



iSeries

IBM Directory Server (LDAP)

Version 5 Release 3





@server

iSeries

IBM Directory Server (LDAP)

Version 5 Release 3

Note

Before using this information and the product it supports, be sure to read the information in "Notices," on page 231.

Seventh Edition (August 2005)

This edition applies to version 5, release 3, modification 0 of IBM Operating System/400 (product number 5722-SS1) and to all subsequent releases and modifications until otherwise indicated in new editions. This version does not run on all reduced instruction set computer (RISC) models nor does it run on CISC models.

© Copyright International Business Machines Corporation 1998, 2005. All rights reserved.

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

Contents

Chapter 1. IBM Directory Server for iSeries (LDAP)	1
---	----------

Chapter 2. What's new for V5R3	3
---	----------

Chapter 3. Printable PDF	5
---	----------

Chapter 4. Directory Server Concepts	7
---	----------

Directories	7
Distinguished names (DNs)	11
Suffix (naming context)	14
Schema	15
IBM Directory Server schema	16
Common schema support	18
Object classes	18
Attributes	19
Object identifier (OID)	26
The subschema entries	27
The IBMsubschema object class	27
Schema queries	27
Dynamic schema	27
Disallowed schema changes	28
Schema checking	31
iPlanet compatibility	32
Generalized and UTC time	33
Publishing	34
Replication	35
Replication overview	36
Replication terminology	37
Replication agreements	38
How replication information is stored in the server	39
Security considerations for replication information	39
Realms and user templates	40
National language support (NLS) considerations	40
LDAP directory referrals	41
Transactions	41
Directory Server security	41
Auditing	42
Secure Sockets Layer (SSL) and Transport Layer Security with the Directory Server	42
Kerberos authentication with the Directory Server	42
Groups and roles	43
Access control lists	49
Ownership of LDAP directory objects	60
Password policy	61
Authentication	64
Operating system projected backend	67
i5/OS user projected directory information tree	67
LDAP operations	68
Administrator and replica bind DN's	72
i5/OS user-projected schema	72
Read access to projected users	72
Directory Server and i5/OS journaling support	73

Operational attributes	73
Controls and extended operations	74

Chapter 5. Get started with Directory Server	79
---	-----------

Migration considerations	79
Migrate to V5R3 from V5R2 or V5R1	79
Migrate data from V4R3, V4R4, or V4R5 to V5R3	80
Migrating a network of replicating servers	81
Kerberos service name change	83
Plan your Directory Server	84
Configure Directory Server	84
Default configuration for Directory Server	85
Web administration	86
Set up Web administration for the first time	87
Web administration tool	88

Chapter 6. Scenario: MyCo, Inc. sets up a Directory Server	91
---	-----------

Scenario details: Set up the Directory Server	92
Scenario details: Create the directory database	93
Scenario details: Publish the iSeries data to the directory database	95
Scenario details: Enter information into the directory database	96
Scenario details: Test the directory database	97

Chapter 7. Administer Directory Server	99
---	-----------

Start the Directory Server	100
Stop the Directory Server	100
Check the status of the directory server	101
Check jobs on the Directory Server	101
Enable event notification	101
Specify transaction settings	101
Change the port or IP address	102
Set password policy	102
Import an LDIF file	103
Export an LDIF file	103
Specify a server for directory referrals	103
Add and remove Directory Server suffixes	104
Save and restore Directory Server information	104
Work with administrative access for authorized users	105
Track access and changes to the LDAP directory	105
Enable object auditing for the Directory Server	106
Adjust search settings	106
Adjust performance settings	107
Manage replication	107
Create a master-replica topology	107
Create a master-forwarder-replica topology	113
Overview of creating a complex replication topology	114
Create a complex topology with peer replication	115
Manage topologies	117
Modify replication properties	120

Create replication schedules	121
Manage queues	123
Enable SSL on the Directory Server	123
Enable Kerberos authentication on the Directory Server	125
Manage the schema	125
View object classes	126
Add an object class	127
Edit an object class	128
Copy an object class	129
Delete an object class	130
View attributes	130
Add an attribute	131
Edit an attribute	132
Copy an attribute	133
Delete an attribute	134
Copy the schema to other servers	135
l Enable or disable read access to projected users	136
Manage directory entries	136
Browse the tree	137
Add an entry	137
Delete an entry	137
Edit an entry	138
Copy an entry	138
Edit access control lists	139
Add an auxiliary object class	139
Delete an auxiliary class	139
Change group membership	139
Search the directory entries	140
Change binary attributes	142
Manage users and groups	142
Manage users	143
Manage groups	144
Manage realms and user templates	145
Create a realm	146
Create a realm administrator	146
Create a template	147
Add the template to a realm	149
Create groups	149
Add a user to the realm	149
Manage realms	149
Manage templates	150
Manage access control lists (ACLs)	153
Effective ACLs	153
Effective owners	154
Non-filtered ACLs	154
Filtered ACLs	155

Owners	157
Publish information to the directory server	157

Chapter 8. Troubleshoot Directory Server 159

Monitor errors and access with the Directory Server job log	160
Use TRCTCPAPP to help find problems	160
Use the LDAP_OPT_DEBUG option to trace errors	161
Common LDAP client errors	161
ldap_search: Timelimit exceeded	162
[Failing LDAP operation]: Operations error	162
ldap_bind: No such object	162
ldap_bind: Inappropriate authentication	162
[Failing LDAP operation]: Insufficient access [failing LDAP operation]: Cannot contact LDAP server	162
[failing LDAP operation]: Failed to connect to SSL server	163

Chapter 9. Reference 165

Command line utilities	165
ldapmodify and ldapadd	165
ldapdelete	168
ldapexop	170
ldapmodrdn	175
ldapsearch	177
ldapchangepwd	185
ldapdiff	187
Notes about using SSL with the LDAP command line utilities	190
LDAP data interchange format (LDIF)	190
LDIF example	191
Version 1 LDIF support	191
Version 1 LDIF examples	192
Directory Server configuration schema	193
Directory information tree	193
Attributes	202

Chapter 10. Related information 229

Appendix. Notices 231

Trademarks	233
Terms and conditions for downloading and printing information	233

Chapter 1. IBM Directory Server for iSeries (LDAP)

IBM® Directory Server for iSeries™ (hereafter referred to as Directory Server) provides a Lightweight Directory Access Protocol (LDAP) server on the iSeries server. LDAP runs over Transmission Control Protocol/Internet Protocol (TCP/IP) and is popular as a directory service for both Internet and non-Internet applications.

The following topics provide you with information to help you understand and use the Directory Server on your iSeries server:

Chapter 2, “What’s new for V5R3,” on page 3

Information about the changes and improvements made to Directory Server since the last release.

Chapter 3, “Printable PDF,” on page 5

A PDF version of this information topic.

Chapter 4, “Directory Server Concepts,” on page 7

Information about Directory Server concepts.

Chapter 5, “Get started with Directory Server,” on page 79

Information related to configuring the Directory Server.

Chapter 6, “Scenario: MyCo, Inc. sets up a Directory Server,” on page 91

An example of how to set up an LDAP directory on the Directory Server.

Chapter 7, “Administer Directory Server,” on page 99

Information about working with the Directory Server.

Chapter 8, “Troubleshoot Directory Server,” on page 159

Information to help you solve problems. Includes suggestions for collecting service data and solving specific problems.

Chapter 9, “Reference,” on page 165

Reference material related to Directory Server such as command line utilities and LDIF information.

Chapter 10, “Related information,” on page 229

Additional information related to Directory Server.

Chapter 2. What's new for V5R3

Directory Server for iSeries (formerly known as IBM Directory Server for iSeries) has the following enhancements and new features for V5R3:

- **Administration and user accessibility:** The new IBM Directory Server Web Administration Tool replaces the IBM Directory Management Tool. The Web administration tool includes the functionality to administer the user entries, the directory server processes, and the directory tree from one common Web interface. LDAP protocol is now used to query and update the configuration options of the Directory Server.
- **Dynamic Groups:** Dynamic groups allow a group to be created where the members are entries that match a search filter.
- **Nested Groups:** Nested groups allow a group to be created whose members include all of the members of other groups.
- **Password policy:** The directory server now supports a password policy that includes password syntax rules, password history, and disabling entries after too many attempts to use incorrect passwords.
- **Filter-based access controls:** Authority to entries can now be specified using filter-based access control. For example, you can specify permissions to entries with `departmentNumber=abc` or grant access to specific types of entries.
- **Replication:** Replication improvements include the ability to have multiple master servers (peer servers), replication of subtrees, improved scheduling and control of replication, improved monitoring, and more robust replication function.
- **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. The `ldapsearch` command has been enhanced with new parameters to allow the search results to be sorted. There are also new LDAP APIs for sorting search results.
- **Paged Search:** The paged results control allows 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. The `ldapsearch` command has been enhanced with new parameters to allow the search results to be paged. There are also new LDAP APIs for paging search results.
- **Command line utilities:** The following command line utilities are new:
 - `ldapexop` - 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.
 - `ldapdiff` - synchronizes a replica server with its master.
 - `ldapchangepwd` - sends modify password requests to an LDAP server.
- **Performance:** Performance is improved for all operations. In addition, all operations are now allowed to be performed simultaneously by multiple clients.
- **Special Characters in Distinguished Names (DN):** A DN may now contain the following special characters: commas, equals, plus, less than, greater than, pound, semicolon, backslash, and quotation marks.
- **Matching Rules for String Attributes:** If an attribute is defined with one of the two string syntaxes, Directory String or IA5 String, the server will now honor the matching behavior specified in the schema for the attribute, correcting an error in previous releases. You can define an attribute to be case sensitive or to ignore case when matching. Previously the server allowed a matching rule to be defined, but ignored it. Internally the server treated IA5 String as case sensitive, and Directory String as case insensitive. If your server had defined attributes as IA5 String with `caseIgnoreMatch`, or DirectoryString with `caseExactMatch`, the server will now behave correctly for those attributes.



| **What's new as of 30 October 2005**

| **"Read access to projected users" on page 72**

| This is a new topic that provides information about the capability to enable or disable read access to
| projected users.

How to see what's new or changed:

To help you see where technical changes have been made, this information uses:

- The  image to mark where new or changed information begins.
- The  image to mark where new or changed information ends.

Chapter 3. Printable PDF

To view or download the PDF version of this document, select Directory Server (LDAP) (about 2700 KB).

Other information


To view or print PDFs of related manuals and Redbooks™, see Chapter 10, “Related information,” on page 229.

Saving PDF files

To save a PDF file on your workstation for viewing or printing:

1. Right-click the PDF file in your browser (right-click the link above).
2. Click the option that saves the PDF locally.
3. Navigate to the directory in which you want to save the PDF file.
4. Click **Save**.

Downloading Adobe Reader

You need Adobe Reader installed on your system to view or print these PDFs. You can download a free copy from the Adobe Web site (www.adobe.com/products/acrobat/readstep.html)  .

Chapter 4. Directory Server Concepts

Directory Server 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. For concepts and considerations related to Directory Server, see the following:

- “Directories”
- “Distinguished names (DNs)” on page 11
- “Suffix (naming context)” on page 14
- “Schema” on page 15
- “Publishing” on page 34
- “Replication” on page 35
- “Realms and user templates” on page 40
- “National language support (NLS) considerations” on page 40
- “LDAP directory referrals” on page 41
- “Transactions” on page 41
- “Directory Server security” on page 41
- “Operating system projected backend” on page 67
- “Directory Server and i5/OS journaling support” on page 73
- “Operational attributes” on page 73
- “Controls and extended operations” on page 74

Directories

The Directory Server allows access to a type of database that stores information in a hierarchical structure similar to the way that the i5/OS™ integrated file system is organized.

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 multiple servers, 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.

The LDAP directory server model is based on entries (which are also referred to as objects). Each entry consists of one or more attributes, such as a name or address, and a type. The types typically consist of mnemonic strings, such as `cn` for common name or `mail` for e-mail address.

The example directory in Figure 1 on page 9 shows an entry for Tim Jones that includes `mail` and `telephoneNumber` attributes. Some other possible attributes include `fax`, `title`, `sn` (for surname), and `jpegPhoto`.

Each directory has a schema, which is a set of rules that determine the structure and contents of the directory. You can view the schema using the Web administration tool. For more information about the schema, see “Schema” on page 15.

Each directory entry has a special attribute called `objectClass`. This attribute controls which attributes are required and allowed in an entry. In other words, the values of the `objectClass` attribute determine the schema rules the entry must obey.

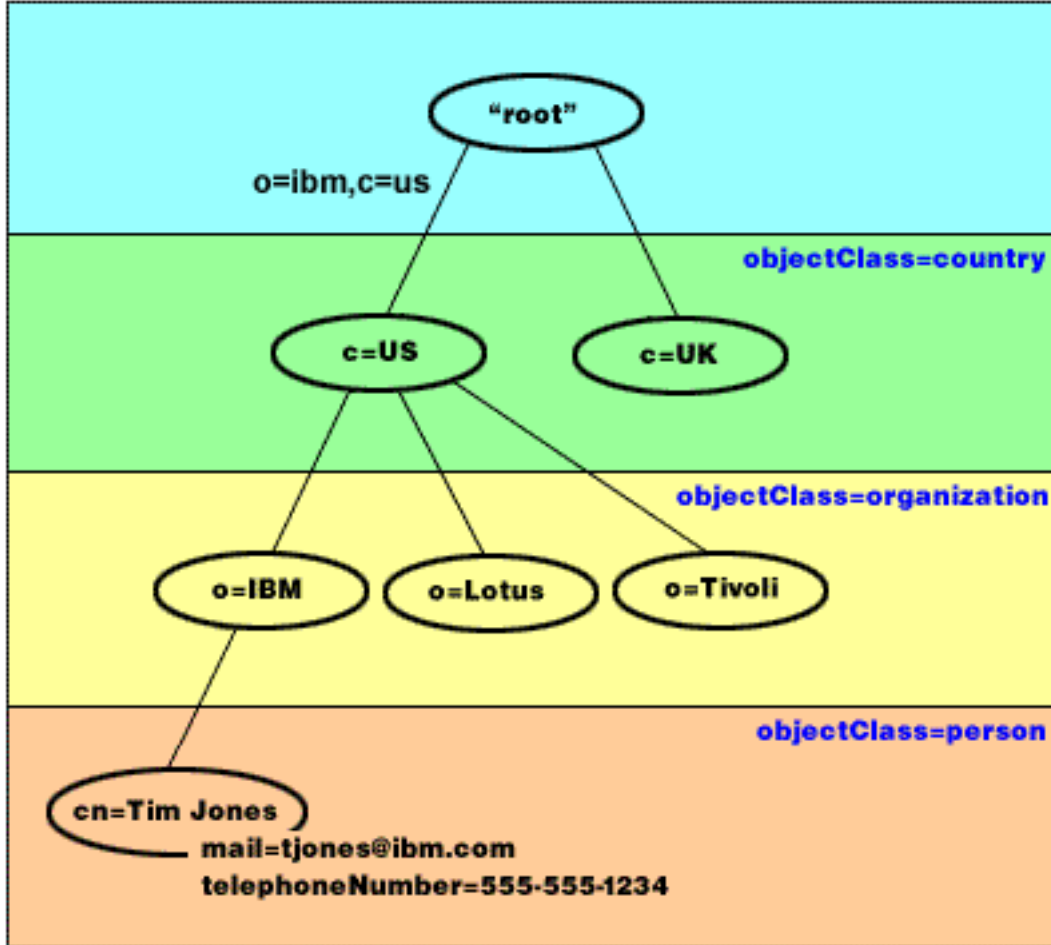
In addition to the attributes defined by the schema, entries also have a set of attributes that are maintained by the server. These attributes, known as operational attributes, include such things as when the entry was created and access control information. For more information about operational attributes, see “Operational attributes” on page 73.

Traditionally, LDAP directory entries are arranged in a hierarchical structure that reflects political, geographic, or organizational boundaries (see Figure 1 on page 9). Entries that represent countries or regions appear at the top of the hierarchy. Entries representing states or national organizations occupy the second level down in the hierarchy. The entries below that can then represent people, organizational units, printers, documents, or other items.

LDAP refers to entries with Distinguished Names (DNs). Distinguished names consist of the name of the entry itself as well as the names, in order from bottom to top, of the objects above it in the directory. For example, the complete DN for the entry in the bottom left corner of Figure 1 on page 9 is `cn=Tim Jones, o=IBM, c=US`. Each entry has at least one attribute that is used to name the entry. This naming attribute is called the Relative Distinguished Name (RDN) of the entry. The entry above a given RDNTM is called its parent Distinguished Name. In the example above, `cn=Tim Jones` names the entry, so it is the RDN. `o=IBM, c=US` is the parent DN for `cn=Tim Jones`. For more information about DN, see “Distinguished names (DNs)” on page 11.

To give an LDAP server the capability to manage part of an LDAP directory, you specify the highest level parent distinguished names in the configuration of the server. These distinguished names are called suffixes. The server can access all objects in the directory that are below the specified suffix in the directory hierarchy. For example, if an LDAP server contained the directory shown in Figure 1 on page 9, it would need to have the suffix `o=ibm, c=us` specified in its configuration in order to be able to answer client queries regarding Tim Jones.

LDAP Directory Structure



RV4Q100-1

Figure 1. LDAP directory structure

You are not limited to the traditional hierarchy when structuring your directory. The domain component structure, for example, is gaining popularity. With this structure, entries are composed of the parts of TCP/IP domain names. For example, dc=ibm,dc=com may be preferable to o=ibm,c=us.

Say that you want to create a directory using the domain component structure that will contain employee data such as names, telephone numbers, and email addresses. You use the suffix or naming context based on the TCP/IP domain. This directory might be visualized as something similar to the following:

```

/
|
+- ibm.com
   |
   +- employees
      |
      +- Tim Jones
         |
         | 555-555-1234
         | tjones@ibm.com
      +- John Smith
         |
         | 555-555-1235
         | jsmith@ibm.com
  
```

When entered in the Directory Server this data might actually look similar to the following:

```

# suffix ibm.com
dn: dc=ibm,dc=com
objectclass: top
objectclass: domain
dc: ibm

# employees directory
dn: cn=employees,dc=ibm,dc=com
objectclass: top
objectclass: container
cn: employees

# employee Tim Jones
dn: cn=Tim Jones,cn=employees,dc=ibm,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
objectclass: publisher
objectclass: ePerson
cn: Tim Jones
cn: "Jones, Tim"
sn: Jones
givenname: Tim
telephonenumber: 555-555-1234
mail: tjones@ibm.com

# employee John Smith
dn: cn=John Smith,cn=employees,dc=ibm,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
objectclass: publisher
objectclass: ePerson
cn: John Smith
cn: "Smith, John"
sn: Smith
givenname: John
telephonenumber: 555-555-1235
mail: jsmith@ibm.com

```

You will notice that the each entry contains attribute values called objectclass. The objectclass values define what attributes are allowed in the entry, such as telephonenumber or givenname. The allowed object classes are defined in the schema. The schema is a set of rules that defines the type of entries allowed in the database.

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 on 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.

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 an entry for a person:

```
dn: cn=Tim Jones,o=ibm,c=us
objectclass: top
objectclass: person
cn: Tim Jones
sn: Jones
telephonenumber: 555-555-1234
```

DN escaping rules

Some characters have special meaning in a DN. For example, = (equals) separates an attribute name and value, and , (comma) separates attribute=value pairs. The special characters are , (comma), = (equals), + (plus), < (less than), > (greater than), # (number sign), ; (semicolon), \ (backslash), and " (quotation mark, ASCII 34).

A special character can be escaped in an attribute value to remove the special meaning. 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 the 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
```

- 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.

Pseudo DNs

Pseudo DNs are used in access control definition and evaluation. The LDAP directory supports 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. For more information on access control, see "Directory Server security" on page 41.

Three pseudo DNs are supported by Directory Server:

- 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:rwc

AclEntry: group: CN=ANYBODY: normal:rsc

AclEntry: group: CN=AUTHENTICATED: sensitive:rsc

User Binding as	Would receive
cn=personA, c=US	normal:rsc:sensitive:rsc:critical:rwc
cn=personB, c=US	normal:rsc:sensitive:rsc
Anonymous	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:rsc

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

User Binding as	Would receive
cn=personA, c=US	object:ad:critical:rwsc
cn=personB, c=US	normal:rsc:sensitive:rsc
Anonymous	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 -b "cn=mike+ou=austin,o=ibm,c=us" "(objectclass=*)"
```

The server supports a DN normalization extended operation. DN normalization extended operations normalize DNs using the server schema. This extended operation might be useful for applications that use DNs. For more information about extended operations, see "Controls and extended operations" on page 74.

Distinguished name syntax

The formal syntax for a Distinguished Name (DN) 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.

The following example is a distinguished name written using a notation that is designed to be convenient for common forms of names. First is a name containing three components. The first of the components is a compound RDN. A compound 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
```

Suffix (naming context)

A suffix (also known as a naming context) 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.

The specific entry that matches the suffix must be added to the directory. The entry you create must use an objectclass that contains the naming attribute used. You can use the Web administration tool or the Qshell ldapadd utility to create the entry corresponding to this suffix. For more information see, "Manage directory entries" on page 136 or "ldapmodify and ldapadd" on page 165.

Conceptually, there is a global LDAP name space. In the global LDAP name space, you might see DNs like:

- cn=John Smith,ou=Rochester,o=IBM
- cn=Jane Doe,o=My Company,c=US
- cn=system administrator,dc=myco,dc=com

The suffix "o=IBM" tells the server that only the first DN is in a name space held by the server. Attempts to reference objects that are not within one of the suffixes result in a no such object error or a referral to another directory server.

A server can have multiple suffixes. The Directory Server has several predefined suffixes that hold data specific to our implementation:

- cn=schema contains the LDAP accessible representation of the schema
- cn=changelog holds the server change log, if enabled

- cn=localhost contains non-replicated information that controls some aspects of the server operation, for example, replication configuration objects
- cn=pwdpolicy contains the server-wide password policy
- the "os400-sys=system-name.mydomain.com" suffix provides LDAP accessible to i5/OS objects, currently limited to user profiles and groups

The Directory Server comes pre-configured with a default suffix, dc=system-name,dc=domain-name, to make it easier to get started with the server. There is no requirement that you use that suffix. You can add your own suffixes, and delete the pre-configured suffix.

There are two commonly used naming conventions for suffixes. One is based on the TCP/IP domain for your organization. The other is based on the organization's name and location.

For example, given a TCP/IP domain of mycompany.com, you might choose a suffix like dc=mycompany,dc=com, where the dc attribute refers to the domain component. In this case the top level entry you create in the directory might look like the following (using LDIF, a text file format for representing LDAP entries):

```
dn: dc=mycompany,dc=com
objectclass: domain
dc: mycompany
```

The domain objectclass also has some optional attributes you might want to use. View the schema or edit the entry you have created using the Web administration tool to see the additional attributes that you can use. For more information see, "Manage the schema" on page 125.

If your company name is My Company and it is located in the United States, you might choose a suffix like one of the following:

```
o=My Company
o=My Company,c=US
ou=Widget Division,o=My Company,c=US
```

Where ou is the name for the organizationalUnit objectclass, o is the organization name for the organization objectclass, and c is a standard two letter country abbreviation used to name the country object class. In this case the top level entry you create might look like:

```
dn: o=My Company,c=US
objectclass: organization
o: My Company
```

Applications that you use may require that specific suffixes be defined, or that a particular naming convention be used. For example, if your directory is used to manage digital certificates, you might be required to structure part of your directory so that entry names match the subject DNs of the certificates that it holds.

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.

For additional information about how to add or remove a suffix, see "Add and remove Directory Server suffixes" on page 104.

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. An 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 “Manage directory entries” on page 136 for additional information about how to manage entries.

For more information related to schema, see the following:

- “IBM Directory Server schema”
- “Common schema support” on page 18
- “Object classes” on page 18
- “Attributes” on page 19
- “Object identifier (OID)” on page 26
- “The subschema entries” on page 27
- “The IBMsubschema object class” on page 27
- “Schema queries” on page 27
- “Dynamic schema” on page 27
- “Disallowed schema changes” on page 28
- “Schema checking” on page 31
- “iPlanet compatibility” on page 32
- “Generalized and UTC time” on page 33

IBM Directory Server schema

The schema for the Directory Server is predefined, however, you can modify the schema, if you have additional requirements. For more information about how to modify the schema, see “Manage the schema” on page 125.

The Directory Server 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 “Directory Server APIs” topic 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.

The Directory Server defines a single schema for the entire server, accessible through 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` (For more information about `objectClasses`, see “Object classes” on page 18.)
- `attributeTypes` (For more information about `attributeTypes`, see “Attributes” on page 19.)
- `IBMAttributeTypes` (For more information about `IBMAttributeTypes`, see “The `IBMAttributeTypes` attribute” on page 22.)
- `matching rules` (For more information about `matching rules`, see “Matching rules” on page 23).
- `ldap syntaxes` (For more information about `ldap syntaxes`, see “Attribute syntax” on page 25).

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 'subschemata'
                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 'subschemataSubentry'
                 EQUALITY distinguishedNameMatch
                 SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
                 NO-USER-MODIFICATION
                 SINGLE-VALUE USAGE directoryOperation )

attributeTypes=( 2.5.21.5 NAME 'attributeTypes'
                 EQUALITY objectIdentifierFirstComponentMatch
                 SYNTAX 1.3.6.1.4.1.1466.115.121.1.3
                 USAGE directoryOperation )

attributeTypes=( 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' )
ldapSyntaxes=( 1.3.6.1.4.1.1466.115.121.1.7 DESC 'Boolean' )
ldapSyntaxes=( 1.3.6.1.4.1.1466.115.121.1.12 DESC 'DN' )
ldapSyntaxes=( 1.3.6.1.4.1.1466.115.121.1.15 DESC 'Directory String' )
ldapSyntaxes=( 1.3.6.1.4.1.1466.115.121.1.24 DESC 'Generalized Time' )
ldapSyntaxes=( 1.3.6.1.4.1.1466.115.121.1.26 DESC 'IA5 String' )
ldapSyntaxes=( 1.3.6.1.4.1.1466.115.121.1.27 DESC 'INTEGER' )
ldapSyntaxes=( 1.3.6.1.4.1.1466.115.121.1.50 DESC 'Telephone Number' )
ldapSyntaxes=( 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 )
matchingRules=( 2.5.13.0 NAME 'objectIdentifierMatch'
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )
matchingRules=( 2.5.13.30 NAME 'objectIdentifierFirstComponentMatch'
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )
matchingRules=( 2.5.13.4 NAME 'caseIgnoreSubstringsMatch'
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.58 )
```

The schema information can be modified through the `ldap_modify` API. Consult the “Directory Server APIs” topic for additional information. With the DN `cn=schema` you can add, delete or replace an attribute type or an object class. See “Dynamic schema” on page 27 and “Manage the schema” on page 125 for more information. 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 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.

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 Directory Server have the prefix 'ibm-'.

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 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.

Attributes

Each directory entry has a set of attributes associated with it through its 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 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, you can create, edit, copy, or delete attributes definitions to suit the needs of your organization.

Attributes can be defined as either single-valued or multi-valued. Multi-valued attributes are not ordered, so an application should not depend on the set of values for a given attribute being returned in particular order. If you need an ordered set of values, consider putting the list of values in a single attribute value:

```
preferences: 1st-pref 2nd-pref 3rd-pref
```

Or consider including order information in the value:

```
preferences: 2 yyy
preferences: 1 xxx
preferences: 3 zzz
```

Multi-valued attributes are useful when an entry is known by several names. For example, cn (common name) is multi-valued. An entry could be defined like:

```
dn: cn=John Smith,o=My Company,c=US
objectclass: inetorgperson
sn: Smith
cn: John Smith
cn: Jack Smith
cn: Johnny Smith
```

This allows searches for John Smith and Jack Smith to return the same information.

Binary attributes contain an arbitrary byte string, for example a JPEG photo, and cannot be used to search for entries.

Boolean attributes contain the strings TRUE or FALSE.

DN attributes contain LDAP distinguished names. The values do not need to be the DNs of existing entries, but they must have a valid DN syntax.

Directory String attributes contain a text string using UTF-8 characters. The attribute can be case exact or case ignore with respect to values used in search filters (based on the matching rule defined for the attribute), though the value is always returned as originally entered.

Generalized Time attributes contain a string representation of a year 2000 safe date and time using GMT times with an optional GMT time zone offset. See “Generalized and UTC time” on page 33 for more details on the syntax of these values.

IA5 String attributes contain a text string using the IA5 character set (7-bit US ASCII). The attribute can be case exact or case ignore with respect to values used in search filters (based on the matching rule defined for the attribute), though the value is always returned as originally entered. IA5 String also allows the use of a wild card character for substring searches.

Integer attributes contain the text string representation of the value. For example, 0 or 1000.

Telephone Number attributes contain a text representation of a telephone number. The Directory Server does not impose any particular syntax on these values. The following are all valid values: (555)555-5555, 555.555.5555, and +1 43 555 555 5555.

UTC Time attributes use an earlier, non-year 2000 safe, string format for representing dates and times. See “Generalized and UTC time” on page 33 for more details.

For more information, see the following:

- “Common subschema elements”
- “The objectclass attribute” on page 21
- “The attributetypes attribute” on page 21
- “The IBMAttributeTypes attribute” on page 22
- “Matching rules” on page 23
- “Indexing rules” on page 24
- “Attribute syntax” on page 25

Common subschema elements

The following elements are used to define the grammar of the subschema attribute values:

- alpha = 'a' - 'z', 'A' - 'Z'
- number = '0' - '9'

- `anh = alpha / number / '-' / ''`
- `anhstring = 1 * anh`
- `keystring = alpha [anhstring]`
- `numericstring = 1 * number`
- `oid = descr / numericoid`
- `descr = keystring`
- `numericoid = numericstring *("." numericstring)`
- `woid = whsp oid whsp ; set of oids of either form (numeric OIDs or names)`
- `oids = woid / ("(" oidlist ")")`
- `oidlist = woid *("$" woid) ; object descriptors used as schema element names`
- `qdescrs = qdescr / (whsp "(" qdescrlist ")" whsp)`
- `qdescrlist = [qdescr *(qdescr)]`
- `whsp """ descr """ whsp`

The objectclass attribute

The objectclasses attribute lists the object classes supported by the server. Each value of this attribute represents a separate object class definition. Object class definitions can be added, deleted, or modified by appropriate modifications of the objectclasses attribute of the `cn=schema` entry. Values of the objectclasses attribute have the following grammar, as defined by RFC 2252:

```
ObjectClassDescription = "(" whsp
    numericoid whsp ; Objectclass identifier
    [ "NAME" qdescrs ]
    [ "DESC" qdstring ]
    [ "OBSOLETE" whsp ]
    [ "SUP" oids ] ; Superior objectclasses
    [ ( "ABSTRACT" / "STRUCTURAL" / "AUXILIARY" ) whsp ] ; default is structural
    [ "MUST" oids ] ; AttributeTypes
    [ "MAY" oids ] ; AttributeTypes
    whsp ")"
```

For example, the definition of the person objectclass is:

```
( 2.5.6.6 NAME 'person' DESC 'Defines entries that generically represent people. ' STRUCTURAL
SUP top MUST ( cn $ sn ) MAY ( userPassword $ telephoneNumber $ seeAlso $ description ) )
```

- The OID for this class is 2.5.6.6
- The name is "person"
- It is a structural object class
- It inherits from the object class "top"
- The following attributes are required: `cn`, `sn`
- The following attributes are optional: `userPassword`, `telephoneNumber`, `seeAlso`, `description`

For more information about how to change the object classes supported by the server, see "Manage the schema" on page 125.

The attributetypes attribute

The attributetypes attribute lists the attribute supported by the server. Each value of this attribute represents a separate attribute definition. Attribute definitions can be added, deleted, or modified by appropriate modifications of the attributetypes attribute of the `cn=schema` entry. Values of the attributetypes attribute have the following grammar, as defined by RFC 2252:

```
AttributeTypeDescription = "(" whsp
    numericoid whsp ; AttributeType identifier
    [ "NAME" qdescrs ] ; name used in AttributeType
    [ "DESC" qdstring ] ; description
```

```
[ "OBSOLETE" whsp ]
[ "SUP" woid ] ; derived from this other AttributeType
[ "EQUALITY" woid ; Matching Rule name
[ "ORDERING" woid ; Matching Rule name
[ "SUBSTR" woid ] ; Matching Rule name
[ "SYNTAX" whsp noidlen whsp ]
[ "SINGLE-VALUE" whsp ] ; default multi-valued
[ "COLLECTIVE" whsp ] ; default not collective
[ "NO-USER-MODIFICATION" whsp ]; default user modifiable
[ "USAGE" whsp AttributeUsage ]; default userApplications
whsp ")"
```

```
AttributeUsage =
  "userApplications" /
  "directoryOperation" /
  "distributedOperation" / ; DSA-shared
  "dSAOperation" ; DSA-specific, value depends on server
```

The matching rules and syntax values must be one the values defined by the following:

- “Matching rules” on page 23
- “Attribute syntax” on page 25

Only “userApplications” attributes can be defined or modified in the schema. The “directoryOperation”, “distributedOperation” and “dSAOperation” attributes are defined by the server and have specific meaning to the server operation.

