Sun OpenDS Standard Edition 2.0 Administration Guide

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 820–6169
July 2009

090717@22510

Contents

Configuring the Directory Server

The easiest way to access the server configuration is by using the `dsconfig` command. Basic server configuration can also be performed with the Control Panel. The topics in this section describe how to configure the server using both of these methods.

This section covers the following topics:

Managing Administration Traffic to the Server

The directory server includes a special connection handler, the administration connector, to manage administration traffic to the server. The administration connector enables the separation of *user* traffic and *administration* traffic to simplify monitoring, and to ensure that administrative commands take precedence over commands that manipulate user data.

Overview of the Administration Connector

The administration connector is based on the LDAP protocol and uses LDAP over SSL by default. All command-line utilities that access the administrative suffixes use the administration connector. This includes the following commands:

- `backup`
- `dsconfig`
- `dsreplication`
- `export-ldif`
- `import-ldif`
- `manage-account`
- `manage-tasks`

- `restore`
- `status`
- `stop-ds`
- `uninstall`

The administration connector is always present and enabled. You cannot disable or delete the connector using `dsconfig`, however, you can use `dsconfig` to manipulate the following properties of the connector:

- `listen-address`. The address on which the directory server listens for administration traffic.
- `listen-port`. The default port of the administration connector is 4444. You can change this port during setup if required. If you use the default port, you do not need to specify a port when running the administration commands (the default port is assumed). If you change the port, you must specify the new port when running the administration commands.
- **Security-related properties.** Traffic using the administration connector is always secured. As with the LDAPS connection handler, the administration connector is configured with a self-signed certificate during server setup. This self-signed certificate is generated the first time the server is started. You can manage the administration connector certificate using external tools, such as `keytool`.

  The security-related properties include the following:

  - `ssl-cert-nickname`
  - `key-manager-provider`
  - `trust-manager-provider`

  When you run the administration commands, you are prompted as to how you want to trust the certificate. If you run the administration commands in non-interactive mode, you must specify the `-X` or `--trustAll` option to trust the certificate, otherwise the command will fail.

Accessing Administrative Suffixes

The directory server *administrative suffixes* include the following:

- `cn=config`
- `cn=monitor`
- `cn=tasks`
- `cn=backups`
- `cn=ads-truststore`
- `cn=schema`
- `cn=admin data`

In general, direct LDAP access to the administrative suffixes (using the `ldap*` utilities) is discouraged, with the exception of the `cn=monitor` suffix. In most cases, it is preferable to use the dedicated administrative command-line utilities to access these suffixes.

If you must use the ldap* commands to access the administrative suffixes, you should use the administration connector port (with the --useSSL or -Z option). Using the administration connector ensures that monitoring data is not polluted and that server administration takes precedence over user traffic. The same recommendations apply if you are accessing the administrative suffixes using an LDAP browser.

▼ To Configure the Administration Connector

This example displays the default properties of the administration connector, and changes the listen port of the connector to 5555.

1 View the default properties of the administration connector, using the dsconfig command.

```
$ dsconfig -D "cn=directory manager" -w password -n \
get-administration-connector-prop
Property               : Value(s)
-----------------------:---------------
key-manager-provider   : Administration
listen-address         : 0.0.0.0
listen-port            : 4444
ssl-cert-nickname      : admin-cert
trust-manager-provider : Administration
```

2 Change the listen port, using the dsconfig command.

```
$ dsconfig -D "cn=directory manager" -w password -n \
 set-administration-connector-prop --set listen-port:5555
```

Note – You must restart the server for changes to this property to take effect.

Configuring the Directory Server With dsconfig

The topics in this section are intended for administrators or users who want to configure and manage a directory server instance. These topics provide an overview of the dsconfig command-line utility and its use in server configuration. For more information, see "dsconfig" in *Sun OpenDS Standard Edition 2.0 Command-Line Usage Guide*.

Overview of the dsconfig Command

The dsconfig command-line utility provides a simple mechanism for accessing the directory server configuration. dsconfig presents the server configuration as a set of components, each of which can be managed through one or more subcommands.

dsconfig can also be used interactively. In interactive mode, dsconfig functions much like a wizard, walking you through the server configuration. For more information, see "Using dsconfig in Interactive Mode" on page 11.

Note –

- dsconfig can only be used to configure a *running* directory server instance. Offline configuration is not supported by dsconfig.
- Like the other administration commands, dsconfig uses the administration connector to access the server. For more information, see "Managing Administration Traffic to the Server" on page 5. All of the examples in this section assume that the administration connector is listening on the default port (4444) and that the command is accessing the server running on the local host. If this is not the case, the --port and --hostname options must be specified.

---

dsconfig and Certificate Checking

dsconfig accesses the server over a secured connection with certificate authentication. If you run dsconfig in interactive mode, you are prompted as to how you want to trust the certificate.

If you run dsconfig in non-interactive mode (that is, with the -n option), specification of the trust store parameters depends on whether you run the command locally or remotely.

- **Running dsconfig locally.** (The command is launched on the server that you are administering.) If you do not specify the trust store parameters, the server uses the local instance trust store by default. Unless you specify otherwise, the local instance trust is *install-dir*/OpenDS-*version*/config/admin-truststore.
- **Running dsconfig remotely.** (The command is launched on a different server to the one you are administering.) You *must* specify the trust store parameters or the -X (--trustAll) option. The easiest way to specify the trust store parameters is to run the command once in interactive mode and to save the certificate that is presented by the server in your trust store.

```
$ dsconfig

>>>> >>>> Specify OpenDS LDAP connection parameters

Directory server hostname or IP address [host1.example.com]:
Directory server administration port number [4444]:

How do you want to trust the server certificate?

    1)  Automatically trust
    2)  Use a truststore
    3)  Manually validate
```

```
Enter choice [3]: 3

Administrator user bind DN [cn=Directory Manager]:

Password for user 'cn=Directory Manager':

Server Certificate:

User DN  : CN=host1.example.com, O=Administration Connector Self-Signed Certificate
Validity : From 'Wed Apr 29 11:13:21 MEST 2009'
            To 'Fri Apr 29 11:13:21 MEST 2011'
Issuer   : CN=host1.example.com, O=Administration Connector Self-Signed Certificate


Do you trust this server certificate?

    1)  No
    2)  Yes, for this session only
    3)  Yes, also add it to a truststore
    4)  View certificate details

Enter choice [2]: 3

Truststore path: /local/instances/certificates/jctruststore

Password for keystore '/local/instances/certificates/jctruststore':

...
```

When you have saved the certificate in the trust store, you can specify those trust store parameters in non-interactive mode.

```
$ dsconfig list-connection-handlers -n --trustStorePath
/local/instances/certificates/jctruststore --trustStorePasswordFile
/local/instances/certificates/jctruststore.pin -w password
Connection Handler       : Type : enabled : listen-port : use-ssl
-------------------------:------:---------:-------------:--------
JMX Connection Handler   : jmx  : false   : 1689        : false
LDAP Connection Handler  : ldap : true    : 1389        : false
LDAPS Connection Handler : ldap : false   : 636         : true
LDIF Connection Handler  : ldif : false   : -
```

dsconfig Subcommands

dsconfig provides an intuitive list of subcommands to manage various elements of the configuration.

For example, the following five subcommands are used to manage connection handlers:

Subcommand	Function
`dsconfig create-connection-handler` *options*	Creates connection handlers
`dsconfig delete-connection-handler` *options*	Deletes connection handlers
`dsconfig get-connection-handler-prop` *options*	Displays the properties of a connection handler
`dsconfig list-connection-handlers` *options*	Lists the existing defined connection handlers
`dsconfig set-connection-handler-prop` *options*	Modifies the properties of a connection handler

Using these subcommands, you can add, delete, list, view, and modify connection handlers. The `dsconfig` command presents similar subcommands for other components, which follow similar naming conventions:

Subcommand	Function
`dsconfig create-`*component options*	Creates a new component
`dsconfig delete-`*component options*	Deletes an existing component
`dsconfig get-`*component*`-prop` *options*	Displays the properties of a component
`dsconfig list-`*component*`s` *options*	Lists the existing defined components
`dsconfig set-`*component*`-prop` *options*	Modifies the properties of a component

Not all types of components can be created and deleted. For example, a directory server has only a single global configuration. For this reason, the global configuration is managed with only two subcommands:

Subcommand	Function
`dsconfig get-global-configuration-prop` *options*	Displays the global configuration properties
`dsconfig set-global-configuration-prop` *options*	Modifies the global configuration properties

The configurable properties of all components can be queried and modified to change the behavior of the component. For example, an LDAP connection has properties that determine its IP listener address, its port, and its SSL configuration.

`dsconfig` Advanced Properties

There are a number of directory server properties that are considered *advanced* properties. The advanced properties are not displayed by default. The advanced properties have default values that apply in most cases. If you want to modify the values or the advanced properties, use `--advanced` before the subcommand. For example:

```
$ dsconfig --advanced get-extension-prop
```

Using dsconfig in Interactive Mode

Unless you specify all configuration parameters and the -n (--no-prompt) option, dsconfig runs in interactive mode. Interactive mode functions like a wizard, walking you through all aspects of the server configuration. interactive mode is a good approach to start using dsconfig.

When you run dsconfig in interactive mode, you can specify that you want the equivalent command (including all your selections) to be displayed, or to be written to a file. The following example shows how to use the --displayCommand option to display the equivalent non-interactive command when configuring the trust manager. Note that the equivalent command is displayed at the point at which the command has been applied and validated by the directory server.

```
$ dsconfig -D "cn=directory manager" -w password --displayCommand
...
The TrustStore Manager Provider was modified successfully

The equivalent non-interactive command-line is:
dsconfig --hostname "localhost" --port "4444" --bindDN "cn=directory
manager" --bindPassword ****** --trustAll
set-trust-manager-provider-prop --provider-name "PKCS12" --set
"enabled:true"
```

To copy the equivalent command to a file, use the --commandFilePath option, as shown in the following example:

```
$ dsconfig -D "cn=directory manager" -w password --commandFilePath /tmp/filename
```

Getting Help With dsconfig

The dsconfig command has extensive online help that is accessed using the --help option.

Global Usage

Use the following command to display dsconfig's global usage:

```
$ dsconfig --help
```

Finding the Correct Subcommand

The global usage information does not include the list of available subcommands. To retrieve the list of subcommands, use one of the --help-*xxx* options, where *xxx* determines the group of subcommands to be displayed.

Note – Use the `--help-all` option used to display all of the available subcommands.

For example, to find all the subcommands relating to caching and back-end configuration, use the following command:

```
$ dsconfig --help-core-server
```

Getting Help for an Individual Subcommand

When you have determined which subcommand you want, you can get more detailed help on that subcommand by using the subcommand's `--help` option as follows:

```
$ dsconfig create-monitor-provider --help
```

Displaying a Summary of a Component's Properties

The `dsconfig` command has built-in documentation for all of the components and their properties. This documentation can be accessed by using the `list-properties` subcommand. For example, a summary of the properties associated with a work queue can be displayed by using the following command:

```
$ dsconfig list-properties -c work-queue
```

Note – If the `-c` option is not specified, a summary of the properties for all components is displayed.

Displaying Detailed Help on a Property

The summary table displays only brief usage information for each property. More detailed information are available using the verbose mode of the `list-properties` subcommand:

```
$ dsconfig list-properties -c work-queue --property num-worker-threads -v
```

Note – If the `--property` option is not specified, verbose help is provided for all the work-queue properties.

Configuring a Directory Server Instance

The `dsconfig` command is the recommended utility for accessing the server configuration. Accessing the configuration directly over LDAP, using the `ldap*` utilities is discouraged.

To Display the Properties of a Component

Each component has one or more properties that can be displayed by using the component's `get-`*`xxx`*`-prop` subcommand. Each component is associated with a single LDAP entry in the server configuration, and each property is associated with a single LDAP attribute.

- **To display the properties of the default LDAP connection handler, run the following command:**

```
$ dsconfig -D "cn=directory manager" -w password -n get-connection-handler-prop \
  --handler-name "LDAP Connection Handler"

Property : Value(s)
------------------------:---------------------------------------------------------
allow-ldap-v2 : true
allow-start-tls : false
allowed-client : -
denied-client : -
enabled : true
keep-stats : true
key-manager-provider : -
listen-address : 0.0.0.0
listen-port : 1389
ssl-cert-nickname : server-cert
ssl-cipher-suite : -
ssl-client-auth-policy : optional
ssl-protocol : -
trust-manager-provider : -
use-ssl : false
```

Note – The `dsconfig` command displays the default values or behavior for properties that have not been customized.

