permission for access are grouped together in classes. Attributes are assigned to an access class within the schema files. The three user-modifiable access classes are :

- Normal
- Sensitive
- Critical

bulkload
A command line utility that is used for bulk-loading large amounts of data in LDIF format.

cascading replication
A cascading replication is a replication topology in which there are multiple tiers of servers. A peer/master server replicates to a small set of read-only servers which in turn replicate to other servers. Such a topology off-loads replication work from the master servers.

consumer server
A consumer server is a server which receives changes through replication from another [supplier] server.

directory schema
Entries in a directory are made up of a collection of attributes and their associated values. Attributes might have one or more values. In order to identify a particular value in an entry, the attribute type name is specified along with the value, as in ″cn=John Doe″. This is referred to as an attribute:value pair. 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 that 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 might be present.

distinguished names (DNs)
Every entry in a 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
```

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 first component of the DN is referred to as the Relative Distinguished Name (RDN). It identifies an entry distinctly from any other entries that have the same parent.

dynamic groups
Dynamic groups are groups that are defined using a search expression. When an attribute is added to a directory entry, causing it to match the search expression, the entry automatically becomes a member of the group. In addition, simple, efficient methods are provided for client applications to:

- Test whether a specific entry is a member of a specific group.
- List all the members of a specific group.
- List all the groups to which a specific entry belongs .

The search expressions can be used in combination with other group attributes.

These groups can be used for access control.

entryOwner
Each object has an associated entryOwner attribute. The entryOwner attribute might be a user or a group, similar to what is allowed within the aclEntry. However, the entryOwner subject has certain privileges over the object. Entry owners are in

essence the administrators for particular objects. They have full access on that particular object, similar to the administrator DN. The administrator has full permission on any object in the database.

forwarding server
A forwarding server is a read-only server that replicates all changes sent to it. This contrasts to a peer/master server in that a peer/master server does not replicate changes sent to it from another peer/master server; it only replicates changes that are originally made on the peer/master server.

groups
There are two type of groups:

- Normal groups
- Groups to be used for access control

Normal groups have an object class of 'GroupOfNames', 'GroupOfUniqueNames' or a user defined group. Access control groups have an object class of 'AccessGroup'.

Each group object contains a multivalued attribute consisting of member DNs. Groups cannot contain group DNs.

gsk6ikm
The gsk6ikm utility is used to create public-private key pairs and certificate requests, receive certificate requests into a key database, and manage keys in a key database. gsk6ikm utilizes a graphical user interface. It provides you with the information you need to perform a task. If you make an error, it issues a message and prompts you again for the information.

indexing rules
Index rules attached to attributes make it possible to retrieve information faster. The IBM Directory Server provides the following indexing rules:

- Equality
- Approximate
- Substring
- Reverse

See "Indexing rules" on page 87.

ldapadd
The LDAP modify-entry and LDAP add-entry tool ldapmodify is a shell-accessible interface to the ldap_modify and ldap_add library calls. **ldapadd** is implemented as a renamed version of **ldapmodify**. When invoked as ldapadd the **-a** (add new entry) flag is turned on automatically.

ldapdelete
The LDAP delete-entry tool ldapdelete is a shell-accessible interface to the ldap_delete library call. ldapdelete opens a connection to an LDAP server and binds and deletes one or more entries. If one or more dn arguments are provided, entries with those Distinguished Names (DN) are deleted. Each DN should be a string-represented DN.

ldapmodify
The LDAP modify-entry and LDAP add-entry tools ldapmodify is a shell-accessible interface to the ldap_modify and ldap_add library calls. **ldapadd** is implemented as a renamed version of **ldapmodify**. When invoked as ldapadd the **-a** (add new entry) flag is turned on automatically.

ldapmodrdn
LDAP modify-entry RDN tool ldapmodrdn is a shell-accessible interface to the ldap_modrdn library call. **ldapmodrdn** opens a connection to an LDAP server and binds and modifies the RDN of entries. The entry information is read from standard input, from a file, through the use of the **- f** option, or from the command-line pair DN and RDN.

ldapsearch
The LDAP search tool ldapsearch is a shell-accessible interface to the ldap_search library call. **ldapsearch** opens a connection to an LDAP server and binds and performs a search using the filter . The filter should conform to the string representation for LDAP filters.

LDIF LDAP Data Interchange Format (LDIF), as used by the **ldapmodify**, **ldapadd**, and **ldapsearch** command-line utilities is used to represent LDAP entries in a standard portable text form.

The LDIF tool **ldif** is a shell-accessible utility that converts arbitrary data values to LDIF. It reads input values from standard input and produces LDIF records.

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

matching rules
: Matching rules describe how to perform a comparison. Supported matching rules are:

```
caseExactIA5Match
caseExactMatch
caseExactOrderingMatch
caseExactSubstringsMatch
caseIgnoreIA5Match
caseIgnoreMatch
caseIgnoreOrderingMatch
caseIgnoreSubstringsMatch
distinguishedNameMatch
distinguishedNameOrderingMatch
generalizedTimeMatch
generalizedTimeOrderingMatch
integerFirstComponentMatch
integerMatch
objectIdentifierFirstComponentMatch
objectIdentifierMatch
octetStringMatch
telephoneNumberMatch
telephoneNumberSubstringsMatch
uTCTimeMatch
```

multiple values
: Multiple values are used to assign more than one value to an attribute. The attribute can have multiple values, for example, to accommodate a maiden and married last name. To add multiple values to an attribute, click **Multiple values**, then add one value per line. If an attribute contains multiple values, the field displays as a drop-down list.

nested groups
: Nesting of groups enables the creation of hierarchical relationships that can be used to define inherited group membership. A nested group is defined as a child group entry whose DN is referenced by an attribute contained within a parent group entry. A new attribute has been defined to explicitly distinguish nested groups from ordinary members.

Nested subtree
: A nested subtree is a subtree within another subtree of the directory.

object class definitions
: Every entry contains an objectClass attribute that identifies what type of information the entry contains. The object class dictates which other attributes can be present in an entry. The directory schema defines the valid attribute types and object classes that can 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 might be present.

object class types
: Object classes can be structural, for example, **person**; abstract, for example **top**; or auxiliary, for example **ePerson**.

ownerPropagate
: 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 this particular object.

ownerSource
: Each object also has an associated ownerSource attribute. This contains the DN of the entry in which the owner values are defined. This attribute is maintained by the server but can be retrieved for administrative purposes.

quiesce
: To put the server into a state in which it does not accept client updates, except for those done by the administrator and accompanied by replication management control.

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

The general format for a referral is: **ldap[s]://hostname:port**. Typically the format for a referral to a nonsecure server is: **ldap://hostname:389** and to a secure SSL server is: **ldaps://hostname:636**. See "Referrals" on page 38 for additional information.

relative distinguished name (RDN)
The relative distinguished name (RDN) is the first component of the distinguished name (DN). For example, if the entries DN is cn=John Doe,ou=Test,o=IBM,c=US, the RDN is cn=John Doe.

replicas
A replica is a server that runs a copy of the directory. This replicated server can keep a copy of the entire directory or just one tree of that directory. Any update to a replica server is referred to the master server. If the master server fails, you have a copy of the directory trees on the replica server. Using the replica server also improves the response time.

replicated subtree
A replicated subtree is a portion of the DIT that is replicated from one server to another. Under this design, a given subtree can be replicated to some servers and not to others. A subtree can be writable on a given server, while other subtrees may be read-only.

replication agreement
A replication agreement is information contained in the directory that defines the ″connection″ or ″replication path″ between two servers. One server is called the supplier (the one that sends the changes) and the other is the consumer (the one that receives the changes). The agreement contains all the information needed for making a connection from the supplier to the consumer and scheduling replication

roles Roles are like groups, but contain special permissions granted by the Administrator.

Secure Sockets Layer (SSL)
The IBM Directory Server has the ability to protect LDAP access with Secure Sockets Layer security. When using SSL to secure LDAP communications with the IBM Directory Server, the SASL authentication mechanism, known as External, is used to authenticate the server, or the server and the client based on X.509 certificates.

sorted search
The sorted search control allows a client to receive search results sorted based on a list of criteria, where each criteria represents a sort key. This moves the responsibility of sorting from the client application to the server, where it might be done more efficiently. For example, you might want to sort a list of employees by surname, common name, and telephone number. Instead of building the search list twice so it can be sorted (once at the server and then again at the client after all the results are returned), the search list is built only once, and then sorted, before returning the results to the client application.

suffixes
A suffix is a DN that identifies the top entry in a locally held directory hierarchy. Because of the relative naming scheme used in LDAP, this DN is also the suffix of every other entry within that directory hierarchy. A directory server might have multiple suffixes, each identifying a locally held directory hierarchy.

supplier server
A supplier server is a server that sends changes to another [consumer] server.

syntax Syntax refers to the required format for data. Supported syntaxes are:

```
IBM Attribute Type Description
     Matching Rule Description
     Name Form Description
     Attribute Type Description
     Object Class Description
     DIT Structure Rule Description
     DIT Content Rule Description
     LDAP Syntax Description
     OID
     Matching Rule Use Description
     Boolean - TRUE/FALSE
     Binary - octet string
     INTEGER - integral number
     Generalized Time
     IA5 String - case-sensitive string
     Directory String - case-insensitive
                        string
     UTC time
     Telephone Number
     DN - distinguished name
```

Index

Printed in U.S.A.