For example, the “description” attribute has the following definition:

```
( 2.5.4.13 NAME 'description' DESC 'Attribute common to CIM and LDAP schema to provide lengthy
description of a directory object entry.' EQUALITY caseIgnoreMatch SUBSTR
caseIgnoreSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE userApplications )
```

- Its OID is 2.5.4.13
- Its name is “description”
- Its syntax is 1.3.6.1.4.1.1466.115.121.1.15 (Directory String)

For more information about how to change the attribute types supported by the server, see “Manage the schema” on page 125.

The IBMAttributeTypes attribute

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. Directory Server 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. A single index is used to implement multiple matching rules. The Directory Server assigns a length of 500 when one is not provided by the user. The server 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

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

Equality matching rules		
Matching Rule	OID	Syntax
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

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

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 33.

Indexing rules

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

An attribute syntax defines the allowable values for an attribute. The server uses the syntax definition for an attribute to validate data and determine how to match values. For example, a "Boolean" attribute can only have the values "TRUE" and "FALSE"..

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

Syntax	OID
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)."	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 33.	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.


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**.

It is absolutely critical that private OIDs are obtained from legitimate authorities. There are two basic strategies for obtaining legitimate OIDs:

- Register the objects with an authority. This strategy can be convenient, for example, if you need a small number of OIDs.
- Obtain an arc (an arc is an individual subtree of the OID tree) from an authority and assign your own OIDs as needed. This strategy may be preferred if many OIDs are needed, or OID assignments are not stable.

The American National Standards Institute (ANSI) is the registration authority for organization names in the United States under the global registration process established by International Standards Organization (ISO) and International Telecommunication Union (ITU). More information about

organization name registration can be found at the ANSI Web site  (www.ansi.org). The ANSI OID arc for organizations is 2.16.840.1. ANSI will assign a number (NEWNUM), creating a new OID arc: 2.16.840.1.NEWNUM.

In most countries or regions, the national standards association maintains an OID registry. As with the ANSI arc, these are generally arcs assigned under the OID 2.16. It may take some investigation to find the OID authority for a particular country or region. The national standards organization for your country or region may be an ISO member. The names and contact information of ISO members can be found at the ISO Web site  (www.iso.ch).

The Internet Assigned Numbers Authority (IANA) assigns private enterprise numbers, which are OIDs, in the arc 1.3.6.1.4.1. IANA will assign a number (NEWNUM) so that the new OID arc will be

1.3.6.1.4.1.NEWNUM. These numbers can be obtained from the IANA Web site  (www.iana.org).

Once your organization has been assigned an OID, you can define your own OIDs by appending to the end of the OID. For example, suppose your organization has been assigned the fictional OID 1.1.1. No other organization will be assigned an OID that starts with "1.1.1". You might create a range for LDAP by appending ".1" to form 1.1.1.1. You might further subdivide this into ranges for objectclasses (1.1.1.1.1), attribute types (1.1.1.1.2), and so on, and assign OID 1.1.1.1.2.34 to the attribute "foo".

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:

```
( 1.3.18.0.2.6.174
NAME 'ibmSubSchema'
DESC 'IBM specific object class that stores all the attributes and object classes for a given directory
server.'
SUP 'subschema'
STRUCTURAL 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 "Directory Server APIs" topic 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 entry, specify the schema attribute that defines the schema entry (objectclasses or attributetypes), and for its value, the OID in parentheses. For example, to delete the attribute with OID <attr-oid>:

```
dn: cn=schema
changetype: modify
delete: attributetypes
attributetypes: ( <attr-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 inherits from the attribute "name" and has a length of 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);

```

The LDIF version of the above would be:

```

dn: cn=schema
changetype: modify
add: attributetypes
attributetypes: ( 20.20.20 NAME 'bar' SUP name )
-
add: ibmattributetypes
ibmattributetypes: (20.20.20 LENGTH 20)

```

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.

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
- os400-usrprf
- referral

- replicaObject
- top

Attributes:

- aclEntry
- aclPropagate
- aclSource
- aliasedObjectName, aliasedentryName
- businessCategory
- cn, commonName
- createTimestamp
- creatorsName
- description
- dn, distinguishedName
- entryOwner
- hasSubordinates
- ibm-entryChecksum
- ibm-entryChecksumOp
- ibm-entryUuid
- member
- modifiersName
- modifyTimestamp
- name
- o, organizationName, organization
- objectClass
- os400-acgcde
- os400-astlvl
- os400-atnpgm
- os400-audlvl
- os400-aut
- os400-ccsid
- os400-chridctl
- os400-cntryid
- os400-curlib
- os400-dlvry
- os400-docpwd
- os400-dspsgninf
- os400-eimassoc
- os400-gid
- os400-groupmember
- os400-grpaut
- os400-grpauttyp
- os400-grpprf
- os400-homedir
- os400-IaspStorageInformation
- os400-inlmnu

- os400-inlpgm
- os400-invalidSignonCount
- os400-jobd
- os400-kbdbuf
- os400-langid
- os400-lclpwdmgt
- os400-lmtcpb
- os400-lmtdevssn
- os400-locale
- os400-maxstg
- os400-msgq
- os400-objaud
- os400-outq
- os400-owner
- os400-password
- os400-passwordExpirationDate
- os400-passwordLastChanged
- os400-previousSignon
- os400-profile
- os400-prtdev
- os400-ptylmt
- os400-pwdexp
- os400-pwdexpitv
- os400-setjobatr
- os400-sev
- os400-spcaut
- os400-spcenv
- os400-srtseq
- os400-status
- os400-storageUsed
- os400-storageUsedOnIasp
- os400-supgrpprf
- os400-sys os400-text
- os400-uid
- os400-usrcls
- os400-usropt
- 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 the schema checking by changing the schema checking level. This is done through iSeries Navigator by changing the value of the **Schema checking** field on the **Database/Suffixes** page of the Directory Server properties. See "Directory Server configuration schema" on page 193 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.
- If the schema checking level is set to **Version 3 (strict)** all structural superclasses must be provided. For example, to create an entry with objectclass `inetorgperson`, the following objectclasses must be specified: `person`, `organizationalperson`, and `inetorgperson`.

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.

iPlanet compatibility

The parser used by the 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 Directory Server 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 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 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 Directory Server describes it:

```
( 2.5.4.3 NAME ( 'cn' 'commonName' ) SUP name )
```

The 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.

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
```

Publishing

i5/OS provides the ability to have the system publish certain kinds of information to an LDAP directory. That is, the system will create and update LDAP entries representing various types of data.

i5/OS has built-in support for publishing the following information to a LDAP server:

Users

When you configure i5/OS to publish the information type Users to the Directory Server, it automatically exports entries from the system distribution directory to the Directory Server. It uses the QGLDSSDD application program interface (API) to do this. This also keeps the LDAP directory synchronized with changes that are made in the system distribution directory. For information about the QGLDSSDD API, see "Directory Server APIs" in the Programming topic.

Publishing users is useful for providing LDAP search access to information from the system distribution directory (for example to provide LDAP address book access to LDAP-enabled POP3 mail clients like Netscape Communicator or Microsoft Outlook Express).

Published users may also be used to support LDAP authentication with some users published from the system distribution directory, and other users added to the directory by other means. A published user has a uid attribute that names the user profile, and has no userPassword attribute. When a bind request is received for an entry like this, the server calls i5/OS security to validate the

uid and password as a valid user profile and password for that profile. If you want to use LDAP authentication, and would like existing i5/OS users to be able to authenticate using their i5/OS passwords, while non-i5/OS users are added to the directory manually, you should consider this feature.

System information

When you configure i5/OS to publish the information type System to the Directory Server, the following types of information are published:

- Basic information about this machine and the operating system release.
- Optionally, you can select one or more printers to publish, in which case the system will automatically keep the LDAP directory synchronized with changes that are made to those printers on the system.

Printer information that can be published includes:

- Location
- Speed in pages per minutes
- Support for duplex and color
- Type and model
- Description

This information comes from the device description on the system being published. In a network environment, users can use this information to help select a printer. The information is first published when a printer is selected to be published, and it is updated when a printer writer is stopped or started, or the printer device description is changed.

Printer shares

When you configure i5/OS to publish printer shares, information about the selected iSeries Netserver printer shares are published to the configured Active Directory server. Publishing print shares to an Active Directory allows users to add iSeries printers to their Windows 2000 desktop with the Windows 2000's Add Printer wizard. In order to do this in the Add Printer wizard, specify that you want to find a printer in the Windows 2000 Active Directory. You must publish print shares to a directory server which supports Microsoft's Active Directory schema.

TCP/IP Quality of Service

The TCP/IP Quality of Service (QOS) server can be configured to use a shared QOS policy defined in an LDAP directory using an IBM defined schema. The TCP/IP QOS publishing agent is used by the QOS server to read the policy information; it defines the server, authentication information, and where in the directory the policy information is stored.

You can also create an application to publish or search for other kinds of information in a LDAP directory using this framework by defining additional publishing agents and making use of the directory publishing APIs. For more information, see "Directory Server APIs" in the Programming topic.

Replication

Replication is a technique used by directory servers to improve performance and reliability. The replication process keeps the data in multiple directories synchronized.

For information about how to manage replication see "Manage replication" on page 107. For more information about replication see the following:

- "Replication overview" on page 36

- “Replication terminology” on page 37
- “Replication agreements” on page 38
- “How replication information is stored in the server” on page 39
- “Security considerations for replication information” on page 39

Replication overview

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.

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 topology 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-subordinate 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 1. Server roles

Directory	Description
Master/peer	<p>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.</p> <p>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.</p> <p>Notes:</p> <ol style="list-style-type: none"> 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.
Cascading (forwarding)	<p>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.</p>
Replica (read-only)	<p>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.</p>

If the replication fails, it is repeated even if the master is restarted. The Manage Queues window in the Web administration tool can be used to check for failing replication.

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. 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 replication 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 terminology

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 replication agreement.

Forwarding server

A read-only server that replicates all changes sent to it by a master or peer. Client update requests are referred to the master or peer 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 replication topology related information 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.

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. Directory servers prior to V5R3 do not have a server id.
- 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 ID from 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.

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.

How replication information is stored in the server

Replication information is stored in the directory in three places:

- The server configuration, which contains information about how other servers can authenticate to this server to perform replication (for example, who this server allows to act as a supplier).
- In the directory at the top of a replicated subtree. If "o=my company" is the top of a replicated subtree, an object named "ibm-replicagroup=default" will be created directly beneath it (ibm-replicagroup=default,o=my company). Beneath the "ibm-replicagroup=default" object will be additional objects that describe the servers holding replicas of the subtree and the agreements between the servers.
- An object named "cn=replication,cn=localhost" is used to contain replication information that is used only by one server. For example, the object containing the credentials used by a supplier server are needed only by the supplier server. Credentials can be placed under "cn=replication,cn=localhost" making them accessible only by that server.

Security considerations for replication information

Review the security considerations for the following objects:

- `ibm-replicagroup=default`: Access controls on this object control who can view or change the replication information stored here. By default, this object inherits the access control from its parent. You should consider setting access control on this object to restrict access to the replication information. For example, you could define a group that contains users that will be managing replication. This group could be made the owner of the "ibm-replicagroup=default" object and other users given no access to the object.
- `cn=replication,cn=localhost`: There are two security considerations for this object:
 - Access control on this object controls who is allowed to view or update objects stored here. The default access control allows anonymous users to read most information except for passwords and requires administrator authority to add, change, or delete objects.

- Objects stored in "cn=localhost" are never replicated to other servers. You can place replication credentials in this container on the server that uses the credential and they will not be accessible to other servers. Alternately, you may chose to place credentials under the "ibm-replicagroup=default" object so that multiple servers share the same credentials.

Realms and user templates

The realm and template objects found in the Web administration tool are used in order to relieve the user of the need to understand some of the underlying LDAP issues.

A realm identifies a collection of users and groups. It specifies information, in a flat directory structure, such as where users are located and where groups are located. A realm defines a location for users (for example, "cn=users,o=acme,c=us") and creates users as immediate subordinates of that entry (for example John Doe is created as "cn=John Doe,cn=users,o=acme,c=us"). You can define multiple realms and give them familiar names (for example Web Users). The familiar name can be used by the people that are creating and maintaining the users.

A template describes what a user looks like. It specifies the objectclasses that are used when creating users (both the structural objectclass and any auxiliary classes that you want). A template also specifies the layout of the panels used to create or edit users (for example, names of tabs, default values, and attributes to appear on each tab).

When you add a new realm, you are creating an ibm-realm object in the directory. The ibm-realm object keeps track of the properties of the realm such as where users and groups are defined, and what template to use. The ibm-realm object can point to an existing directory entry that is the parent of users, or it can point to itself (the default), making it the container for new users. For example, you could have an existing cn=users,o=acme,c=us container, and create a realm named users elsewhere in the directory (maybe a container object called cn=realms,cn=admin stuff,o=acme,c=us) that identifies cn=users,o=acme,c=us as the location for users and groups. This creates an ibm-realm object:

```
dn: cn=users,cn=realms,cn=admin stuff,o=acme,c=us
objectclass: top
objectclass: ibm-realm
objectclass: ibm-staticGroup
ibm-realmUserTemplate: cn=users template,cn=realms,cn=admin stuff,o=acme,c=us
ibm-realmUserContainer: cn=users,o=acme,c=us
ibm-realmGroupContainer: cn=users,o=acme,c=us
ibm-realmAdminGroup: cn=users,cn=realms,cn=admin stuff,o=acme,c=us
ibm-realmUserSearchFilter:
cn: users
```

Or, if there was no existing cn=users,o=acme,c=us object, you could create the realm users under o=acme,c=us and have it point to itself.

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 are responsible for managing the users and groups within that realm.

For more information about how to manage realms and user templates, see "Manage realms and user templates" on page 145.

National language support (NLS) considerations

Be aware of the following NLS considerations:

- Data is transferred between LDAP servers and clients in UTF-8 format. All ISO 10646 characters are allowed.
- The Directory Server uses the UTF-16 mapping method to store data in the database.

- The server and the client do case insensitive string comparisons. The uppercase algorithms will not be correct for all languages (locales).

For more information about UCS-2, see “Globalization” in the Planning topic.

LDAP directory referrals

Referrals allow Directory Servers to work in teams. If the DN that a client requests is not in one directory, the server can automatically send (refer) the request to any other LDAP server.

Directory Server allows you to use two different types of referrals. You can specify default referral servers, where the LDAP server will refer clients whenever any DN is not in the directory. You can also use your LDAP client to add entries to the directory server that have the objectClass referral. This allows you to specify referrals that are based on what specific DN a client requests.

Note: With Directory Server, referral objects must contain only a distinguished name (dn), an objectClass (objectClass), and a referral (ref) attribute. See “ldapsearch” on page 177 for an example that illustrates this restriction.

Referral servers are closely related to replica servers. Because data on replica servers cannot be changed from clients, the replica refers any requests to change directory data to the master server.

Transactions

You can configure your Directory Server to allow clients to use transactions. (For more information about configuring transaction settings, see “Specify transaction settings” on page 101.) A transaction is a group of LDAP directory operations that are treated as one unit. None of the individual LDAP operations that make up a transaction are permanent until all operations in the transaction have completed successfully and the transaction has been committed. If any of the operations fail or the transaction is cancelled, the other operations are undone. This capability can help users to keep LDAP operations organized. For example, a user might set up a transaction on his client that will delete several directory entries. If the client loses its connection to the server part way through the transaction, none of the entries are deleted. Therefore the user can simply start the transaction over rather than having to check to see which entries were successfully deleted.

The following LDAP operations may be part of a transaction:

- add
- modify
- modify RDN
- delete

Note: Do not include changes to the directory schema (the cn=schema suffix) in transactions. Though it is possible to include them, they cannot be backed out if the transaction fails. This could cause your directory server to experience unpredictable problems.

Directory Server security

See the following for more information about Directory Server security:

- “Auditing” on page 42
- “Secure Sockets Layer (SSL) and Transport Layer Security with the Directory Server” on page 42
- “Kerberos authentication with the Directory Server” on page 42
- “Groups and roles” on page 43
- “Access control lists” on page 49
- “Ownership of LDAP directory objects” on page 60


- “Password policy” on page 61
- “Authentication” on page 64

Auditing

Directory Server supports OS/400 security auditing. Auditable items include the following:

- Binds to and unbinds from the directory server.
- Changes to permissions of LDAP directory objects.
- Changes in ownership of LDAP directory objects.
- Creation of, deletion of, searches of, and changes to LDAP directory objects.
- Changes to the password of administrator and update distinguished names (DNs)
- Changes to the passwords of users.
- File imports and exports.

You may need to make changes to your i5/OS auditing settings before auditing of directory entries will work. If the QAUDCTL system value has *OBJAUD specified, you can enable object auditing through

iSeries Navigator. For more information about auditing, see *Security - Reference*  or the “Security auditing” topic.

Secure Sockets Layer (SSL) and Transport Layer Security with the Directory Server

To make communications with your Directory Server more secure, Directory Server can use Secure Sockets Layer (SSL) security.

To use SSL with Directory Server, you must have one of the Cryptographic Access Provider products (5722-ACx) installed on your system. If you want to use SSL from iSeries Navigator, you must also have one of the Client Encryption products (5722-CEX) installed on your PC. You need this software if you want to do any of the following:

- To configure and administer Directory Server from your workstation using an SSL connection. This includes tasks that you perform from iSeries Navigator.
- To use an SSL connection with applications that you create with the LDAP client application program interfaces (APIs).

SSL is the standard for Internet security. You can use SSL to communicate with LDAP clients, as well as with replica LDAP servers. You can use client authentication in addition to server authentication to provide additional security to your SSL connections. Client authentication requires that the LDAP client present a digital certificate that confirms the clients identity to the server before a connection is established.

To use SSL, you must have Digital Certificate Manager (DCM), option 34 of i5/OS installed on your system. DCM provides an interface for you to create and manage digital certificates and certificate stores. See the “Digital Certificate Manager” topic for information about digital certificates and using DCM. For information about SSL on iSeries, see the “Secure Sockets Layer (SSL)” topic. For information about TLS on the iSeries server, see Supported SSL and Transport Layer Security (TLS) protocols.

Kerberos authentication with the Directory Server

Directory Server allows you to use Kerberos authentication. Kerberos is a network authentication protocol that uses secret key cryptography to provide strong authentication to client/server applications.

To enable Kerberos authentication, you must have one the Cryptographic Service Provider products (5722AC2 or 5722AC3) installed on your system. You must also have the network authentication service configured.

The Kerberos support of Directory Server provides support for the GSSAPI SASL mechanism. This enables both Directory Server and Windows 2000 LDAP clients to use Kerberos authentication with the Directory Server.

The **Kerberos principal name** that the server uses has the following form:

```
service-name/host-name@realm
```

service-name is ldap (ldap must be lower case), host-name is the fully qualified TCP/IP name of the system, and realm is the default realm specified in the systems Kerberos configuration.

For example, for a system named my-as400 in the acme.com TCP/IP domain, with a default Kerberos realm of ACME.COM, the LDAP server Kerberos principal name would be ldap/my-as400.acme.com@ACME.COM. The default Kerberos realm is specified in the Kerberos configuration file (by default, /QIBM/UserData/OS400/NetworkAuthentication/krb5.conf) with the default_realm directive (default_realm = ACME.COM). The directory server cannot be configured to use Kerberos authentication if a default realm has not been configured.

When Kerberos authentication is used, the Directory Server associates a distinguished name (DN) with the connection that determines access to directory data. You can choose to have the server DN associated with one of the following methods:

- The server can create a DN based on the Kerberos ID. When you choose this option, a Kerberos identity of the form principal@realm generates a DN of the form ibm-kn=principal@realm. ibm-kn= is equivalent to ibm-kerberosName=.
- The server can search the directory for a distinguished name (DN) that contains an entry for the Kerberos principal and realm. When you choose this option, the server searches the directory for an entry that specifies this Kerberos identity.

You must have a key table (keytab) file that contains a key for the LDAP service principal. See the Information Center topic Network authentication service under Security for more information about Kerberos on the iSeries server. The Configuring network authentication service section contains information about adding information to key table files.

Groups and roles

A group is a list, a collection of names. A group can be used in **aclentry**, **ibm-fliterAclEntry**, and **entryowner** attributes to control access or in application-specific uses such as a mailing list; see "Access control lists" on page 49. Groups can be defined as either static, dynamic, or nested. For information about how to work with groups, see "Manage users and groups" on page 142.

Roles are similar to groups in that they are represented in the directory by an object. Additionally, roles contain a group of DNs.

See the following for more information:

- "Static groups" on page 44
- "Dynamic groups" on page 44
- "Nested groups" on page 45
- "Hybrid groups" on page 46
- "Determining group membership" on page 46
- "Group object classes for nested and dynamic groups" on page 48
- "Group attribute types" on page 48
- "Roles" on page 49

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 the **groupOfNames** or **groupOfUniqueNames** structural objectclasses must have at least one member. A group using the **accessGroup** or **accessRole** structural objectclasses can 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 181 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 **memberURLs** 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 can be extended such that group membership is described by a combination of static, dynamic, and nested member types. For example:

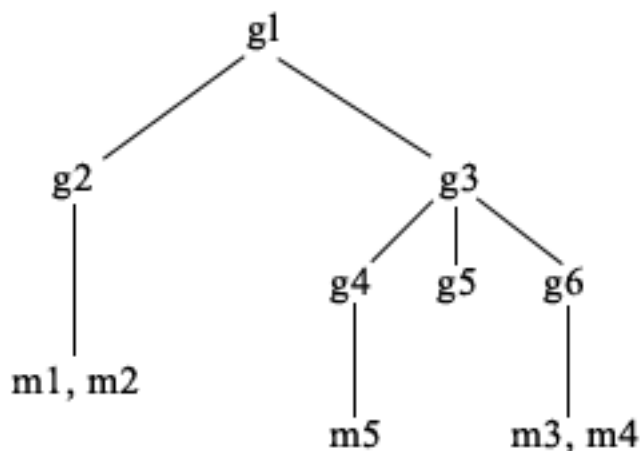
```
dn: cn=Group 10, cn=Groups, o=IBM, c=US
objectclass: groupOfURLs
objectclass: ibm-nestedGroup
objectclass: ibm-staticGroup
objectclass: top
cn: Group 10
description: Group composed of static, dynamic, and nested members.
memberURL: ldap:///cn=Austin, cn=Employees, o=IBM, c=US??one?objectClass=person
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 `acentry`, 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
acentry: access-id:cn=user1,cn=users,o=ibm,c=us:normal:rsc
acentry: 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 for nested and dynamic groups

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. The **ibm-allGroups** attribute can be used in a compare request to determine if an entry is a member of given group. For example, to determine if "cn=john smith,cn=users,o=my company" is a member of the group "cn=system administrators, o=my company":

```
rc = ldap_compare_s(ld, "cn=john smith,cn=users,o=my company, "ibm-allgroups",
"cn=system administrators,o=my company");
```

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. The **ibm-allMembers** attribute can be used in a compare request to determine if a DN is a member of given group. For example, to determine if "cn=john smith,cn=users,o=my company" is a member of the group "cn=system administrators, o=my company":

```
rc = ldap_compare_s(ld, "cn=system administrators,o=my company, "ibm-allmembers",
"cn=john smith,cn=users,o=my company");
```

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.

member

Identifies the distinguished names for each member of the group. For example: `member: cn=John Smith, dc=ibm, dc=com`.

memberURL

Identifies a URL associated with each member of a group. Any type of labeled URL can be used. For example: `memberURL: ldap:///cn=jsmith,dc=ibm,dc=com`.

uniquemember

Identifies a group of names associated with an entry where each name was given a `uniqueIdentifier` to ensure its uniqueness. A value for the `uniqueMember` attribute is a DN followed by the `uniqueIdentifier`. For example: `uniqueMember: cn=John Smith, dc=ibm, dc=com 17`.

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'.

Access control lists

Access control lists (ACLs) 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. Changes to each entry and attribute in the directory can be controlled by using ACLs. An ACL for a given entry or attribute can be inherited from its parent entry or can be explicitly defined.

It is best to design your access control strategy by creating groups of users that you will use when setting the access for objects and attributes. Set ownership and access at the highest level in the tree possible and let the controls inherit down the tree.

The operational attributes associated with access control, such as `entryOwner`, `ownerSource`, `ownerPropagate`, `aclEntry`, `aclSource` and `aclPropagate` are unusual in that they are logically associated with each object, but can have values that depend on other objects higher in the tree. Depending on 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 defines the access rights given to a specified subject with respect to the operations they can perform on the objects to which they apply. The `aclEntry` and `aclPropagate` attributes apply to the ACI definition. The `entryOwner` information defines which subjects can define the ACI for the associated entry object. The `entryOwner` and `ownerPropagate` attributes apply to the `entryOwner` definition.

There are two kinds of access control lists that you can choose from: filter-based ACLs and non-filtered ACLs. Non-filtered 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.

Using ACLs, 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
- ibm-filterAclEntry
- ibm-filterAclInherit
- entryOwner
- ownerPropagate

For information about how to work with ACLs, see “Manage access control lists (ACLs)” on page 153. For additional information, see the following:

- “Filtered ACLs”
- “The access control attribute syntax” on page 51
- “AclEntry and ibm-filterAclEntry” on page 52
- “EntryOwner” on page 54
- “Propagation” on page 54
- “Access evaluation” on page 55
- “Define the ACIs and entry owners” on page 57
- “Modify the ACI and entry owner values” on page 57
- “Delete the ACI/entry owner values” on page 60
- “Retrieve the ACI/entry owner values” on page 60
- “Subtree replication considerations” on page 60

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.

Filter-based ACLs 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 is 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"

<objectPermissions> ::= <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.

Operation	Permission Needed
ldapadd	add (on parent)
ldapdelete	delete (on object)
ldapmodify	write (on attributes being modified)
ldapsearch	<ul style="list-style-type: none">• 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 (see "Propagation" on page 54).

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** cannot 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 50 for more information about 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.

The directory administrator is the entry owner. The pseudo group `cn=anybody` (all users) is granted read, search, and compare access to attributes in the normal access class.

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.

In filter-based ACL mode, if no filter-based ACL applies, the default ACL applies (`cn=anybody` is granted read, search, and compare access to attributes in the normal access class). This situation can occur when the entry being accessed does not match any of the filters specified in the **ibm-filterAclEntry** values. You may want to specify a default filter ACL like the following if you do not want this default access control to apply:

```
ibm-filterAclEntry: group:cn=anybody:(objectclass=*):
```

This example grants no access. Change it to provide the access you want applied.

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 cannot 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.attribute1: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).

Define 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
```

Modify 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:

```
acIEntry: group:cn=Dept XYZ,o=IBM:normal:grant:rsc
```

with a modification:

```
dn: cn=some entry
changetype: modify
add: acIEntry
acIEntry: group:cn=Dept XYZ,o=IBM:normal:deny:r:critical:deny::sensitive
:grant:r
```

yields an acIEntry of:

```
acIEntry: group:cn=Dept XYZ,o=IBM:normal:grant:sc:normal:deny:r:critical
:grant::sensitive:grant:r
```

For example, given the following ACI:

```
Ibm-filterAcIEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):normal
:grant:rsc
```

with a modification:

```
dn: cn=some entry
changetype: modify
add: ibm-filterAcIEntry
ibm-filterAcIEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):normal
:deny:r:critical:deny::sensitive:grant:r
```

yields an acIEntry of:

```
ibm-filterAcIEntry: 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:

```
acIEntry: group:cn=Dept XYZ,o=IBM:object:grant:ad
acIEntry: group:cn=Dept XYZ,o=IBM:normal:grant:rwc
```

```
dn: cn = some entry
changetype: modify
delete: acIEntry
acIEntry: group:cn=Dept XYZ,o=IBM:object:grant:ad
```

yields a remaining ACI on the server of :

```
acIEntry: group:cn=Dept XYZ,o=IBM:normal:grant:rwc
```

Given an ACI of:

```
ibm-filterAcIEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):object
:grant:ad
ibm-filterAcIEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):normal
:grant:rwc
```

```
dn: cn = some entry
changetype: modify
delete: ibm-filterAcIEntry
ibm-filterAcIEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):object
:grant:ad
```

yields a remaining ACI on the server of:

```
ibm-filterAcIEntry: group:cn=Dept XYZ,o=IBM:(cn=Manager XYZ):normal
:grant:rwc
```

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.

Delete 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.

Retrieve 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.

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.

Ownership of LDAP directory objects

Each object in your LDAP directory has at least one owner. Object owners have the power to delete the object. Owners and the server administrator are the only users that can change the ownership properties

and the access control list (ACL) attributes of an object. Ownership of objects can be either inherited or explicit. That is, to assign ownership you can do one of the following:

- Explicitly set up ownership for a specific object.
- Specify that objects inherit their owners from objects higher up in the LDAP directory hierarchy.

Directory Server allows you to specify multiple owners for the same object. You can also specify that an object owns itself. To do this you include the special DN `cn=this` in the list of object owners. For example, assume that the object `cn=A` has the owner `cn=this`. Any user will have owner access to the `cn=A` object if he connects to the server as `cn=A`.

For more information about how to work with ownership properties, see “Manage directory entries” on page 136.

Password policy

With the use of LDAP servers for authentication, it is important that a LDAP server support policies regarding password expiration, failed login attempts, and password rules. Directory Server provides configurable support for all three of these kinds of policies. This policy is applied to all directory entries having a `userPassword` attribute. You cannot define one policy for one set of users, and different policies for other sets of users. Directory Server also provides a mechanism for clients to be informed of password policy related conditions (password expires in three days), and a set of operational attributes that an administrator can use to search for such things as users with expired passwords or locked out accounts.

For more information about how to work with password policy properties, see “Set password policy” on page 102.

Configuration

You can configure behavior of the server with respect to passwords in the following areas:

- A global “on/off” switch for enabling or disabling password policy
- Rules for changing passwords, including:
 - Users can change their own passwords. Note that this policy applies in addition to any access control. That is, access control must give a user authority to change the `userPassword` attribute, as well as password policy allowing users to change their own passwords. If this policy is disabled, users cannot change their own passwords. Only an administrator or other user with authority to change the `userPassword` attribute can change the password for an entry.
 - Passwords must be changed after reset. If this policy is enabled, when a password is changed by anybody other than that user, the password is marked as reset and must be changed by the user before he can perform other directory operations. A bind request with a reset password is successful. To be notified that the password must be reset, the application must be password policy aware.
 - Users must send old password when changing password. If this policy is enabled, a password can be changed only by a modify request that includes both a delete of the `userPassword` attribute (with the old value) and an add of the new `userPassword` value. This ensures that only a user who knows their password can change it. The administrator, or other users authorized to change the `userPassword` attribute can always set the password.
- Rules for password expiration, including:
 - Passwords never expire, or passwords expire a configurable time after they were last changed.
 - Do not warn users when a password expires, or warn users a configurable time before the password expires. To be warned of approaching password expiration, the application must be password policy aware.
 - Allow a configurable number of grace logins after the user’s password has expired. A password policy aware application will be notified of the number of remaining grace logins. If no grace logins are allowed, a user cannot authenticate or change their own password once it has expired.
- Rules for password validation, including:

- A configurable password history size, which tells the server to keep a history of the last N passwords and reject passwords that have been previously used.
- Password syntax checking, including a setting for how the server should behave when passwords are hashed. This setting affects whether the server should ignore the policy under either of the following conditions:
 - The server is storing hashed passwords.
 - A client presents a hashed password to the server (this can happen when transferring entries between servers via an LDIF file if the source server stores hashed passwords).

In either of these cases the server may not be able to apply all syntax rules. The following syntax rules are supported: Minimum length, minimum number of alphabetic characters, minimum number of numeric or special characters, number of repeated characters, and number of characters in which the password must differ from the previous password.

- Rules for failed logins, including:
 - A minimum time allowed between password changes, which prevents users from quickly cycling through a set of passwords to get back to their original password.
 - A maximum number of failed login attempts before the account is locked.
 - A configurable password lockout duration. After this time, a previous locked account can be used. This can help to lockout a hacker attempting to crack a password, while aiding a user that has forgotten their password.
 - A configurable time for which the server keeps track of failed login attempts. If the maximum number of failed login attempts occurs within this time, the account is locked. Once this time has expired, the server discards information about previous failed login attempts for the account.

The password policy settings for the directory server are stored in the object "cn=pwdpolicy", which looks like:

```
cn=pwdpolicy
objectclass=container
objectclass=pwdPolicy
objectclass=ibm-pwdPolicyExt
objectclass=top
cn=pwdPolicy
pwdExpireWarning=0
pwdGraceLoginLimit=0
passwordMaxRepeatedChars=0
pwdSafeModify=false
pwdattribute=userpassword
pwdinhistory=0
pwdchecksyntax=0
passwordminotherchars=0
passwordminalphachars=0
pwdminlength=0
passwordmindiffchars=0
pwdminage=0
pwdmaxage=0
pwdallowuserchange=true
pwdlockoutduration=0
ibm-pwdpolicy=true
pwdlockout=true
pwdmaxfailure=2
pwdfailurecountinterval=0
pwdmustchange=false
```

Password policy aware applications

The Directory Server for iSeries password policy support includes a set of LDAP controls which can be used by a password policy aware application to receive notification of additional password policy related conditions.

An application can be informed of the following warning conditions:

- Time remaining before password expiration
- Number of grace logins remaining after the password has expired

An application can also be informed of the following error conditions:

- Password has expired
- Account is locked
- Password has been reset and must be changed
- User is not allowed to change their password
- Old password must be supplied when changing password
- New password violates syntax rules
- New password is too short
- Password has been changed too recently
- New password is in history

Two controls are used. A password policy request control is used to inform the server that the application wishes to be informed of password policy related conditions. This control must be specified by the application on all operations for which it is interested, typically the initial bind request and any password change requests. If the password policy request control is present, a password policy response control is returned by the server when any of the above error conditions are present.

The Directory Server client APIs include a set of APIs which can be used by C applications to work with these controls. These APIs are:

- `ldap_parse_pwdpolicy_response`
- `ldap_pwdpolicy_err2string`

For applications not using these APIs, the controls are defined below. You must use the capabilities provided by the LDAP client APIs being used to process the controls. For example, the Java Naming and Directory Interface (JNDI) has built-in support for some well-known controls, and also provides a framework for supporting controls that JNDI does not recognize.

Password Policy Request Control

Control name: 1.3.6.1.4.1.42.2.27.8.5.1
Control criticality: FALSE
Control value: None

Password Policy Response Control

Control name: 1.3.6.1.4.1.42.2.27.8.5.1 (same as the request control)
Control criticality: FALSE
Control value: A BER encoded value defined in ASN.1 as follows:

```
PasswordPolicyResponseValue ::= SEQUENCE {
  warning [0] CHOICE OPTIONAL {
    timeBeforeExpiration [0] INTEGER (0 .. MaxInt),
    graceLoginsRemaining [1] INTEGER (0 .. maxInt) }
  error [1] ENUMERATED OPTIONAL {
    passwordExpired (0),
    accountLocked (1),
    changeAfterReset (2),
    passwordModNotAllowed (3),
    mustSupplyOldPassword (4),
    invalidPasswordSyntax (5),
    passwordTooShort (6),
    passwordTooYoung (7),
    passwordInHistory (8) } }
```

Like other LDAP protocol elements, the BER encoding uses implicit tagging.

Password policy operational attributes

The Directory Server maintains a set of operational attributes for each entry that has a userPassword attribute. These attributes can be searched by authorized users, either used in search filters, or returned by the search request. These attributes are:

- pwdChangedTime - A GeneralizedTime attribute containing the time the password was last changed.
- pwdAccountLockedTime - A GeneralizedTime attribute containing the time at which the account was locked. If the account is not locked, this attribute is not present.
- pwdExpirationWarned - A GeneralizedTime attribute containing the time at which the password expiration warning was first sent to the client.
- pwdFailureTime - A multi-valued GeneralizedTime attribute containing the times of previous consecutive login failures. If the last login was successful, this attribute is not present.
- pwdGraceUseTime - A multi-valued GeneralizedTime attribute containing the times of the previous grace logins.
- pwdReset - A Boolean attribute containing the value TRUE if the password has been reset and must be changed by the user.

Replication of Password Policy

Password policy information is replicated by supplier servers to consumers. Changes to the entry cn=pwdpolicy are replicated as global changes, like changes to the schema. Password policy state information for individual entries is also replicated, so that, for example, if an entry is locked on a supplier server, that action will be replicated to any consumers. Password policy state changes on a read-only replica do not replicate to any other servers, however.

Authentication

Access control within the Directory Server is based on the distinguished name (DN) associated with a given connection. That DN is established as the result of a bind to (logging into) the Directory Server.

When the Directory Server is first configured, the following identities can be used to authenticate to the server:

- anonymous
- the directory administrator (cn=admin by default)
- a projected i5/OS user profile (see “Operating system projected backend” on page 67)

It is a good idea to create additional users that can be given authority to manage different parts of the directory without requiring that you share the directory administrator identity.

From an LDAP perspective, there are two frameworks for authenticating to LDAP:

- Simple bind, in which an application provides a DN and the clear text password for that DN
- Simple Authentication and Security Layer (SASL), which provides several additional authentication methods, including CRAM-MD5, EXTERNAL, GSSAPI, and OS400-PRFTKN.

Simple bind (and CRAM-MD5)

To use a simple bind, the client must supply the DN of an existing LDAP entry and a password which matches the userPassword attribute for that entry. For example, you could create an entry for John Smith as follows:

```
sample.ldif:
    dn: cn=John Smith,cn=users,o=acme,c=us
    objectclass: inetorgperson
```

```
cn: John Smith
sn: smith
userPassword: mypassword
```

```
ldapadd -D cn=administrator -w secret -f sample.ldif
```

You can now use the DN "cn=John Smith,cn=users,o=acme,c=us" in access control, or make it a member of a group used in access control.

Several predefined objectclasses allow userPassword to be specified, including (but not limited to): person, organizationalperson, inetorgperson, organization, organizationalunit, and others.

The Directory Server passwords are case sensitive. If you create an entry with the userPassword value secret, a bind that specifies the password SECRET will fail.

When using a simple bind, the client sends the clear text password to the server as part of the bind request. This makes the password susceptible to protocol level snooping. An SSL connection could be used to protect the password (all information sent over an SSL connection is encrypted). Or the CRAM-MD5 SASL method can be used.

The CRAM-MD5 method requires that the server have access to the clear text password (password protection is set to none, which really means the password is stored in decryptable form and returned on searches as clear text). The client sends the DN to the server. The server retrieves the userPassword value for the entry and generates a random string. The random string is sent to the client. Both the client and the server hash the random string using the password as the key, and the client sends the result to the server. If the two hashed strings match, the bind request is successful, and the password was never sent to the server.

In order to use CRAM-MD5 the server must be configured so that password protection is None and the QRETSVRSEC (Retain server security data) system value must be 1 (Retain data).

Binding as a published user

The Directory Server provides a means to have an LDAP entry whose password is that of an i5/OS user profile on the same system. To do this, the entry must:

- have a UID attribute, whose value is the name of an i5/OS user profile
- not have a userPassword attribute

When the server receives a bind request for an entry that has a UID value but no userPassword, the server calls i5/OS security to validate that the UID is a valid user profile name and that the specified password is the correct password for that user profile. Such an entry is called a published user in reference to publishing of the system distribution directory (SDD) to LDAP, which creates such entries.

Binding as a projected user

An LDAP entry representing an i5/OS user profile is referred to as a projected user. You can use the DN of a projected user along with the correct password for that user profile in a simple bind. For example, the DN for user JSMITH on system my-system.acme.com would be:

```
os400-profile=JSMITH,cn=accounts,os400-sys=my-system.acme.com
```

SASL EXTERNAL bind

If an SSL or TLS connection is used with client authentication (for example, the client has a private certificate), the SASL EXTERNAL method can be used. This method tells the server to get the client's identity from an external source, in this case the SSL connection. The server gets the public portion of the

client certificate (sent to the server as part of establishing the SSL connection) and extracts the subject DN. That DN is assigned by the LDAP server to the connection.

For example, given a certificate assigned to:

```
common name: John Smith
organization unit: Engineering
organization: ACME
locality: Minneapolis
state: MN
country: US
```

The subject DN would be:

```
cn=John Smith,ou=Engineering,o=acme,l=Minneapolis,st=MN,c=US
```

Note that the cn, ou, o, l, st, and c elements are used in the order shown to generate the subject DN.

SASL GSSAPI bind

The SASL GSSAPI bind mechanism is used to authenticate to the server using a Kerberos ticket. This is useful when the client has done a KINIT or other form of Kerberos authentication (for example, Windows 2000 domain login). In this case, the server validates the client's ticket and then gets the Kerberos principal and realm names; for example, principal jsmith in realm acme.com, normally expressed as jsmith@acme.com. The server can be configured to map this identity to a DN in one of two ways:

- Generate a pseudo DN of the form `ibm-kn=jsmith@acme.com`
- Search for an entry having the `ibm-securityidentities` auxiliary class and an `altsecurityidentities` value of the form `KERBEROS:<principal>@<realm>`.

An entry that could be used for jsmith@acme.com might look like:

```
dn: cn=John Smith,cn=users,o=acme,c=us
objectclass: inetorgperson
objectclass: ibm-securityidentities
cn: John Smith
sn: Smith
altsecurityidentities: kerberos:jsmith@acme.com
```

For information about how to enable Kerberos authentication, see "Enable Kerberos authentication on the Directory Server" on page 125.

OS400-PRFTKN bind

The OS400-PRFTKN SASL bind mechanism is used to authenticate to the server using a profile token (refer to the Generate Profile Token API). When this mechanism is used, the server validates the profile token and associates the DN of the projected user profile with the connection (for example, `os400-profile=JSMITH,cn=accounts,os400-system=my-as400.mycompany.com`). If the application already has a profile token, this mechanism avoids the need to get the user profile name and user password to perform a simple bind. To use this mechanism, use the `ldap_sasl_bind` s API, specifying a null DN, OS400-PRFTKN for the mechanism, and a `berval` (binary data that is encoded using simplified basic encoding rules) containing the 32-byte profile token for the credentials.

LDAP as an authentication service

LDAP is commonly used to provide an authentication service. You can configure a Web server to authenticate to LDAP. By setting up multiple Web servers (or other applications) to authenticate to LDAP, you can establish a single user registry for those applications, rather than defining users over and over for each application or Web server instance.

How does this work? In short, the Web server prompts the user for a user name and password. The Web server takes this information and then does a search in the LDAP directory for an entry with that user name (for example, you might configure the Web server to map the user name to the LDAP 'uid' or 'mail' attributes). If it finds exactly one entry, the Web server then sends a bind request to the server using the DN of the entry it just found and the user provided password. If the bind is successful, the user is now authenticated. SSL connections may be used to protect the password information from protocol level snooping.

The Web server may also keep track of the DN that was used so that a given application can use that DN, perhaps by storing customization data in that entry, another entry associated with it, or in a separate database using the DN as a key to find the information.

A common alternative to using a bind request is to use the LDAP compare operation. For example `ldap_compare(ldap_session, dn, "userPassword", enteredPassword)`. This allows the application to use a single LDAP session, rather than starting and ending sessions for each authentication request.

Operating system projected backend

The system projected backend has the ability to map i5/OS objects as entries within the LDAP-accessible directory tree. The projected objects are LDAP representations of i5/OS objects instead of actual entries stored in the LDAP server database. User profiles are the only objects being mapped or projected as entries within the directory tree. The mapping of user profile objects is referred to as the i5/OS user projected backend.

LDAP operations are mapped to the underlying i5/OS objects and LDAP operations perform operating system functions in order to access these objects. All LDAP operations performed on the user profiles are done under the authority of the user profile associated with the client connection.

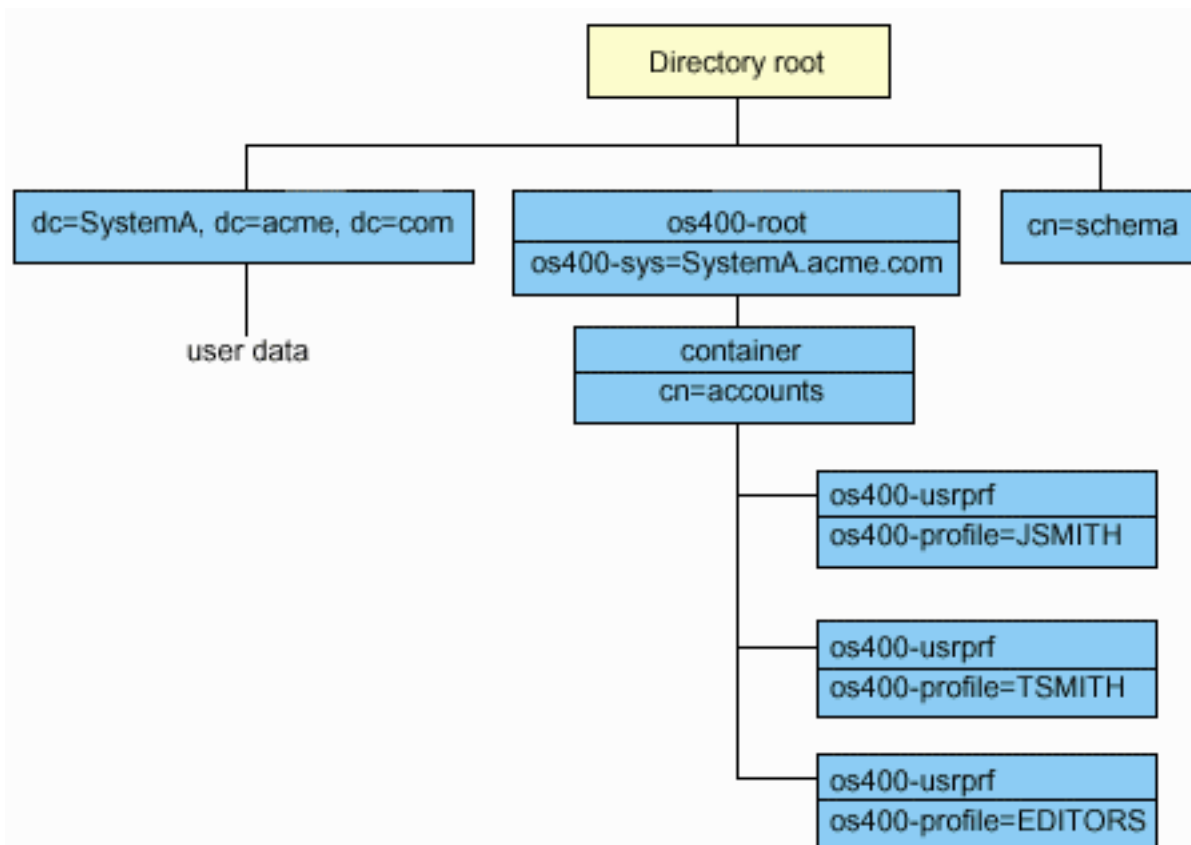
For more detailed information about the operating system projected backend, see the following:

- "i5/OS user projected directory information tree"
- "LDAP operations" on page 68
- "Administrator and replica bind DNs" on page 72
- "i5/OS user-projected schema" on page 72
- "Read access to projected users" on page 72

i5/OS user projected directory information tree

The figure below shows a sample directory information tree (DIT) for the user projected backend. The figure shows both individual and group profiles. In the figure, JSMITH and TSMITH are user profiles, which is indicated internally by the group identifier (GID), `GID=*NONE` (or 0); EDITORS is a group profile, which is indicated internally by a non-zero GID.

The suffix `dc=SystemA,dc=acme,dc=com` is included in the figure for reference. This suffix represents the current database backend which is managing other LDAP entries. The suffix `cn=schema` is the current server-wide schema being used.



The root of the tree is a suffix, which defaults to `os400-sys=SystemA.acme.com`, where *SystemA.acme.com* is the name of your system. The objectclass is `os400-root`. Although the DIT cannot be modified or deleted, you may reconfigure the system objects' suffix. However, you must ensure that the current suffix is not being used in ACLs or elsewhere on the system where entries would need to be modified should the suffix be changed.

In the previous figure, the container, `cn=accounts`, is shown below the root. This object cannot be modified. A container is placed at this level in anticipation of other kinds of information or objects that may be projected by the operating system in the future. Below the `cn=accounts` container are the user profiles that are projected as `objectclass=os400-usrprf`. The user profiles are referred to as projected user profiles and are known to LDAP in the form `os400-profile=JSMITH,cn=accounts,os400-sys=SystemA.acme.com`.

LDAP operations

The following are the LDAP operations that can be performed using the projected user profiles.

Bind

An LDAP client can bind (authenticate) to the LDAP server using a projected user profile. This is accomplished by specifying the projected user profile distinguished name (DN) for the bind DN and the correct i5/OS user profile password for authentication. An example of a DN used in a bind request would be `os400-profile=jsmith,cn=accounts,os400-sys=systemA.acme.com`.

A client must bind as a projected user to access information in the system projected backend.

Two additional mechanisms are available to authenticate to the directory server as an i5/OS user:

- GSSAPI SASL bind. If i5/OS is configured to use Enterprise Identity Mapping (EIM), the directory server queries EIM to determine if there is an association to a local i5/OS user profile from the initial Kerberos identity. If there is such an association, the server will associate the user profile with the connection and it can be used to access the system projection backend. For more information about EIM, see the EIM topic.
- OS400-PRFTKN SASL bind. A profile token can be used to authenticate to the directory server. The server associates the profile token user profile with the connection.

The server performs all of the operations using the authority of that user profile. The projected user profile DN can also be used in LDAP ACLs like other LDAP entry DN's. The simple bind method is the only bind method that is allowed when a projected user profile is specified on a bind request.

Search

The system projected backend supports some basic search filters. You can specify the objectclass, os400-profile, and os400-gid attributes in search filters. The os400-profile attribute supports wildcards. The os400-gid attribute is limited to specifying (os400-gid=0), which is an individual user profile, or !(os400-gid=0), which is a group profile. You can retrieve all attributes of a user profile except the password and similar attributes.

For certain filters, only the DN objectclass and os400-profile values are returned. However, subsequent searches can be conducted to return more detailed information.

The following table describes the behavior of the system projected backend for search operations.

Table 2. System projected backend behavior for search operations

Search requested	Search base	Search scope	Search filter	Comments
Return information for os400-sys=SystemA, (optionally) for the containers under it, and (optionally) for the objects in those containers.	os400-sys=SystemA.acme.com	base, sub, or one	objectclass=* objectclass=os400-root objectclass=container objectclass=os400-usrprf	Return the appropriate attributes and their values based on the scope and filter specified. Hardcoded attributes and their values are returned for the system objects' suffix and the container under it.
Return all user profiles.	cn=accounts, os400-sys=SystemA.acme.com	one or sub	os400-gid=0	Only the distinguished name (DN), objectclass, and os400-profile values are returned for projected user profiles. If any other filter is specified, LDAP_UNWILLING_TO_PERFORM is returned.
Return all group profiles.	cn=accounts, os400-sys=SystemA.acme.com	one or sub	(!(os400-gid=0))	Only the distinguished name (DN), objectclass, and os400-profile values are returned for projected user profiles. If any other filter is specified, LDAP_UNWILLING_TO_PERFORM is returned.

Table 2. System projected backend behavior for search operations (continued)

Search requested	Search base	Search scope	Search filter	Comments
Return all user and group profiles.	cn=accounts, os400- sys=SystemA.acme.com	one or sub	os400-profile=*	Only the distinguished name (DN), objectclass, and os400-profile values are returned for projected user profiles. If any other filter is specified, LDAP_UNWILLING_TO_PERFORM is returned.
Return information for a specific user or group profile such as the user profile JSMITH.	cn=accounts, os400- sys=SystemA.acme.com	one or sub	os400-profile=JSMITH	Other attributes to be returned can be specified.
Return information for a specific user or group profile such as the user profile JSMITH.	os400-profile=JSMITH, cn=accounts, os400- sys=SystemA.acme.com	bas, sub, or one	objectclass=os400-usrprf objectclass=* os400-profile=JSMITH	Other attributes to be returned can be specified. Even though a scope of one level can be specified, the search results would return no values because there is nothing below the user profile JSMITH in the DIT.
Return all user and group profiles starting with A.	cn=accounts, os400- sys=SystemA.acme.com	one or sub	os400-profile=A*	Only the distinguished name (DN), objectclass, and os400-profile values are returned for projected user profiles. If any other filter is specified, LDAP_UNWILLING_TO_PERFORM is returned.
Return all group profiles starting with G.	cn=accounts, os400- sys=SystemA.acme.com	one or sub	(&(!(os400-gid=0)) (os400-profile=G*))	Only the distinguished name (DN), objectclass, and os400-profile values are returned for projected user profiles. If any other filter is specified, LDAP_UNWILLING_TO_PERFORM is returned.

Table 2. System projected backend behavior for search operations (continued)

Search requested	Search base	Search scope	Search filter	Comments
Return all user profiles starting with A.	cn=accounts, os400- sys=SystemA.acme.com	one or sub	(&(os400-gid=0) (os400-profile=A*))	Only the distinguished name (DN), objectclass, and os400-profile values are returned for projected user profiles. If any other filter is specified, LDAP_UNWILLING_TO_PERFORM is returned.

Compare

The LDAP compare operation can be used to compare an attribute value of a projected user profile. The os400-aut and os400-docpwd attributes cannot be compared.

Add and modify

You can create user profiles using the LDAP add operation and you can also modify user profiles using the LDAP modify operation.

Delete

User profiles can be deleted using the LDAP delete operation. To specify the behavior of the DLTUSRPRF OWNBOBJOPT and PGPOPT parameters, two LDAP server controls are now provided. These controls can be specified on the LDAP delete operation. Refer to the Delete User Profile (DLTUSRPRF) command for more information about the behavior of these parameters.

The following are the controls and their object identifiers (OIDs) that can be specified on the LDAP delete client operation.

- os400-dltusrprf-ownobjopt 1.3.18.0.2.10.8

The control value is a string of the following form:

- controlValue ::= ownObjOpt [newOwner]
- ownObjOpt ::= *NODLT / *DLT / *CHGOWN

The ownObjOpt control value specifies the action to be taken if the user profile owns any objects. The value of *NODLT indicates not to delete the user profile if the user profile owns any objects. The *DLT value indicates to delete the owned objects and the *CHGOWN value indicates to transfer ownership to another profile.

The newOwner value specifies the profile to which ownership is transferred. This value is required when ownObjOpt is set to *CHGOWN.

Examples of the control value are the following:

- *NODLT: specifies that the profile cannot be deleted if it owns any objects
- *CHGOWN SMITH: specifies to transfer the ownership of any objects to the SMITH user profile.
- The object identifier (OID) is defined in ldap.h as LDAP_OS400_OWNOBJOPT_CONTROL_OID.
 - os400-dltusrprf-pgpopt 1.3.18.0.2.10.9

The control value is defined as a string of the following form:

```
controlValue ::=pgpOpt [ newPgp [ newPgpAut ] ]
pgpOpt ::= *NOCHG / *CHGPGP
```

```
newPgp ::= *NONE / user-profile-name
newPgpAut ::= *OLDPGP / *PRIVATE / *ALL / *CHANGE / *USE / *EXCLUDE
```

The `pgpOpt` value specifies the action to be taken if the profile being deleted is the primary group for any objects. If `*CHGPGP` is specified, `newPgp` must also be specified. The `newPgp` value specifies the primary group profile name or `*NONE`. If a new primary group profile is specified, the `newPgpAut` value may also be specified. The `newPgpAut` value specifies the authority to the objects that the new primary group is given.

Examples of the control value are the following:

- `*NOCHG`: specifies that the profile cannot be deleted if it is the primary group for any objects.
- `*CHGPGP *NONE`: specifies to remove the primary group for the objects.
- `*CHGPGP SMITH *USE`: specifies to change the primary group to the SMITH user profile and to grant `*USE` authority to the primary group.

If either of these controls is not specified on the delete, the defaults currently in effect for the `QSYS/DLTUSRPRF` command are used instead.

ModRDN

You cannot rename projected user profiles because this is not supported by the operating system.

Import and Export APIs

The `QgldImportLdif` and `QgldExportLdif` APIs do not support importing or exporting data within the system projected backend.

Administrator and replica bind DN

You can specify a projected user profile as the configured administrator or replica bind DN. The password of the user profile is used. Projected user profiles can also become LDAP administrators if they are authorized to the Directory Server Administrator function identifier (`QIBM_DIRSRV_ADMIN`). Multiple user profiles can be granted administrator access.

For more information, see “Work with administrative access for authorized users” on page 105.

i5/OS user-projected schema

The object classes and attributes from the projected backend can be found in the server-wide schema. The names of the LDAP attributes are in the format `os400-nnn`, where *nnn* is typically the keyword of the attribute on the user profile commands. For example, the `os400-usrcls` attribute corresponds to the `USRCLS` parameter of the `CRTUSRPRF` command. The values of the attributes correspond to the parameter values accepted by the `CRTUSRPRF` and `CHGUSRPRF` commands, or the values displayed when displaying a user profile. Use the Web administration tool or another application to view the definitions of the `os400-usrprf` objectclass and the associated `os400-xxx` attributes.

| Read access to projected users

| **Note:** PTF 5722SS1-SI19805 is required to enable or disable read access to projected users.

| By default, the system projection backend provides read access to user profile information to authorized users. Read access to projected users can be enabled or disabled by a configuration setting in the
| `/QIBM/UserData/OS400/DirSrv/ibmslapd.conf` file. The following line can be added to the `cn=Front
| End, cn=Configuration` stanza of the configuration file to disable search and compare operations to the
| user projected backend:

```
| ibm-slapdSetEnv: IBMSLAPDOS400USRPRJREAD=FALSE
```

| If the value FALSE is specified, read access to projected user information is disabled. If the value is
| anything other than FALSE such as TRUE or the setting is not present in the configuration file, read access
| to projected user information is enabled.

| For information on how to modify the configuration file, see “Enable or disable read access to projected
| users” on page 136.

Directory Server and i5/OS journaling support

Directory Server uses i5/OS database support to store directory information. Directory Server uses commitment control to store directory entries in the database. This requires i5/OS journaling support.

When the server or the LDIF import tool is started for the first time, the following are built:

- A journal
- A journal receiver
- Any database tables needed initially

The journal QSQJRN is built in the database library that you have configured. The journal receiver QSQJRN0001 is initially created in the database library that you have configured.

Your environment, directory size and structure, or save and restore strategy may dictate some differences from the defaults, including how these objects are managed and the size threshold used. You can change journaling command parameters if necessary. LDAP journaling is set up by default to delete old receivers. If the change log is configured and you want to keep old receivers, execute the following command from an i5/OS command line:

```
CHGJRN JRN(QUSRDIRCL/QSQJRN) DLTRCV(*NO)
```

If the change log is configured, you can delete its old journal receivers with the following command:

```
CHGJRN JRN(QUSRDIRCL/QSQJRN) DLTRCV(*YES)
```

For information about journaling commands, see “OS/400 commands” in the Programming topic.

Operational attributes

There are several attributes that have special meaning to the Directory Server known as operational attributes. These are attributes that are maintained by the server and either reflect information the server manages about an entry or affect server operation. These attributes have special characteristics:

- The attributes are not returned by a search operation unless they are specifically requested (by name) in the search request
- The attributes are not part of any object class. The server controls what entries have the attributes.

The following sets of operational attributes are supported by the Directory Server:

- `creatorsName`, `createTimestamp`, `modifiersName`, `modifyTimestamp`. Present on every entry. These attributes show the bind DN and time when an entry was first created or last modified. You can use these attributes in search filters, for example, to find all entries modified after a specified time. These attributes cannot be modified by any user.
- `ibm-entryuuid`. Present on every entry that is created while the server is at V5R3 or later. This attribute is a universally unique string identifier assigned to each entry by the server when the entry is created. It is useful for applications that need to distinguish between identically named entries on different servers. The attribute uses the DCE UUID algorithm to generate an ID that is unique across all entries on all servers using a timestamp, adapter address, and other information.
- `entryowner`, `ownersource`, `ownerpropagate`, `aclentry`, `aclsource`, `aclpropagate`, `ibm-filteracl`, `ibm-filteraclinherit`, `ibm-effectiveAcl`. For more information, see “Access control lists” on page 49.

- hasSubordinates. Present on every entry and has the value TRUE if the entry has subordinates.
- numSubordinates. Present on every entry and contains the number of entries which are children of this entry.
- pwdChangedTime, pwdAccountLockedTime, pwdExpirationWarned, pwdFailureTime, pwdGraceUseTime, pwdReset, pwdHistory. (password policy attributes).
- subschemasubentry - Present on every entry and identifies the location of the schema for that part of the tree. This is useful for servers with multiple schemas if you want to find the schema that you can use in that part of the tree.

Controls and extended operations

Controls

Controls provide additional information to the server to control how it interprets a given request. For example, a delete subtree control can be specified on a LDAP delete request, indicating that the server should delete the entry and all its subordinate entries, rather than deleting just the entry specified. A control consists of three parts:

- The control type, which is an OID identifying the control.
- A criticality indicator, which specifies how the server should behave if it does not support the control. This is a Boolean value. FALSE indicates the control is not critical, and the server should ignore it if it doesn't support it. TRUE indicates the control is critical and the entire request should be failed (with an unsupported critical extension error) if the server cannot honor the control.
- An optional control value, which contains other information specific to the control. The content of the control value is specified using ASN.1 notation. The value itself is the BER encoding of the control data.

The following controls are supported by the Directory Server:

Name	OID	Earliest OS/400 release	Earliest IBM Directory Server version	Description
Manage DSA IT	2.16.840.1.1137.30.3.4.2	V4R5	V3.2	Treat referral entries as regular entries.
Transaction	1.3.18.0.2.10.5	V4R5	V3.2	Mark an operation as part of a transaction.
OS/400 DLTUSRPRF OWNBJOPT	1.3.18.0.2.10.8	V5R2		OS/400 delete user profile option for object owner. See "Operating system projected backend" on page 67 for details.
OS/400 DLTUSRPRF PGPOPT	1.3.18.0.2.10.9	V5R2		OS/400 delete user profile option for primary group. See "Operating system projected backend" on page 67 for details.

Name	OID	Earliest OS/400 release	Earliest IBM Directory Server version	Description
Sorted search	1.2.840.113556.1.4.473 (request) and 1.2.840.113556.1.4.474 (response)	V5R2 with PTF	V4.1	Sort search results before returning the entries to the client.
Paged search	1.2.840.113556.1.4.319	V5R2 with PTF	V4.1	Return search results in pages to the client instead of all at once.
Tree Delete control	1.2.840.113556.1.4.805	V5R3	V5.1	This control is attached to a Delete request to indicate that the specified entry and all descendant entries are to be deleted. User must be a directory administrator. The entry to be deleted cannot be a replication context.
Password policy	1.3.6.1.4.1.42.2.27.8.5.1	V5R3	V5.1	Return extra password policy error information to the client.
Server administration	1.3.18.0.2.10.15	V5R3	V5.1	Permits the administrator to perform repair operations that would normally be refused (for example: update a read-only replica, update a quiesced server, or set certain operational attributes).

Extended operations

Extended operations are used to start additional operations beyond the core LDAP operations. For example, extended operations have been defined to group a set of operations into a single transaction. An extended operation consists of:

- The request name, an OID which identifies the specific operation.
- An optional request value, which contains other information specific to the operation. The content of the request value is specified using ASN.1 notation. The value itself is the BER encoding of the request data.

Extended operations typically have an extended response. The response consists of:

- The components of the standard LDAP result (error code, matched DN, and error message)
- The response name, an OID which identifies the type of response

- An optional value, which contains other information specific to the response. The content of the response value is specified using ASN.1 notation. The value itself is the BER encoding of the response data.

The following extended requests are supported by the Directory Server:

Name	OID	Earliest OS/400 release	Earliest IBM Directory Server version	Description
Register for events	1.3.18.0.2.12.1	V4R5	V3.2	
Unregister for events	1.3.18.0.2.12.3	V4R5	V3.2	
Begin transaction	1.3.18.0.2.12.5	V4R5	V3.2	
End transaction	1.3.18.0.2.12.6	V4R5	V3.2	
DN normalize request	1.3.18.0.2.12.30	V5R3	V5.1	

Additional extended operations are defined which are not intended to be started by a client. These operations are used through the ldapexop utility or through operations performed by the Web Administration tool. These operations, and the authority required to start them are listed below:

Name	OID	Earliest OS/400 release	Earliest IBM Directory Server version	Description
Control replication	1.3.18.0.2.12.16	V5R3	V5.1	This operation performs the requested action on the server it is issued to and cascades the call to all consumers beneath it in the replication topology. The client must be the directory administrator or have write authority to ibm-replicagroup=default object for the associated replication context.
Control replication queue	1.3.18.0.2.12.17	V5R3	V5.1	This operation marks items as already replicated for a specified agreement. This operation is allowed only when the client has write authority to the replication agreement.

Name	OID	Earliest OS/400 release	Earliest IBM Directory Server version	Description
Quiesce or unquiesce	1.3.18.0.2.12.17	V5R3	V5.1	This operation puts the subtree into a state where it does not accept client updates (or terminates this state), except for those from clients authenticated as a directory administrator where the Server Administration control is present. The client must be authenticated as the directory administrator or have write authority to the ibm-replicagroup=default object for the associated replication context.
End transaction	1.3.18.0.2.12.19	V5R3	V5.1	
Cascading control replication	1.3.18.0.2.12.15	V5R3	V5.1	This operation performs the requested action on the server it is issued to and cascades the call to all consumers beneath it in the replication topology. The client must be the directory administrator or have write authority to ibm-replicagroup=default object for the associated replication context.
Update configuration	1.3.18.0.2.12.28	V5R3	V5.1	This operation is used to cause the server to reread specified settings from its configuration. The operation is allowed only when the client is the directory administrator.

Chapter 5. Get started with Directory Server

Directory Server is automatically installed when you install i5/OS. The Directory Server includes a default configuration. To get started with the Directory Server, do the following:

1. If you are installing V5R3 and were using Directory Server on a previous release, then review the migration considerations. For more information, see “Migration considerations.”
2. Plan your Directory Server. For more information, see “Plan your Directory Server” on page 84.
3. To customize Directory Server settings, run the Directory Server Configuration wizard. For more information see “Configure Directory Server” on page 84.
4. Start the server. For more information, see “Start the Directory Server” on page 100
5. Use the Web administration tool to create or edit LDAP directories. For more information see “Web administration” on page 86.
6. Refer to the information in the Chapter 7, “Administer Directory Server,” on page 99 section to find more information about how to perform various Directory Server tasks.

Migration considerations

Directory Server is automatically installed when you install i5/OS. The first time the server is started, it automatically migrates any existing configuration and data. This may cause a long delay before the server is started the first time.

If you have a Directory Server running under V5R2 or V5R1, see “Migrate to V5R3 from V5R2 or V5R1.”

If you have a Directory Server running under V4R3, V4R4, or V4R5, you can migrate your data to V5R3. For more information, see “Migrate data from V4R3, V4R4, or V4R5 to V5R3” on page 80.

If you have a network of replicating servers, see “Migrating a network of replicating servers” on page 81 for more information.

If you are using Kerberos, see “Kerberos service name change” on page 83.

Migrate to V5R3 from V5R2 or V5R1

V5R3 of OS/400 introduces new features and capabilities to Directory Server. These changes affect both the LDAP directory server and the graphical user interface (GUI) of iSeries Navigator. To take advantage of the new GUI features, you need to install iSeries Navigator on a PC that can communicate over TCP/IP to your iSeries server. iSeries Navigator is a component of iSeries Access for Windows. If you have an earlier version of iSeries Navigator installed, you should upgrade to V5R3.

V5R3 of OS/400 supports upgrades from V5R1 and V5R2. When you upgrade to V5R3 of OS/400, both the LDAP directory data and the directory schema files are automatically migrated to conform to V5R3 formats.

When you upgrade to V5R3 of OS/400, you should be aware of some migration issues:

- When you upgrade to V5R3, Directory Server automatically migrates your schema files to V5R3 and deletes the old schema files. However, if you have deleted or renamed the schema files, Directory Server cannot migrate them. You may receive an error or Directory Server may assume that the files have already been migrated.
- Directory Server migrates directory data to the V5R3 format the first time that you start the server or import an LDIF file. Plan to allow some time for this migration to complete.

After you upgrade to V5R3, you should start your server once to migrate existing data before importing new data. If you try to import data before starting the server once and you do not have enough authority, the import may fail.

- Following migration, the LDAP directory server will automatically start when TCP/IP starts. If you do not want the directory server to start automatically, use iSeries Navigator to change the setting.

Migrate data from V4R3, V4R4, or V4R5 to V5R3

V5R3 of OS/400 does not support direct upgrades from V4R3, V4R4, or V4R5. If you want to migrate a V4R3, V4R4, or V4R5 Directory Server to V5R3, you can follow either of the following procedures:

- “Upgrade OS/400 from V4R3, V4R4, or V4R5 to an interim release”
- “Save the database library and install V5R3” on page 81

Before you start, read the following:

- When you upgrade from V4R3 to any later release, you should be aware of the following issues:
 - **Migrating the key ring file to a key database:**

The LDAP directory server also used a key ring file for its own SSL connections in V4R3. Beginning with V4R4 it uses the system certificate store. If your server was set up to use SSL in V4R3, the contents of the key ring file will be migrated to the system certificate store.

- **Two stream files have been removed:**

The following stream files used by Directory Server in V4R3 are no longer needed and are automatically removed when you install a later release:

```
/QIBM/ProdData/OS400/DirSrv/qgldcert.kyr  
/QIBM/ProdData/OS400/DirSrv/qgldcert.sth
```

You do not need to take any action with these files. This is mentioned only so that you are not concerned if you notice that they are no longer present on your system.

- V4R4 and earlier releases of Directory Server did not take time zones into account when creating time stamp entries. Beginning with V4R5, the time zone is used in all additions and modifications to the directory. Therefore, if you upgrade data from V4R4 or earlier, Directory Server adjusts existing `createtimestamp` and `modifytimestamp` attributes to reflect the correct time zone. It does this by subtracting the time zone that is currently defined on the iSeries system from the time stamps that are stored in the directory. Note that if the current time zone is not the same time zone that was active when the entries were originally created or modified, the new time stamp values will not reflect the original time zone.
- If you are upgrading data from V4R4 or earlier, be aware that the directory data will require approximately twice as much storage space than it required previously. This is because in V4R4 or earlier versions, Directory Server supported only the IA5 character set and saved data in ccsid 37 (single byte format). Directory Server supports the full ISO 10646 character set. After you upgrade, you should start your server once to migrate existing data before importing new data. If you try to import data before starting the server once and you do not have enough authority, the import may fail.
- Also be aware that there may be additional issues associated with upgrading to the current release from other releases.

Upgrade OS/400 from V4R3, V4R4, or V4R5 to an interim release

Though upgrades from V4R3, V4R4, and V4R5 of OS/400 to V5R3 are not supported, the following upgrades are supported:

- V4R3 and V4R4 upgraded to V4R5
- V4R4 and V4R5 upgraded to V5R1
- V4R5 and V5R1 upgraded to V5R2
- V5R1 and V5R2 upgraded to V5R3

One way to migrate your Directory Server server is to upgrade to an interim release (V5R1 or V5R2), then to V5R3. For detailed information about OS/400 installation procedures, see *Software Installation*



. Follow these general steps to perform the migration:

1. Note any changes that you have made to the schema files in the /QIBM/UserData/OS400/DirSrv directory. The schema files are migrated automatically.
2. For V5R3, do the install of V4R5.
3. For V4R4 or V4R5, do the install of V5R1 or V5R2.
4. Do the install to V5R3.
5. Start the Directory Server if not already started.
6. Use the Web administration tool to modify the schema files for any user changes that you noted in step 1.
7. Restart the Directory Server.

Save the database library and install V5R3

You can migrate your Directory Server server by saving the database library that Directory Server uses in V4R3, V4R4, or V4R5 and then restoring it after installing V5R3. This saves you the step of installing an interim release. However, the server's settings are not migrated, so you must reconfigure the server

settings. For detailed information about OS/400 installation procedures, see *Software Installation*  . Follow these general steps to perform the migration:

1. Note any changes that you have made to the schema files in the /QIBM/UserData/OS400/DirSrv directory. The schema files are not migrated automatically, so if you want to keep your changes you will need to manually implement them again.
2. Note the various configuration settings in the Directory Server properties, including the database library name.
3. Save the database library that is specified in the Directory Server configuration. If you configured the change log, then will also need to save the QUSRDIRCL library.
4. Note the publishing configuration.
5. Install V5R3 of OS/400 on the system.
6. Use EZ-Setup to configure the Directory Server.
7. Restore the database library that you saved in step 3. If you saved the QUSRDIRCL library in step 3, restore it now.
8. Use the Web administration tool to modify the schema files for any user changes that you noted in step 1.
9. Use iSeries Navigator to reconfigure Directory Server. Specify the database library that was previously configured and that was saved and restored in previous steps
10. Use iSeries Navigator to reconfigure publishing.
11. Restart the Directory Server.

Migrating a network of replicating servers

The first time that the master server is started, it migrates the information in the directory that controls replication. The entries with objectclass replicaObject under cn=localhost are replaced with entries used by the new replication model (for more information, see "Replication" on page 35). The master server is configured to replicate all the suffixes in the directory. The agreement entries are created with the attribute ibm-replicationOnHold set to true. This allows updates made to the master to be accumulated for the replica until the replica is ready.

These entries are referred to as the replication topology. The new master can be used with replicas running prior versions; data related to the new features will not be replicated to the back-level servers. It is necessary to export the replication topology entries from the master and add them to each replica after

the replica server has been migrated. To export the entries, use the Qshell command line tool “ldapsearch” on page 177 and save the output to a file. The search command is similar to the following:

```
ldapsearch -h master-server-host-name -p master-server-port \  
-D master-server-admin-DN -w master-server-admin-password \  
-b ibm-replicagroup=default,suffix-entry-DN \  
-L "(|(objectclass=ibm-replicaSubEntry)(objectclass=ibm-replicationAgreement))" \  
> replication.topology.ldif
```

This command creates an output LDIF file named replication.topology.ldif in the current working directory. The file contains only the new entries.

Note: Do not include the following suffixes:

- cn=changelog
- cn=localhost
- cn=pwdpolicy
- cn=schema
- cn=configuration

Include only user-created suffixes.

Repeat the command for each suffix entry on the master, but replace “>” with “>>” to append the data to the output file for subsequent searches. After the file is complete, copy it to the replica servers.

Add the file to the replica servers after they have been successfully migrated; do not add the file to servers running previous versions of the directory server. You must start and stop the server before you add the file.

To start the server, use the **Start** option in iSeries Navigator. For more information, see “Start the Directory Server” on page 100.

To stop the server, use the **Stop** option in iSeries Navigator. For more information see, “Stop the Directory Server” on page 100.

When you add the file to a replica server, be sure that the replica server is not started. To add the data, use the **Import File** option in iSeries Navigator.

After the replication topology entries are loaded, start the replica server and resume replication. You can resume replication in one of the following ways:

- On the master server, use **Manage Queues in Replication Management** in the Web administration tool.
- Use the **ldapexop** command line utility. For example:

```
ldapexop -h master-server-host-name -p master-server-port \  
-D master-server-admin-DN -w master-server-admin-password \  
-op controlrepl -action resume -ra replica-agreement-DN
```

This command resumes replication for the server defined in the entry with the specified DN.

To determine which replica agreement DN corresponds to a replica server, look in the replication.topology.ldif file. The master server will log a message that replication has started for that replica and a warning that the replica server’s ID in the agreement does not match the replica’s server ID. To update the replica agreement to use the correct server ID, use **Replication Management** in the Web administration tool, or the command line tool **ldapmodify**. For example:


```

ldapmodify -c -h master-server-host-name -p master-server-port \
  -D master-server-admin-DN -w master-server-admin-password
dn: replica-agreement-DN
changetype: modify
replace: ibm-replicaConsumerID
ibm-replicaConsumerID: replica-server-ID

```

You can enter these commands directly on the command line, or you can save the commands in an LDIF file and supply them to the command with the `-i file` option. Use **End Previous Request** to stop the command.

Migration for this replica is complete.

To continue to use a replica running a previous version, it is still necessary to resume replication using the command line tool **ldapexop** or **Replication Management** in the Web administration tool for that replica. If a replica running a previous version is migrated later, use the command line tool **ldapdiff** to synchronize the directory data. This will ensure that entries or attributes that were not replicated are updated on the replica.

Kerberos service name change

In V5R3, the service name used by the directory server and client APIs for GSSAPI authentication (Kerberos) are changed. This change is incompatible with the service name used prior to V5R3 (V5R2M0 PTF 5722SS1-SI08487 includes the same change).

Previous to this release, the i5/OS directory server and client APIs have used a service name of the form LDAP/dns-host-name@Kerberos-realm when the GSSAPI mechanism (Kerberos) is used for authentication. This name does not comply with the standards that define GSSAPI authentication, which state that the principal name should start with lower case "ldap". As a result, the both the i5/OS directory server and client APIs may not interoperate with other vendor's products. This is particularly true if the Kerberos key distribution center (KDC) has case sensitive principal names. The LDAP service provider for JNDI, a commonly used Java LDAP client API, is an example of a client included with i5/OS that uses the correct service name.

V5R3M0 changes the service name to comply with the standards. This, however, introduces its own compatibility problems.

- A directory server configured to use GSSAPI authentication will not start installing this release. This is because the keytab file used by the server has credentials using the old service name (LDAP/mysys.ibm.com@IBM.COM), while the server is looking for credentials using the new service name (ldap/mysys.ibm.com@IBM.COM).
- A directory server or LDAP application using the LDAP APIs at V5R3M0 may not be able to authenticate with older i5/OS servers or clients. To correct this, you should do the following:
 1. If the KDC uses case sensitive principal names, create an account using the correct service name (ldap/mysys.ibm.com@IBM.COM).
 2. Update the keytab file used by the i5/OS Directory Server to contain credentials for the new service name. You might also want to delete the old credentials. You can use the Qshell keytab utility to update the keytab file. By default, the directory server uses the /QIBM/UserData/OS400/NetworkAuthentication/keytab/krb5.keytab file. The V5R3M0 Network Authentication Service (Kerberos) wizard in iSeries Navigator also creates keytab entries using the new service name.
 3. Update V5R2M0 i5/OS systems where GSSAPI is used by applying PTF 5722SS1-SI08487.

Alternately, you can choose to have the directory server and client APIs continue to use the old service name. This might be desirable when you are using Kerberos authentication in a mixed network of

systems running with and without the PTFs. To do this, set the LDAP_KRB_SERVICE_NAME environment variable. You can set this for the entire system (required to set service name for the server) using the following command:

```
ADDENVVAR ENVVAR(LDAP_KRB_SERVICE_NAME)
```

or in QSH (to affect LDAP utilities run from this QSH session):

```
export LDAP_KRB_SERVICE_NAME=1
```

Plan your Directory Server

Before you install Directory Server and begin to configure your LDAP directory, you should take a few minutes to plan the directory. Important things to consider include the following:

- **Organize the directory.** Plan the structure of your directory and determine what suffixes and attributes your server will require. For more information see, “Directories” on page 7, “Suffix (naming context)” on page 14, and “Attributes” on page 19.
- **Decide how large your directory will be.** You can then estimate how much storage you need. The size of the directory depends on the following:
 - The number of attributes in the servers schema.
 - The number of entries on the server.
 - The type of information that you store on the server.

For example, an empty directory that uses the default Directory Server schema requires approximately 10 MB of storage space. A directory that uses the default schema and which contains 1000 entries of typical employee information requires about 30 MB of storage space. This number will vary depending on the exact attributes that you used. It will also increase greatly if you stored large objects, such as pictures, in the directory.

- **Decide what security measures you will take.**

Directory server allows you to apply a password policy to ensure that users change their passwords periodically, and that the passwords meet the organization’s syntactic password requirements.

Directory Server supports the use of Secure Sockets Layer (SSL) and Digital Certificates as well as Transport Layer Security (TLS) for communication security. Kerberos authentication is also supported.

Directory Server allows you to control access to directory objects with access control lists (ACLs). You can also use i5/OS security auditing to protect the directory.

Additionally decide what password policy to apply.

- **Choose an administrator DN and password.** The default administrator DN is cn=admin. This is the only identity that authority to create or modify directory entries when the server is initially configured. You can use the default administrator DN or select a different DN. You also need to create a password for the administrator DN.
- **Install prerequisite software for the Directory Server Web administration tool.** In order to use the Directory Server Web administration tool, the following prerequisite products must be installed on the iSeries server.
 - IBM HTTP Server for iSeries (5722-DG1)
 - IBM WebSphere® Application Server - Express (5722-IWE Base and Option 2)

See the IBM HTTP Server topic for more information about IBM HTTP Server for iSeries and IBM WebSphere Application Server - Express.

Configure Directory Server

1. If your system has not been configured to publish information to another LDAP server and no LDAP servers are known to the TCP/IP DNS server, then Directory Server is automatically installed with a limited default configuration. See “Default configuration for Directory Server” on page 85 for more information. Directory Server provides a wizard to assist you in configuring the Directory Server for

your specific needs. You may run this wizard as part of EZ-Setup or run the wizard later from iSeries Navigator. Use this wizard when you initially configure the directory server. You may also use the wizard to reconfigure the directory server.

Note: When you use the wizard to reconfigure the directory server, you start configuring from scratch. The original configuration is deleted rather than changed. However, the directory data is not deleted, but instead remains stored in the library that you selected on installation (QUSRDIRDB by default). The change log also remains intact, in the QUSRDIRCL library by default.

If you want to start completely from scratch, clear those two libraries before starting the wizard.

If you want to change the directory server configuration, but not clear it completely, right-click **Directory** and select **Properties**. This does not delete the original configuration.

You must have *ALLOBJ and *IOSYSCFG special authorities to configure the server. If you want to configure OS/400 security auditing, you must also have *AUDIT special authority.

2. To start the Directory Server Configuration Wizard, take these steps:
 - a. In iSeries Navigator, expand **Network**.
 - b. Expand **Servers**.
 - c. Click **TCP/IP**.
 - d. Right-click **Directory** and select **Configure**.

Note: If you have already configured the directory server, click **Reconfigure** rather than **Configure**.

3. Follow the instructions in the Configure Directory Server wizard to configure your Directory Server.

Note: You may also want to put the library that stores the directory data in a user auxiliary storage pool (ASP) rather than the system ASP. However, this library cannot be stored in an Independent ASP and any attempt to configure, reconfigure, or start the server with a library that exists in an Independent ASP will fail.

4. When the wizard is finished, your Directory Server has a basic configuration. If you are running Lotus® Domino® on your system, then port 389 (the default port for the LDAP server) may already be in use by the Domino LDAP function. You must do one of the following:
 - Change the port that Lotus Domino uses. See “Host Domino LDAP and Directory Server on the same iSeries ” in the E-mail topic for more information.
 - Change the port that Directory Server uses. See “Change the port or IP address” on page 102 for more information.
 - Use specific IP addresses. See “Change the port or IP address” on page 102 for more information.
5. Create entries corresponding to the suffix or suffixes that you have configured. For more information, see “Add and remove Directory Server suffixes” on page 104.

You may want to do some or all of the following before continuing:

- Import data to the server, see “Import an LDIF file” on page 103.
- Enable Secure Sockets Layer (SSL) security, see “Enable SSL on the Directory Server” on page 123.
- Enable Kerberos authentication, see “Enable Kerberos authentication on the Directory Server” on page 125.
- Set up a referral, see “Specify a server for directory referrals” on page 103.

Default configuration for Directory Server

The Directory Server is automatically installed when you install OS/400. This installation includes a default configuration. The directory server uses the default configuration when all of the following are true:

- Administrators have not run the Directory Server Configuration Wizard or changed directory settings with the properties pages.
- Directory Server publishing is not configured.
- The Directory Server cannot find any LDAP DNS information.

If the Directory Server uses the default configuration, then the following occur:

- The Directory Server automatically starts when TCP/IP starts.
- The system creates a default administrator, cn=Administrator. It also generates a password that is used internally. If you need to use an administrator password later, you can set a new one from the Directory Server property page.
- A default suffix is created that is based on the system's IP name. A system objects' suffix is also created based on the system name. For example, if your system's IP name is mary.acme.com, the suffix is dc=mary,dc=acme,dc=com.
- The Directory Server uses the default data library QUSRDIRDB. The system creates it in the system ASP.
- The server uses port 389 for non-secure communications. If a digital certificate has been configured for LDAP, secure sockets layer (SSL) is enabled and port 636 is used for secure communications.

Web administration

One or more Directory Servers can be administered through the Web administration console. The Web administration console allows you to:

- Add or change the list of Directory Servers that can be administered.
- Administer a Directory Server using the Web administration tool.
- Change Web administration console attributes.

To use the Web administration console, do the following:

1. If this is the first time that you are using Directory Server Web administration, you need to first set up Web administration (see "Set up Web administration for the first time" on page 87) and then continue with the next step.
2. Log on to the Directory Server Web administration by doing one of the following:
 - From iSeries Navigator, select your server and click **Network > Servers > TCP/IP**, right click **Directory**, and click **Server Administration**.
 - From the iSeries Tasks page (http://your_server:2001) click **IBM Directory Server**.
3. If you want to administer a Directory Server, do the following:
 - a. Select the Directory Server that you want to administer in the **LDAP Hostname** field.
 - b. Enter the administrator login DN that you use to bind to the directory server.
 - c. Enter the administrator password.
 - d. Click **Login**. The IBM Directory Server Web Administration Tool page is displayed. For more information about the IBM Directory Server Web Administration Tool page, see "Web administration tool" on page 88.
4. If you want to add or change the list of Directory Servers that can be administered, or change the Web administration console attributes, do the following:
 - a. Select the **Console Admin** in the **LDAP Hostname** field.
 - b. Enter the console administrator login.
 - c. Enter the console administrator password.
 - d. Click **Login**. The IBM Directory Server Web Administration Tool page is displayed. For more information about the IBM Directory Server Web Administration Tool page, see "Web administration tool" on page 88.
 - e. Click **Console administration** and then select one of the following:

- **Change console administrator login** to change the name of the console administrator login.
- **Change console administrator password** to change the console administrator's password.
- **Manage console servers** to change which Directory Servers can be administered by the Web administration console.
- **Manage console properties** to change the properties of the Web administration console.

Set up Web administration for the first time

Do the following to set up the Directory Server Web Administration Tool for the first time.

1. Install IBM WebSphere Application Server - Express (5722-IWE Base and Option 2) and associated prerequisite software if they are not already installed. See the IBM HTTP Server topic for more information.
2. Enable the system application server instance in the HTTP ADMIN server instance.
 - a. Start the HTTP ADMIN server instance by doing one of the following.
 - In iSeries Navigator, click **Network** -> **Servers** -> **TCP/IP** and right-click **HTTP Administration**. Then click **Start**.
 - On an i5/OS command line type `STRTCPSVR SERVER(*HTTP) HTTPSVR(*ADMIN)`.
 - b. Log on to the IBM Web Administration for iSeries. Use an i5/OS user profile and password to logon to the iSeries Tasks page (http://your_server:2001), then click **IBM Web Administration for iSeries**.
 - c. From the HTTP Server Administration *your_server* page, click the **Manage** tab and then click the **HTTP Servers** tab. Make sure **ADMIN – Apache** is selected in the **Server** drop-down list and that **Include /QIBM/UserData/HTTPA/admin/conf/admin-cust.conf** is selected in the **Server Area** drop-down list.
 - d. From the options in the left pane of the page, click **General Server Configuration**.

Note: You might need to expand the section **Server Properties** in order to see the **General Server Configuration** option.

- e. Set **Start the system application server instance when the 'Admin' server is started** to **Yes**.
- f. Click **OK**.
- g. Restart the HTTP ADMIN server instance by clicking on the restart button (the second button under the **HTTP Servers** tab). You can also stop and start the HTTP ADMIN server instance using the iSeries Navigator or an i5/OS[®] command line.

You can stop the HTTP ADMIN server instance by doing one of the following.

- In iSeries Navigator click **Network** -> **Servers** -> **TCP/IP** and right-click **HTTP Administration**. Then click **Stop**.
- On an OS/400 command line type `ENDTCPSVR SERVER(*HTTP) HTTPSVR(*ADMIN)`.

You can start the HTTP ADMIN server instance by doing one of the following.

- In iSeries Navigator click **Network** -> **Servers** -> **TCP/IP** and right-click **HTTP Administration**. Then click **Start**.
- On an i5/OS command line type `STRTCPSVR SERVER(*HTTP) HTTPSVR(*ADMIN)`.

See the IBM HTTP Server topic for more information.

3. Log on to the Directory Server Web Administration Tool.
 - a. Bring up the **Login page** by doing one of the following.
 - From iSeries Navigator, select your server and click **Network** -> **Servers** -> **TCP/IP**, right click **IBM Directory Server**, and click **Server Administration**.
 - From the iSeries Tasks page (http://your_server:2001) click **IBM Directory Server for iSeries**.
 - b. Select **Console Admin** in the **LDAP Hostname** field.
 - c. Type superadmin in the **Username** field.

- d. Type **secret** in the **Password** field.
 - e. Click **Login**. The IBM Directory Server Web Administration Tool page is displayed.
4. Change the console administration login.
 - a. Click **Console administration** in the left pane to expand the section, and then click **Change console administrator login**.
 - b. Type a new console administration login name in the **Console administrator login** field.
 - c. Type the current password (secret) in the **Current password** field.
 - d. Click **OK**.
 5. Change the console administration password. Click **Change console administrator password** in the left pane.
 6. Add the Directory Server that you want to administer. Click **Manage console servers** in the left pane.

Note: When adding an i5/OS Directory Server, the **Administration port** is not used and will be ignored.
 7. If you want to change console properties. Click **Manage console properties** in the left pane.
 8. Click **Logout**. When the Logout successful screen appears, click the **here** link to return to the Web administration login page.

After you have configured the console for the first time, you can return to the console at any time to:

- Change the console administrator login and password.
- Change which Directory Servers that can be administered by the Web administration tool.
- Change console properties.

Web administration tool

Once you have logged on to the Web administration tool, you will find an application window consisting of five parts:

Banner area

The banner area is located at the top of the panel and contains the application name and IBM logo.

Navigation area

The navigation area, located on the left side of the panel, displays expandable categories for various server content tasks such as:

User properties

This task allows you to change the current user's password.

Schema management

This task allows you to work with object classes, attributes, matching rules, and syntaxes.

Directory management

This task allows you to work with directory entries.

Replication management

This task allows you to work with credentials, topology, schedules, and queues.

Realms and templates

This task allows you to work with user templates and realms.

Users and groups

This task allows you to work with users and groups in the defined realms. For example, if you want to create a new Web user, the **Users and groups** task works with a single group objectclass, `groupOfNames`. You cannot tailor the group support.

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. The icon on the left-hand side of the server status area indicates the current status of the server. Next to the icon is the name of the server being administered. The icon on the right-hand side of the server status area provides a link to the online help.

Task status area

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

Chapter 6. Scenario: MyCo, Inc. sets up a Directory Server

Situation

As the administrator of your company's computer systems, you would like to place employee information such as telephone numbers and e-mail addresses for your organization into a central LDAP repository.

Objectives

In this scenario, MyCo, Inc. wants to configure a Directory Server and create a directory database that contains employee information such as name, e-mail address, and telephone number.

The objectives of this scenario are as follows:

- To make employee information available anywhere on the company network to employees using a Lotus Notes[®] or Microsoft Outlook Express mail client.
- To allow managers to change employee data in the directory database, while not allowing non-managers to change employee data.
- To allow the iSeries server to be able to publish employee data into the directory database.

Details

The Directory Server will run on the iSeries server called myiSeries.

The following example illustrates the information that MyCo, Inc. wants to include into its directory database for each employee.

Name: Jose Alvarez
Department: DEPTA
Telephone number: 999 999 9999
Email address: jalvarez@my_co.com

The directory structure for this scenario might be visualized as something similar to the following:

```
/
|
+- my_co.com
  |
  +- employees
    |
    +- Jose Alvarez
      |
      DEPTA
      999-555-1234
      jalvarez@my_co.com
    +- John Smith
      |
      DEPTA
      999-555-1235
      jsmith@my_co.com
    + Managers group
      Jose Alvarez
      myiSeries.my_co.com
  .
  .
  .
```

All employees (managers and non-managers) exist in the employees directory tree. Managers also belong to the managers group. Members of the managers group have authority to change employee data.

The iSeries server (myiSeries) also needs to have authority to change employee data. In this scenario, the iSeries server is placed in the employees directory tree and is made a member of the managers group.

If you want to keep the employee entries separate from the iSeries server entry, you can create another directory tree (for example: computers) and add the iSeries server there. The iSeries server will need to have the same authority as the managers.

Prerequisites and assumptions

The Web Administration tool is properly configured and running. See “Web administration” on page 86 for more information.

Setup steps

Complete the following tasks:

1. “Scenario details: Set up the Directory Server.”
2. “Scenario details: Create the directory database” on page 93.
3. “Scenario details: Publish the iSeries data to the directory database” on page 95.
4. “Scenario details: Enter information into the directory database” on page 96.
5. “Scenario details: Test the directory database” on page 97.

Scenario details: Set up the Directory Server

Step 1: Configure the Directory Server

Note: You must have *ALLOBJ and *IOSYSCFG special authorities to configure the server.

1. In iSeries Navigator click **Network** → **Servers** → **TCP/IP**.
2. Click **Configure system as Directory server** in the **Server Configuration tasks** window at the bottom right of iSeries Navigator.
3. The **Directory Server Configuration Wizard** appears.
4. Click **Configure a local LDAP directory server** on the **IBM Directory Server Configuration Wizard - Welcome** window.
5. Click **Next** on the **IBM Directory Server Configuration Wizard - Welcome** window.
6. Select **No** on the **IBM Directory Server Configuration Wizard - Specify Settings** window. This allows you to configure the LDAP server without the default settings.
7. Click **Next** on the **IBM Directory Server Configuration Wizard - Specify Settings** window.
8. Uncheck **System-generated** on the **IBM Directory Server Configuration Wizard - Specify Administrator DN** window and enter the following:

Administrator DN	cn=administrator
Password	secret
Confirm password	secret

Note: Any and all passwords specified in this scenario are for example purposes only. To prevent a compromise to your system or network security, you should never use these passwords as part of your own configuration.

9. Click **Next** on the **IBM Directory Server Configuration Wizard - Specify Administrator DN** window.

10. Type `dc=my_co,dc=com` in the **Suffix** field on the **IBM Directory Server Configuration Wizard - Specify Suffixes** window.
11. Click **Add** on the **IBM Directory Server Configuration Wizard - Specify Suffixes** window.
12. Click **Next** on the **IBM Directory Server Configuration Wizard - Specify Suffixes** window.
13. Select **Yes, use all IP addresses** on the **IBM Directory Server Configuration Wizard - Select IP Addresses** window.
14. Click **Next** on the **IBM Directory Server Configuration Wizard - Select IP Addresses** window.
15. Select **Yes** on the **IBM Directory Server Configuration Wizard - Specify TCP/IP Preference** window.
16. Click **Next** on the **IBM Directory Server Configuration Wizard - Specify TCP/IP Preference** window.
17. Click **Finish** on the **IBM Directory Server Configuration Wizard - Summary** window.
18. Right-click on **IBM Directory Server** and click **Start**.

Step 2: Configure the Directory server Web Administration tool

1. Point your browser to `http://myiSeries.my_co.com:9080/IDSWebApp/IDSjsp/Login.jsp`, where `myiSeries.my_co.com` is your iSeries server.
2. A login page should appear. Click the **LDAP Hostname** list and select **Console Admin**. Type `superadmin` for the username and `secret` for the password. Click **Logon**.
3. Configure the Web Administration tool to connect to the LDAP server on your iSeries. Select **Console administration** → **Manage console servers** in the left hand navigation.
4. Click **Add**.
5. In the **Add server** field, type `myiSeries.my_co.com`.
6. Click **Ok**. The new server appears in the list under **Manage console servers**.
7. Click **logout** in the left hand navigation.
8. At the login page of the Web administration tool click the **LDAP Hostname** list and select the server you just configured (`myiSeries.my_co.com`).
9. In the **Username** field type `cn=administrator`, and in the **Password** field type `secret`. Click **Login**. You should see the main page of the IBM Directory Server Web Administration tool.

Scenario details: Create the directory database

Before you can begin to enter data, you must create a place for the data to be stored.

Step 1: Create a base DN object

1. Click **Directory management** → **Manage entries**. You see a listing of the objects in the base level of the directory. Since the server is new, you see only the structural objects which contain the configuration information.
2. You want to add a new object to contain the MyCo, Inc. data. First click **Add...** on the right side of the window. In the next window, scroll within the **Object class** list to select **domain** and click **Next**.
3. You do not want to add any auxiliary object classes, so click **Next** again.
4. In the **Enter the attributes** window, enter the data that corresponds with the suffix that you created earlier in the wizard. Leave the **Object class** drop down list on **domain**. Type `dc=my_co` in the **Relative DN** field. Type `dc=com` in the **Parent DN** field. Type `my_co` in the **dc** field.
5. Click **Finish** at the bottom of the window. Back in the base level you should see the new base DN.

Step:2 Create a user template

You will create a user template as an aid to adding the MyCo, Inc. employee data.

1. Click **Realms and templates** → **Add user template**.
2. In the **User template name** field, type `Employee`.

3. Click on the **Browse...** button next to the **Parent DN** field. Click the base DN you created in the previous section, **dc=my_co,dc=com**, and click **Select**, on the right of the window.
4. Click **Next**.
5. In the **Structural object class** drop-down
6. list, choose **inetOrgPerson** and click **Next**.
7. In the **Naming attribute** drop-down list, select **cn**.
8. In the **Tabs** list, select **Required** and click **Edit**.
9. The **Edit tab** window is where you choose which fields to include in the user template. **sn** and **cn** are required.
10. In the **Attributes** list, select **departmentNumber** and click **Add >>>**.
11. Select **telephoneNumber** and click **Add >>>**.
12. Select **mail** and click **Add >>>**.
13. Select **userPassword** and click **Add >>>**.
14. Click **OK** and then **Finish** to create the user template.

Step:3 Create a realm

1. In the Web Administration tool, click **Realms and templates** —> **Add realm**.
2. In the **Realm name** field, type **employees**.
3. Click **Browse...** to the right of the **Parent DN** field.
4. Select the parent DN you created, **dc=my_co,dc=com**, and click **Select** on the right side of the window.
5. Click **Next**.
6. In the next window you only need to change the **User template** drop-down list. Select the user template you created, **cn=employees,dc=my_co,dc=com**.
7. Click **Finish**.

Step:4 Create a manager group

1. Create the manager group.
 - a. Click **Users and groups** —> **Add group**.
 - b. In the **Group name** field, type **managers**.
 - c. Ensure that **employees** is selected in the **Realm** pull down list.
 - d. Click **Finish**.
2. Configure the manager group administrator for the **employees** realm.
 - a. Click **Realms and templates** —> **Manage realms**.
 - b. Select the realm that you created, **cn=employees,dc=my_co,dc=com**, and click **Edit**.
 - c. To the right of the **Administrator group** field, click **Browse...**
 - d. Select **dc=my_co,dc=com** and click **Expand**.
 - e. Select **cn=employees** and click **Expand**.
 - f. Select **cn=managers** and click **Select**.
 - g. In the **Edit realm** window, click **OK**.
3. Give the manager group authority over the **dc=my_co,dc=com** suffix.
 - a. Click **Directory management** —> **Manage entries**.
 - b. Select **dc=my_co,dc=com** and click **Edit ACL...**
 - c. In the **Edit ACL** window, click the **Owners** tab.
 - d. Select the **Propagate owner** check box. Everyone who is a member of the managers group will be made an owner of the **dc=my_co,dc=com** data tree.
 - e. In the **Type** pull down list, select **Group**.

- f. In the **DN (Distinguished name)** field, type `cn=managers,cn=employees,dc=my_co,dc=com`.
- g. Click **Add**.
- h. Click **Ok**.

Step:5 Add a user as a manager

1. In the Web Administration tool, click **Users and groups** —> **Add user**.
2. Select the realm you created, **employees**, in the **Realm** drop-down menu, and click **Next**.
3. In the **cn** field, type Jose Alvarez.
4. In the ***sn** (surname) field type Alvarez.
5. In the ***cn** (complete name) field, type Jose Alvarez. **cn** is used to create the entry's DN. ***cn** is an attribute of the object.
6. In the **telephoneNumber** field type 999 555 1234.
7. In the **departmentNumber** field type DEPTA.
8. In the **mail** field type jalvarez@my_co.com.
9. In the **userPassword** field type secret.
10. Click the **User groups** tab.
11. In the **Available groups** list, select **managers** and click **Add** —>.
12. At the bottom of the window, click **Finish**.
13. Log out of the Web administration tool by clicking **Log out** in the left hand navigation.

Scenario details: Publish the iSeries data to the directory database

Configure publishing to allow your iSeries server to automatically enter user information into the LDAP directory. User information from the system distribution directory is published into the LDAP directory.

Note: Users created with iSeries Navigator are given both a user profile and an system distribution directory user entry. If you use CL commands to create users, you must create both a user profile (**CRTUSRPRF**) and an system distribution directory user entry (**WRKDIRE**). If your users exist only as user profiles and you want them to be published to the LDAP directory, you must create system distribution directory user entries for them.

Step:1 Create the iSeries server as a Directory Server user

1. Login to the Web Administration tool (http://myiSeries.my_co.com:9080/IDSWebApp/IDSjsp/Login.jsp) as the administrator.
 - a. Select **myiSeries.my_co.com** in the **LDAP Hostname** list.
 - b. Type `cn=admin` in the **Username** field
 - c. Type `secret` in the **Password** field.
 - d. Click **Login**.
2. Select **Users and groups** —> **Add user**.
3. Select **employees** in the **Realm** list.
4. Click **Next**.
5. Type `myiSeries.my_co.com` in the **cn** field.
6. Type `myiSeries.my_co.com` in the ***sn** field.
7. Type `myiSeries.my_co.com` in the ***cn** field.
8. Type `secret` in the **userPassword** field.
9. Click the **User groups** tab.
10. Select the group **managers**.
11. Click **Add** —>.

12. Click **Finish**.

Step:2 Configure the iSeries server to publish data

1. In iSeries Navigator, right-click on your iSeries in the left hand navigation and select **Properties**.
2. In the **Properties** dialog box, choose the **Directory Server** tab.
3. Select **Users** and click **Details**.
4. Select the **Publish user information** check box.
5. In the **Where to publish** section, click the **Edit** button. A window appears.
6. Type `myiSeries.my_co.com`.
7. In the **Under DN** field, type `cn=employees,dc=my_co,dc=com`.
8. In the **Server connection** section, ensure that the default port number, **389**, is entered in the **Port** field. In the **Authentication method** drop-down list, choose **Distinguished name** and enter `cn=myiSeries,cn=employees,dc=my_co,dc=com` in the **Distinguished name** field.
9. Click **Password**.
10. Type `secret` in the **Password** field.
11. Type `secret` in the **Confirm Password** field.
12. Click **OK**.
13. Click the **Verify** button. This ensures that you have entered all the information correctly and that the iSeries can connect to the LDAP directory.
14. Click **OK**.
15. Click **OK**.

Scenario details: Enter information into the directory database

As the manager, Jose Alvarez now adds and updates data for individuals in his department. He needs to add some additional information about Jane Doe. Jane Doe is a user on the iSeries server and her information was published. Jose Alvarez also needs to add information about John Smith. John Smith is not a user on the iSeries server. Jose Alvarez does the following:

Step 1: Login to the Web Administration tool

Log into the Web Administration tool. (http://myiSeries.my_co.com:9080/IDSWebApp/IDSjsp/Login.) by doing the following:

1. Select `myiSeries.my_co.com`, in the **LDAP Hostname** list.
2. Type `cn=Jose Alvarez,cn=myco employees,dc=my_co,dc=com` in the **Username** field.
3. Type `secret` in the **password** field.
4. Click **Logon**.

Step 2: Modify employee data

1. Click **Users and groups** —> **Manage users**.
2. Select **employees** in the **Realm** list and click **View users**.
3. Select **Jane Doe** in the users list and click **Edit**.
4. Type `DEPTA` in the **departmentNumber** field.
5. Click **OK**.
6. Click **Close**.

Step 3: Add employee data

1. Click **Users and groups** —> **Add user**.
2. Select **employees** in the **Realm** pull down menu and click **Next**.

3. In the **cn** field, type John Smith.
4. In the ***sn** field type Smith.
5. In the ***cn** field, type John Smith.
6. In the **telephoneNumber** field type 999 555 1235.
7. In the **departmentNumber** field type DEPTA.
8. In the **mail** field type jsmith@my_co.com.
9. Click **Finish** at the bottom of the window.

Scenario details: Test the directory database

After you have entered the employee data into the directory database, test the directory database and Directory Server by doing one of the following:

Search the directory database using your e-mail address book

Information in an LDAP directory can be easily searched by LDAP enabled programs. Many e-mail clients can search LDAP directory servers as part of their address book function. The following are example procedures to configure Lotus Notes 6 and Microsoft Outlook Express 6. The procedure for most other e-mail clients will be similar.

Lotus Notes

1. Open your address book.
2. Click **Actions** → **New** → **Account**.
3. Type myiSeries in the **Account name** field.
4. Type myiSeries.my_co.com in the **Account server name** field.
5. Select **LDAP** in the **Protocol** field.
6. Click the **Protocol Configuration** tab.
7. Type dc=my_co,dc=com in the **Search base** field.
8. Click **Save and close**.
9. Click **Create** → **Mail** → **Memo**.
10. Click **Address...**
11. Select myiSeries in the **Choose address book** field.
12. Type Alvirez in the **Search for** field.
13. Click **Search**. The data for Jose Alvirez appears

Microsoft Outlook Express

1. Click **Tools** → **Accounts**.
2. Click **Add** → **Directory Service**.
3. Type the Web address of the iSeries in the **Internet Directory (LDAP) server** field (myiSeries.my_co.com).
4. Uncheck the **My LDAP server requires me to log on** check box
5. Click **Next**.
6. Click **Next**.
7. Click **Finish**.
8. Select myiSeries.my_co.com (the directory service that you just configured) and click **Properties**.
9. Click **Advanced**.
10. Type dc=my_co,dc=com in the **Search base** field.
11. Click **Ok**.

12. Click **Close**.
13. Type Ctrl+E to open the **Find People** window.
14. Select myiSeries.my_co.com from the **Look in** list.
15. Type Alvirez in the **Name** field.
16. Click **Find now**. The data for Jose Alvirez appears.

Search the directory database using the ldapsearch command line command

1. On the character-based interface enter the CL command **QSH** to open a Qshell session.
2. Enter the following to retrieve a list of all the LDAP entries in the database.

```
ldapsearch -h myiSeries.my_co.com -b dc=my_co,dc=com objectclass=*
```

Where:

-h is the name of the host machine running the LDAP server.

-b is the base DN to search under.

objectclass=*

returns all of the entries in the directory.

This command returns something like the following:

```
dc=my_co,dc=com
dc=my_co
objectclass=domain
objectclass=top
```

```
cn=MyCo employee,dc=my_co,dc=com
```

```
.
.
.
```

```
cn=Jose Alvirez,cn=MyCo Employees,dc=my_co,dc=com
```

```
sn=Alvirez
departmentNumber=DEPTA
mail=jalvirez@my_co.com
telephoneNumber=999 999 9999
objectclass=top
objectclass=inetOrgPerson
objectclass=organizationalPerson
objectclass=person
cn=Jose Alvirez
```

```
.
.
.
```

The first line of each entry is called the distinguished name (DN). DNs are like the complete file name of each entry. Some of the entries do not contain data and are only structural. Those with the line **objectclass=inetOrgPerson** correspond to the entries you created for people. Jose Alvirez's DN is **cn=Jose Alvirez,cn=MyCo Employees,dc=my_co,dc=com**.

Chapter 7. Administer Directory Server

To administer the Directory Server, you must have the following authority sets:

- To configure the server or change the server configuration: All Object (*ALLOBJ) and I/O System Configuration (*IOSYSCFG) special authorities
- To start or stop the server: Job Control (*JOBCTL) authority and object authority to the End TCP/IP (ENDTCP), Start TCP/IP (STRTCP), Start TCP/IP Server (STRTCP SVR), and End TCP/IP Server (ENDTCP SVR) commands
- To set auditing behavior for the directory server: Audit (*AUDIT) special authority
- To view the server job log: Spool Control (*SPLCTL) special authority

To manage directory objects (including access control lists, object ownership, and replicas), connect to the directory with either the administrator DN or another DN that has the proper LDAP authority. If authority integration is being used, an administrator can also be a projected user (see “Operating system projected backend” on page 67) that has authority to the Directory Server Administrator function ID (see “Work with administrative access for authorized users” on page 105).

General administration tasks

- “Start the Directory Server” on page 100
- “Stop the Directory Server” on page 100
- “Check the status of the directory server” on page 101
- “Check jobs on the Directory Server” on page 101
- “Enable event notification” on page 101
- “Specify transaction settings” on page 101
- “Change the port or IP address” on page 102
- “Set password policy” on page 102
- “Import an LDIF file” on page 103
- “Export an LDIF file” on page 103
- “Specify a server for directory referrals” on page 103
- “Add and remove Directory Server suffixes” on page 104
- “Save and restore Directory Server information” on page 104
- “Work with administrative access for authorized users” on page 105
- “Track access and changes to the LDAP directory” on page 105
- “Enable object auditing for the Directory Server” on page 106
- “Adjust search settings” on page 106
- “Adjust performance settings” on page 107
- “Manage replication” on page 107
- “Enable SSL on the Directory Server” on page 123
- “Enable Kerberos authentication on the Directory Server” on page 125
- “Manage the schema” on page 125
- “Enable or disable read access to projected users” on page 136

Directory content tasks

- “Manage directory entries” on page 136
- “Manage users and groups” on page 142

- “Manage realms and user templates” on page 145
- “Manage access control lists (ACLs)” on page 153

Publishing tasks

- “Publish information to the directory server” on page 157

Start the Directory Server

To start the Directory Server, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Start**.

The directory server may take several minutes to start, depending on the speed of your server and the amount of available memory. The first time you start the directory server may take several minutes longer than usual because the server must create new files. Similarly, when starting the directory server for the first time after upgrading from an earlier version of Directory Server, it may take several minutes longer than usual because the server must migrate files. You can check the status of the server periodically (see “Check the status of the directory server” on page 101) to see if it has started yet.

The directory server can also be started from the character-base interface by entering the command `STRTCPSVR *DIRSRV`. Additionally, if you have your directory server configured to start when TCP/IP starts, you can also start it by entering the `STRTCP` command.

Configuration only mode

The directory server can be started in configuration only mode from the character-base interface by entering the command `TRCTCPAPP APP(*DIRSRV) ARGLIST(SAFEMODE)`.

Configuration only mode starts the server with only the `cn=configuration` suffix active and does not depend on successful initialization of the database backends.

Stop the Directory Server

Stopping the directory server affects all applications using the server at the time it is stopped. This includes Enterprise Identity Mapping (EIM) applications that are currently using the directory server for EIM operations. All applications are disconnected from the directory server, however, they are not prevented from attempting to reconnect to the server.

To stop the Directory Server, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Stop**.

The directory server may take several minutes to stop, depending on the speed of your system, the amount of server activity, and the amount of available memory. You can check the status of the server periodically (see “Check the status of the directory server” on page 101) to see if it has started yet.

Note: The directory server can also be stopped from a 5250 session by entering the commands `ENDTCPSVR *DIRSRV`, `ENDTCPSVR *ALL`, or `ENDTCP`. `ENDTCPSVR *ALL` and `ENDTCP` also affect any other TCP/IP servers that run on your system. `ENDTCP` will also end TCP/IP itself.

Check the status of the directory server

iSeries Navigator displays the status of the directory server in the **Status** column in the right frame.

To check the status of the directory server, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**. iSeries Navigator displays the status of all TCP/IP servers, including the directory server, in the **Status** column. To update the status of the servers, click the **View** menu and select **Refresh**.
4. To view more information about the status of the directory server, right-click **Directory** and select **Status**. This will show you the number of active connections, as well as other information such as past and current activity levels.

Besides providing additional information, viewing status through this option can save time. You can refresh the status of the directory server without taking the additional time that is required to check the status of the other TCP/IP servers.

Check jobs on the Directory Server

At times you may want to monitor specific jobs on the Directory Server. To check server jobs, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Server Jobs**.


Enable event notification

Directory Server supports event notification, which allows clients to register with the LDAP server to be notified when a specified event, such as something being added to the directory, occurs.

To enable event notification for your server, follow these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click **Events**.
6. Select **Allow clients to register for event notification**.

You can also specify the maximum registrations allowed for each connection and the maximum total registrations that the server allows.

For additional information about event notification, see the Event notification section of the IBM Directory Server Version 5.1 Programming Reference  .

Specify transaction settings

Directory Server supports transactions, which allows a group of LDAP directory operations to be treated as one unit. For more information, see “Transactions” on page 41.

To configure your servers transaction settings, follow these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.

5. Click **Transactions**.
6. Specify your transaction settings.

Note: Transaction settings can impact your LDAP servers performance, so you may want to experiment with different settings.

Change the port or IP address

The Directory Server uses the following default ports:

- 389 for unsecured connections.
- 636 for secured connections (if you have used Digital Certificate Manager to enable Directory Server as an application that can use a secure port).

Note: By default, all IP addresses defined on the local system are bound to the server.

If you are already using these ports for another application, you can either assign a different port to Directory Server, or you can use different IP addresses for the two servers, if the applications support binding to a specific IP address.

For an example of the Domino LDAP server conflicting with the Directory Server, see Host Domino LDAP and Directory Server on the same iSeries

To change the ports that the Directory Server uses, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click the **Network** tab.
6. Enter the appropriate port numbers, then click **OK**.

To change the IP address on which the directory server accepts connections, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click the **Network** tab.
6. Click the **IP Addresses...** button.
7. Select **Use selected IP addresses** and select the IP addresses for the server to use when accepting connections.

Set password policy

To set the password policy, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click the **Password** tab.
6. Enter the password policy information. Optionally click **Password Validation and Lockout** to specify additional password policy information, then click **OK**.
7. Click **OK**.

Note: You can also use the ldapmodify utility (see “ldapmodify and ldapadd” on page 165) to set password policy.

For more information about password policy, see “Password policy” on page 61.

Import an LDIF file

You can transfer information between different Directory Servers by using LDAP Data Interchange Format (LDIF) files. See “LDAP data interchange format (LDIF)” on page 190 for more information. Before you begin this procedure, transfer the LDIF file to your iSeries server as a stream file.

To import an LDIF file to the Directory Server, take these steps:

1. If the directory server is started, stop it. See “Stop the Directory Server” on page 100 for information about stopping the directory server.
2. In iSeries Navigator, expand **Network**.
3. Expand **Servers**.
4. Click **TCP/IP**.
5. Right-click **Directory** and select **Tools**, then **Import File**.

Optionally you can have the server replicate the newly imported data when it is next started by selecting **Replicate imported data**. This is useful when adding new entries to an existing directory tree on a master server. If you are importing data to initialize a replica (or peer) server, typically you will not want to have the data replicated, as it may already exist on the servers for which this server is a supplier.

Note: You can also use the ldapadd utility (see “ldapmodify and ldapadd” on page 165) to import LDIF files.

Export an LDIF file

You can transfer information between different Directory Servers by using LDAP Data Interchange Format (LDIF) files, see “LDAP data interchange format (LDIF)” on page 190. You can export all or part of your LDAP directory to an LDIF file.

To export an LDIF file from the directory server, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Tools**, then **Export File**.

Note: If you do not specify a fully qualified path for the LDIF file to export data into, the file will be created in the home directory specified in your i5/OS user profile.

Notes:

1. Be sure to set authority to the LDIF file to prevent unauthorized access to directory data. To do this, right-click on the file in iSeries Navigator, then select **Permissions**.
2. You can also create a full or partial LDIF file with the ldapsearch utility, see “ldapsearch” on page 177. Use the -L option and redirect the output to a file.

Specify a server for directory referrals

To assign referral servers for the directory server, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory**, then select **Properties**.

5. Select the **General** properties page.
6. In the **New referral** field, specify the URL of the referral server.
7. At the prompt, specify the name of the referral server in URL format. The following are examples of acceptable LDAP URLs:
 - ldap://test.server.com
 - ldap://test.server.com:400
 - ldap://9.9.99.255

Note: If the referral server does not use the default port, specify the correct port number as part of the URL, as port 400 is specified in the second example above.

8. Click **Add**.
9. Click **OK**.

Add and remove Directory Server suffixes

Adding a suffix to the Directory Server allows the server to manage that part of the directory tree.

Note: You cannot add a suffix that is under another suffix already on the server. For example, if `o=ibm, c=us` were a suffix on your server, you cannot add `ou=rochester, o=ibm, c=us`.

To add a suffix to the directory server, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click the **Database/Suffixes** tab.
6. In the **New suffix** field, type the name of the new suffix.
7. Click **Add**.
8. Click **OK**.

Note: Adding a suffix points the server to a section of the directory, but does not create any objects. If an object that corresponds to the new suffix did not previously exist, you must create it just as you would any other object.

To remove a suffix from the Directory Server, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click the **Database/Suffixes** tab.
6. Click on the suffix that you want to remove to select it.
7. Click **Remove**.

Note: You can choose to delete a suffix without deleting the directory objects under it. This makes the data inaccessible from the directory server. However, you can later regain access to the data by adding back the suffix.

Save and restore Directory Server information

Directory Server stores information in the following locations:

- The database library (QUSRDIRDB by default), which contains the directory servers contents.
- The QDIRSRV2 library, which is used to store publishing information.
- The QUSRSYS library, which stores various items in objects beginning with QGLD (specify QUSRSYS/QGLD* to save them).

- If you configure the directory server to log directory changes, a database library called QUSRDIRCL that the change log uses.

If the contents of the directory change regularly, you should save your database library and the objects in it on a regular basis. Configuration data is also stored in the following directory:

/QIBM/UserData/OS400/Dirsrv/

You should also save the files in that directory whenever you change the configuration or apply PTFs.

See Backup and Recovery, SC41-5304  for information about saving and restoring OS/400 data.

Work with administrative access for authorized users

You can grant administrator access to user profiles that have been given access to the Directory Server Administrator (QIBM_DIRSrv_ADMIN) function identifier (ID).

For example, if the user profile JOHNSMITH is granted access to the Directory Server Administrator function ID and the Grant administrator access to authorized users option is selected from the Directory property dialog, the JOHNSMITH profile then has LDAP administrator authority. When this profile is used to bind to the directory server using the following DN, `os400-profile=JOHNSMITH,cn=accounts,os400-sys=systemA.acme.com`, the user has administrator authority. The system objects' suffix in this example is `os400-sys=systemA.acme.com`. For more information about projected users, see "Operating system projected backend" on page 67.

To select this option, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Right-click **Directory** and select **Properties**.
4. On the **General** tab under **Administrator information**, select the **Grant administrator access to authorized users** option.

To set the Directory Server Administrator authority function ID in a user profile, take these steps:

1. In iSeries Navigator, right-click the system name and select **Application Administration**.
2. Click the **Host Applications** tab.
3. Expand **Operating System/400**.
4. Click **Directory Server Administrator** to highlight the option.
5. Click the **Customize** button.
6. Expand **Users, Groups, or Uses not in a group**, whichever is appropriate for the user you want.
7. Select a user or group to be added to the **Access allowed** list.
8. Click the **Add** button.
9. Click **OK** to save the changes.
10. Click **OK** on the **Application Administration** dialog.

Track access and changes to the LDAP directory

You may want to track access and changes to your LDAP directory. You can use the LDAP directories change log to keep track of changes to the directory. The change log is located under the special suffix `cn=changeLog`. It is stored in the QUSRDIRCL library.

To enable the change log, follow these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.

3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click the **Change Log** tab.
6. Select **Log directory changes**.
7. (optional) In the **Maximum entries** field specify the maximum number of entries for the change log to keep. In the **Maximum age** field specify how long change log entries are retained.

Note: Though these parameters are optional, you should strongly consider specifying either a maximum number of entries or maximum age. If you do not specify either, the change log will keep all entries and may become very large.

The `changeLogEntry` object class is used to represent the changes applied to the directory server. The set of changes is given by the ordered set of all entries within the `changelog` container as defined by `changeNumber`. The change log information is read-only.

Any user who is on the Access Control List for the `cn=changelog` suffix can search on the entries in the change log. You should only execute searches on the change log suffix, `cn=changelog`. Do not attempt to add, change, or delete to the change log suffix, even if you have authority to do so. This will cause unpredictable results.

Example:

The following example uses the `ldapsearch` command line utility to retrieve all change log entries logged on the server:

```
ldapsearch -h ldaphost -D cn=admininistrator -w password -b cn=changelog (changetype=*)
```

Enable object auditing for the Directory Server

Directory Server supports OS/400 security auditing. If the `QAUDCTL` system value has `*OBJAUD` specified, you can enable object auditing through iSeries Navigator.

To enable object auditing for Directory Server, follow these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click the **Auditing** tab.
6. Select the auditing setting that you want to use for your server.

Changes to auditing settings will take effect as soon as you click **OK**. There is no need to restart the Directory Server. For more information, see “Directory Server security” on page 41

Adjust search settings

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 cancelled 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.

To adjust the search values of the directory server, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.

3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click the **Search** tab.

Adjust performance settings

You can adjust the performance settings of your Directory Server by changing any of the following:

- The ACL cache size, the entry cache size, the maximum number of searches to store in the filter cache, and the largest search to cache in the filter cache.
- The servers transaction settings
- The number of database connections and server threads

To adjust cache values of the directory server, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click the **Performance** tab.

To adjust transaction values of the directory server, take these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click the **Transactions** tab.

You can also adjust the performance of the directory server by changing the number of database connections and server threads that the server uses. To change this value, follow these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click the **Database/Suffixes** tab.

Manage replication

To manage replication, expand the **Replication management** category of the Web administration tool. For more information about replication concepts, see “Replication” on page 35.

See the following for more information:

- “Create a master-replica topology”
- “Create a master-forwarder-replica topology” on page 113
- “Overview of creating a complex replication topology” on page 114
- “Create a complex topology with peer replication” on page 115
- “Manage topologies” on page 117
- “Modify replication properties” on page 120
- “Create replication schedules” on page 121
- “Manage queues” on page 123

Create a master-replica topology

To define a basic master-replica 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. See “Create a master server (replicated subtree)” on page 108.

2. Create credentials to be used by the supplier. See “Create credentials” on page 109.
3. Create a replica server. See “Create a replica server” on page 110.
4. Export the topology from the master to the replica. See “Copy data to the replica” on page 111.
5. Change the replica’s configuration to identify who is authorized to replicate changes to it, and add a referral to a master. See “Add the supplier information to the replica” on page 112.

Note:

If the entry at the root of the subtree that you want to be replicated is not a suffix in the server, before you can use the **Add subtree** function, you must ensure that its ACLs are defined as follows:

For non-filtered ACLs:

```
ownertime: <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-filtered ACLs, select that tab and select the checkbox to specify if the ACLs are explicit or not for both ACLs and owners. Ensure that **Propagate ACLs** and **Propagate owner** are checked. If you want to add Filtered 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 “Manage access control lists (ACLs)” on page 153 for more detailed information.

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.

Create a master server (replicated subtree)

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**.

1. Click **Add subtree**.
2. Enter the DN of the root entry 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.
3. The master server referral URL is displayed 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.

4. Click **OK**.
5. The new server is displayed on the Manage topology panel under the heading **Replicated subtrees**.

Create 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 these locations:

- **cn=replication,cn=localhost**, which keeps the credentials only on the current server.

Note: In most replication cases, locating credentials in **cn=replication,cn=localhost** is preferred because it provides greater security than replicated credentials located on the subtree. However, there are certain situations in which credentials located on **cn=replication,cn=localhost** are not available.

If you are trying to add a replica under a server, for example **serverA** and you are connected to a different server with the Web administration tool, **serverB**, the **Select credentials** field does not display the option **cn=replication,cn=localhost**. This is because you cannot read the information or update any information under **cn=localhost** of the **serverA** when you are connected to **serverB**.

The **cn=replication,cn=localhost** option is only available when the server under which you are trying to add a replica is the same server that you are connected to with the Web administration tool.

- Within the replicated subtree, in which case the credentials are replicated with the rest of the subtree. 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 “Create a master server (replicated subtree)” on page 108 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**, **cn=** is prefilled in the field for you.

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 the server 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. Enter your Kerberos bind DN.
 - b. Enter the bind password.
 - c. Reenter the bind password to confirm it.
 - d. If you want, enter a brief description of the credentials. No other information is necessary. See “Enable Kerberos authentication on the Directory Server” on page 125 for additional information.
 - e. 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/master.our.org.com@SOME.REALM. The realm value is case insensitive. 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. Reenter the key file password to confirm it.
 - d. Enter the key label.
 - e. If you want, enter a brief description.
 - f. Click **Finish**.

See "Enable SSL on the Directory Server" on page 123 for additional information.

5. On the server where you created the credentials, set the Allow server security information to be retained (QRETSVRSEC) system value to 1 (retain data). Since the replication credentials are stored in a validation list, this allows the server to retrieve the credentials from the validation list when it connects to the replica.

Create 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 that all servers be at the same release.
- 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 these 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. Credentials placed in the replicated subtree are created beneath the **ibm-replicagroup=default** entry for that subtree.

Placing credentials in **cn=replication,cn=localhost** is considered more secure.

- 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 “Create credentials” on page 109 for additional information on agreement credentials.

2. Specify a replication schedule from the drop-down list or click **Add** to create one. See “Create replication schedules” on page 121
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.
5. A message is displayed noting that additional actions must be taken. Click **OK**.

Note: If you are adding more servers as additional replicas or are creating a complex topology, do not proceed with “Copy data to the replica” or “Add the supplier information to the replica” on page 112 until you have finished defining the topology on the master server. If you create the *masterfile.ldif* after you have completed the topology, it contains the directory entries of the master server and a complete copy of the topology agreements. When you load this file on each of the servers, each server then has the same information.

Copy data to the replica

After creating the replica, you must now export the topology from the master to the replica.

1. On the master server create an LDIF file for the data. To copy all the data contained on the master server, do the following:
 - a. In iSeries Navigator, expand **Network**.
 - b. Expand **Servers**.
 - c. Click **TCP/IP**.
 - d. Right-click **Directory** and select **Tools**, then **Export File**.
 - e. Specify the output LDIF file name (for example `masterfile.ldif`), optionally specify a subtree to export (for example `subtreeDN`), and click **OK**.
2. On the machine where you are creating the replica, do the following:
 - a. Ensure that the replicated suffixes are defined in the replica server's configuration.
 - b. Stop the replica server.
 - c. Copy the LDIF file the replica and do the following:
 - 1) In iSeries Navigator, expand **Network**.
 - 2) Expand **Servers**.
 - 3) Click **TCP/IP**.
 - 4) Right-click **Directory** and select **Tools**, then **Import File**.
 - 5) Specify the input LDIF file name (for example `masterfile.ldif`), optionally specify if you want to replicate data, and click **OK**.

The replication agreements, schedules, credentials (if stored in the replicated subtree) and entry data are loaded on the replica.

- d. Start the server.

Add 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. Expand **Replication management** in the navigation area and click **Manage replication properties**.
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 `ibm-kn=ldap/<yourservername@yourrealm>`. If the credentials has a principal name such as `<myprincipal@myrealm>`, use that as the DN. In either case a password is not needed.
 - **SSL w/ EXTERNAL bind** - Specify the subject DN for the certificate and no password

See "Create credentials" on page 109.
 6. Click **OK**.

7. You must restart the replica for the changes to take effect.

See “Modify replication properties” on page 120 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 “Manage queues” on page 123 for more detailed information. The replica now receives updates from the master.

Create a master-forwarder-replica topology

To define a master-forwarder-replica topology, you must:

1. Created a master server and a replica server. See “Create a master-replica topology” on page 107.
2. Create a new replica server for the original replica. See “Create a new replica server.”
3. Copy data to the replicas. See “Copy data to the replica” on page 111.

Create a new replica server

If you have set up a replication topology (see “Create a master server (replicated subtree)” on page 108) 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 that all servers be at the same release.
 - 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 “Create credentials” on page 109 for additional information on agreement credentials.

- b. Specify a replication schedule from the drop-down list or click **Add** to create one. See “Create replication schedules” on page 121.
- 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 “Copy data to the replica” on page 111 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 “Add the supplier information to the replica” on page 112 for information on how to do this.

The server roles are represented by icons in the Web administration tool. Your topology is now:

- server1 (master)
 - server2 (forwarder)
 - server3 (replica)

Overview of creating a complex replication topology

Use this high level overview as a guide for setting up a complex replication topology.

1. Start all peer server or replicas-to-be. This is required for the Web administration tool to gather information from the servers.
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. This prevents updates from being made while copying data to the other servers.
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. Stop the replica servers and import the data for the replicated subtree on to each replica and peer master. Then restart the servers.
15. Manage the replication properties on each replica and peer master to set the credentials to be used by suppliers.