To List Components

Where more than one instance of a component can exist (for example, it is possible to have more than one connection handler), a summary of the instances can be obtained by using the component's `list-`*`xxx`*`s` subcommand.

- **To list all of the available connection handlers, run this command:**

```
$ dsconfig -D "cn=directory manager" -w password -n list-connection-handlers

Connection Handler : Type : enabled : listen-port : use-ssl
------------------------:------:---------:-------------:--------
JMX Connection Handler : jmx : false : 1689 : false
LDAP Connection Handler : ldap : true : 1389 : false
LDAPS Connection Handler : ldap : true : 1636 : true
LDIF Connection Handler : ldif : true : - : -
```

To Modify the Properties of a Component

The properties of a component can be modified by using the component's `set-`*xxx*`-prop` subcommand. Multiple properties can be modified at the same time by using multiple occurrences of the `--set` option. The following example uses the `set-connection-handler-prop` subcommand to modify the properties of a connection handler.

Note – Many components have a Java class property that specifies the name of a Java class to be used as the implementation of the component. Do not modify this property, as doing so could prevent your server from operating correctly. These properties are treated as *advanced* properties and hidden from view unless you run `dsconfig` with the `--advanced` option.

- **To configure the LDAP connection handler to accept LDAPv2 connections, run this command:**

```
$ dsconfig -D "cn=directory manager" -w password -n set-connection-handler-prop \
  --handler-name="LDAP Connection Handler" --set allow-ldap-v2:true
```

To Modify the Values of a Multi-Valued Property

You can set multiple values for a property by using the `--set` and `--add` options in successive `dsconfig` commands.

Note – You cannot use the `--set` and `--add` options simultaneously in the same command.

To set more than one value for a property that currently has no values, use the `--set` option to set the first value, and the `--add` option (in a separate command) for subsequent values. You cannot use the `--add` option if the property does not have an existing value, either a default value or a value that you have already set.

Note – Many components have a Java class property that specifies the name of a Java class to be used as the implementation of the component. Do not modify this property, as doing so could prevent your server from operating correctly. These properties are treated as *advanced* properties and hidden from view unless you run `dsconfig` with the `--advanced` option.

The following example sets multiple values for the `allowed-client` property.

- **To restrict connections through the LDAP connection handler to specific clients, run these commands:**

```
$ dsconfig -D "cn=directory manager" -w password -n set-connection-handler-prop \
  --handler-name "LDAP Connection Handler"  --set allowed-client:myhost
$ dsconfig -D "cn=directory manager" -w password -n set-connection-handler-prop \
```

```
  --handler-name "LDAP Connection Handler" \
  --add allowed-client:myhost.example --add allowed-client:myhost.example.com
```

To Create a Component

New instances of a component can be created by using the component's `create-`*xxx* subcommand. Often there are several *subtypes* of the component. For example, there are currently three types of connection handler: LDAP, JMX, and LDIF. Because all of these are created by using the same subcommand, you must specify the type of component that you want to create. Do this by using the subcommand's `-t` or `--type`.

When you create a new component, you must specify the component's mandatory properties. The mandatory properties depend on the type of component that is being created. For example, an LDAP connection handler might have different mandatory properties to a JMX connection handler. If a mandatory property is left undefined, `dsconfig` enters interactive mode and prompts you for the undefined properties. If you include the `-n` (non-interactive) option, `dsconfig` fails to create the component and displays an error message indicating which properties need to be defined.

1. **Display the types of connection handler that can be created by accessing the help for the connection handler component.**

```
$ dsconfig create-connection-handler --help

Usage: dsconfig create-connection-handler {options}
Creates Connection Handlers

Global Options:
See "dsconfig --help"

SubCommand Options:
--handler-name {NAME}
The name of the new Connection Handler
--set {PROP:VALUE}
Assigns a value to a property where PROP is the name of the property and
VAL is the single value to be assigned. Specify the same property multiple
times in order to assign more than one value to it
-t, --type {TYPE}
The type of Connection Handler which should be created. The value for TYPE
can be one of: custom | jmx | ldap | ldif
```

2. **Create a new LDAP connection handler, specifying values for the mandatory `enabled` and the `listen-port` properties.**

```
$ dsconfig -D "cn=directory manager" -w password -n create-connection-handler \
  -t ldap --handler-name "My LDAP Connection Handler"

The LDAP Connection Handler could not be created because the following
```

```
mandatory properties were not defined:

Property     Syntax
---------------------------------
enabled      false | true
listen-port  1 <= INTEGER <= 65535
```

▼ To Delete a Component

Existing instances of a component can be removed using the component's `delete-`*xxx*.

- **Delete the LDAP connection handler that was created in the previous example:**

```
$ dsconfig -D "cn=directory manager" -w password -X -n delete-connection-handler \
  --handler-name "My LDAP Connection Handler"
```

Configuring the Connection Handlers

Connection handlers are responsible for handling all interaction with client applications, including accepting connections, reading requests, and sending responses.

The following sections describe how to configure the connection handlers by using the `dsconfig` command. These sections provide examples on only a few aspects of the configuration. For details about all the configuration properties, use the following command: `$ dsconfig list-properties -c connection-handler`.

For information about configuring secure connections, see "Configuring SSL and StartTLS for LDAP and JMX" on page 59.

▼ To Display All Connection Handlers

The following connection handlers are currently available for use in the directory server:

- **LDAP connection handler.** This connection handler is used to interact with clients using LDAP. It provides full support for LDAPv3 and limited support for LDAPv2.
- **LDAPS connection handler.** This connection handler is used to interact with clients using LDAP over SSL.
- **LDIF connection handler.** This connection handler is used to process changes in the server using internal operations.
- **JMX connection handler.** This connection handler allows interactions with clients using the Java Management Extensions (JMX) framework and the Remote Method Invocation (RMI) protocol.

- **To display all configured connection handlers, along with their basic properties, use the `dsconfig list-connection-handlers` command.**

```
$ dsconfig -D "cn=directory manager" -w password -n \
  list-connection-handlers

Connection Handler : Type : enabled : listen-port : use-ssl
-------------------------:------:---------:-------------:--------
JMX Connection Handler : jmx : false : 1689 : false
LDAP Connection Handler : ldap : true : 1389 : false
LDAPS Connection Handler : ldap : true : 1636 : true
LDIF Connection Handler : ldif : true : - : -
```

Configuring the LDAP Connection Handler

The following command displays the properties of the LDAP connection handler:

```
$ dsconfig "cn=directory manager" -w password -n get-connection-handler-prop \
  --handler-name "LDAP Connection Handler"

Property : Value(s)
-----------------------:------------
allow-ldap-v2 : true
allow-start-tls : false
allowed-client : -
denied-client : -
enabled : true
keep-stats : true
key-manager-provider : -
listen-address : 0.0.0.0
listen-port : 1389
ssl-cert-nickname : server-cert
ssl-cipher-suite : -
ssl-client-auth-policy : optional
ssl-protocol : -
trust-manager-provider : -
use-ssl : false
```

▼ To Control Which Clients Have LDAP Access to the Directory Server

You can specify a list of clients that may or may not access the directory server over LDAP. To do this, set the `allowed-client` or `denied-client` property of the LDAP connection handler. These properties take an IP address or subnetwork with subnetwork mask as values.

By default, these properties are not set and all clients are allowed access. Changes to these properties take effect immediately but do not interfere with connections that are already established.

This example permits access only to clients in the subnet mask `255.255.255.10`.

- **Run the `dsconfig` command as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n set-connection-handler-prop \
  --handler-name "LDAP Connection Handler" --set allowed-client:255.255.255.10
```

Configuring the LDIF Connection Handler

The LDIF connection handler is enabled by default. This connection handler can be used to process changes in the server using internal operations. The changes to be processed are read from an LDIF file.

The following command displays the default properties of the LDIF connection handler:

```
$ dsconfig -D "cn=directory manager" -w password -n get-connection-handler-prop \
  --handler-name "LDIF Connection Handler"

Property : Value(s)
---------------:------------------------
allowed-client : -
denied-client : -
enabled : true
ldif-directory : config/auto-process-ldif
poll-interval : 5 s
```

The `ldif-directory` property specifies the directory in which the LDIF files are located. The connection handler checks for the existence of any files in this directory, at an interval specified by the `poll-interval` property. The connection handler then processes the changes contained in those files as internal operations and writes the result to an output file with comments indicating the result of the processing.

▼ To Enable the JMX Alert Handler Through the LDIF Connection Handler

This example demonstrates how to enable the JMX alert handler through the LDIF connection handler.

1 Check the status of the JMX alert handler (disabled by default).

```
$ dsconfig -D "cn=directory manager" -w password -n get-alert-handler-prop \
  --handler-name "JMX Alert Handler"
Property            : Value(s)
--------------------:---------
disabled-alert-type : -
enabled             : false
enabled-alert-type  : -
```

2. **Create an LDIF file in the default LDIF directory that enables the JMX alert handler.**

```
$ cd ../config/
$ mkdir auto-process-ldif
$ cd auto-process-ldif/
$ cat > disable-jmx.ldif << EOM
> dn: cn=JMX Alert Handler,cn=Alert Handlers,cn=config
> changetype: modify
> replace: ds-cfg-enabled
> ds-cfg-enabled: true
> EOM
$
```

3. **After a period of time longer than `poll-interval`, recheck the status of the JMX alert handler.**

```
$ dsconfig -D "cn=directory manager" -w password -n get-alert-handler-prop \
  --handler-name "JMX Alert Handler" -n
Property            : Value(s)
--------------------:---------
disabled-alert-type : -
enabled             : true
enabled-alert-type  : -
```

Configuring the JMX Connection Handler

The following command displays the default properties of the JMX connection handler:

```
$ dsconfig -D "cn=directory manager" -w password get-connection-handler-prop \
  --handler-name "JMX Connection Handler" -n

Property : Value(s)
---------------------:------------
allowed-client : -
denied-client : -
enabled : false
key-manager-provider : -
listen-port : 1689
ssl-cert-nickname : server-cert
use-ssl : false
```

▼ To Change the Port on Which the Server Listens for JMX Connections

This example changes the port on which the server listens for JMX connections to 1789.

- **Use the `dsconfig` command as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n set-connection-handler-prop \
  --handler-name "JMX Connection Handler" --set listen-port:1789
```

Configuring Plug-Ins With dsconfig

Plug-ins are responsible for providing custom logic in the course of processing an operation or at other well-defined points within the directory server. The dsconfig command is used to manage the configuration of the directory server. For information about using dsconfig, see "Configuring the Directory Server With dsconfig" on page 7.

Overview of Plug-In Types

The dsconfig plugin-type property can be used to configure a plug-in to use one or more of the numerous plug-in types supported by the server. Usually a plug-in was written to perform a specific processing action for each of its default plug-in types. For this reason, a new default plug-in type cannot be added to a plug-in's configuration without changing the plug-in's underlying source code to add support for that plug-in type. A well-written plug-in checks the plug-in types passed to it from the configuration manager when it is enabled, and fails to start if it sees a plug-in type that it does not support.

Therefore, you can only remove one or more of the default plug-in type values from a plug-in's configuration. Care should be taken when doing this, because usually a plug-in has been engineered to support its default plug-in types for a reason. Removing one or more plug-in types might endanger the safe operation of the directory server.

Most of the plug-ins support more than one type, and multiple plug-ins are sometimes defined with the same plug-in type. The order in which these plug-ins are invoked during processing is undefined. If a specific order is required (for example, if the processing performed by one plug-in depends on the result of another), you can specify the order in which the plug-ins are invoked. For more information, see "To Configure Plug-In Invocation Order" on page 22.

Modifying the Plug-In Configuration

The following sections show various examples of managing plug-in configuration using dsconfig. dsconfig uses the administration connector to access the server. All of the examples in this section assume that the administration connector is listening on the default port (4444) and that the command is accessing the server running on the local host. If this is not the case, the --port and --hostname options must be specified.

dsconfig always accesses the server over a secured connection with certificate authentication. If you run dsconfig in interactive mode, you are prompted as to how you want to trust the certificate. If you run dsconfig in non-interactive mode (that is, with the -n option, you must specify the -X or --trustAll option, otherwise the command will fail.