Create a complex topology with 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.

To add additional masters (peers), you first add the server as a read-only replica of the existing masters (see “Create a replica server” on page 110), initialize the directory data, and then promote the server to be a master (see “Move or promote a server” on page 118).

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 of the Web administration tool to set ACLs for the replication information associated with the newly created replicated subtree (see “Edit access control lists” on page 120).

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 “Manage queues” on page 123 for more detailed information. The replica now receives updates from the master.

Use peer replication only in environments where the pattern of directory updates is 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.

To define a peer-forwarder-replica topology, consisting of two peer-master servers, two forwarding servers, and four replicas you must:

1. Created a master server and a replica server. See “Create a master-replica topology” on page 107.
2. Create two additional replica servers for the master server. See “Create a replica server” on page 110.
3. Create two replicas under each of the two newly created replica servers.

- Promote the original replica to a master. See “Promote a server to be a peer.”

Note: The server that you want to promote to a master must be a leaf replica with no subordinate replicas.

- Copy the data from the master to the new master and replicas. See “Copy data to the replica” on page 111.

Promote a server to be a peer

Using the forwarding topology created in “Create a master-forwarder-replica topology” on page 113, 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).

- Connect Web Administration to the master (server1).
- Expand the Replication management category in the navigation area and click **Manage topology**.
- Select the subtree that you want to replicate and click **Show topology**.
- Click the arrow next to the **Replication topology** selection to expand the list of servers.
- Click the arrow next to the **server1** selection to expand the list of servers.
- Click the arrow next to the **server2** selection to expand the list of servers.
- Click **server1** and click **Add replica**. Create server4. See “Create a replica server” on page 110. Follow the same procedure to create server5. The server roles are represented by icons in the Web administration tool. Your topology is now:

- server1 (master)
 - server2 (forwarder)
 - server3 (replica)
 - server4 (replica)
 - server5 (replica)

- Click **server2** and click **Add replica** to create server6.
- Click **server4** and click **Add replica** to create server7. Follow the same procedure to create server8. Your topology is now:

- server1 (master)
 - server2 (forwarder)
 - server3 (replica)
 - server6 (replica)
 - server4 (forwarder)
 - server7 (replica)
 - server8 (replica)
 - server5 (replica)

- Select **server5** and click **Move**.

Note: The server you want to move must be a leaf replica with no subordinate replicas.

- Select **Replication topology** to promote the replica to a master. Click **Move**.
- The **Create additional supplier agreements panel** is displayed. Peer replication requires each master to be a supplier and consumer to each of the other masters in the topology and to each of the first level replicas, server2 and server4. Server5 already is a consumer of server1, it now needs to become a supplier to server1, server2, and server4. Ensure that the supplier agreement boxes are checked for:

Table 3.

	Supplier	Consumer
✓	server5	server1

Table 3. (continued)

	Supplier	Consumer
✓	server5	server2
✓	server5	server4

Click **Continue**.

Note: 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 from the existing sets of credentials or create new credentials. See “Create credentials” on page 109

13. Click **OK**. Your topology is now:

- server1 (master)
 - server2 (forwarder)
 - server3 (replica)
 - server6 (replica)
 - server4 (forwarder)
 - server7 (replica)
 - server8 (replica)
 - server5 (master)
- server5 (master)
 - server1 (master)
 - server2 (forwarder)
 - server4 (forwarder)

14. Copy data from server1 to the all the servers. See “Copy data to the replica” on page 111 for information on how to do that.

Manage topologies

Topologies are specific to the replicated subtrees.

- “View the topology”
- “Add a replica” on page 118
- “Edit an agreement” on page 118
- “Move or promote a server” on page 118
- “Demote a master” on page 118
- “Replicate a subtree” on page 119
- “Edit a subtree” on page 119
- “Remove a subtree” on page 119
- “Quiesce the subtree” on page 120
- “Edit access control lists” on page 120

View 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.

Add a replica

See “Create a replica server” on page 110.

Edit an agreement

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 “Create credentials” on page 109.
- Replication schedules - see “Create replication schedules” on page 121.
- 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**.

Move or promote a server

1. Select the server that you want 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 from the existing sets of credentials or create new credentials. See “Create credentials” on page 109.
4. The **Create additional supplier agreements** is displayed. Select the supplier agreements appropriate for the role of the server. For example, if a replica server is being promoted to be a peer server, you must select to create supplier agreements with all the other servers and their first level replicas. These agreements enable the promoted server to act as a supplier to the other servers and their replicas. Existing supplier agreements from the other servers to the newly promoted server are still in effect and do not need to be recreated.
5. Click **OK**.

The change in the topology tree reflects the moving of the server.

See “Create a complex topology with peer replication” on page 115 for more information.

Demote 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 “Create a replica server” on page 110 for instructions.

Replicate 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**.

Edit 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.

Remove 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` entry is empty.

Quiesce 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**.

Edit 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 and to work with ACLs, see “Manage access control lists (ACLs)” on page 153.

See “Access control lists” on page 49 for additional information.

Modify replication properties

Expand the **Replication management** category in the navigation area and click **Manage replication properties**. You must log into the Web administration tool as a projected i5/OS user with ***ALLOBJ** and ***IOSYSCFG** special authorities for the Manage replication properties to be shown.

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.

Note: The supplier DN can be the DN of a projected i5/OS user profile. The projected i5/OS user profile must not have LDAP administrative authority. The user cannot be a user with ***ALLOBJ** and ***IOSYSCFG** special authorities and cannot have been granted administrative authority through the directory server administrator application ID.

For more information, see the following:

- “Add supplier information”
- “Edit supplier information” on page 121
- “Remove supplier information” on page 121

Add 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 "Create credentials" on page 109.

5. Click **OK**.

The subtree of the supplier is added to the Supplier information list.

Edit 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**.

Remove 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.

Create 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 “Create a daily schedule” for more information on setting up daily schedules.
5. When you are finished, click **OK**.

Create 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 4.

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.

Manage 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 187 for more information.

Enable SSL on the Directory Server

If you have Digital Certificate Manager installed on your system, you can use Secure Sockets Layer (SSL) security to protect access to your Directory Server. Before enabling SSL on the directory server, you may find it helpful to read "Secure Sockets Layer (SSL) and Transport Layer Security with the Directory Server" on page 42.

To use an SSL connection when you administer your Directory Server from iSeries Navigator, or to use SSL with the Windows LDAP client, you must have one of the Client Encryption products (5722CE2 or 5722CE3) installed on your PC.

To enable SSL on your LDAP server, do the following:

1. **Associate a certificate with the Directory Server**

- a. If you want to manage your Directory Server through an SSL connection from iSeries Navigator, see the iSeries Access for Windows User's Guide (it is optionally installed on your PC when you installed iSeries Navigator). If you are planning to allow both SSL and non-SSL connections to the directory server, you may choose to skip this step.
- b. Start IBM Digital Certificate Manager. See Start Digital Certificate Manager in the Digital Certificate Manager topic for more information.
- c. If you need to obtain or create certificates, or otherwise setup or change your certificate system, do so now. See Digital Certificate Manager for information about setting up a certificate system. There are two server applications and one client application associated with Directory Server. They are:

Directory Server application

The Directory Server application is the server itself.

Directory Server publishing application

The Directory Server publishing application identifies the certificate used by publishing.

Directory Server client application

The Directory Server client application identifies the default certificate used by applications using the LDAP client ILE APIs.

- d. Click the **Select a Certificate Store** button.
- e. Select ***SYSTEM**. Click **Continue**.
- f. Enter the appropriate password for *SYSTEM certificate store. Click **Continue**.
- g. When the left navigational menu reloads, expand **Manage Applications**.
- h. Click **Update certificate assignment**.
- i. On the next screen, select **Server** application. Click **Continue**.
- j. Select the **Directory Server server**.
- k. Click **Update Certificate Assignment** to assign a certificate to the Directory Server to use to establish its identity to iSeries Access for Windows clients.

Note: If you choose a certificate from a CA whose CA certificate is not in your iSeries Access for Windows client's key database, you will need to add it in order to use SSL. Finish this procedure before beginning that one.

- l. Select a certificate from the list to assign to the server.
 - m. Click **Assign New Certificate**.
 - n. DCM reloads to the **Update Certificate Assignment** page with a confirmation message. When you are finished setting up the certificates for the Directory Server, click **Done**.
2. **Associate a certificate for the Directory Server publishing.** (optional step) If you also want to enable publishing from the system to a Directory Server through an SSL connection, you may want to also associate a certificate with the Directory Server publishing. This identifies the default certificate and trusted CAs for applications using the LDAP ILE APIs that do not specify their own application id or an alternate key database.
- a. Start IBM Digital Certificate Manager.
 - b. Click the **Select a Certificate Store** button.
 - c. Select ***SYSTEM**. Click **Continue**.
 - d. Enter the appropriate password for *SYSTEM certificate store. Click **Continue**.
 - e. When the left navigational menu reloads, expand **Manage Applications**.

- f. Click **Update certificate assignment**.
- g. On the next screen, select **Client** application. Click **Continue**.
- h. Select the **Directory Server publishing**.
- i. Click **Update Certificate Assignment** to assign a certificate to the Directory Server publishing that will establish its identity.
- j. Select a certificate from the list to assign to the server.
- k. Click **Assign new certificate**.
- l. DCM reloads to the **Update Certificate Assignment** page with a confirmation message.

Note: These steps assume that you are already publishing information to the Directory Server with a non-SSL connection. See “Publish information to the directory server” on page 157 for complete information about setting up publishing.

3. **Associate a certificate for the Directory Server client.** (optional step) If you have other applications that use SSL connections to a Directory Server, you must also need to associate a certificate with a the Directory Server client.
 - a. Start IBM Digital Certificate Manager.
 - b. Click the **Select a Certificate Store** button.
 - c. Select ***SYSTEM**. Click **Continue**.
 - d. Enter the appropriate password for ***SYSTEM** certificate store. Click **Continue**.
 - e. When the left navigational menu reloads, expand **Manage Applications**.
 - f. Click **Update certificate assignment**.
 - g. On the next screen, select **Client** application. Click **Continue**.
 - h. Select the **Directory Server client**.
 - i. Click **Update Certificate Assignment** to assign a certificate to the Directory Server client that will establish its identity.
 - j. Select a certificate from the list to assign to the server.
 - k. Click **Assign New Certificate**.
 - l. DCM reloads to the **Update Certificate Assignment** page with a confirmation message.

After SSL is enabled, you can change the port that the Directory Server uses for secured connections.

Enable Kerberos authentication on the Directory Server

If you have Network Authentication Service configured on your system, you can set up your Directory Server to use Kerberos authentication. Kerberos authentication applies to the users and the administrator. Before enabling Kerberos on the directory server, you may find it helpful to read an overview on using Kerberos with Directory Server.

To enable Kerberos authentication, follow these steps:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Properties**.
5. Click the **Kerberos** tab.
6. Check **Enable Kerberos authentication**.
7. Specify other settings on the **Kerberos** page as appropriate to your situation. See the page’s online help for information about individual fields.

Manage the schema

For more information about schema, see “Schema” on page 15.

The schema can be managed using the Web administration tool, or an LDAP application like ldapmodify in combination with LDIF files. When you are first defining new objectclasses or attributes, it may be most convenient to use the Web administration tool. If you need to copy the new schema to other servers (perhaps as part of a product or tool you are deploying), the ldapmodify utility may be more useful, see “Copy the schema to other servers” on page 135 for more information.

See the following for more information:

- “View object classes”
- “Add an object class” on page 127
- “Edit an object class” on page 128
- “Copy an object class” on page 129
- “Delete an object class” on page 130
- “View attributes” on page 130
- “Add an attribute” on page 131
- “Edit an attribute” on page 132
- “Copy an attribute” on page 133
- “Delete an attribute” on page 134

View 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.

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.

Command line

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

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

Add an object class

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 26. 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 classes” on page 18 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.

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>))
```

Edit an object class

Not all schema changes are allowed. See “Disallowed schema changes” on page 28 for change restrictions.

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 classes” on page 18 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.

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>) )
```

Copy an object class

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 classes” on page 18 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.

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

Delete an object class

Not all schema changes are allowed. See “Disallowed schema changes” on page 28 for change restrictions.

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.

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>)
```

View attributes

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

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 25 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.

Command line

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

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

Add an attribute

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

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 26. 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 25 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.
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 23 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 24 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.

Command line

The following example adds an attribute type definition for an attribute called “myAttribute”, with Directory String syntax (see “Attribute syntax” on page 25) and Case Ignore Equality matching (see “Matching rules” on page 23). 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 and ldapadd” on page 165 for more information about this command.

Edit an attribute

Not all schema changes are allowed. See “Disallowed schema changes” on page 28 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.

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.

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 <admin> -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 and ldapadd" on page 165 for more information about this command.

Copy an attribute

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

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 25 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.
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 23 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 24 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.

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 )
```

Delete an attribute

Not all schema changes are allowed. See “Disallowed schema changes” on page 28 for change restrictions.

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

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.

Command line

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

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

```
dn: cn=schema
changetype: modify
delete: attributetypes
attributetypes: (<myAttribute-oid>)
```

See “**ldapmodify and ldapadd**” on page 165 for more information about this command.

Copy the schema to other servers

To copy a schema to other servers, do the following:

1. Use the **ldapsearch** utility to copy the schema into a file:

```
ldapsearch -b cn=schema -L "(objectclass=*)" > schema.ldif
```
2. The schema file will include all objectclasses and attributes. Edit the LDIF file to include only the schema elements you want, or, you may be able to filter the **ldapsearch** output using a tool like **grep**. Be sure to put attributes before the objectclasses that reference them. For example, you might end up with the following file (note that each continued line has a single space at the end, and the continuation line has at least one space at the beginning of the line).

```
attributetypes: ( myattr1-oid NAME 'myattr1' DESC 'Some piece of
information.' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 EQUALITY 2.5.13.2
USAGE userApplications )
IBMAattributetypes: ( myattr1-oid DBNAME( 'myattr1' 'myattr1' )
ACCESS-CLASS normal LENGTH 500 )
attributetypes: ( myattr2-oid NAME 'myattr2' DESC 'Some piece of
information.' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 EQUALITY 2.5.13.2
USAGE userApplications )
IBMAattributetypes: ( myattr2-oid DBNAME( 'myattr2' 'myattr2' )
ACCESS-CLASS normal LENGTH 500 )
objectclasses: ( myobject-oid NAME 'myobject' DESC 'Represents
something.' SUP 'top' STRUCTURAL MUST ( cn ) MAY ( myattr1 $ myattr2 ) )
```

3. Insert lines before each objectclasses or attributetype line to construct LDIF directives to add these values to the entry **cn=schema**. Each object class and attribute must be added as an individual modification.

```
dn: cn=schema
changetype: modify
add: attributetypes ibmattributetypes
attributetypes: ( myattr1-oid NAME 'myattr1' DESC 'Some piece of
information.' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 EQUALITY 2.5.13.2
USAGE userApplications )
IBMAattributetypes: ( myattr1-oid DBNAME( 'myattr1' 'myattr1' )
ACCESS-CLASS normal LENGTH 500 )

dn: cn=schema
```

```

changetype: modify
add: attributetypes ibmattributetypes
attributetypes: ( myattr2-oid NAME 'myattr2' DESC 'Some piece of
information.' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 EQUALITY 2.5.13.2
USAGE userApplications )
IBMATtributetypes: ( myattr2-oid DBNAME( 'myattr2' 'myattr2' )
ACCESS-CLASS normal LENGTH 500 )

```

```

dn: cn=schema
changetype: modify
add: objectclasses
objectclasses: ( myobject-oid NAME 'myobject' DESC 'Represents
something.' SUP 'top' STRUCTURAL MUST ( cn ) MAY ( myattr1 $ myattr2 ) )

```

4. Load that schema on other servers using the ldapmodify utility:

```
ldapmodify -D cn=administrator -w <password> -f schema.ldif
```

Enable or disable read access to projected users

Note: PTF 5722SS1-SI19805 is required to enable or disable read access to projected users.

To prohibit search and compare operations to the user projected backend, do the following:

1. End the directory server. Enter `ENDTCPSVR *DIRSRV`.
2. Edit the file `/QIBM/UserData/OS400/DirSrv/ibmslapd.conf`. For example, enter `EDTF '/QIBM/UserData/OS400/DirSrv/ibmslapd.conf'`.
3. Search for the text `cn=Front End`.
4. Insert a new line containing the text `ibm-slapdSetEnv: IBMSLAPDOS400USRPRJREAD=FALSE` immediately after the line that contains the text `cn=Front End`. In the following example, the second line was inserted:

```

dn: cn=Front End, cn=Configuration
ibm-slapdSetEnv: IBMSLAPDOS400USRPRJREAD=FALSE
cn: Front End

```
5. Save the file and exit the editor. For example, press F2 to save the file, followed by F3 to exit the editor if using EDTF.
6. Restart the directory server. Enter `STRTCPSVR *DIRSRV`.

For more information, see “Read access to projected users” on page 72.

Manage directory entries

To manage directory entries, expand the **Directory management** category in the navigation area of the Web administration tool.

See the following for more information:

- “Browse the tree” on page 137
- “Add an entry” on page 137
- “Delete an entry” on page 137
- “Edit an entry” on page 138
- “Copy an entry” on page 138
- “Edit access control lists” on page 139
- “Add an auxiliary object class” on page 139
- “Delete an auxiliary class” on page 139
- “Change group membership” on page 139
- “Search the directory entries” on page 140

- “Change binary attributes” on page 142

Browse 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.

Add 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 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 “Change binary attributes” on page 142 for information about 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 “Access control lists” on page 49 for information about 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.

Delete 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.

Edit 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 “Change binary attributes” on page 142 for information about 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.

Copy 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.

Edit access control lists

To view ACL properties using the Web administration tool utility and to work with ACLs, see “Manage access control lists (ACLs)” on page 153.

See “Access control lists” on page 49 for additional information.

Add 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.

Delete 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 an 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.

Change 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.
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.

Search 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 either the **Search filters** or **Options** tab.

Note: Binary entries, for example passwords, are not searchable.

Search filters

Select one 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 "Adjust search settings" on page 106 for additional information about searches.

Change 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.

Manage users and groups

To manage users and groups, expand the **Users and groups** category in the navigation area of the Web administration tool.

See the following for more information:

- “Manage users” on page 143
- “Manage groups” on page 144

Manage users

After you have set up your realms and templates, you can populate them with users. See the following:

- “Add users”
- “Find users within the realm”
- “Edit a user’s information”
- “Copy a user”
- “Remove a user” on page 144

Add 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**.

Find 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 ending with smith.
4. You can perform the following operations on a selected user:
 - **Edit** - See “Edit a user’s information.”
 - **Copy** - See “Copy a user.”
 - **Delete** - See “Remove a user” on page 144.
5. When you are done, click **OK**.

Edit 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**.

Copy 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**.

Remove 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.

Manage groups

After you have set up your realms and templates, you can create groups. See the following:

- “Add groups”
- “Find groups within the realm”
- “Edit a group’s information”
- “Copy a group” on page 145
- “Remove a group” on page 145

Add 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 and roles” on page 43 for additional information.

Find 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 “Edit a group’s information.”
 - **Copy** - See “Copy a group” on page 145.
 - **Delete** - See “Remove a group” on page 145.
5. When you are done, click **Close**.

Edit 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**.

Copy 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.

Remove 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.

Manage realms and user templates

To manage realms and user templates click **Realms and templates** in the navigation area of the Web administration tool. Use realms and user templates to make it easier to for others to enter data into the directory. For more information about realms and user template concepts, see "Realms and user templates" on page 40.

See the following for more information:

- "Create a realm" on page 146
- "Create a realm administrator" on page 146
- "Create a template" on page 147
- "Add the template to a realm" on page 149
- "Create groups" on page 149
- "Add a user to the realm" on page 149
- "Manage realms" on page 149
- "Manage templates" on page 150

Create a realm

For more information about realms and user template concepts, see “Realms and user templates” on page 40.

To create a realm, do the following:

1. Expand the **Realms and templates** category in the navigation area of the Web administration tool.
2. 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**. This entry can be a suffix or an entry elsewhere in the directory. You can also click **Browse** to select the location of the subtree that you want.
3. Click **Next** to continue or click **Finish**.
4. 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.
5. Click **Finish** to create the realm.

Create a realm administrator

To create a realm administrator, you must first create an administration group for the realm by doing the following:

1. Create the realm administration group.
 - a. Expand the **Directory management** category in the navigation area of the Web administration tool.
 - b. Click **Manage entries**.
 - c. Expand the tree and select the realm you just created, **cn=realm1,o=ibm,c=us**.
 - d. Click **Edit ACL**.
 - e. Click the **Owners** tab.
 - f. Ensure that **Propagate owner** is checked.
 - g. Enter the DN for the realm, **cn=realm1,o=ibm,c=us**.
 - h. Change the **Type** to group.
 - i. Click **Add**.
2. Create the administrator entry. If you do not already have a user entry for the administrator, you must create one.
 - a. Expand the **Directory management** category in the navigation area of the Web administration tool.
 - b. Click **Manage entries**.
 - c. 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 accidentally delete him or herself. In this example the location might be **o=ibm,c=us**.
 - d. Click **Add**.
 - e. Select the **Structural object class**, for example **inetOrgPerson**.
 - f. Click **Next**.
 - g. Select any auxiliary object class you want to add.
 - h. Click **Next**.
 - i. Enter the required attributes for the entry. For example,
 - **RDN** cn=JohnDoe
 - **DN** o=ibm,c=us
 - **cn** John Doe

- **sn** Doe
- j. On the **Other attributes** tab ensure that you have assigned a password.
 - k. When you are done, click **Finish**.
3. Add the administrator to the administration group.
 - a. Expand the **Directory management** category in the navigation area of the Web administration tool.
 - b. Click **Manage entries**.
 - c. Expand the tree and select the realm you just created, **cn=realm1,o=ibm,c=us**.
 - d. Click **Edit attributes**.
 - e. Click the **Members** tab.
 - f. Click **Members**.
 - g. In the **Members** field enter the DN of the administrator, in this example **cn=John Doe,o=ibm,c=us**.
 - h. Click **Add**. The DN is displayed in the **Members** list.
 - i. Click **OK**.
 - j. Click **Update**. The DN is displayed in the **Current members** list.
 - k. Click **OK**.
 4. You have created an administrator that can manage entries within the realm.

Create 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 "Edit a template" on page 152.
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.

Add 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**.

Create 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 and roles” on page 43 for additional information.

Add 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**.

Manage 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. For more information, see the following:

- “Add a realm”
- “Edit a realm” on page 150
- “Remove a realm” on page 150
- “Edit ACLs on the realm” on page 150

Add 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.

Edit 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.

Remove 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.

Edit ACLs on the realm

To view ACL properties using the Web administration tool utility and to work with ACLs, see “Manage access control lists (ACLs)” on page 153.

See “Access control lists” on page 49 for additional information.

Manage 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. For more information, see the following:

- “Add a user template”
- “Edit a template” on page 152
- “Remove a template” on page 152
- “Edit ACLs on the template” on page 152

Add 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 “Edit a template” on page 152.
 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.

Edit 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 151 for information about how to modify the tabs.
- When you are done, click **Finish**.

Remove 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.

Edit 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 “Manage access control lists (ACLs).”

See “Access control lists” on page 49 for additional information.

Manage access control lists (ACLs)

For more information about access control lists, see “Access control lists” on page 49.

To view ACL properties using the Web administration tool and to work with ACLs, do the following:

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” on page 154
- “Non-filtered ACLs” on page 154
- “Filtered ACLs” on page 155
- “Owners” on page 157

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.
- 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 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 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 filter** 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.
Using cn=this with objects that propagate their ownership to other objects makes it easy to create a directory subtree in which every object is owned by itself.
- 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.

Publish information to the directory server

You can configure your system to publish certain information into an Directory Server on the same system or on a different system as well as user defined information. OS/400 automatically publishes this information to the Directory Server when you use iSeries Navigator to change this information on OS/400. Information that you can publish includes system (systems and printers), print shares, user information, and TCP/IP Quality of service policies (for more information see "Publishing" on page 34).

If the parent DN to which the data is being published does not exist, Directory Server automatically creates it. You may have also installed other OS/400 applications which publish information in an LDAP directory. Additionally, you can call application program interfaces (APIs) from your own programs to publish other types of information to the LDAP directory.

Note: You can also publish OS/400 information to a directory server that is not on running on OS/400 if you configure that server to use the IBM schema.

To configure your system to publish OS/400 information into a directory server, take these steps:

1. In iSeries Navigator, right-click on your system and select **Properties**.
2. Click the **Directory Server** tab.
3. Click on the types of information that you want to publish.

Tip: If you plan to publish more than one type of information to the same location, you can save time by selecting multiple information types to configure at one time. Operations Navigator will then use the values you enter when you configure the one information type as default values when you configure subsequent information types.

4. Click **Details**.
5. Click the **Publish system information** check box.
6. Specify the **Authentication method** that you want the server to use, as well as the appropriate authentication information.

7. Click the **Edit** button next to the **(Active) Directory server** field. In the dialog that pops up, enter the name of the directory server where you want to publish OS/400 information, then click **OK**.
8. In the **Under DN** field, enter the parent distinguished name (DN) where you want information added on the directory server.
9. Fill in the fields in the **Server connection** frame that are appropriate to your configuration.

Note: To publish OS/400 information to the directory server using SSL or Kerberos, you need to first have your directory server configured to use the appropriate protocol. See “Kerberos authentication with the Directory Server” on page 42 for more information about SSL and Kerberos.

10. If your directory server does not use the default port, enter the correct port number in the **Port** field.
11. Click **Verify** to ensure that the parent DN exists on the server and that the connection information is correct. If the directory path does not exist, a dialog will prompt you to create it.

Note: If the parent DN does not exist, and you do not create it, then publishing will not be successful.

12. Click **OK**.

Note: You can also publish i5/OS information to a directory server that is on a different platform. You must publish user and system information to a directory server that uses a schema compatible with the IBM Directory Server schema. For more information about the IBM Directory Schema, see “IBM Directory Server schema” on page 16.

APIs for publishing OS/400 information to the directory server

Directory Server provides built-in support for publishing user and system information. These items are listed on the **Directory Server** page of the systems **Properties** dialog. You can use LDAP server configuration and publishing APIs to enable the OS/400 programs that you write to publish other types of information. These types of information then appear on the **Directory Server** page as well. Like users and systems, they are initially disabled, and you configure them using the same procedure. The program that adds the data to the LDAP directory is called the publishing agent. The type of information that is published, as it appears on the **Directory Server** page, is called the agent name.

The following APIs will allow you to incorporate publishing into your own programs:

QgldChgDirSvrA

An application uses the CSVR0500 format to initially add an agent name that is marked as a disabled entry. Instructions for users of the application should instruct them to use iSeries Navigator to go to the Directory Server property page to configure the publishing agent. Examples of agent names are the systems and users agent names automatically available on the **Directory Server** page.

QgldLstDirSvrA

Use this APIs LSVR0500 format to list what agents are currently available on your system.

QgldPubDirObj

Use this API to do the actual publishing of information.

For detailed information about these APIs, see the Lightweight Directory Access Protocol (LDAP) topic under Programming in the iSeries Information Center.

Chapter 8. Troubleshoot Directory Server

Unfortunately, even reliable servers such as the Directory Server sometimes have problems. When your Directory Server has problems, the following information can help you figure out what is wrong and how to fix the trouble.

You can find return codes for LDAP errors in the ldap.h file, which is located on your system in QSYSINC/H.LDAP.

“Monitor errors and access with the Directory Server job log” on page 160

When you get an error on your Directory Server and want more details, another action to take is to view the QDIRSRV job log.

“Use TRCTCPAPP to help find problems” on page 160

For reproducible errors, you can use Trace TCP/IP Application (TRCTCPAPP APP(*DIRSRV)) command to run a trace of the errors.

“Use the LDAP_OPT_DEBUG option to trace errors” on page 161

Trace problems with clients that are using the LDAP C APIs.

“Common LDAP client errors” on page 161

Knowing the causes of common LDAP client errors can help you to solve problems with your server.

For additional information about common Directory Server problems, see the Directory Server home page



(www.iseries.ibm.com/ldap).

Directory Server uses several Structured Query Language (SQL) servers which are iSeries QSQRV jobs. When an SQL error occurs, the QDIRSRV job log will usually contain the following message:

```
SQL error -1 occurred
```

In these instances the QDIRSRV job log will refer you to the SQL server job logs. However, in some cases QDIRSRV may not contain this message and this referral, even if an SQL server is the cause of the problem. In these instances, it will help you to know what SQL server jobs the server started, so that you know in which QSQRV job logs to look for additional errors.

When the Directory Server starts normally, it generates messages similar to the following:

```
Job . . . : QDIRSRV      User . . . : QDIRSRV      System:  MYISERIES
Number . . . : 174440

>> CALL PGM(QSYS/QGLDSVR)
Job 057448/QUSER/QSQRV used for SQL server mode processing.
Job 057340/QUSER/QSQRV used for SQL server mode processing.
Job 057448/QUSER/QSQRV used for SQL server mode processing.
Job 057166/QUSER/QSQRV used for SQL server mode processing.
Job 057279/QUSER/QSQRV used for SQL server mode processing.
Job 057288/QUSER/QSQRV used for SQL server mode processing.
Directory Server started successfully.
```

The messages refer to the QSQRV jobs that were started for the server. The number of messages may differ on your server depending on the configuration and the number of QSQRV jobs needed to accomplish server startup.

On the directory servers **Database/Suffixes** Properties page in iSeries Navigator you specify the total number of SQL servers that Directory Server uses for directory operations after server startup. Additional SQL servers are started for replication.

Monitor errors and access with the Directory Server job log

Viewing the job log for your Directory Server can alert you to errors and help you to monitor server access. The job log contains:

- Messages about server operation and any problems within the server such as SQL server jobs or replication failures.
- Security related messages reflecting operations by clients such as wrong passwords.
- Messages giving details about client errors such as missing required attributes.

You may not want to log the client errors unless you are debugging client problems. You can control the logging of client errors on the **General** properties tab of the Directory Server in iSeries Navigator.

If your server is started, take these steps to view the QDIRSRV job log:

1. In iSeries Navigator, expand **Network**.
2. Expand **Servers**.
3. Click **TCP/IP**.
4. Right-click **Directory** and select **Server Jobs..**
5. From the **File** menu, choose **Job Log**.

If your server is stopped, take these steps to view the QDIRSRV job log:

1. In iSeries Navigator, expand **Basic Operations**.
2. Click **Printer Output**.
3. QDIRSRV appears in the **User** column of iSeries Navigators right panel. To view the job log, double-click on **Qpjoblog** to the left of QDIRSRV in the same row.

Note: iSeries Navigator may be configured to show only spooled files. If QDIRSRV does not appear on the list, click **Printer Output**, then choose **Include** from the **Options** menu. Specify **All** in the **User** field, then click **OK**.

Note: Directory Server uses other system resources to perform some tasks. If an error occurs with one of those resources, the job log will indicate where to go for information. In some cases Directory Server may not be able to determine where to look. In those cases, look in the Structured Query Language (SQL) servers job log to see if the problem was related to SQL servers.

Use TRCTCPAPP to help find problems

Your server provides a communication trace to collect data on a communications line, such as a local area network (LAN) or a wide area network (WAN) interface. The average user may not understand the entire contents of the trace data. However, you can use the trace entries to determine whether a data exchange between two points actually took place.

The Trace TCP/IP Application (TRCTCPAPP) command with the *DIRSRV option can be used on the Directory Server to aid in finding problems with clients or applications.

For more detailed information about the uses of the TRCTCPAPP command with LDAP as well as the restrictions on required authorities, see TRCTCPAPP (Trace TCP/IP Application) Command Description.

For general information about using communications trace, see Communications trace.

Use the LDAP_OPT_DEBUG option to trace errors

You can use the LDAP_OPT_DEBUG option of the `ldap_set_option()` API to trace problems with clients that are using the LDAP C APIs. The debug option has multiple debug level setting that you can use to aid in troubleshooting problems with these applications.

The following is an example of enabling the client trace debug option.

```
int debugvalue= LDAP_DEBUG_TRACE | LDAP_DEBUG_PACKETS;
ldap_set_option( ld, LDAP_OPT_DEBUG, &debugvalue);
```

An alternate way of setting the debug level is to configure the numerical value of the LDAP_DEBUG environment variable, for the job in which the client application runs, to the same numerical value that the debugvalue would be if the `ldap_set_option()` API is used.