To Display the List of Plug-Ins

This example shows a directory server configured with the current supported plug-ins. For a description of these plug-ins and their purpose, see "The Plug-In Configuration" in *Sun OpenDS Standard Edition 2.0 Configuration Reference*.

- **Use `dsconfig` to display the list of plug-ins that are currently configured.**

```
$ dsconfig -D cn="Directory Manager" -w password -n list-plugins

Plugin                          : Type                            : enabled
--------------------------------:---------------------------------:--------
7-Bit Clean                     : seven-bit-clean                 : false
Entry UUID                      : entry-uuid                      : true
LastMod                         : last-mod                        : true
LDAP Attribute Description List : ldap-attribute-description-list : true
Password Policy Import          : password-policy-import          : true
Profiler                        : profiler                        : true
Referential Integrity           : referential-integrity           : false
UID Unique Attribute            : unique-attribute                : false
```

The output of the command shows (from left to right):

- **Plug-in.** The name of the plug-in, usually descriptive of what it does.
- **Type.** The type of plug-in. It is possible to have more then one plug-in of a specific type.
- **Enabled.** Plug-ins can either be enabled of disabled. Disabled plug-ins remain in the server configuration but do not perform any processing.

To Create a New Plug-In

The easiest way to configure plug-ins is to use `dsconfig` in interactive mode. Interactive mode walks you through the plug-in configuration, and is therefore not documented here.

This example creates a new Password Policy Import Plug-in by using `dsconfig` in non-interactive mode.

- **Run the `dsconfig` command to create and enable a new Password Policy Import plug-in.**

```
$ dsconfig -D "cn=directory manager" -w password -n create-plugin \
  --type password-policy-import --plugin-name "My Password Policy Import Plugin" \
  --set enabled:true
```

To Enable or Disable a Plug-In

You can enable or disable a plug-in by setting the `enabled` property to `true` or `false`. This example disables the Password Policy Import plug-in created in the previous example.

- **Run the `dsconfig` command to disable the new Password Policy Import plug-in.**

```
$ dsconfig -D cn="Directory Manager" -w password -n set-plugin-prop \
  --plugin-name "My Password Policy Import Plugin" --set enabled:false
```

To Display and Configure Plug-In Properties

To display the properties of a plug-in, use the `get-plugin-prop` subcommand. To change the properties of a plug-in, use the `set-plugin-prop` subcommand. This example displays the properties of the plug-in created in the previous example, then enables the plug-in and sets the default authentication password storage scheme to Salted SHA-512.

1. **Display the plug-in properties.**

```
$ dsconfig -D cn="Directory Manager" -w password -n get-plugin-prop \
  --plugin-name "My Password Policy Import Plugin"
Property                            : Value(s)
------------------------------------:---------
default-auth-password-storage-scheme : -
default-user-password-storage-scheme : -
enabled                             : false
```

2. **Enable the plug-in and set the default authentication password storage scheme to Salted SHA-512.**

```
$ dsconfig -D cn="Directory Manager" -w password -n set-plugin-prop \
  --plugin-name "My Password Policy Import Plugin" --set enabled:true\
  --set default-auth-password-storage-scheme:"Salted SHA-512"
```

3. **(Optional) Display the plug-in properties again to verify the change.**

```
$ dsconfig -D cn="Directory Manager" -w password -n get-plugin-prop \
  --plugin-name "My Password Policy Import Plugin"
Property                            : Value(s)
------------------------------------:---------------
default-auth-password-storage-scheme : Salted SHA-512
default-user-password-storage-scheme : -
enabled                             : true
```

To Configure Plug-In Invocation Order

By default, the order in which plug-ins are invoked is undefined. You can specify that plug-ins be invoked in a specific order by using the `set-plugin-root-prop --set` *plugin-type*`:`*value* subcommand. The *value* in this case is the plug-in order, expressed as a comma-delimited list of plug-in names. The plug-in order string should also include a single asterisk element, which is a wildcard that will match any plug-in that is not explicitly named.

This example specifies that the Entry UUID plug-in should be invoked before any other pre-operation add plug-ins.

1. **(Optional) Display the current plug-in invocation order.**

```
$ dsconfig -D cn="Directory Manager" -w password -n get-plugin-root-prop
Property                                  : Value(s)
------------------------------------------:---------
plugin-order-intermediate-response        : -
plugin-order-ldif-export                  : -
plugin-order-ldif-import                  : -
plugin-order-post-connect                 : -
...
```

2. **Set the plug-in order.**

```
$ dsconfig -D cn="Directory Manager" -w password -n set-plugin-root-prop \
  --set plugin-order-pre-operation-add:"Entry UUID,*"
```

Note – Plug-in order values are not validated. Values that do not match defined plug-ins are ignored.

Configuring Commands As Tasks

Certain command-line utilities can be used to schedule tasks to run within the directory server as well as to perform their functions locally. Tasks that can be scheduled support the options used to connect to the directory server to interact with the task back end.

Utilities That Can Schedule Tasks

The following utilities can schedule tasks:

- `import-ldif`
- `export-ldif`
- `backup`
- `restore`
- `stop-ds`
- `stop-ds --restart`

Controlling Which Tasks Can Be Run

You can control the tasks that can be run by setting the `allowed-tasks` advanced global configuration property. By default, all tasks supported by the tasks back end are allowed. To prevent a task from being run, remove its value from the `allowed-tasks` property. For example, to prevent the server from being stopped using a task, run the following command:

```
$ dsconfig -D "cn=directory manager" -w password -n set-global-configuration-prop \
  --remove allowed-task:org.opends.server.tasks.ShutdownTask
```

Scheduling and Configuring Tasks

The procedures in this section indicate how to schedule a task, how to configure task notification, and how to configure task dependencies. All of the examples in this section assume that the commands are being run on the local host, using the default administration port (4444), and the local certificate configuration. If you are running the commands remotely, you might need to specify the certificate parameters. For more information, see "Managing Administration Traffic to the Server" on page 5.

▼ To Schedule a Task

To schedule a task, invoke the required utility with the options used to connect to the directory server, an optional start time, and any options that will be used as arguments for the task execution.

If the `-t` or `--start` option is provided, the utility exits immediately after scheduling the task. To schedule a task for immediate execution and have the utility exit immediately after scheduling the task, specify `0` as the value for the start time.

If the `-t` or `--start` option is omitted, the utility schedules the task for immediate execution and tracks the task's progress, printing log messages as they are available and exiting when the task has completed.

- **Schedule the `export-ldif` task to start at 12:15 on September 24th, 2008.**

```
$ export-ldif -D "cn=directory manager" -w password \
  -l /ldif-files/example.ldif --start 20080924121500 userRoot
```

▼ To Schedule a Recurring Task

To schedule a recurring task, invoke the required utility with the options used to connect to the directory server, specifying the recurring task schedule, and any options that will be used as arguments for the task execution. The following commands can be scheduled as recurring tasks:

- `import-ldif`
- `export-ldif`
- `backup`
- `restore`

The `--recurringTask` option specifies a recurring task schedule that is used by the task scheduler to determine when and how often a recurring task should run. The pattern used to specify the schedule is based on UNIX `crontab`(5) scheduling patterns and rules and includes the following five integer pattern fields, separated by blank spaces:

- Minute [0,59]
- Hour [0,23]
- Day of the month [1,31]
- Month of the year [1,12]
- Day of the week [0,6] (with 0=Sunday)

Each of these patterns can be either an asterisk (meaning all valid values), an element, or a list of elements separated by commas. An element is either a number or two numbers separated by a dash (meaning an inclusive range).

The task scheduler spawns regular task iterations according to the specified schedule.

- **Schedule the task using the `--recurringTask` option.**

  The following command schedules a backup task to execute at the beginning of every hour.

```
$ backup -D "cn=directory manager" -w password --recurringTask \
  "00 * * * *" --backupDirectory /example/backup --backUpAll --backupID "Hourly Backup"
```

Example 1 Recurring Task Example

This example shows an export task that is scheduled to run every 15 minutes, every Sunday.

```
$ export-ldif -D "cn=directory manager" -w password --recurringTask \
  "0,15,30,45 * * * 0" -l PATH/export-recurring.ldif -n userRoot
Recurring Export task  ExportTask-a614e45d-6ba5-4c29-a8e1-d518c20e46ab scheduled
successfully
```

▼ To Configure Task Notification

The task scheduling options of a utility enable you to notify an administrator when a task completes or if an error occurs during the task's execution. To use the notification facility, an SMTP server must be configured for the directory server.

1. **Specify an SMTP server by setting the `smtp-server` global configuration property.**

   The following command configures the SMTP server named `mailserver.example.com`:

```
$ dsconfig -D "cn=directory manager" -w password -n set-global-configuration-prop \
  --set smtp-server:mailserver.example.com
```

2 **Use the `completionNotify` and `errorNotify` options to specify the email address to which the task notification should be sent.**

The following command schedules a `backup` task and specifies that `admin@example.com` should be notified when the task completes, or when an error occurs:

```
$ backup -D "cn=directory manager" -w password -a -d /tmp/backups \
  --start 20080924121500 --completionNotify admin@example.com \
  --errorNotify admin@example.com
Backup task 20080924121500 scheduled to start Sep 24, 2008 12:15:00 PM SAST
```

▼ To Configure Task Dependencies

Certain tasks might require that another task be completed before the task begins. The task dependency options of a utility enable you to specify that the task depends on another task, and what the task should do should the other task fail.

- **Schedule the task and specify the `dependency` and `failedDependencyAction`.**

The following example schedules a backup task that depends on another task, and specifies that the backup should be canceled should the other task fail:

```
$ backup -D "cn=directory manager" -w password -a -d /tmp/backups \
  --start 2008102914530410 --dependency 20080924121500 \
  --failedDependencyAction cancel
Backup task 2008102914530410 scheduled to start Oct 29, 2008 14:53:04 PM SAST
```

Managing and Monitoring Scheduled Tasks

The `manage-tasks` utility can be used to obtain a list of scheduled tasks, to display task status, and to cancel scheduled tasks. The following procedures provide examples of managing scheduled tasks. For more information, see "manage-tasks" in *Sun OpenDS Standard Edition 2.0 Command-Line Usage Guide*.

▼ To Obtain Information About Scheduled Tasks

1 **Display a summary of all scheduled tasks.**

```
$ manage-tasks -D "cn=directory manager" -w password -n -s
ID                Type    Status
-----------------------------------------------
2008100912550010  Backup  Completed successfully
2008100912554710  Backup  Completed successfully
2008100912560510  Backup  Waiting on start time
2008100912561410  Backup  Waiting on start time
```

2. **Display additional information on a particular task, specified by its task ID.**

```
$ manage-tasks -D "cn=directory manager" -w password -n -i 2008100912550010

Task Details
------------------------------------------------------
ID                        2008100912550010
Type                      Backup
Status                    Completed successfully
Scheduled Start Time      Immediate execution
Actual Start Time         Oct 9, 2008 12:55:00 PM SAST
Completion Time           Oct 9, 2008 12:55:01 PM SAST
Dependencies              None
Failed Dependency Action  None
Email Upon Completion     None Specified
Email Upon Error          None Specified

Backup Options
----------------------------
Backup All        true
Backup Directory  ../backups

Last Log Message
------------------------------------------------------------------------------
[09/Oct/2008:12:55:01 +0200] severity="NOTICE" msgCount=4 msgID=10944795
message="The backup process completed successfully"
```

To Cancel a Scheduled Task

- **Run the `manage-tasks` utility with the `-c` or `--cancel` option.**

  The following command cancels a particular task, specified by its task ID:

```
$ manage-tasks -D "cn=directory manager" -w password -n -c 2008100912561410
```

To Cancel a Recurring Task

You can cancel an entire recurring task, in which case both the recurring task and its next scheduled iteration are canceled. Alternatively, you can cancel only the next scheduled task iteration, in which case future recurring task iterations will be spawned by the task scheduler.