An example of enabling the client trace using the LDAP_DEBUG environment variable is the following:

```
ADDENVVAR ENVVAR(LDAP_DEBUG) VALUE(0x0003)
```

After running the client that produces the problem you are having, type the following at the iSeries prompt:

```
DMPUSRTRC ClientJobNumber
```

where ClientJobNumber is the number of the client job.

To display this information interactively, type the following at the iSeries prompt:

```
DSPPFM QAP0ZDMP QP0Znnnnnn
```

where QAP0ZDMP contains a zero and nnnnnn is the job number.

To save this information in order to send the information to service, take the following steps:

1. Create a SAVF file using the create SAVF (CRTSAVF) command.
2. Type the following at the iSeries command prompt.

```
SAVOBJ OBJ(QAP0ZDMP LIB(QTEMP) DEV(*SAVF) SAVF(xxx)
```

where QAP0ZDMP contains a zero and xxx is the name that you specified for the SAVF file.

Common LDAP client errors

Knowing the causes of common LDAP client errors can help you to solve problems with your server. For a complete list of LDAP client error conditions, see the “Directory Server APIs” topic under Programming in the iSeries Information Center.

The client error messages have the following format:

```
[Failing LDAP operation]:[LDAP client API error conditions]
```

Note: The explanation of these errors assumes that the client is communicating with an LDAP server on i5/OS. A client communicating with a server on a different platform might get similar errors, but the causes and resolutions would most likely be different.

Common messages include the following:

- “ldap_search: Timelimit exceeded” on page 162

- “[Failing LDAP operation]: Operations error”
- “ldap_bind: No such object”
- “ldap_bind: Inappropriate authentication”
- “[Failing LDAP operation]: Insufficient access”
- “[failing LDAP operation]: Cannot contact LDAP server”
- “[failing LDAP operation]: Failed to connect to SSL server” on page 163

ldap_search: Timelimit exceeded

This error occurs when ldapsearches are performing slowly. To correct this error, you can do one or both of the following:

- Increase the search time limit for the Directory Server. See “Adjust performance settings” on page 107 for information about doing this.
- Reduce the activity on your system. You can also reduce the number of active LDAP client jobs running.

[Failing LDAP operation]: Operations error

Several things can generate this error. To get information about the cause of this error for a particular instance, look at the QDIRSRV job logs (as described in “Monitor errors and access with the Directory Server job log” on page 160) and the Structured Query Language (SQL) server job logs (as described in Chapter 8, “Troubleshoot Directory Server,” on page 159).

ldap_bind: No such object

A common cause of this error is that a user makes a typing mistake when performing an operation. Another common cause is when the LDAP client attempts to bind with a DN that does not exist. This often occurs when the user specifies what he or she mistakenly thinks is the administrator DN. For example, the user may specify QSECOFR or Administrator, when the actual administrator DN may be something like cn=Administrator.

For details about the error, look at the QDIRSRV job log as described in “Monitor errors and access with the Directory Server job log” on page 160.

ldap_bind: Inappropriate authentication

The server returns Invalid credentials when the password or bind DN is incorrect. The server returns inappropriate authentication when the client attempts to bind as one of the following:

- An entry that does not have a userpassword attribute
- An entry that represents an i5/OS user, which has a UID attribute and not a userpassword attribute. This causes a compare to be done between the password specified and the i5/OS user password, which do not match.
- An entry that represents a projected user and a bind method other than simple has been requested.

This error is usually generated when the client attempts to bind with a password that is not valid. To obtain details about the error, look at the QDIRSRV job log as described in “Monitor errors and access with the Directory Server job log” on page 160.

[Failing LDAP operation]: Insufficient access

This error is usually generated when the binding DN does not have authority to do the operation (such as an add or delete) that the client requests. To get information about the error, look at the QDIRSRV job log as described in “Monitor errors and access with the Directory Server job log” on page 160.

[failing LDAP operation]: Cannot contact LDAP server

The most common causes of this error include the following:

- An LDAP client makes a request before the LDAP server on the specified system is up and in select wait status.
- The user specifies a port number that is not valid. For example, the server is listening on port 386, but the client request attempts to use port 387.

To get information about the error, look at the QDIRSRV job log as described in “Monitor errors and access with the Directory Server job log” on page 160. If the Directory Server started successfully, the message Directory Server started successfully will be in the QDIRSRV job log.

[failing LDAP operation]: Failed to connect to SSL server

This error occurs when the LDAP server rejects the client connection because a secure socket connection cannot be established. This can be caused by any of the following:

- The Certificate Management support rejects the clients attempt to connect to the server. Use Digital Certificate Manager to make sure that your certificates are set up properly, then restart the server and try to connect again.
- The user may not have read access to the *SYSTEM certificate store (by default /QIBM/userdata/ICSS/Cert/Server/default.kdb).

For i5/OS C applications, additional SSL error information is available. See “Directory Server APIs” in the Programming topic for details.

Chapter 9. Reference

See the following for additional reference information.

- “Command line utilities”
- “LDAP data interchange format (LDIF)” on page 190
- “Directory Server configuration schema” on page 193

Command line utilities

This section describes the utilities that can be run from the Qshell command environment on i5/OS. See the following commands for more information:

- “ldapmodify and ldapadd”
- “ldapdelete” on page 168
- “ldapexop” on page 170
- “ldapmodrdn” on page 175
- “ldapsearch” on page 177
- “ldapchangepwd” on page 185
- “ldapdiff” on page 187
- “Notes about using SSL with the LDAP command line utilities” on page 190

Note that some strings need to be contained in quotation marks in order to be processed correctly in the Qshell command environment. This generally pertains to strings that are DN's, search filters, and the list of attributes to be returned by ldapsearch. See the following list for some examples.

- Strings that contain spaces: "cn=John Smith,cn=users"
- Strings that contain wildcard characters: "*"
- Strings that contain parentheses: "(objectclass=person)"

For more information about the Qshell command environment, see the “Qshell” topic.

ldapmodify and 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, ldap_add, ldap_delete, and ldap_modrdn application programming interfaces (APIs). **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 value.
- c** Continuous operation mode. Errors are reported, but **ldapmodify** continues with modifications. Otherwise the 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. Use the **-C charset** option if the input string codepage is different from the job codepage value. Refer to the `ldap_set_iconv_local_charset()` API to see supported charset values.
- d debuglevel**
Set the LDAP debugging level to *debuglevel*.
- 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 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 system keyring file will be used, if present.

This parameter effectively enables the **-Z** switch. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.
- 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. Valid mechanisms are:
 - CRAM-MD5 - protects the password sent to the server.
 - EXTERNAL - uses the SSL certificate. Requires **-Z**.
 - GSSAPI - uses the user's Kerberos credentials

- 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 certificate/private key pair has been designated as the default for the key database file. 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. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.
- 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.
- w** *passwd* | ?
Use *passwd* as the password for authentication. Use the ? to generate a password prompt.
- Z** Use a secure SSL connection to communicate with the LDAP server. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

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. See “LDAP data interchange format (LDIF)” on page 190 for more information about the LDIF format.

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/newentry
```

adds a new entry for John Doe, using the values from the file /tmp/newentry.

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.

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.

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 application programming interface (API).

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 a 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 the **-C charset** option if the input string codepage is different from the job codepage value. Refer to the `ldap_set_iconv_local_charset()` API to see supported charset values.
- d debuglevel**
Set the LDAP debugging level to *debuglevel*.
- 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 the server administration control.
- K keyfile**
Specify the name of the SSL key database file. If the key database file is not in the current directory, specify the fully-qualified key database filename.

If the utility cannot locate a key database, it will use a hard-coded set of default trusted certificate authority roots. The key database file typically contains one or more certificates of certification authorities (CAs) that are trusted by the client. These types of X.509 certificates are also known as trusted roots.

This parameter effectively enables the **-Z** switch. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.
- m mechanism**
Use *mechanism* to specify the SASL mechanism to be used to bind to the server. 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. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

-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.

-w *passwd* | ?

Use *passwd* as the password for authentication. Use the ? to generate a password prompt.

-Z Use a secure SSL connection to communicate with the LDAP server. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

dn Specifies one or more DN arguments. Each DN should be a string-represented DN.

Examples

The following command,

```
ldapdelete -D cn=administrator -w secret "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.

Notes

If no DN arguments are provided, the **ldapdelete** command waits to read a list of DNs from standard input.

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.

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 | controlqueue | controlrepl |
quiesce | readconfig}
```

Description

The **ldapexop** utility is a command-line interface that provides the capability to bind to a directory server 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 the **-C** *charset* option if the input string codepage is different from the job codepage value. Refer to the `ldap_set_iconv_local_charset()` API to see supported charset values.

-d *debuglevel*

Set the LDAP debugging level to *debuglevel*.

-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 command syntax and usage information.

-K *keyfile*

Specify the name of the SSL key database file. If the key database file is not in the current directory, specify the fully-qualified key database filename.

If the utility cannot locate a key database, the system key database is used. The key database file typically contains one or more certificates of certification authorities (CAs) that are trusted by the client. These types of X.509 certificates are also known as trusted roots.

This parameter effectively enables the **-Z** switch. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

-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. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

-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 command syntax and usage information.

-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.

-Z Use a secure SSL connection to communicate with the LDAP server. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

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
```

- **controlqueue:** control queue replication extended operation. This operation allows you to delete or remove pending changes from the list of replication changes that have queued up and were not run because of replication failures. This operation is useful when the replica data is manually fixed. You would then use this operation to skip doing some of the queued up failures.

-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"
```

- **quiesce:** quiesce or unquiesce subtree replication 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
```

Note: The following entries marked with:

- ¹ take effect immediately
- ² take effect on new operations
- ³ take effect as soon as the password is changed (no readconfig required)
- ⁴ are supported by the command line utility on i5/OS, but are not supported by the Directory Server on i5/OS

```
cn=Configuration
ibm-slapdadmindn2
ibm-slapdadminpw2, 3, 4
ibm-slapderrorlog1, 4
ibm-slapdpwncryption1
ibm-slapdsizelimit1
ibm-slapdsysloglevel1, 4
ibm-slapdtimelimit1
cn=Front End, cn=Configuration
ibm-slapdaclcache1
ibm-slapdaclcachesize1
ibm-slapdentrycachesize1
ibm-slapdfiltercachebypasslimit1
ibm-slapdfiltercachesize1
ibm-slapdidletimeout1
cn=Event Notification, cn=Configuration
ibm-slapdmaxeventsperconnection2
ibm-slapdmaxeventstotal2
cn=Transaction, cn=Configuration
ibm-slapdmaxnumoftransactions2
ibm-slapdmaxoppertransaction2
ibm-slapdmaxtimelimitoftransactions2
cn=ConfigDB, cn=Config Backends, cn=IBM SecureWay, cn=Schemas, cn=Configuration
ibm-slapdreadonly2
cn=Directory, cn=RDBM Backends, cn=IBM SecureWay, cn=Schemas, cn=Configuration
ibm-slapdbulkloaderrors1, 4
ibm-slapdclierrors1, 4
ibm-slapdpagedresallownonadmin2
ibm-slapdpagedreslmt2
ibm-slapdpagesizelmt2
ibm-slapdreadonly2
ibm-slapdsortkeylimit2
ibm-slapdsortsrchallownonadmin2
ibm-slapdsuffix2
```

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.

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` application programming interface (API).

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 “Distinguished names (DNs)” on page 11 for information about RDNs (Relative Distinguished Names) and DN (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 the **-C charset** option if the input string codepage is different from the job codepage value. Refer to the `ldap_set_iconv_local_charset()` API to see supported `charset` values. 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`.

-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. If the key database file is not in the current directory, specify the fully-qualified key database filename.

If the utility cannot locate a key database, it will use a hard-coded set of default trusted certificate authority roots. The key database file typically contains one or more certificates of certification authorities (CAs) that are trusted by the client. These types of X.509 certificates are also known as trusted roots.

This parameter effectively enables the **-Z** switch. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

-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. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

-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.

-Z Use a secure SSL connection to communicate with the LDAP server. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

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.

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.

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 [attrs...]
```

Description

ldapsearch is a command-line interface to the `ldap_search` application programming interface (API).

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 Directory Server APIs for more information about 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 character 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 the **-C charset** option if the input string codepage is different from the job codepage value. Refer to the ldap_set_iconv_local_charset() API to see supported charset values. 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. 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.
- 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 exit.
- 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. If the key database file is not in the current directory, specify the fully-qualified key database filename.

If the utility cannot locate a key database, it will use a hard-coded set of default trusted certificate authority roots. The key database file typically contains one or more certificates of certification authorities (CAs) that are trusted by the client. These types of X.509 certificates are also known as trusted roots.

This parameter effectively enables the **-Z** switch. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

-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.

For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

-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 matching rule OID attribute is not supported by the Directory Server, however other LDAP servers may support this attribute.
- 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 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.

These parameters can be useful when the client has limited resources or when it is connected through a low-bandwidth connection. In general, it allows you to control the rate at which data is returned from a search request. Instead of receiving all of the results at once, you can get them a few entries (a page) at a time. In addition, you can control the duration of the delay taken between each page request, giving the client time to process the results.

```
-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. .


-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. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

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> <filterlist>  
<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, if the `filtertype` is `'='` then `<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 are 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
```

The command:

```
ldapsearch -L -s one -b "c=US" "o=university*" o description
```

performs a one-level search at the c=US level for all organizations whose organizationName begins with university. Search results 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: UF1
description: Shaper of young minds

...

```

The 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 (ibm-slapdDN), when used for sorted searches, sorts the search results 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

```

The command:

```
ldapsearch -h hostname -o sn -b "o=ibm,c=us" "title=engineer"
```

returns all entries in an IBM employee directory whose title is "engineer", with the results sorted by surname.

The command:

```
ldapsearch -h hostname -o -sn -o cn -b "o=ibm,c=us" "title=engineer"
```

returns all entries in an IBM employee directory whose title is "engineer", with the results sorted by surname (in descending order) and then by common name (in ascending order).

The command:

```
ldapsearch -h hostname -q 5 -T 3 -b o=ibm,c=us "title=engineer"
```

returns five entries per page, with a delay of 3 seconds between pages for all entries in an IBM employee directory whose title is "engineer".

This example demonstrates searches where a referral object is involved. As discussed in "LDAP directory referrals" on page 41, Directory Server LDAP directories may contain referral objects, provided that they contain only the following:

- A distinguished name (dn).
- An objectClass (objectClass).
- A referral (ref) attribute.

Assume that 'System_A' holds the referral entry:

```
dn: cn=Barb Jensen, ou=Rochester, o=Big Company, c=US
ref: ldap://System_B:389/cn=Barb Jensen,
    ou=Rochester, o=Big Company, c=US
objectclass: referral
```

All attributes associated with the entry should reside on 'System_B'.

System_B contains an entry:

```
dn: cn=Barb Jensen, ou=Rochester, o=Big Company, c=US
cn: Barb Jensen
objectclass: organizationalPerson
sn: Jensen
telephonenumber: (800) 555 1212
```

When a client issues a request to 'System_A', the LDAP server on System_A responds to the client with the URL:

```
ldap://System_B:389/cn=Barb Jensen,
ou=Rochester, o=Big Company, c=US
```

The client uses this information to issue a request to System_B. If the entry on System_A contains attributes in addition to dn, objectclass, and ref, the server ignores those attributes (unless you specify the **-R** flag to indicate not to chase referrals).

When the client receives a referral response from a server, it issues the request again, this time to the server to which the returned URL refers. The new request has the same scope as the original request. The results of this search vary depending on the value you specify for the scope of the search (**-b**).

If you specify **-s base**, as shown here:

```
ldapsearch -h System_A -b 'ou=Rochester, o=Big Company, c=US'
-s base 'sn=Jensen'
```

the search returns all attributes for all entries with 'sn=Jensen' that reside in 'ou=Rochester, o=Big Company, c=US' on both System_A and System_B.

If you specify `-s sub`, as shown here:

```
ldapsearch -h System_A -b 'ou=Rochester, o=Big Company, c=US'
-s sub 'sn=Jensen'
```

the search returns all attributes for all entries with 'sn=Jensen' that reside in or below 'ou=Rochester, o=Big Company, c=US' on both System_A and System_B.

If you specify `-s one`, as shown here:

```
ldapsearch -h System_A -b 'ou=Rochester, o=Big Company, c=US'
-s one 'sn=Jensen'
```

the search returns no entries on either system. Instead, the server returns the referral URL to the client:

```
ldap://System_B:389/cn=Barb Jensen,
ou=Rochester, o=Big Company, c=US
```

The client in turn submits a request:

```
ldapsearch -h System_B -b 'ou=Rochester, o=Big Company, c=US'
-s one 'sn=Jensen'
```

This does not give any results either, because the entry

```
dn: cn=Barb Jensen, ou=Rochester, o=Big Company, c=US
```

resides at

```
ou=Rochester, o=Big Company, c=US
```

A search with `-s one` attempts to find entries in the level immediately below

```
ou=Rochester, o=Big Company, c=US
```

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.

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. Allows the password for a directory entry to be changed.

Options

`-C charset`

Specifies that the DN's supplied as input to the **ldapdelete** utility are represented in a local

character set, as specified by *charset*. Use the **-C** *charset* option if the input string codepage is different from the job codepage value. Refer to the `ldap_set_iconv_local_charset()` API to see supported *charset* values.

-d *debuglevel*

Set the LDAP debugging level to *debuglevel*.

-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. If the key database file is not in the current directory, specify the fully-qualified key database filename.

If the utility cannot locate a key database, it will use a hard-coded set of default trusted certificate authority roots. The key database file typically contains one or more certificates of certification authorities (CAs) that are trusted by the client. These types of X.509 certificates are also known as trusted roots.

This parameter effectively enables the **-Z** switch. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

-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.

-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. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.

-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.
- Z** Use a secure SSL connection to communicate with the LDAP server. For Directory Server on i5/OS if you use **-Z** and do not use **-K** or **-N**, the certificate associated with the Directory Services Client application ID will be used.
- ?** 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

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.

ldapdiff

The LDAP replica synchronization tool.

Note: This command could run for a long time depending on the number of entries (and attributes for those entries) that are replicated.

Synopsis

(Compares and synchronizes data entries between two servers within a replication environment.)

```
ldapdiff -b baseDN -sh host -ch host [-a] [-C countnumber]
[-cD dn] [-cK keyStore] [-cw password] [-cN keyLabel]
[-cp port] [-cP keyStorePwd] [-cZ] [-F] [-L filename] [-sD dn] [-sK keyStore]
[-sw password] [-sN keyLabel] [-sp port] [-sP keyStorePwd]
[-sZ] [-v]
```

or

(Compares the schema between two servers.)

```
ldapdiff -S -sh host -ch host [-a] [-C countnumber] [-cD dn]
[-cK keyStore] [-cw password] [-cN keyLabel] [-cp port]
[-cP keyStorePwd] [-cZ] [-L filename] [-sD dn]
[-sK keyStore] [-sw password] [-sN keyLabel] [-sp port]
[-sP keyStorePwd] [-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 this parameter is not specified, or the value is an empty string (**-sK""**) the system keystore is used. If the key database file is not in the current directory, specify the fully-qualified key database filename.
- sN** *keyLabel*
Specify the label associated with the client certificate in the key database file. If a label is specified without specifying a keystore, the label is an application identifier in the Digital Certificate Manager (DCM). The default label (application id) is QIBM_GLD_DIRSRV_CLIENT. 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 is required. *keyLabel* is not required if a default certificate/private key pair has been designated. Similarly, *keyLabel* 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. The password is not used if there is a stash file for the keystore being used.
- 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.

-sZ Use a secure SSL connection to communicate with the LDAP server.

Options for a replication consumer

The following options apply to the consumer server and are denoted by an initial 'c' in the option name. For convenience, if **-cZ** is specified without specifying values for **-cK**, **-cN** or **-cP**, these options use the same value specified for the supplier SSL options. To override the supplier options and use the defaults setting, specify **-cK "" -cN "" -cP ""**.

-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 value is an empty string (**-sK""**) the system keystore is used. If the key database file is not in the current directory, specify the fully-qualified key database filename.

-cN keyLabel
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 a label is specified without specifying a keystore, the label is an application identifier in the Digital Certificate Manager (DCM). The default label (application id) is QIBM_GLD_DIRSRV_CLIENT. If the LDAP server is configured to perform client and server authentication, a client certificate is required. *keyLabel* is not required if a default certificate/private key pair has been designated. Similarly, *keyLabel* 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.

-cw password | ?
Use *password* as the password for authentication. Use the ? to generate a password prompt.

-cZ Use a secure SSL connection to communicate with the LDAP server.

Examples

```
ldapdiff -b <baseDN> -sh <supplierhostname> -ch <consumerhostname> [options]
```

or

```
ldapdiff -S -sh <supplierhostname> -ch <consumerhostname> [options]
```

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.

Notes about using SSL with the LDAP command line utilities

To use the Secure Sockets Layer (SSL) features of the command line utilities, you must have installed one of the Cryptographic Access Provider Products (5722-ACx).

“Secure Sockets Layer (SSL) and Transport Layer Security with the Directory Server” on page 42 discusses using SSL with the Directory Server LDAP server. This information includes managing and creating trusted Certificate Authorities with Digital Certificate Manager.

Some of the LDAP servers accessed by the client use server authentication only. For these servers, you only need to define one or more trusted root certificates in the certificate store. With server authentication, the client can be assured that the target LDAP server has been issued a certificate by one of the trusted Certificate Authorities (CAs). In addition, all LDAP transactions that flow over the SSL connection with the server are encrypted. This includes the LDAP credentials that are supplied on application program interfaces (APIs) that are used to bind to the directory server. For example, if the LDAP server is using a high-assurance Verisign certificate, you should do the following:

1. Obtain a CA certificate from Verisign.
2. Use DCM to import it into your certificate store.
3. Use DCM to mark it as trusted.

If the LDAP server is using a privately issued server certificate, the servers administrator can supply you with a copy of the servers certificate request file. Import the certificate request file into your certificate store and mark it as trusted.

If you use the shell utilities to access LDAP servers that use both client authentication and server authentication, you must do the following:

- Define one or more trusted root certificates in the system certificate store. 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. This includes the LDAP credentials that are supplied on application program interfaces (APIs) that are used to bind to the directory server.
- Create a key pair and request a client certificate from a CA. After receiving the signed certificate from the CA, receive the certificate into the key ring file on the client.