1. **Use the manage-tasks command to display the summary of scheduled tasks.**

```
$ manage-tasks -D "cn=directory manager" -w password -n -s

 ID                                          Type    Status

------------------------------------------------------------------------------
 Hourly Backup                               Backup  Recurring
 Hourly Backup - Wed Jan 14 13:00:00 SAST 2009  Backup  Waiting on start time
```

2 **Run the `manage-tasks` utility with the `-c` or `--cancel` option.**

- **Cancel the entire recurring task by specifying its task ID.**

```
$ manage-tasks -D "cn=directory manager" -w password -n -c "Hourly Backup"
Task Hourly Backup canceled
```

- **Cancel the next scheduled task by specifying its task ID.**

```
$ manage-tasks -D "cn=directory manager" -w password -n \
  -c "Hourly Backup - Wed Jan 14 13:00:00 SAST 2009 "
Task Hourly Backup - Wed Jan 14 13:00:00 SAST 2009  canceled
```

Managing the Directory Server With the Control Panel

The Control Panel is a graphical user interface that displays server status information and enables you to perform basic server administration. The following topics describe the tasks that can be performed by using the Control Panel:

- Starting and stopping the server (see "Starting and Stopping the Directory Server" in *Sun OpenDS Standard Edition 2.0 Installation Guide*)
- Configuring Java Settings (see"To Configure Java Settings With the Control Panel" in *Sun OpenDS Standard Edition 2.0 Installation Guide*)
- "Managing Directory Data With the Control Panel" on page 200
- "Importing and Exporting Entries With the Control Panel" on page 107
- "Backing Up and Restoring Directory Data With the Control Panel" on page 134
- "Managing Indexes With the Control Panel" on page 194
- "Managing the Schema With the Control Panel" on page 391
- "Monitoring the Directory Server With the Control Panel" on page 337

▼ To Start the Control Panel

1 **Start the control-panel application.**

- **From a graphical file browser, navigate to the `bin` folder beneath the folder where you installed the directory server, and then double-click on the icon for the `control-panel` command.**

- **From a command line in a terminal window, run the `control-panel` command.**
  - UNIX and Linux: *install-dir*`/bin/control-panel`
  - Windows: *install-dir*`\bat\control-panel`

The Authentication Required window is displayed, with fields for the bind DN and password of an administrative user. The default value for the bind DN is for the root DN user, `cn=Directory Manager`.

2 **Enter the password for the administrative user (and a DN if needed), and click OK.**

The OpenDS Control Panel is displayed.

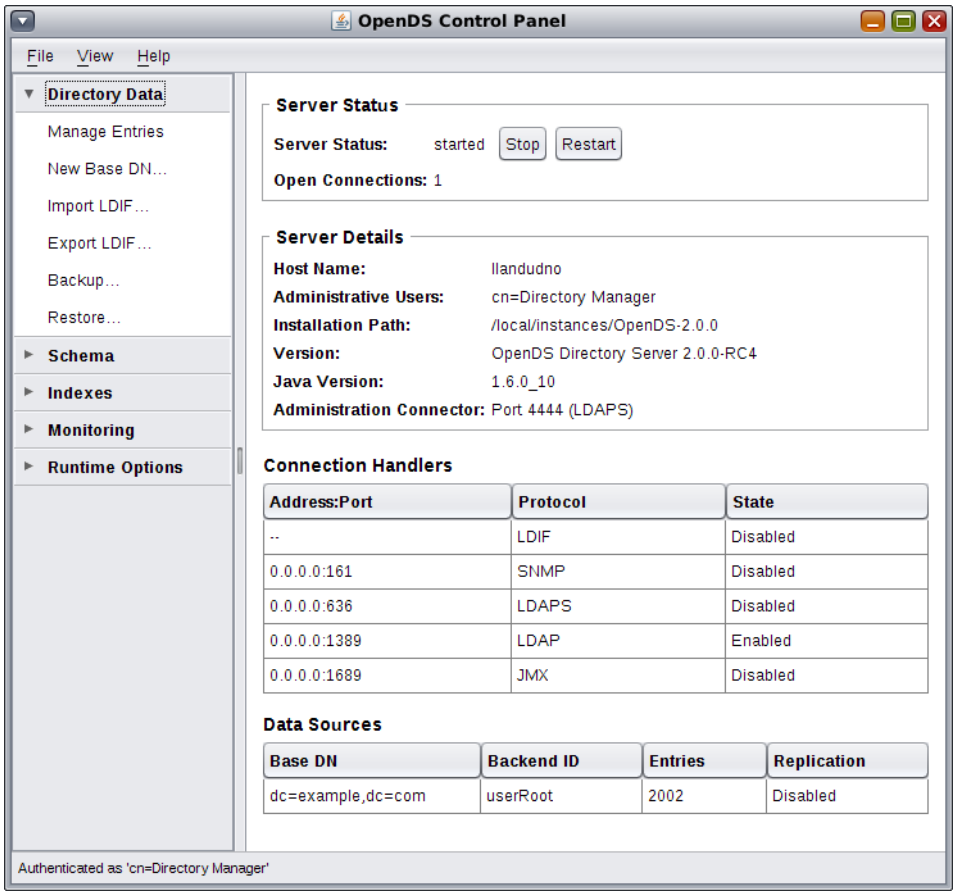


FIGURE 1 The OpenDS Control Panel

▼ To Specify the Trust Manager Provider and Trust Store Algorithm Used by the Control Panel

The Control Panel uses a trust manager provider to determine whether to trust certificates that are presented to it. By default, the Control Panel uses the SunJSEE trust manager provider, and the SunX509 trust store algorithm. If these defaults are not suitable to your platform, you can

specify the trust manager provider and trust store algorithm by setting the relevant Java properties and running the `dsjavaproperties` command.

Note – You can also use the Control Panel itself to set these properties. For more information, see "To Configure Java Settings With the Control Panel" in *Sun OpenDS Standard Edition 2.0 Installation Guide*.

The following example configures the Control Panel to use the trust manager provider and keystore algorithm for the IBM JDK.

1 **Edit the** *install-dir*`/config/java.properties` **file and add the following line:**

```
control-panel.java-args=-Dorg.opends.admin.trustmanagerprovider=IBMJSSE
-Dorg.opends.admin.trustmanageralgo=IBMX509
```

2 **Run the** `dsjavaproperties` **command.**

```
$ dsjavaproperties
```

Configuring and Testing the DSML Gateway

The Directory Services Markup Language (DSML) is a SOAP-based mechanism that can communicate with directory servers using an XML-based representation instead of the LDAP protocol. Sun OpenDS Standard Edition 2.0 supports the use of DSML through a web application that acts as a DSML-to-LDAP gateway, in which clients communicate with the gateway using DSML, but the gateway communicates with the directory server through LDAP.

Deploying the DSML Gateway

In most cases, the DSML gateway can be deployed like any other web application. The following sections describe how to deploy the DSML gateway in common application containers:

- "Deploying the DSML Gateway in Apache Tomcat" on page 31
- "Deploying the DSML Gateway in Glassfish" on page 31
- "Deploying the DSML Gateway in Sun Java System Web Server 7" on page 33

Deploying the DSML Gateway in Apache Tomcat

You can deploy the DSML gateway in Apache Tomcat by using one of the following methods:

- Copy the WAR file containing the DSML gateway into the `webapps` directory, and rename the file based on the name that you want to use for the application context. For example, if you want the application context to be `/dsml`, then rename the file to `dsml.war`. Restart Tomcat, and it automatically creates a directory with the name of the specified context (for example, `webapps/dsml`). The DSML gateway is deployed in that directory and should be available for use.
- Manually create a directory below `webapps`with the name that you want to use for the application context (for example, `webapps/dsml`). Go into that directory and unpack the contents of the DSML gateway WAR file into it using this command:

  ```
  jar -xvf path-to-DSML.war
  ```

  Restart Tomcat, and the DSML gateway becomes available for use.

See "Configuring the DSML Gateway" on page 35 and "Confirming the DSML Gateway Deployment" on page 36 for information about completing your DSML deployment.

Deploying the DSML Gateway in Glassfish

You can deploy the DSML Gateway in Glassfish using either the autodeploy feature or the graphical administration interface.

▼ To Deploy the DSML Gateway Using Autodeploy

1. **Make sure that the Glassfish server is running.**
2. **Copy the WAR file for the DSML gateway into the `autodeploy` directory below the desired domain (for example, `domains/domain1/autodeploy`).**
3. **If necessary, rename the file in the process so that the name before the `.war`extension matches what you want the application context to be.**

   For example, if you want the application context to be `/dsml`, then re name it to `dsml.war` before putting it in the `autodeploy`directory.

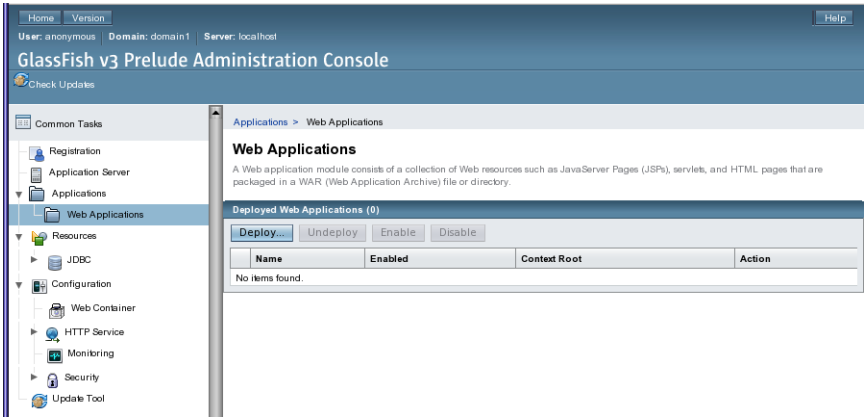
   Glassfish automatically detects the new WAR file and makes it available for use, usually in a few seconds.

   See "Configuring the DSML Gateway" on page 35 and "Confirming the DSML Gateway Deployment" on page 36 for information about completing your DSML deployment.

▼ To Deploy the DSML Gateway Using the Administration GUI

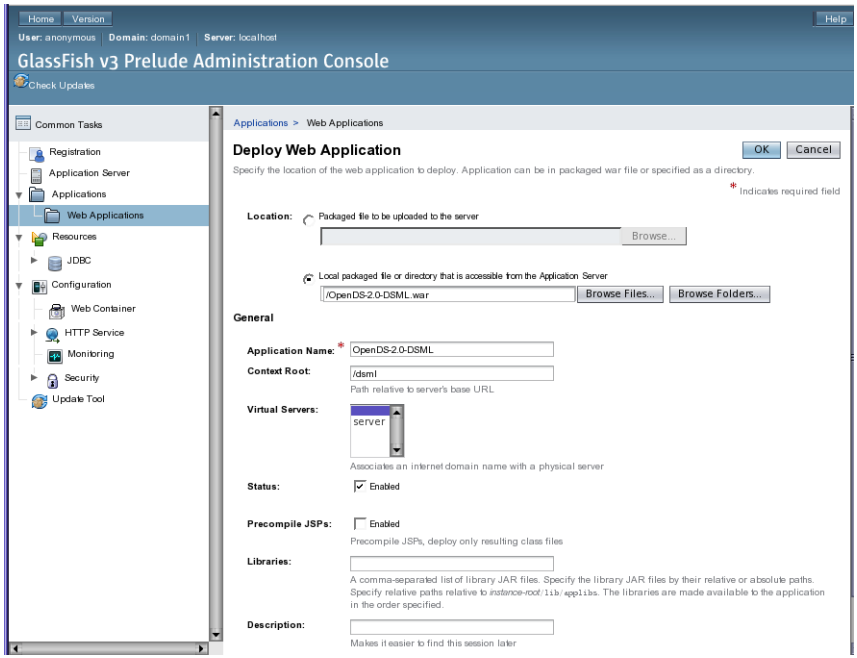
1 Log in to the Glassfish administrative interface and click on the Web Applications link.

The available web applications are displayed. Unless you have previously installed web applications, the list is empty.



2 Click the Deploy button.

The Deploy Enterprise Applications/Modules page appears in the browser, displaying fields where you specify either the WAR file to the server or specify the path to a WAR file already on the server system.



3 **Specify the WAR file to use, and also specify values in the other fields as needed. The preceding figure shows typical values.**

4 **Click the OK button, and the DSML gateway is deployed.**

See "Configuring the DSML Gateway" on page 35 and "Confirming the DSML Gateway Deployment" on page 36 for information about completing your DSML deployment.

Deploying the DSML Gateway in Sun Java System Web Server 7

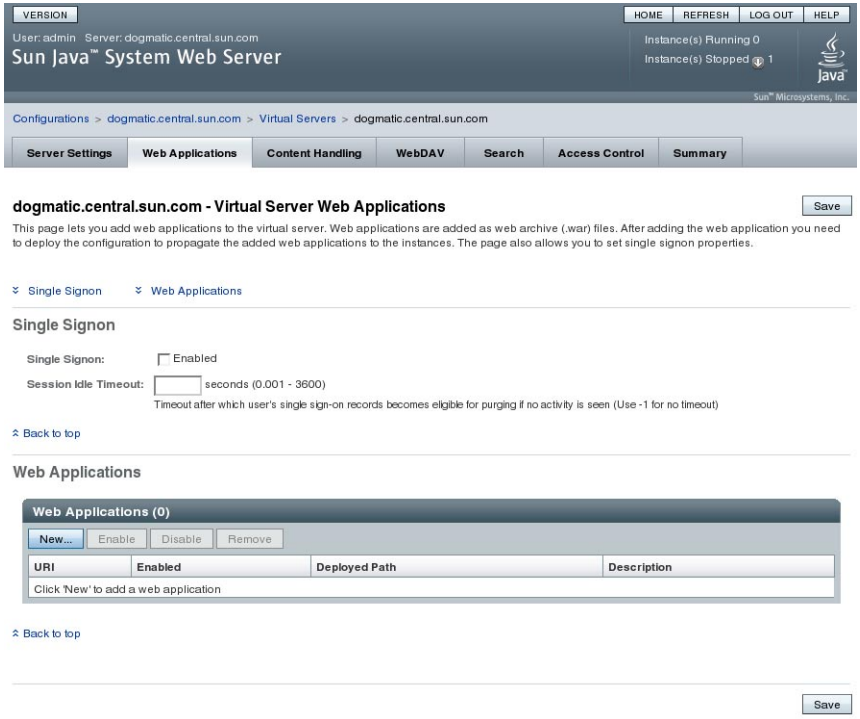
Deploy the DSML gateway in the Sun Java System Web Server 7 using the graphical administrative interface.

▼ To Use the Sun Java System Web Server 7 Graphical Administrative Interface

1 **Using the Sun Java System Web Server 7 graphical administrative interface, log in to the server instance that you want to administer, and choose the desired virtual server.**

By default, only one virtual server is present.

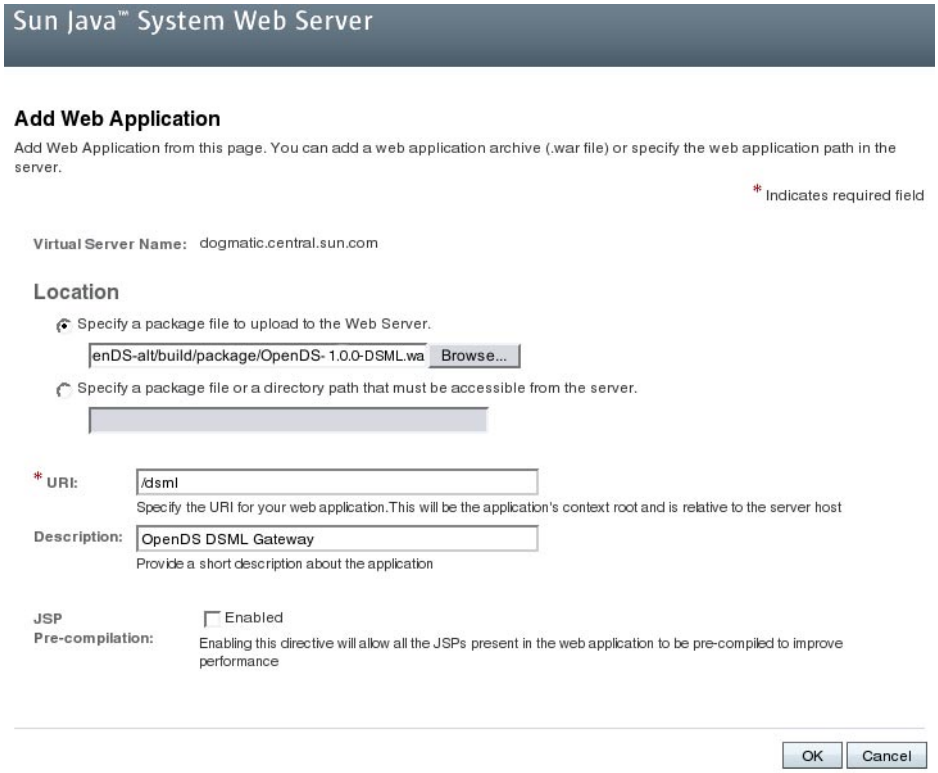
2 **Click the Web Application tab. The Virtual Server Web Applications page appears, as shown in the following figure.**



3 **Click the New button in the Web Applications section.**

A window appears, requesting information about the application.

4 **Choose the path to the WAR file, either on the local system or the server system, and specify the URI to use to access the application. Complete the fields in the window as shown in the following figure.**



5 **Click the OK button.**

6 **Click the Deployment Pending link in the upper-right corner of the administration console, and then click the Deploy button on the window that opens.**

The DSML gateway is deployed and ready to use.

See "Configuring the DSML Gateway" on page 35 and "Confirming the DSML Gateway Deployment" on page 36 for information about completing your DSML deployment.

Configuring the DSML Gateway

The `WEB-INF/web.xml` file includes initialization parameters that can be used to specify the address (in the `ldap.host` parameter) and port number (in the `ldap.port` parameter) of the directory server to which DSML requests should be forwarded. By default, the DSML gateway is configured to communicate with a directory server on the same system, that is, `localhost`) on port `389`. If you need to change the host address and port number, edit the `web.xml` file and restart the web container.

Confirming the DSML Gateway Deployment

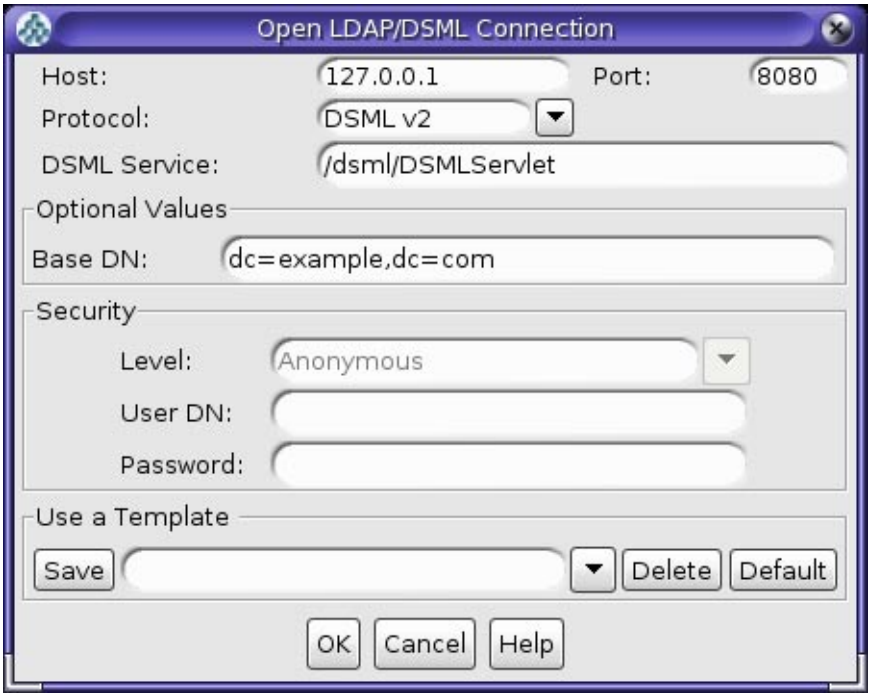
After the DSML gateway has been deployed and configured, you can communicate with it by using any DSMLv2 client. The following sections describe two ways to accomplish this:

- "Confirming the DSML Gateway Deployment with JXplorer" on page 36
- "Confirming the DSML Gateway Deployment with the Directory Server Resource Kit" on page 37

Confirming the DSML Gateway Deployment with JXplorer

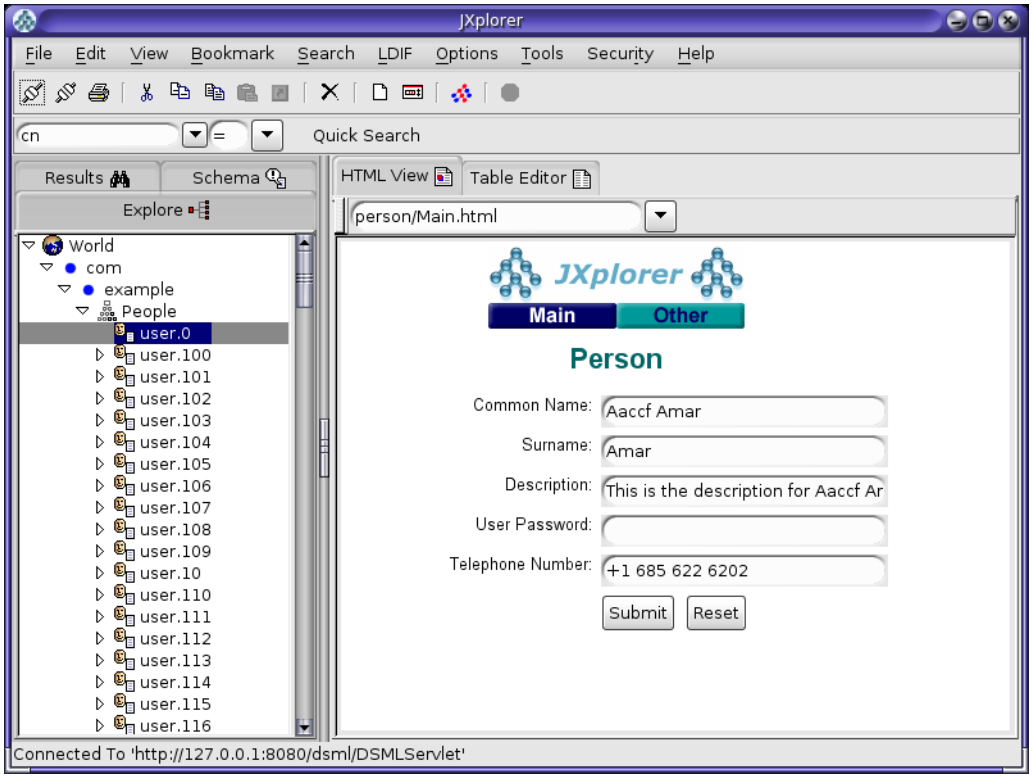
The JXplorer tool is a Java-based LDAP browser that can be used to browse, search, and edit the contents of a Sun OpenDS Standard Edition instance This tool can communicate using both LDAP and DSML. Although JXplorer's DSML support does not allow authentication (and therefore is restricted to the set of operations available to anonymous users), it is still possible to use it to verify that the DSML gateway is functioning as expected.

To do this, start JXplorer and chose the Connect option from the File menu. The Open LDAP/DSML Connection window opens with fields for connection information. The following figure shows typical entries.



Enter the address and port number of the Web application on which the DSML gateway is running and choose `DSMLv2` from the Protocol list. Specify the path to the DSMLServlet in the DSML Service field, and provide an appropriate base DN value for your directory.

Click the OK button to connect the directory server and display a JXplorer window where you can search and browse the tree (with the limitations imposed for anonymous users)



Confirming the DSML Gateway Deployment with the Directory Server Resource Kit

The Directory Server Resource Kit (DSRK) is a collection of utilities that may be used in conjunction with directory servers. It is originally intended for use with the Sun Java System Directory Server, but in most cases the applications also work with Sun OpenDS Standard Edition. The most recent version of the DSRK is included as part of DSEE 6.0, and it contains `dsmlsearch` and `dsmlmodify` tools that can interact with a directory server using DSML rather than LDAP.

Note that even though an older version of these DSML tools was provided with earlier versions of the Directory Server Resource Kit, the version provided with DSEE 6 is strongly recommended because it is easier to use.