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:: IGJ1Z21ucyB3aXRoIGEgc3BhY2U=
```

Multiple entries within the same LDIF file are separated by a blank line. Multiple blank lines are considered a logical end-of-file.

For more information, see the following:

- "LDIF example"
- "Version 1 LDIF support"
- "Version 1 LDIF examples" on page 192

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/4AAQSkZJRgABAAAAQABAAD/2wBDABALD
A4MChAODQ4SERATGCgaGBYWGDEjJR0oOjM9PDkzODdASFxOQ
ERXRTc4UG1RV19iZ2hnPk1xeXBkeFxlZ2P/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 "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.

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 ("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.

Directory Server configuration schema

This information describes the Directory Information Tree (DIT) and the attributes that are used to configure the `ibmslapd.conf` file. In previous releases the directory configuration settings were stored in a proprietary format in the configuration file. The directory settings are now stored using the LDIF format in the configuration file.

The configuration file is named `ibmslapd.conf`. The schema used by the configuration file is also now available. Attribute types can be found in the `v3.config.at` file, and object classes are in the `v3.config.oc` file. Attributes can be modified using the `ldapmodify` command. For more information about the `ldapmodify` command, see “`ldapmodify` and `ldapadd`” on page 165.

- “Directory information tree”
- “Attributes” on page 202

Directory information tree

`cn=Configuration`

- `cn=Admin`
- `cn=Event Notification`
- `cn=Front End`
- `cn=Kerberos`
- `cn=Master Server`
- `cn=Referral`
- `cn=Schema`
 - `cn=IBM Directory`
 - `cn=Config Backends`
 - `cn=ConfigDB`
 - `cn=RDBM Backends`
 - `cn=Directory`
 - `cn=ChangeLog`
 - `cn=LDCF Backends`
 - `cn=SchemaDB`
- `cn=SSL`
 - `cn=CRL`
- `cn=Transaction`

`cn=Configuration`

DN `cn=Configuration`

Description

This is the top-level entry in the configuration DIT. It holds data of global interest to the server, although in practice it also contains miscellaneous items. Every attribute in the this entry comes from the first section (global stanza) of `ibmslapd.conf`.

Number

1 (required)

Object Class

ibm-slapdTop

Mandatory Attributes

- cn
- ibm-slapdAdminDN
- ibm-slapdAdminPW
- ibm-slapdErrorLog
- ibm-slapdPort
- ibm-slapdPwEncryption
- ibm-slapdSizeLimit
- ibm-slapdSysLogLevel
- ibm-slapdTimeLimit
- objectClass

Optional Attributes

- ibm-slapdACLAccess
- ibm-slapdACIMechanism
- ibm-slapdConcurrentRW (Deprecated)
- ibm-slapdMaxPendingChangesDisplayed
- ibm-slapdServerId
- ibm-slapdSupportedWebAdmVersion
- ibm-slapdVersion

cn=Admin

DN cn=Admin, cn=Configuration

Description

Global configuration settings for IBM Admin Daemon

Number

1 (required)

Object Class

ibm-slapdAdmin

Mandatory Attributes

- cn
- ibm-slapdErrorLog
- ibm-slapdPort

Optional Attributes

- ibm-slapdSecurePort

cn=Event Notification

DN cn=Event Notification, cn=Configuration

Description

Global event notification settings for Directory Server

Number

0 or 1 (optional; needed only if you want to enable event notification)

Object Class

ibm-slapdEventNotification

Mandatory Attributes

- cn
- ibm-slapdEnableEventNotification
- objectClass

Optional Attributes

- ibm-slapdMaxEventsPerConnection
- ibm-slapdMaxEventsTotal

cn=Front End

DN cn=Front End, cn=Configuration

Description

Global environment settings that the server applies at startup.

Number

0 or 1 (optional)

Object Class

ibm-slapdFrontEnd

Mandatory Attributes

- cn
- objectClass

Optional Attributes

- ibm-slapdACLCache
- ibm-slapdACLCacheSize
- ibm-slapdDB2CP
- ibm-slapdEntryCacheSize
- ibm-slapdFilterCacheBypassLimit
- ibm-slapdFilterCacheSize
- ibm-slapdPlugin
- ibm-slapdSetenv
- ibm-slapdIdleTimeOut

cn=Kerberos

DN cn=Kerberos, cn=Configuration

Description

Global Kerberos authentication settings for Directory Server.

Number

0 or 1 (optional)

Object Class

ibm-slapdKerberos

Mandatory Attributes

- cn
- ibm-slapdKrbEnable
- ibm-slapdKrbRealm

- ibm-slapdKrbKeyTab
- ibm-slapdKrbIdentityMap
- ibm-slapdKrbAdminDN
- objectClass

Optional Attributes

- None

cn=Master Server

DN cn=Master Server, cn=Configuration

Description

When configuring a replica, this entry holds the bind credentials and referral URL of the master server.

Number

0 or 1 (optional)

Object Class

ibm-slapdReplication

Mandatory Attributes

- cn
- ibm-slapdMasterPW (Mandatory if not using Kerberos authentication.)

Optional Attributes

- ibm-slapdMasterDN
- ibm-slapdMasterPW (Optional if using Kerberos authentication.)
- ibm-slapdMasterReferral
- objectClass

cn=Referral

DN cn=Referral, cn=Configuration

Description

This entry contains all the referral entries from the first section (global stanza) of ibmslapd.conf. If there are no referrals (there are none by default), this entry is optional.

Number

0 or 1 (optional)

Object Class

ibm-slapdReferral

Mandatory Attributes

- cn
- ibm-slapdReferral
- objectClass

Optional Attributes

- None

cn=Schemas

DN cn=Schemas, cn=Configuration

Description

This entry serves as a container for the schemas. This entry is not really necessary because the schemas can be distinguished by the object class `ibm-slapdSchema`. It is included to improve the readability of the DIT.

Only one schema entry is currently allowed: `cn=IBM Directory`.

Number

1 (required)

Object Class

Container

Mandatory Attributes

- `cn`
- `objectClass`

Optional Attributes

- None

cn=IBM Directory

DN `cn=IBM Directory, cn=Schemas, cn=Configuration`

Description

This entry contains all the schema configuration data from the first section (global stanza) of `ibmslapd.conf`. It also serves as a container for all the backends which use the schema. Multiple schemas are not currently supported, but if they were, then there would be one `ibm-slapdSchema` entry per schema. Note that multiple schemas are assumed to be incompatible. Therefore, a backend can be associated with a single schema only.

Number

1 (required)

Object Class

`ibm-slapdSchema`

Mandatory Attributes

- `cn`
- `ibm-slapdSchemaCheck`
- `ibm-slapdIncludeSchema`
- `objectClass`

Optional Attributes

- `ibm-slapdSchemaAdditions`

cn=Config Backends

DN `cn=Config Backends, cn=IBM Directory, cn=Schemas, cn=Configuration`

Description

This entry serves as a container for the Config backends.

Number

1 (required)

Object Class

Container

Mandatory Attributes

- `cn`

- objectClass

Optional Attributes

None

cn=ConfigDB

DN cn=ConfigDB, cn=Config Backends, cn=IBM Directory, cn=Schemas, cn=Configuration

Description

Configuration backend for IBM Directory server configuration

Number

0 - n (optional)

Object Class

ibm-slapdConfigBackend

Mandatory Attributes

- ibm-slapdSuffix
- ibm-slapdPlugin

Optional Attributes

- ibm-slapdReadOnly

cn=RDBM Backends

DN cn=RDBM Backends, cn=IBM Directory, cn=Schemas, cn=Configuration

Description

This entry serves as a container for the RDBM backends. It effectively replaces the database rdbm line from ibmslapd.conf by identifying all sub-entries as DB2 backends. This entry is not really necessary because the RDBM backends can be distinguished by object class ibm-slapdRdbmBackend. It is included to improve the readability of the DIT.

Number

0 or 1 (optional)

Object Class

Container

Mandatory Attributes

- cn
- objectClass

Optional Attributes

- None

cn=Directory

DN cn=Directory, cn=RDBM Backends, cn=IBM Directory, cn=Schemas, cn=Configuration

Description

This entry contains all the database configuration settings for the default RDBM database backend.

Although multiple backends with arbitrary names can be created, the Server Administration assumes that "cn=Directory" is the main directory backend, and that "cn=Change Log" is the optional changelog backend. Only the suffixes displayed in "cn=Directory" are configurable through the Server Administration (except for the changelog suffix, which is set transparently by enabling changelog).

Number

0 - n (optional)

Object Class

ibm-slapdRdbmBackend

Mandatory Attributes

- cn
- ibm-slapdDbInstance
- ibm-slapdDbName
- ibm-slapdDbUserID
- objectClass

Optional Attributes

- ibm-slapdBulkloadErrors
- ibm-slapdChangeLogMaxEntries
- ibm-slapdCLIErrors
- ibm-slapdDBAlias
- ibm-slapdDB2CP
- ibm-slapdDbConnections
- ibm-slapdDbLocation
- ibm-slapdPagedResAllowNonAdmin
- ibm-slapdPagedResLmt
- ibm-slapdPageSizeLmt
- ibm-slapdPlugin
- ibm-slapdReadOnly
- ibm-slapdReplDbConns
- ibm-slapdSortKeyLimit
- ibm-slapdSortSrchAllowNonAdmin
- ibm-slapdSuffix
- ibm-slapdUseProcessIdPw

Note: If you are using **ibm-slapdUseProcessIdPw**, you must modify the schema to make **ibm-slapdDbUserPW** optional.

cn=Change Log

DN cn=Change Log, cn=RDBM Backends, cn=IBM Directory, cn=Schemas, cn=Configuration

Description

This entry contains all the database configuration settings for the change log backend.

Number

0 - n (optional)

Object Class

ibm-slapdRdbmBackend

Mandatory Attributes

- cn
- ibm-slapdDbInstance
- ibm-slapdDbName
- ibm-slapdDbUserID

- objectClass

Optional Attributes

- ibm-slapdBulkloadErrors
- ibm-slapdChangeLogMaxEntries
- ibm-slapdCLIErrors
- ibm-slapdDBAlias
- ibm-slapdDB2CP
- ibm-slapdDbConnections
- ibm-slapdDbLocation
- ibm-slapdPagedResAllowNonAdmin
- ibm-slapdPagedResLmt
- ibm-slapdPageSizeLmt
- ibm-slapdPlugin
- ibm-slapdReadOnly
- ibm-slapdReplDbConns
- ibm-slapdSortKeyLimit
- ibm-slapdSortSrchAllowNonAdmin
- ibm-slapdSuffix
- ibm-slapdUseProcessIdPw

Note: If you are using **ibm-slapdUseProcessIdPw**, you must modify the schema to make **ibm-slapdDbUserPW** optional.

cn=LDCF Backends

DN cn=LDCF Backends, cn=IBM Directory, cn=Schemas, cn=Configuration

Description

This entry serves as a container for the LDCF backends. It effectively replaces the database ldcf line from ibmslapd.conf by identifying all sub-entries as LDCF backends. This entry is not really necessary because the LDCF backends can be distinguished by the object class ibm-slapdLdcfBackend. It is included to improve the readability of the DIT.

Number

1 (required)

Object Class

Container

Mandatory Attributes

- cn
- objectClass

Optional Attributes

- ibm-slapdPlugin

cn=SchemaDB

DN cn=SchemaDB, cn=LDCF Backends, cn=IBM Directory, cn=Schemas, cn=Configuration

Description

This entry contains all the database configuration data from the ldcf database section of ibmslapd.conf.

Number

1 (required)

Object Class

ibm-slapdLdcfBackend

Mandatory Attributes

- cn
- objectClass

Optional Attributes

- ibm-slapdPlugin
- ibm-slapdSuffix

cn=SSL

DN cn=SSL, cn=Configuration

Description

Global SSL connection settings for Directory Server.

Number

0 or 1 (optional)

Object Class

ibm-slapdSSL

Mandatory Attributes

- cn
- ibm-slapdSecurity
- ibm-slapdSecurePort
- ibm-slapdSslAuth
- objectClass

Optional Attributes

- ibm-slapdSslCertificate
- ibm-slapdSslCipherSpec

Note: `ibm-slapdSslCipherSpecs` is now deprecated. Use `ibm-slapdSslCipherSpec` instead. If you use `ibm-slapdSslCipherSpecs`, the server will convert to the supported attribute.

- ibm-slapdSslKeyDatabase
- ibm-slapdSslKeyDatabasePW

cn=CRL

DN cn=CRL, cn=SSL, cn=Configuration

Description

This entry contains certificate revocation list data from the first section (global stanza) of `ibmslapd.conf`. It is only needed if `"ibm-slapdSslAuth = serverclientauth"` in the `cn=SSL` entry and the client certificates have been issued for CRL validation.

Number

0 or 1 (optional)

Object Class

ibm-slapdCRL

Mandatory Attributes

- cn
- ibm-slapdLdapCrlHost
- ibm-slapdLdapCrlPort
- objectClass

Optional Attributes

- ibm-slapdLdapCrlUser
- ibm-slapdLdapCrlPassword

cn=Transaction

DN cn = Transaction, cn = Configuration

Description

Specifies Global transaction support settings. Transaction support is provided using the plugin:

extendedop /QSYS.LIB/QGLDTRANEX.SRVPGM tranExtOpInit 1.3.18.0.2.12.5
1.3.18.0.2.12.6

The server (**slapd**) loads this plugin automatically at startup if **ibm-slapdTransactionEnable = TRUE**. The plugin does not need to be explicitly added to **ibmslapd.conf**.

Number

0 or 1 (optional; required only if you want to use transactions.)

Object Class

ibm-slapdTransaction

Mandatory Attributes

- cn
- ibm-slapdMaxNumOfTransactions
- ibm-slapdMaxOpPerTransaction
- ibm-slapdMaxTimeLimitOfTransactions
- ibm-slapdTransactionEnable
- objectClass

Optional Attributes

- None

Attributes

- cn
- ibm-slapdACIMechanism
- ibm-slapdACLAccess
- ibm-slapdACLCache
- ibm-slapdACLCacheSize
- ibm-slapdAdminDN
- ibm-slapdAdminPW
- ibm-slapdBulkloadErrors
- ibm-slapdChangeLogMaxEntries
- ibm-slapdCLIErrors
- ibm-slapdConcurrentRW
- ibm-slapdDB2CP
- ibm-slapdDBAlias

- ibm-slapdDbConnections
- ibm-slapdDbInstance
- ibm-slapdDbLocation
- ibm-slapdDbName
- ibm-slapdDbUserID
- ibm-slapdDbUserPW
- ibm-slapdEnableEventNotification
- ibm-slapdEntryCacheSize
- ibm-slapdErrorLog
- ibm-slapdFilterCacheBypassLimit
- ibm-slapdFilterCacheSize
- ibm-slapdIdleTimeOut
- ibm-slapdIncludeSchema
- ibm-slapdKrbAdminDN
- ibm-slapdKrbEnable
- ibm-slapdKrbIdentityMap
- ibm-slapdKrbKeyTab
- ibm-slapdKrbRealm
- ibm-slapdLdapCrlHost
- ibm-slapdLdapCrlPassword
- ibm-slapdLdapCrlPort
- ibm-slapdLdapCrlUser
- ibm-slapdMasterDN
- ibm-slapdMasterPW
- ibm-slapdMasterReferral
- ibm-slapdMaxEventsPerConnection
- ibm-slapdMaxEventsTotal
- ibm-slapdMaxNumOfTransactions
- ibm-slapdMaxOpPerTransaction
- ibm-slapdMaxPendingChangesDisplayed
- ibm-slapdMaxTimeLimitOfTransactions
- ibm-slapdPagedResAllowNonAdmin
- ibm-slapdPagedResLmt
- ibm-slapdPageSizeLmt
- ibm-slapdPlugin
- ibm-slapdPort
- ibm-slapdPwEncryption
- ibm-slapdReadOnly
- ibm-slapdReferral
- ibm-slapdReplDbConns
- ibm-slapdReplicaSubtree
- ibm-slapdSchemaAdditions
- ibm-slapdSchemaCheck
- ibm-slapdSecurePort
- ibm-slapdSecurity

- `ibm-slapdServerId`
- `ibm-slapdSetenv`
- `ibm-slapdSizeLimit`
- `ibm-slapdSortKeyLimit`
- `ibm-slapdSortSrchAllowNonAdmin`
- `ibm-slapdSslAuth`
- `ibm-slapdSslCertificate`
- `ibm-slapdSslCipherSpec`
- `ibm-slapdSslKeyDatabase`
- `ibm-slapdSslKeyDatabasePW`
- `ibm-slapdSslKeyRingFile`
- `ibm-slapdSuffix`
- `ibm-slapdSupportedWebAdmVersion`
- `ibm-slapdSysLogLevel`
- `ibm-slapdTimeLimit`
- `ibm-slapdTransactionEnable`
- `ibm-slapdUseProcessIdPw`
- `ibm-slapdVersion`
- `objectClass`

cn

Description

This is the X.500 common Name attribute, which contains a name of an object.

Syntax

Directory string

Maximum Length

256

Value Multi-valued

ibm-slapdACIMechanism

Description

Determines which ACL model the server uses. (Supported only on i5/OS as of v3.2, ignored on other platforms.)

- 1.3.18.0.2.26.1 = IBM SecureWay® v3.1 ACL model
- 1.3.18.0.2.26.2 = IBM SecureWay v3.2 ACL model

Default

1.3.18.0.2.26.2 = IBM SecureWay v3.2 ACL model

Syntax

Directory string

Maximum Length

256

Value Multi-valued.

ibm-slapdACLAccess

Description

Controls whether access to ACLs is enabled. If set to TRUE, access to ACLs is enabled. If set to FALSE, access to ACLs is disabled.

Default

TRUE

Syntax

Boolean

Maximum Length

5

Value Single-valued

ibm-slapdACLCache**Description**

Controls whether or not the server caches ACL information.

- If set to TRUE, the server caches ACL information.
- If set to FALSE, the server does not cache ACL information.

Default

TRUE

Syntax

Boolean

Maximum Length

5

Value Single-valued

ibm-slapdACLCacheSize**Description**

Maximum number of entries to keep in the ACL Cache.

Default

25000

Syntax

Integer

Maximum Length

11

Value Single-valued

ibm-slapdAdminDN**Description**

The administrator bind DN for Directory Server.

Default

cn=root

Syntax

DN

Maximum Length

Unlimited

Value Single-valued

ibm-slapdAdminPW

Description

The administrator bind Password for Directory Server.

Default

secret

Syntax

Binary

Maximum Length

128

Value Single-valued

ibm-slapdBulkloadErrors

Description

File path or device on ibmslapd host machine to which bulkload error messages will be written.

Default

/var/bulkload.log

Syntax

Directory string with case-exact matching

Maximum Length

1024

Value Single-valued

ibm-slapdChangeLogMaxEntries

Description

This attribute is used by a changelog plug-in to specify the maximum number of changelog entries allowed in the RDBM database. Each changelog has its own changeLogMaxEntries attribute.

Minimum = 0 (unlimited)

Maximum = 2,147,483,647 (32-bit, signed integer)

Default

0

Syntax

Integer

Maximum Length

11

Value Single-valued

ibm-slapdCLIErrors

Description

File path or device on ibmslapd host machine to which CLI error messages will be written.

Default

/var/db2cli.log

Syntax

Directory string with case-exact matching

Maximum Length

1024

Value Single-valued**ibm-slapdConcurrentRW****Description**

Setting this to TRUE allows searches to proceed simultaneously with updates. It allows for 'dirty reads', that is, results that might not be consistent with the committed state of the database.

Attention: This attribute is deprecated.

Default

FALSE

Syntax

Boolean

Maximum Length

5

Value Single-valued**ibm-slapdDB2CP****Description**

Specifies the code page of the directory database. 1208 is the code page for UTF-8 databases.

Syntax

Directory string with case-exact matching

Maximum Length

11

Value Single-valued**ibm-slapdDBAlias****Description**

The DB2 database alias.

Syntax

Directory string with case-exact matching

Maximum Length

8

Value Single-valued**ibm-slapdDbConnections****Description**

Specify the number of DB2 connections the server will dedicate to the DB2 backend. The value must be between 5 & 50 (inclusive).

Note: ODBCCONS environment variable overrides the value of this directive.

If `ibm-slapdDbConnections` (or `ODBCCONS`) is less than 5 or greater than 50, the server will use 5 or 50 respectively. 1 additional connection will be created for replication (even if no replication is defined). 2 additional connections will be created for the change log (if change log is enabled).

Default
15

Syntax
Integer

Maximum Length
50

Value Single-valued

ibm-slapdDbInstance

Description
Specifies the DB2 database instance for this backend.

Default
ldapdb2

Syntax
Directory string with case-exact matching

Maximum Length
8

Value Single-valued

Note: All `ibm-slapdRdbmBackend` objects must use the same `ibm-slapdDbInstance`, `ibm-slapdDbUserID`, `ibm-slapdDbUserPW` and DB2 character set.

ibm-slapdDbLocation

Description
The file system path where the backend database is located.

Syntax
Directory string with case-exact matching

Maximum Length
1024

Value Single-valued

ibm-slapdDbName

Description
Specifies the DB2 database name for this backend.

Default
ldapdb2

Syntax
Directory string with case-exact matching

Maximum Length
8

Value Single-valued

ibm-slapdDbUserID

Description
Specifies the user name with which to bind to the DB2 database for this backend.

Default

ldapdb2

Syntax

Directory string with case-exact matching

Maximum Length

8

Value Single-valued

Note: All `ibm-slapdRdbmBackend` objects must use the same `ibm-slapdDbInstance`, `ibm-slapdDbUserID`, `ibm-slapdDbUserPW` and DB2 character set.

ibm-slapdDbUserPW**Description**

Specifies the user password with which to bind to the DB2 database for this backend. The password can be plain text or imask encrypted.

Default

ldapdb2

Syntax

Binary

Maximum Length

128

Value Single-valued

Note: All `ibm-slapdRdbmBackend` objects must use the same `ibm-slapdDbInstance`, `ibm-slapdDbUserID`, `ibm-slapdDbUserPW` and DB2 character set.

ibm-slapdEnableEventNotification**Description**

Specifies whether to enable Event Notification. It must be set to either TRUE or FALSE.

If set to FALSE, the server rejects all client requests to register event notifications with the extended result LDAP_UNWILLING_TO_PERFORM.

Default

TRUE

Syntax

Boolean

Maximum Length

5

Value Single-valued

ibm-slapdEntryCacheSize**Description**

Maximum number of entries to keep in the entry cache.

Default

25000

Syntax

Integer

Maximum Length

11

Value Single-valued**ibm-slapdErrorLog****Description**

Specifies the file path or device on the Directory Server machine to which error messages are written.

Default

/var/ibmslapd.log

Syntax

Directory string with case-exact matching

Maximum Length

1024

Value Single-valued**ibm-slapdFilterCacheBypassLimit****Description**

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.

Default

100

Syntax

Integer

Maximum Length

11

Value Single-valued**ibm-slapdFilterCacheSize****Description**

Specifies the maximum number of entries to keep in the Search Filter Cache.

Default

25000

Syntax

Integer

Maximum Length

11

Value Single-valued**ibm-slapdIdleTimeOut****Description**

Maximum time to keep an LDAP connection open when there is no activity on the connection. The idle time for an LDAP connection is the time (in seconds) between the last activity on the connection and the current time. If the connection has expired, based on the idle time being greater than the value of this attribute, the LDAP server will clean up and end the LDAP connection, making it available for other incoming requests.

Default 300
Syntax Integer
Length 11
Count Single
Usage Directory operation
User Modify Yes
Access Class Critical
Required No

ibm-slapdIncludeSchema

Description
Specifies a file path on the Directory Server server machine containing schema definitions.

Default
/etc/V3.system.at
/etc/V3.system.oc
/etc/V3.config.at
/etc/V3.config.oc
/etc/V3.ibm.at
/etc/V3.ibm.oc
/etc/V3.user.at
/etc/V3.user.oc
/etc/V3.ldapsyntaxes
/etc/V3.matchingrules

Syntax
Directory string with case-exact matching

Maximum Length
1024

Value Multi-valued

ibm-slapdKrbAdminDN

Description
Specifies the Kerberos ID of the LDAP administrator (for example, ibm-kn=admin1@realm1). Used when Kerberos authentication is used to authenticate the administrator when logged onto the Server Administration interface. This might be specified instead of or in addition to adminDN and adminPW.

Default
No preset default is defined.

Syntax
Directory string with case-exact matching

Maximum Length

128

Value Single-valued**ibm-slapdKrbEnable****Description**

Specifies whether the server supports Kerberos. It must be either TRUE or FALSE.

Default

TRUE

Syntax

Boolean

Maximum Length

5

Value Single-valued**ibm-slapdKrbIdentityMap****Description**

Specifies whether to use Kerberos identity mapping. It must be set to either TRUE or FALSE. If set to TRUE, when a client is authenticated with a Kerberos ID, the server searches for all local users with matching Kerberos credentials, and adds those user DNs to the bind credentials of the connection. This allows ACLs based on LDAP user DNs to still be usable with Kerberos.

Default

FALSE

Syntax

Boolean

Maximum Length

5

Value Single-valued**ibm-slapdKrbKeyTab****Description**

Specifies the LDAP server Kerberos keytab file. This file contains the LDAP server private key, that is associated with its Kerberos account. This file is to be protected (like the server SSL key database file).

Default

No preset default is defined.

Syntax

Directory string with case-exact matching

Maximum Length

1024

Value Single-valued**ibm-slapdKrbRealm****Description**

Specifies the Kerberos realm of the LDAP server. It is 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 kerberized server, can only be a member of a single realm.

Default

No preset default is defined.

Syntax

Directory string with case-insensitive matching

Maximum Length

256

Value Single-valued

ibm-slapdLdapCrlHost

Description

Specifies the host name 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.

Default

No preset default is defined.

Syntax

Directory string with case-insensitive matching

Maximum Length

256

Value Single-valued

ibm-slapdLdapCrlPassword

Description

Specifies the password that server-side SSL uses to bind to the LDAP server that contains the Certificate Revocation Lists (CRLs) for validating client x.509v3 certificates. This parameter might 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 (that is, anonymous access), then `ibm-slapdLdapCrlPassword` is not required.

Default

No preset default is defined.

Syntax

Binary

Maximum Length

128

Value Single-valued

ibm-slapdLdapCrlPort

Description

Specifies the port used to connect to 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)

Default
No preset default is defined.

Syntax
Integer

Maximum Length
11

Value Single-valued

ibm-slapdLdapCrlUser

Description
Specifies the bindDN that the server-side SSL uses to bind to the LDAP server that contains the Certificate Revocation Lists (CRLs) for validating client x.509v3 certificates. This parameter might 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 (that is, anonymous access), then `ibm-slapdLdapCrlUser` is not required.

Default
No preset default is defined.

Syntax
DN

Maximum Length
1000

Value Single-valued

ibm-slapdMasterDN

Description
Specifies the bind DN of master server. The value must match the `replicaBindDN` in the `replicaObject` defined for the master server. When Kerberos is used to authenticate to the replica, `ibm-slapdMasterDN` must specify the DN representation of the Kerberos ID (for example, `ibm-kn=freddy@realm1`). When Kerberos is used, `MasterServerPW` is ignored.

Default
No preset default is defined.

Syntax
DN

Maximum Length
1000

Value Single-valued

ibm-slapdMasterPW

Description
Specifies the bind password of master replica server. The value must match `replicaBindDN` in the `replicaObject` defined for the master server. When Kerberos is used to authenticate to the replica, `ibm-slapdMasterDN` must specify the DN representation of the Kerberos ID (for example, `ibm-kn=freddy@realm1`). When Kerberos is used, `MasterServerPW` is ignored.

Default
No preset default is defined.

Syntax

Binary

Maximum Length

128

Value Single-valued

ibm-slapedMasterReferral**Description**

Specifies the URL of the master replica server. For example:

`ldap://master.us.ibm.com`

For security set to SSL only:

`ldaps://master.us.ibm.com:636`

For security set to none and using a nonstandard port:

`ldap://master.us.ibm.com:1389`

Default

none

Syntax

Directory string with case-insensitive matching

Maximum Length

256

Value Single-valued

ibm-slapedMaxEventsPerConnection**Description**

Specifies the maximum number of event notifications which can be registered per connection.

Minimum = 0 (unlimited)

Maximum = 2,147,483,647

Default

100

Syntax

Integer

Maximum Length

11

Value Single-valued

ibm-slapedMaxEventsTotal**Description**

Specifies the maximum total number of event notifications which can be registered for all connections.

Minimum = 0 (unlimited)

Maximum = 2,147,483,647

Default

0

Syntax

Integer

Maximum Length

11

Value Single-valued**ibm-slapdMaxNumOfTransactions****Description**

Specifies the maximum number of transactions per server.

Minimum = 0 (unlimited)

Maximum = 2,147,483,647

Default

20

Syntax

Integer

Maximum Length

11

Value Single-valued**ibm-slapdMaxOpPerTransaction****Description**

Specifies the maximum number of operations per transaction.

Minimum = 0 (unlimited)

Maximum = 2,147,483,647

Default

5

Syntax

Integer

Maximum Length

11

Value Single-valued**ibm-slapdMaxPendingChangesDisplayed****Description**

Maximum number of pending changes to be displayed.

Default

200

Syntax

Integer

Maximum Length

11

Value Single-valued**ibm-slapdMaxTimeLimitOfTransactions****Description**

Specifies the maximum timeout value of a pending transaction in seconds.

Minimum = 0 (unlimited)

Maximum = 2,147,483,647

Default
300

Syntax
Integer

Maximum Length
11

Value Single-valued

ibm-slapdPagedResAllowNonAdmin

Description

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 `FALSE`, the server will process only those client requests submitted by a user with Administrator authority. If a client requests paged results 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.

Default

`FALSE`

Syntax

Boolean

Length

5

Count Single

Usage directoryOperation

User Modify

Yes

Access Class

critical

Objectclass

`ibm-slapdRdbmBackend`

Required

No

ibm-slapdPagedResLmt

Description

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.

Default

3

Syntax

Integer

Length

11

Count Single

Usage directoryOperation

User Modify
Yes

Access Class
critical

Required
No

Objectclass
ibm-slapdRdbmBackend

ibm-slapdPageSizeLmt

Description

Maximum number of entries to return from search for an individual page when paged results control is specified, regardless of any pagesize that might 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.

Default
50

Syntax
Integer

Length
11

Count Single

Usage directoryOperation

User Modify
Yes

Access Class
critical

Required
No

Objectclass
ibm-slapdRdbmBackend

ibm-slapdPlugin

Description

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 plug-in library. The syntax is:

```
keyword filename init_function [args...]
```

The syntax is slightly different for each platform because of library naming conventions.

Most plug-ins are optional, but the RDBM backend plug-in is required for all RDBM backends.

Default
database /bin/libback-rdbm.dll rdbm_backend_init

Syntax
Directory string with case-exact matching

Maximum Length
2000

Value Multi-valued

ibm-slapdPort

Description
Specifies the TCP/IP port used for non-SSL connections. It cannot have the same value as `ibm-slapdSecurePort`. (IP ports are unsigned, 16-bit integers in the range 1 - 65535.)

Default
389

Syntax
Integer

Maximum Length
5

Value Single-valued

ibm-slapdPWEncryption

Description
Specifies the encoding mechanism for the user passwords before they are stored in the directory. It must be specified as `none`, `imask`, `crypt`, or `sha` (you must use the keyword **sha** in order to get SHA-1 encoding). The value must be set to `none` for the SASL `cram-md5` bind to succeed.

Default
`none`

Syntax
Directory string with case-insensitive matching

Maximum Length
5

Value Single-valued

ibm-slapdReadOnly

Description
This attribute is normally applied to only the Directory backend. It specifies whether the backend can be written to. It must be specified as either `TRUE` or `FALSE`. It defaults to `FALSE` if unspecified. If set to `TRUE`, the server returns `LDAP_UNWILLING_TO_PERFORM` (0x35) in response to any client request which changes data in the `readOnly` database.

Default
`FALSE`

Syntax
Boolean

Maximum Length
5

Value Single-valued

ibm-slapdReferral

Description

Specifies the referral LDAP URL to pass back when the local suffixes do not match the request. It is used for superior referral (that is, the suffix is not within the naming context of the server).

Default

No preset default is defined.

Syntax

Directory string with case-exact matching

Maximum Length

32700

Value Multi-valued

ibm-slapdReplDbConns**Description**

Maximum number of database connections for use by replication.

Default

4

Syntax

Integer

Maximum Length

11

Value Single-valued

ibm-slapdReplicaSubtree**Description**

Identifies the DN of a replicated subtree

Syntax

DN

Maximum Length

1000

Value Single-valued

ibm-slapdSchemaAdditions**Description**

The `ibm-slapdSchemaAdditions` attribute is used to identify explicitly which file holds new schema entries. This is set by default to be `/etc/V3.modifiedschema`. If this attribute is not defined, the server reverts to using the last `ibm-slapdIncludeSchema` file as in previous releases.

Before Version 3.2, the last `includeSchema` entry in **`slapd.conf`** was the file to which any new schema entries were added by the server if it received an add request from a client. Normally the last `includeSchema` is the `V3.modifiedschema` file, which is an empty file installed just for this purpose.

Note: The name modified is misleading, for it only stores new entries. Changes to existing schema entries are made in their original files.

Default

`/etc/V3.modifiedschema`

Syntax
Directory string with case-exact matching

Maximum Length
1024

Value Single-valued

ibm-slapdSchemaCheck

Description
Specifies the schema checking mechanism for the add/modify/delete operation. It must be specified as V2, V3, or V3_lenient.

- V2 - Retain v2 and v2.1 checking. Recommended for migration purpose.
- V3 - Perform v3 checking.
- V3_lenient - Not all parent object classes are needed. Only the immediate object class is needed when adding entries.

Default
V3_lenient

Syntax
Directory string with case-insensitive matching

Maximum Length
10

Value Single-valued

ibm-slapdSecurePort

Description
Specifies the TCP/IP port used for SSL connections. It cannot have the same value as ibm-slapdPort. (IP ports are unsigned, 16-bit integers in the range 1 - 65535.)

Default
636

Syntax
Integer

Maximum Length
5

Value Single-valued

ibm-slapdSecurity

Description
Enables SSL connections. Must be none, SSL, or SSLOnly.

- none - server listens on the non-ssl port only.
- SSL - server listens on both the ssl and the non-ssl ports.
- SSLOnly - server listens on the ssl port only.

Default
none

Syntax
Directory string with case-insensitive matching

Maximum Length
7

Value Single-valued

ibm-slapdServerId

Description

Identifies the server for use in replication.

Syntax

IA5 String with case-sensitive matching

Maximum Length

240

Value Single-valued

ibm-slapdSetenv

Description

The server runs **putenv()** for all values of **ibm-slapdSetenv** at startup to modify the server runtime environment. Shell variables (like **%PATH%** or **\$LANG**) are not expanded.

Default

No preset default is defined.

Syntax

Directory string with case-exact matching

Maximum Length

2000

Value Multi-valued

ibm-slapdSizeLimit

Description

Specifies the maximum number of entries to return from search, regardless of any size limit that might have been specified on the client search request (Range = 0...). If a client has passed a limit, then the smaller value of the client values and the value read from **ibmslapd.conf** are used. If a client has not passed a limit and has bound as admin DN, the limit is considered unlimited. If the client has not passed a limit and has not bound as admin DN, then the limit is that which was read from the **ibmslapd.conf** file. 0 = unlimited.

Default

500

Syntax

Integer

Maximum Length

12

Value Single-valued

ibm-slapdSortKeyLimit

Description

The 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 the **ibmslapd.conf** file 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 a

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 a return code of **adminLimitExceeded** - no searching or sorting will be performed.

Default

3

Syntax

cis

Length

11

Count Single

Usage directoryOperation

User Modify

Yes

Access Class

critical

Objectclass

ibm-slapdRdbmBackend

Required

No

ibm-slapdSortSrchAllowNonAdmin

Description

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 FALSE, the server will process only those client requests submitted by a user with Administrator authority. If a client requests sort 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.

Default

FALSE

Syntax

Boolean

Length

5

Count Single

Usage directoryOperation

User Modify

Yes

Access Class

critical

Objectclass

ibm-slapdRdbmBackend

Required

No

ibm-slapdSslAuth

Description

Specifies the authentication type for the ssl connection, either serverauth or serverclientauth.

- serverauth - supports server authentication at the client. This is the default.
- serverclientauth - supports both server and client authentication.

Default

serverauth

Syntax

Directory string with case-insensitive matching

Maximum Length

16

Value Single-valued

ibm-slapedSslCertificate**Description**

Specifies the label that identifies the server Personal Certificate in the key database file. This label is specified when the server private key and certificate are created with the **gsk4ikm** application. If ibm-slapedSslCertificate is not defined, the default private key, as defined in the key database file, is used by the LDAP server for SSL connections.

Default

No preset default is defined.

Syntax

Directory string with case-exact matching

Maximum Length

128

Value Single-valued

ibm-slapedSslCipherSpec

Specifies the method of SSL encryption for clients accessing the server. Must be set to one of the following:

Table 5. Methods of SSL encryption

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

Syntax

IA5 String

Maximum Length

30

ibm-slapedSslKeyDatabase

Description

Specifies the file path to the LDAP server 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.

Default

/etc/key.kdb

Syntax

Directory string with case-exact matching

Maximum Length

1024

Value Single-valued

ibm-slapdSslKeyDatabasePW**Description**

Specifies the password associated with the LDAP server SSL key database file, as specified on the `ibm-slapdSslKeyDatabase` parameter. If the LDAP server key database file has an associated password stash file, then the `ibm-slapdSslKeyDatabasePW` parameter can be omitted, or set to none.

Note: 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`.

Default

none

Syntax

Binary

Maximum Length

128

Value Single-valued

ibm-slapdSslKeyRingFile**Description**

Path to the LDAP server's 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.

Default

key.kdb

Syntax

Directory String with case-sensitive matching

Maximum Length

1024

Value Single-valued

ibm-slapdSuffix**Description**

Specifies a naming context to be stored in this backend.

Note: This has the same name as the object class.

Default
No preset default is defined.

Syntax
DN

Maximum Length
1000

Value Multi-valued

ibm-slapdSupportedWebAdmVersion

Description
This attribute defines the earliest version of the Web administration tool that supports this server of cn=configuration.

Default

Syntax
Directory String

Maximum Length

Value Single-valued

ibm-slapdSysLogLevel

Description
Specifies the level at which debugging and operation statistics are logged in the slapd.errors file. It must be specified as l, m, or h.

- h - high (provides the most information)
- m - medium (the default)
- l - low (provides the least information)

Default
m

Syntax
Directory string with case-insensitive matching

Maximum Length
1

Value Single-valued

ibm-slapdTimeLimit

Description
Specifies the maximum number of seconds to spend on a search request, regardless of any time limit that might have been specified on the client request. If a client has passed a limit, then the smaller value of the client values and the value read from **ibmslapd.conf** are used. If a client has not passed a limit and has bound as admin DN, the limit is considered unlimited. If the client has not passed a limit and has not bound as admin DN, then the limit is that which was read from the **ibmslapd.conf** file. 0 = unlimited.

Default
900

Syntax
Integer

Maximum Length

Value Single-valued

ibm-slapdTransactionEnable

Description

If the transaction plugin is loaded but `ibm-slapdTransactionEnable` is set to `FALSE`, the server rejects all `StartTransaction` requests with the response `LDAP_UNWILLING_TO_PERFORM`.

Default

TRUE

Syntax

Boolean

Maximum Length

5

Value Single-valued

ibm-slapdUseProcessIdPw

Description

If set to `TRUE`, the server ignores the `ibm-slapdDbUserID` and the `ibm-slapdDbUserPW` attributes and uses its own process credentials to authenticate to DB2.

Default

FALSE

Syntax

Boolean

Maximum Length

5

Value Single-valued

ibm-slapdVersion

Description

IBM Slapd version Number

Default

Syntax

Directory String with case-sensitive matching

Maximum Length

Value Single-valued

objectClass

Description

The values of the `objectClass` attribute describe the kind of object which an entry represents.

Syntax

Directory string

Maximum Length




128

Value Multi-valued


Chapter 10. Related information

Listed below are the IBM Redbooks (in PDF format), Web sites, and Information Center topics that relate to the Directory Server topic. You can view or print any of the PDFs.

Redbooks (www.redbooks.ibm.com)

- *Understanding LDAP*, SG24-4986  .
- *Using LDAP for Directory Integration: A Look at IBM SecureWay Directory, Active Directory, and Domino*, SG24-6163  .
- *Implementation and Practical Use of LDAP on the iSeries Server*, SG24-6193  .

Web sites

- IBM Directory Server for iSeries Web site (www.ibm.com/servers/eserver/series/ldap) 
- The Java Naming and Directory Interface (JNDI) Tutorial Web site (java.sun.com/products/jndi/tutorial/) 

Other information

“Directory Server APIs” in the Programming topic.

Appendix. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

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

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

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation

Software Interoperability Coordinator, Department 49XA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

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

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

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

COPYRIGHT LICENSE:

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

SUBJECT TO ANY STATUTORY WARRANTIES WHICH CANNOT BE EXCLUDED, IBM, ITS PROGRAM DEVELOPERS AND SUPPLIERS MAKE NO WARRANTIES OR CONDITIONS EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT, REGARDING THE PROGRAM OR TECHNICAL SUPPORT, IF ANY.

UNDER NO CIRCUMSTANCES IS IBM, ITS PROGRAM DEVELOPERS OR SUPPLIERS LIABLE FOR ANY OF THE FOLLOWING, EVEN IF INFORMED OF THEIR POSSIBILITY:

1. LOSS OF, OR DAMAGE TO, DATA;
2. SPECIAL, INCIDENTAL, OR INDIRECT DAMAGES, OR FOR ANY ECONOMIC CONSEQUENTIAL DAMAGES; OR
3. LOST PROFITS, BUSINESS, REVENUE, GOODWILL, OR ANTICIPATED SAVINGS.

SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO SOME OR ALL OF THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

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

AIX
AIX 5L
e(logo)server
eServer
i5/OS
IBM
iSeries
pSeries
xSeries
zSeries

Intel, Intel Inside (logos), MMX, and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

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

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

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

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

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

Terms and conditions for downloading and printing information

Permissions for the use of the information you have selected for download are granted subject to the following terms and conditions and your indication of acceptance thereof.

Personal Use: You may reproduce this information for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative works of this information, or any portion thereof, without the express consent of IBM.

Commercial Use: You may reproduce, distribute and display this information solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of this information, or reproduce, distribute or display this information or any portion thereof outside your enterprise, without the express consent of IBM.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the information or any data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the information is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations. IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THIS INFORMATION. THE INFORMATION IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

All material copyrighted by IBM Corporation.

By downloading or printing information from this site, you have indicated your agreement with these terms and conditions.



Printed in USA