Using the `dsmlsearch` Command

The `dsmlsearch` command is a DSML-based counterpart to the `ldapsearch` command. `dsmlsearch` operates in a similar manner to `ldapsearch` but there are certain key differences. To see usage information, invoke the command with no arguments, as in the following example:

```
$ ./dsmlsearch
usage: dsmlsearch -h http://host:port -b basedn [options] filter [attributes...]
where:
-h hostURL URL of the directory server
-b basedn  base dn for search
-D binddn  bind dn
-w passwd  bind password (for simple HTTP authentication)
use "-w - " to prompt for a password
-j pwfile  file where password is stored
-s scope   specify the scope of the search
baseObject - For searching only the base entry
singleLevel - For searching only the children
wholeSubtree - For searching the base entry and all childrens
-a deref   specify how aliases are deferenced
neverDerefAliases - Aliases are never dereferenced
derefFindingBaseObj - Dereferenced when finding the base DN
derefAlways - Dereferenced when finding below the base DN
-l seconds specify the maximum number of seconds to wait for the search
-z number  specify the maximum number of entries to return for the search
-f file    specify the name of the file containing the search filter
```

The `dsmlsearch` command differs in usage from `ldapsearch`:

- The `-h` argument is used to provide a URL to use to access the server. It should include the host and port number, as well as the URI for the gateway servlet (for example, `http://127.0.0.1:8080/dsml/DSMLServlet`).
- The `-b` argument is used to specify the search scope, but note that the values you provide are different (`baseObject`instead of `base`, `singleLevel`instead of `one`, and `wholeSubtree`instead of `sub`).
- The results are output in DSML format, which is not as user-friendly or human-readable as the LDIF output provided by `ldapsearch`.

An example usage of this tool is as follows. Note that the DSML output does not contain any line breaks, but line breaks are added here for readability.

```
$ ./dsmlsearch -h http://127.0.0.1:8080/dsml/DSMLServlet \
-b "dc=example,dc=com" -s baseObject \"(objectClass=*)"
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body><dsml:batchResponse xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
<dsml:searchResponse><dsml:searchResultEntry dn="dc=example,dc=com"><dsml:attr
name="objectClass"><dsml:value>domain</dsml:value><dsml:value>top</dsml:value>
</dsml:attr><dsml:attr name="dc"><dsml:value>example</dsml:value></dsml:attr>
</dsml:searchResultEntry><dsml:searchResultDone><dsml:resultCode code="0"/>
</dsml:searchResultDone></dsml:searchResponse></dsml:batchResponse>
</SOAP-ENV:Body></SOAP-ENV:Envelope>
```

Using the `dsmlmodify` Utility

The `dsmlmodify` utility is a DSML-based counterpart to the `ldapmodify` tool, and it can perform add, delete, modify, and modify DN operations over DSML. To see the usage information for this tool, run it with no arguments, as shown in this example:

```
$ ./dsmlmodify
usage: dsmlmodify -h http://host:port [options] -f file
where:
-h hostURL URL of the directory server
-D binddn  bind dn
-w passwd  bind password (for simple HTTP authentication)
use "-w - " to prompt for a password
-j pwfile  file where password is stored
-f file    specify the name of the file containing
the modifications
```

As with the `dsmlsearch` utility, the `-h` argument specifies a URL, and the output is returned in DSML form. Unlike `ldapmodify`, the `dsmlmodify` tool does not accept the changes through standard input. Changes must be specified in a file, and that file must be in DSML format instead of than LDIF, and the changes cannot contain an outer `batchRequest` wrapper. The following example shows a typical input file.

```
<addRequest dn="uid=test.user,dc=example,dc=com">
<attr name="objectClass">
<value>top</value>
<value>person</value>
<value>organizationalPerson</value>
<value>inetOrgPerson</value>
</attr>
<attr name="uid">
<value>test.user</value>
</attr>
<attr name="givenName">
<value>Test</value>
</attr>
```

```
<attr name="sn">
<value>User</value>
</attr>
<attr name="cn">
<value>Test User</value>
</attr>
<attr name="userPassword">
<value>password</value>
</attr>
</addRequest>
<modifyRequest dn="uid=test.user,dc=example,dc=com">
<modification name="description" operation="replace">
<value>This is the new description</value>
</modification>
</modifyRequest>
<modDNRequest dn="uid=test.user,dc=example,dc=com" newrdn="cn=Test User"
 deleteoldrdn="false" newSuperior="ou=People,dc=example,dc=com" />
<delRequest dn="cn=Test User,ou=People,dc=example,dc=com" />
```

The following example shows the output from applying these changes. Line breaks have been added to the output to make it more readable:

```
$ ./dsmlmodify -h http://127.0.0.1:8080/dsml/DSMLServlet \
  -D "cn=Directory Manager" -w password -f /tmp/test.dsml
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body><dsml:batchResponse xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
<dsml:addResponse><dsml:resultCode code="0"/></dsml:addResponse>
<dsml:modifyResponse><dsml:resultCode code="0"/></dsml:modifyResponse>
<dsml:modDNResponse><dsml:resultCode code="0"/></dsml:modDNResponse>
<dsml:delResponse><dsml:resultCode code="0"/><dsml:errorMessage>The number of
entries deleted was 1</dsml:errorMessage></dsml:delResponse></dsml:batchResponse>
</SOAP-ENV:Body></SOAP-ENV:Envelope>

$ dsmlmodify -h http://localhost:8080/dsml/DSMLServlet \
  -D "cn=directory manager" -w password -f /tmp/dsml.ldif
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body><batchResponse xmlns="urn:oasis:names:tc:DSML:2:0:core">
<addResponse><resultCode code="0"/></addResponse>
<modifyResponse><resultCode code="0"/></modifyResponse>
<modDNResponse><resultCode code="0"/></modDNResponse>
<delResponse><resultCode code="0"/></delResponse></batchResponse>
</SOAP-ENV:Body></SOAP-ENV:Envelope>
```

Configuring Security in the Directory Server

The directory server provides several mechanisms to secure traffic to the server. The topics in this section describe these mechanisms, and how to configure them. For information about securing access to the directory data, see "Controlling Access To Data" on page 225.

This section covers the following topics:

Getting SSL Up and Running Quickly

The directory server provides a number of options for configuring and using SSL and StartTLS. The numerous possibilities for configuration might be daunting for those who are unfamiliar with the technology or who just want to get up and running as quickly as possible for testing purposes.

This chapter provides a rough list of the steps that must be performed to allow the directory server to accept SSL-based connections using a self-signed certificate. The chapter also demonstrates how to configure SSL and StartTLS if you install the server using the QuickStart utility. Each configuration step is described in more detail in the chapters that follow.

▼ To Accept SSL-Based Connections Using a Self-Signed Certificate

This procedure assumes the following:

- The directory server is installed on the system on which you are working.
- The Java `keytool` utility is in your path. If it is not, you can add it to your path or provide the complete path to it when invoking the commands.
- The administration connector is listening on the default port (4444) and the `dsconfig` command is accessing the server running on the local host. If this is not the case, the `--port` and `--hostname` options must be specified.

1 Create a private key for the certificate, using the `keytool` utility. For example:

```
$ keytool -genkey -alias server-cert -keyalg rsa \
  -dname "CN=myhost.example.com,O=Example Company,C=US" \
  -keystore config/keystore -storetype JKS
```

Change the value of the `-dname` argument so that it is suitable for your environment:

- The value of the `CN` attribute should be the fully-qualified name of the system on which the certificate is being installed.
- The value of the `O` attribute should be the name of your company or organization.
- The value of the `C` attribute should be the two-character abbreviation for your country.

You are prompted for a password to protect the contents of the keystore and for a password to protect the private key.

2 Generate a self-signed certificate for the key. For example:

```
$ keytool -selfcert -alias server-cert -validity 1825 \
   -keystore config/keystore -storetype JKS
```

When you are prompted for the keystore password, enter the same password that you provided in the previous step.

3 Create a text file named `config/keystore.pin`.

4 Export the public key for the certificate that you created. For example:

```
$ keytool -export -alias server-cert -file config/server-cert.txt -rfc \
   -keystore config/keystore -storetype JKS
```

5 Create a new trust store and import the server certificate into that trust store. For example:

```
$ keytool -import -alias server-cert -file config/server-cert.txt \
  -keystore config/truststore -storetype JKS
```

6 Type `yes` when you are prompted to trust the certificate.

This step is required only if the SSL and StartTLS settings were not specified during installation, or if you want to change those settings.

7 **Use the `dsconfig` command to enable the key manager provider, trust manager provider, and connection handler. For example:**

```
$ dsconfig -D "cn=directory manager" -w password -n set-key-manager-provider-prop \
  --provider-name JKS --set enabled:true
$ dsconfig -D "cn=directory manager" -w password -n set-trust-manager-provider-prop \
  --provider-name "Blind Trust" --set enabled:true
$ dsconfig -D "cn=directory manager" -w password -n set-connection-handler-prop \
  --handler-name "LDAPS Connection Handler" \
  --set "trust-manager-provider:Blind Trust" --set key-manager-provider:JKS \
  --set listen-port:1636 --set enabled:true
```

Port `1636` is the standard LDAPS port, but you might not be able to use this port if it is already taken or if you are a regular user. If you need to accept SSL-based connections on a port other than `1636`, change the `listen-port` property in the last command to the port number being used.

8 **The server should now have a second listener that accepts SSL-based client connections. Test the configuration with the `ldapsearch` command, for example:**

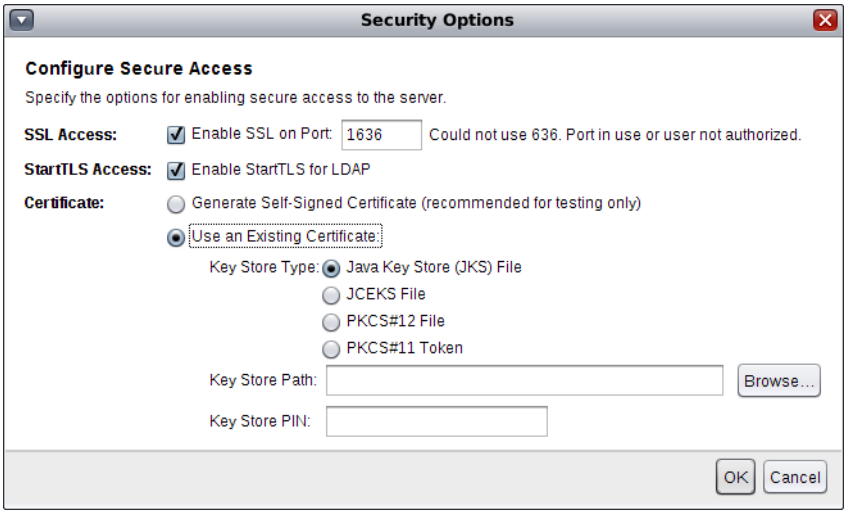
```
$ ldapsearch --port 1636 --useSSL --baseDN "" --searchScope base "(objectClass=*)"
```

You are prompted to trust the server's certificate. On typing `yes`, the root DSE entry should be returned.

Enabling SSL and StartTLS in QuickSetup

The easiest way to get the directory server up and running with SSL, StartTLS, or both, is to use the `setup` utility in GUI mode. This tool can be used to set up the server after you have downloaded it as a zip file. QuickSetup enables you to use a self-signed certificate, or an existing certificate in a JKS keystore, a PKCS#12 file, or a PKCS#11 token.

To access the SSL and StartTLS configuration, click the Configure button in front of the LDAP Secure Access field. The following dialog is displayed:



The fields on this screen include:

- SSL Access — Select this checkbox to indicate that the LDAPS (that is, LDAP over SSL) listener should be enabled. Enter the port number on which the directory server listens for connections.
- StartTLS Access — Select this checkbox to configure whether the LDAP connection handler will allow clients to use the StartTLS extended operation to initiate secure communication over an otherwise insecure connection.
- Certificate — Select one of the following radio buttons to obtain the certificate that the server should use for SSL, StartTLS, or both:
  - Generate Self-Signed Certificate will generate a self-signed certificate that can be used to secure the communication. While this is convenient for testing purposes, many clients will not trust the certificate by default, and you might need to configure it manually.
  - Use an Existing Certificate will use a certificate in an existing JKS keystore, a PKCS #12 file, or a PKCS #11 token. For more information about obtaining certificates, see "Configuring Key Manager Providers" on page 44.

Configuring Key Manager Providers

Key manager providers are ultimately responsible for providing access to the certificate that should be used by the directory server when performing SSL or StartTLS negotiation.

This section covers the following topics:

- "Using the PKCS #12 Key Manager Provider" on page 48
- "Using the PKCS #11 Key Manager Provider" on page 49

For additional information, see "The Key Manager Provider Configuration" in the *Sun OpenDS Standard Edition 2.0 Configuration Reference*.

Key Manager Provider Overview

- JKS keystore, which is the default keystore format used by Java Secure Socket Extension (JSSE)
- PKCS #12 file
- PKCS #11 device, such as a hardware security module or cryptographic accelerator

The process for configuring the directory server to use these key manager providers is described in detail in the following sections.

Using the JKS Key Manager Provider

The JKS keystore is the default keystore used by most JSSE implementations, and is the preferred keystore type in many environments. To configure the server to use this keystore type, you must first obtain a JKS keystore that contains a valid certificate. To do this, you can either generate a self-signed certificate or issue a certificate signing request to an existing Certificate Authority (CA) and import the signed certificate.

All of the steps described here require the use of the `keytool` utility, which is provided with the Java runtime environment. This utility is typically found in the `bin` directory below the root of the Java installation. For more information about using the `keytool` utility, see the official Java documentation (`http://java.sun.com/j2se/1.5.0/docs/tooldocs/solaris/keytool.html`). The `keytool` examples in the following sections use the `keytool` syntax provided with Java 1.5.

Using the JKS key manager provider involves the following:

1. Generating the private key
2. Self-signing the certificate, or using an external certificate authority to sign the certificate
3. Configuring the JKS key manager provider

▼ To Generate the Private Key

Whether you use a self-signed certificate or generate a certificate signing request, you must first generate a private key. You can do this using the `keytool` utility with the `-genkey` option. The following arguments can be used with this option:

- `-alias` *alias*. Specifies the name that should be used to refer to the certificate in the keystore. The default name used by the directory server is `server-cert`.

- `-keyalg` *algorithm*. Specifies the algorithm that should be used to generate the private key. This should almost always be `rsa`.
- `-dname` *subject*. Specifies the subject to use for the certificate. The subject typically contains at least a `CN` attribute, which is the fully-qualified name of the system on which the certificate will be installed, an `O` attribute that specifies the name of the organization (or company), and a `C` attribute that specifies the country in which the certificate will be used.
- `-keystore` *path*. Specifies the path to the keystore file. The file will be created if it does not already exist. The default keystore path used by the directory server is `config/keystore`.
- `-keypass` *password*. Specifies the password that should be used to protect the private key in the keystore. If the password is not provided, you will be prompted for it.
- `-storepass` *password*. Specifies the password that should be used to protect the contents of the keystore. If the password is not provided, you will be prompted for it. The directory server expects the password used for the `-keypass` and `\-storepass` options to be the same.
- `-storetype` *type*. Specifies the keystore type that should be used. For the JKS keystore, the value should always be `JKS`.

- **Use the `keytool -genkey` command to create a private key.**

```
$ keytool -genkey -alias server-cert -keyalg rsa \
  -dname "CN=server.example.com,O=example.com,C=US" \
  -keystore config/keystore -keypass password \
  -storetype JKS -storepass password
```

▼ To Self-Sign the Certificate

If the certificate is to be self-signed, use the `-selfcert` option. The most important arguments for use with this option include:

- `-alias` *alias*. Specifies the name that should be used to refer to the certificate in the keystore. This name should be the same as the value used when creating the private key with the `-genkey` option.
- `-validity` *days*. Specifies the length of time in days that the certificate should be valid. The default validity is 90 days.
- `-keystore` *path*. Specifies the path to the keystore file. The file will be created if it does not already exist.
- `-keypass` *password*. Specifies the password that should be used to protect the private key in the keystore. If this is not provided, then you will be interactively prompted for it.
- `-storepass` *password*. Specifies the password that should be used to protect the contents of the keystore. If this is not provided, then you will be interactively prompted for it.
- `-storetype` *type*. Specifies the keystore type that should be used. For the JKS keystore, the value should always be `JKS`.

- **Use the `keytool -selfcert` command to generate a self-signed certificate.**

```
$ keytool -selfcert -alias server-cert -validity 1825 \
  -keystore config/keystore -keypass password -storetype JKS \
  -storepass password
```

▼ To Sign the Certificate by Using an External Certificate Authority

If the certificate is to be signed by an external certificate authority, you must first generate a certificate signing request (CSR) using the `-certreq` option. The CSR can be submitted to a certificate authority to be signed. The method for doing this, and the method for obtaining the signed certificate, might vary from one certificate authority to another.

When you receive the signed certificate from the Certificate Authority, import it into the keystore with the `-import` option.

1. **Use the `-certreq` option to obtain a certificate signing request.**

```
$ keytool -certreq -alias server-cert -file /tmp/server-cert.csr \
  -keystore config/keystore -keypass password -storetype JKS \
  -storepass password
```

The arguments used with this command are as follows:

- `-alias` *alias*. Specifies the name that should be used to refer to the certificate in the keystore. This name should be the same as the value used when creating the private key with the `-genkey` option.
- `-file` *path*. Specifies the path to the file to which the CSR should be written. If this is not provided, the request will be written to standard output.
- `-keystore` *path*. Specifies the path to the keystore file. The file will be created if it does not already exist.
- `-keypass` *password*. Specifies the password that should be used to protect the private key in the keystore. If this is not provided, you will be interactively prompted for it.
- `-storepass` *password*. Specifies the password that should be used to protect the contents of the keystore. If this is not provided, you will be interactively prompted for it.
- `-storetype` *type*. Specifies the keystore type that should be used. For the JKS keystore, the value should always be `JKS`.

2. **Send the certificate request to an external certificate authority. The certificate authority will send you a signed certificate file. Save the file in `/tmp/server-cert.txt`**

3. **Use the `-import` to import the signed certificate.**

```
$ keytool -import -alias server-cert -file /tmp/server-cert.cert \
  -keystore config/keystore -storetype JKS -storepass password
```

The arguments used with this command are as follows:

- `-alias` *alias*. Specifies the name that should be used to refer to the certificate in the keystore. This name should be the same as the value used when creating the private key with the `-genkey` option.
- `-file` *path*. Specifies the path to the file containing the signed certificate. The file should be in either the DER-encoded binary format or the base64-encoded ASCII format as described in RFC 1421 (`http://www.ietf.org/rfc/rfc1421.txt`).
- `-keystore` *path*. Specifies the path to the keystore file. The file will be created if it doesn't already exist.
- `-storepass` *password*. Specifies the password that should be used to protect the contents of the keystore. If this is not provided, then you will be interactively prompted for it.
- `-storetype` *type*. Specifies the keystore type that should be used. For the JKS keystore, the value should always be `JKS`.

▼ To Configure the JKS Key Manager Provider

When you have created a JKS keystore containing a signed certificate (whether self-signed or signed by an external CA), you can configure the server to use that keystore by creating a key manager provider entry for that keystore.

This example defines an instance of a file-based key manager provider, using `dsconfig` to set the properties of the key manager provider. For details about the properties of the key manager provider, see "File-Based Key Manager Provider Configuration" in the *Sun OpenDS Standard Edition 2.0 Configuration Reference*.

- **Use the `dsconfig` command to create the key manager provider entry.**

```
$ dsconfig -D "cn=Directory Manager" -w password -X -n \
  set-key-manager-provider-prop --provider-name "JKS" \
  --set java-class:org.opends.server.extensions.FileBasedKeyManagerProvider \
  --set enabled:true --set "key-store-type:JKS" --set "key-store-file:/config/keystore" \
  --set "key-store-pin:secret"
```

Using the PKCS #12 Key Manager Provider

PKCS #12 is a standard format for storing certificate information, including private keys. The directory server can use a PKCS #12 file as a certificate keystore if it includes the private key for the certificate.

Because PKCS #12 is a common format for storing certificate information, you might already have a certificate in this format, or the certificate authority (CA) that you use might create certificates in this form. In some cases, it might also be possible to convert an existing certificate into PKCS #12 format. For example, if you already have a certificate in a Network Security Services (NSS) certificate database, then the NSS `pk12util` tool can import it. The following example uses the `pk12util` tool to export a certificate named `server-cert` contained in the database `../../alias/slapd-config-key3.db` to a PKCS #12 file, `/tmp/server-cert.p12`:

```
$ ./pk12util -n server-cert -o /tmp/server-cert.p12 \
  -d ../../alias -P "slapd-config-"
```

To create a new certificate in PKCS #12 format, use the procedure described in "Using the JKS Key Manager Provider" on page 45 for obtaining a certificate in a JKS keystore. The only difference in the process is that you should use `-storetype PKCS12` instead of `-storetype JKS` when you invoke the `keytool` commands. For example, to create a self-signed certificate in a PKCS #12 file, use the following commands:

```
$ keytool -genkey -alias server-cert -keyalg rsa \
  -dname "CN=server.example.com,O=example.com,C=US" \
  -keystore config/keystore.p12 -keypass password \
  -storetype PKCS12 -storepass password
```

Note – The preceding command uses syntax for the `keytool` provided with Java 1.5. If your installation uses Java 1.6, substitute `-genkeypair` for the `-genkey` option.

```
$ keytool -selfcert -alias server-cert -validity 1825 \
  -keystore config/keystore.p12 -keypass password \
  -storetype PKCS12 -storepass password
```

As with JKS, the directory server provides a template key manager provider for use with PKCS #12 certificate files that uses the same set of configuration attributes as the configuration entry for the JKS key manager provider. The only differences are that the value of the `key-store-type` attribute must be `PKCS12`, and the `key-store-file` attribute should refer to the location of the PKCS #12 file rather than a JKS keystore. The following example uses `dsconfig` to configure the PKCS #12 keystore manager provider:

```
$ dsconfig -D "cn=directory manager" -w password -X -n\
  set-key-manager-provider-prop --provider-name "PKCS12" --advanced
```

For a complete list of configurable properties, see "File-Based Key Manager Provider Configuration" in the *Sun OpenDS Standard Edition 2.0 Configuration Reference*.

Using the PKCS #11 Key Manager Provider

PKCS #11 is a standard interface used for interacting with devices capable of holding cryptographic information and performing cryptographic functions. The PKCS #11 interface has two common uses of interest for the directory server:

- Cryptographic accelerators use this interface to allow products to offload their cryptographic processing to an external board (or in some cases, a special module inside the system's CPU or a framework inside the OS kernel), which might provide better performance for those operations.

- Hardware security modules (HSMs) use this interface to provide a secure repository for storing key information. This significantly reduces the likelihood that sensitive key information will be exposed and helps protect the overall integrity of the secure communication mechanisms.

At present, the PKCS #11 support that the directory server provides has been tested and verified only on systems running at least Solaris 10 (on SPARC® and x86/x64 systems) through the use of the Solaris OS cryptographic framework. Any device that plugs into this Solaris cryptographic framework should be supported in this manner. This includes the *softtoken* device, which is simulated in software and is therefore available on all systems supporting the Solaris cryptographic framework regardless of whether they have a hardware device providing PKCS #11 support.

If you do have a third-party PKCS #11 device installed in a Solaris system, it is likely that the Solaris OS cryptographic framework is already configured to access that device. However, if you will simply be using the software token or if you are running on a Sun Fire™ T1000 or T2000 system and want to take advantage of the cryptographic processor included in the UltraSPARC—T1® CPU, you will likely need to initialize the PKCS #11 interface. This should first be accomplished by choosing a PIN to use for the certificate store, which can be done with this command:

```
$ pktool setpin
```

This command prompts you for the current passphrase. If you have not yet used the Solaris OS cryptographic framework, the default passphrase is `changeme`. You are then prompted twice for the new password.

Note – This step should be done while logged in as the user or role that will be used to run the directory server, since each user might have a different set of certificates.

At this point, it should be possible to use the Java `keytool` utility to interact with the Solaris cryptographic framework through PKCS #11. This will work much in the same way as it does when working with JKS or PKCS#12 keystores, with the following exceptions:

- The value of the `-keystore` argument must be `NONE`.
- The value of the `-storetype` argument must be `PKCS11`.
- You should not use the `-keypass` argument, and the tool will not prompt you for that password interactively if you do not provide it.
- The value of the `-storepass` argument must be the passphrase that you chose when using the `pktool setpin` command. Alternately, if you do not provide this argument on the command line, this is the password that you should enter when prompted.

For example, the following commands use the PKCS #11 interface to generate a self-signed certificate through the Solaris cryptographic framework:

```
$ keytool -genkey -alias server-cert -keyalg rsa \
  -dname "CN=server.example.com,O=example.com,C=US" \
  -keystore NONE -storetype PKCS11 -storepass password
```

Note – The preceding command uses syntax for the `keytool` provided with Java 1.5. If your installation uses Java 1.6, substitute `-genkeypair` for the `-genkey` option.

```
$ keytool -selfcert -alias server-cert -validity 1825 \
  -keystore NONE -storetype PKCS11 -storepass password
```

When the certificate is installed in the PKCS #11 keystore, the directory server must be configured to use that keystore. Configure the PKCS #11 keystore provider in the same way as the entry for the JKS and PKCS#12 keystore manager providers, with the exception that the `key-store-file` attribute is not included. However, a PIN is still required and is provided either directly, in a PIN file, through a Java property, or through an environment variable.

The following example uses `dsconfig` to configure the PKCS #11 key manager provider:

```
$ dsconfig -D "cn=directory manager" -w password -X -n \
  set-key-manager-provider-prop --provider-name "PKCS11" --advanced
```

For a complete list of configurable properties, see "PKCS11 Key Manager Provider Configuration" in the *Sun OpenDS Standard Edition 2.0 Configuration Reference*..

Configuring Trust Manager Providers

The directory server uses trust manager providers to determine whether to trust a certificate that is presented to it. Trust managers serve an important role in the overall security of the system by ensuring that the peer (the system at the other end of the connection, whether it is an inbound connection from a client or an outbound connection to another server) is who it claims to be.

This section covers the following topics:

Overview of Certificate Trust Mechanisms

A trust manager provider can improve security whenever SSL or StartTLS is used by thwarting attempts to use forged certificates and foiling man-in-the-middle attacks. The two primary use cases for trust manager providers are as follows:

- A client presents its own certificate to the directory server during the SSL or StartTLS negotiation process, potentially for use in SASL EXTERNAL authentication.
- The directory server attempts to establish an SSL-based connection to an external system, for example for the purpose of synchronization or for proxied or chained operations.

The trust manager has no impact on the strength of the encryption, so only the directory server and its peer will be able to understand the communication. Any third-party observer will be unable to decipher the exchange. However, the trust manager is responsible for ensuring that the peer is who it claims to be so that confidential information is not inadvertently exposed to one peer masquerading as another.

The trust manager considers a number of factors to determine whether a peer certificate should be trusted. This topic describes some of the most common criteria that are taken into account during this process.

One of the simplest trust mechanisms is the validity period for the certificate. All certificates have a specific window during which they should be considered valid, bounded by "notBefore" and "notAfter" time stamps. If the current time is beyond the "notAfter" time stamp, the certificate is expired and trust managers reject it. Similarly, certificates are also typically rejected if the current time is before the "notBefore" time stamp. Most often, the "notBefore" time stamp is set to the time that the certificate was signed, but there are cases in which a certificate might be issued that is not immediately valid. In those cases, it is important to ensure that the peer is not granted access too early.

Another very important factor in deciding whether to trust a peer certificate is the peer certificate chain. When one system presents its certificate to another, it does not present its certificate only, but a chain of certificates that describes all entities involved in the process. When a trust manager is attempting to determine whether to trust a peer, the trust manager first looks in its trust store to determine whether it contains the peer certificate. If that certificate is found, the peer will be trusted (barring rejection for another reason, such as being outside the validity period). If the peer's certificate is not found, the trust manager looks at the next certificate in the chain, which will be the certificate that was used to sign the peer's certificate (also called the issuer certificate). If the trust store contains the issuer's certificate, the server will trust that issuer certificate and will also implicitly trust any certificate that it has signed. This process continues up the certificate chain (looking at the certificate that signed the issuer certificate, and so on) until one of the certificates is found in the trust store or until the root of the chain is reached (in which case, the root certificate will be self-signed and therefore will be its own issuer). If none of the certificates in the peer chain is contained in the trust store, the peer's certificate is rejected.

This process makes it much easier to manage an environment with a large number of certificates (for example, one in which there is a large number of servers or in which many clients use SASL EXTERNAL authentication). It is not necessary for the trust store to have each individual peer certificate. The trust store can contain only one of the certificates in the peer chain. For example, if all of the certificates that might legitimately be presented to the server

were signed by the same issuer, it is necessary to have only that issuer's certificate in the trust store in order to implicitly trust any of the peers.

In some environments, there might be other elements taken into account when deciding to trust a peer certificate chain. For example, there might be a certificate revocation list (CRL) that contains a list of all of the certificates that have been revoked and should no longer be considered valid even if they are still within their validity period and were signed by a trusted issuer. This can be useful, for example, if the certificate belonged to an employee that has left the company or if the private key for the certificate has been compromised. The Online Certificate Status Protocol (OCSP, as described in RFC 2560 (`http://www.ietf.org/rfc/rfc2560.txt`) also provides a similar mechanism, in which the trust manager might ask an OCSP server whether a given certificate is still valid. The directory server currently does not support using CRLs or OCSP when attempting to determine whether a peer certificate chain should be trusted.

Using the Blind Trust Manager Provider

The blind trust manager provider is a simple provider that trusts any certificate that is presented to it. It does not look at the expiration date, who signed the certificate, the subject or alternate names, or any other form of criteria.

The directory server provides a blind trust manager provider that is disabled by default. You can enable the provider by changing the value of the `enabled` attribute to `true`. The blind trust manager provider does not require any other configuration attributes. The following example uses `[dsconfig]` to configure the blind trust manager provider:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X\
  set-trust-manager-provider-prop --provider-name "Blind Trust" --advanced
```

For a list of the configurable properties, see “Blind Trust Manager Provider Configuration” in the *Sun OpenDS Standard Edition 2.0 Configuration Reference*.

Caution – The blind trust manager provider is provided as a convenience for testing purposes only and should never be used in a production server, especially one that is configured to allow SASL EXTERNAL authentication. If a client attempts to use SASL EXTERNAL to authenticate to the directory server using a certificate and the server blindly accepts any certificate that the client presents, the user can create a self-signed certificate that allows it to impersonate any user in the directory.

Using the JKS Trust Manager Provider

Just as the JKS keystore can be used to provide the key material for a key manager provider, it can also be used to provide information that can used by trust manager providers. In general, using a JKS file as a trust store is similar to using it as a keystore. However, because private key information is not accessed when the file is used as a trust store, there is generally no need for a PIN when accessing its contents.

When the JKS trust manager provider determines whether to trust a given peer certificate chain, it considers two factors:

- Is the peer certificate within the validity period?
- Is any certificate in the chain contained in the trust store?

If the peer certificate is not within the validity period or none of the certificates in the peer certificate chain are contained in the trust store, the JKS trust manager rejects that peer certificate.

Use the `keytool -import` utility to import certificates into a JKS trust store. The `-import` option uses these arguments:

- `-alias` *alias*. Specifies the name to give to the certificate in the trust store. Give each certificate a unique name, although the nickname is primarily for managing the certificates in the trust store and has no impact on whether a certificate is trusted.
- `-file` *path*. Specifies the path to the file containing the certificate to import. The file can be in either DER format or in base64-encoded ASCII format, as described in RFC 1421 (`http://www.ietf.org/rfc/rfc1421.txt`).
- `-keystore` *path*. Specifies the path to the file used as the JKS trust store. This path is typically `config/truststore`.
- `-storetype` *type*. Specifies the format of the trust store file. For the JKS trust manager, this must be `JKS`.
- `-storepass` *password*. Specifies the password used to protect the contents of the trust store. If the trust store file does not exist, this value is the password to assign to the trust store, and must be used for future interaction with the trust store. If this option is not provided, the password is interactively requested from the user.

The following command provides an example of importing a certificate into a JKS trust store. If the trust store does not exist, this command creates it before importing the certificate.

```
$ keytool -import -alias server-cert -file /tmp/cert.txt \
  -keystore config/truststore -storetype JKS -storepass password
```

The directory server provides a template JKS trust manager provider. Use `dsconfig` to configure the following properties of the JKS trust manager provider:

- `enabled`. Indicates whether the JKS trust manager provider is enabled. The JKS trust manager provider is not available for use by other server components unless the value of this property is `true`.
- `trust-store-type`. Specifies the format of the trust store. For the JKS trust store provider, the value of this property is `JKS`.
- `trust-store-file`. Specifies the path to the trust store file, which is typically `config/truststore`, although an alternate file can be used if needed. The value of this property can be either an absolute path or a path that is relative to the *install-dir*.

  The following example uses `dsconfig` to configure the JKS trust manager provider:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X \
  set-trust-manager-provider-prop --provider-name "JKS" --advanced
```

For a list of the configurable properties, see "File-Based Trust Manager Provider Configuration" in the *Sun OpenDS Standard Edition 2.0 Configuration Reference*.

Using the PKCS #12 Trust Manager Provider

The PKCS #12 trust manager provider is primarily useful if you already have the peer or issuer certificates to be used in a PKCS #12 file. If you do not have the certificates in this format, use the JKS trust manager provider instead. The Java `keytool` utility does not currently support importing trusted certificates (that is, those with just a public key and no private key information) into a PKCS #12 file.

The directory server provides a template PKCS #12 trust manager provider. Use `dsconfig` to configure the following properties of the PKCS #12 trust manager provider:

- `enabled`. Indicates whether the PKCS #12 trust manager provider is enabled. The trust manager provider is not available for use by other server components unless this property has a value of `true`.
- `trust-store-type`. Specifies the format of the trust store. For the PKCS #12 trust manager provider, the value is `PKCS12`.
- `trust-store-file`. Specifies the path to the trust store file, which is typically `config/truststore.p12`, although an alternate file can be used if needed. The value of this property can be either an absolute path or a path that is relative to the *install-dir*.

A PIN might be required to access the contents of the PKCS #12 file. In this case, one of the following configuration attributes must be used to provide the password. (At the present time, the password must be provided in clear text.)

- `trust-store-pin`. Specifies the PIN needed to access the trust store directly.
- `trust-store-pin-file`. Specifies the path to a file containing the PIN needed to access the trust store. The value of this property can be either an absolute path or a path that is relative to the server root.
- `trust-store-pin-property`. Specifies the name of a Java property that holds the PIN needed to access the trust store.
- `trust-store-pin-environment-variable`. Specifies the name of an environment variable that holds the PIN needed to access the trust store.

```
$ dsconfig -D "cn=directory manager" -w password \
  set-trust-manager-provider-prop \
  --provider-name "PKCS12" --advanced
```

Configuring Certificate Mappers

A *certificate mapper* examines a certificate presented by a client and maps it to the user in the directory that should be associated with that certificate. It is primarily used in the context of processing SASL EXTERNAL authentication, in which case the client wants to authenticate to the server using its SSL certificate rather than a password or some other form of credentials.

The examples in this section use the `dsconfig` command to modify certificate mappers. The `dsconfig` command accesses the server configuration over SSL, using the administration connector. For more information, see "Configuring the Directory Server With `dsconfig`" on page 7.

Using the Subject Equals DN Certificate Mapper

The Subject Equals DN certificate mapper is a simple certificate mapper that expects the subject of the client certificate to be exactly the same as the distinguished name (DN) of the corresponding user entry. Using this certificate mapper is easy because there are no configuration attributes associated with it. However, this mapper is not suitable for many environments because certificate subjects and user DNs are often not the same.

To enable or disable the Subject Equals DN certificate mapper, use `dsconfig` to set its `enabled` property to `true` or `false`. The following example uses `dsconfig` to configure the Subject Equals DN certificate mapper.

```
$ dsconfig -D "cn=directory manager" -w password -n set-certificate-mapper-prop \
  --mapper-name "Subject Equals DN" --advanced
```

Using the Subject Attribute to User Attribute Certificate Mapper

The Subject Attribute to User Attribute certificate mapper attempts to map a client certificate to a user entry based on a set of attributes that they have in common. In particular, it takes the values of a specified set of attributes from the certificate subject and attempts to locate user entries that contain those same values in a corresponding set of attributes.

Use `dsconfig` to set the properties of this certificate mapper:

- `subject-attribute-mapping`. Specifies a multivalued property that is used to map attributes from the certificate subject to attributes in user entries. Values for this attribute consist of the name of the attribute in the certificate subject followed by a colon and the name of the corresponding attribute in the user's entry. For example, the value `e:mail` maps the `e` attribute from the certificate subject to the `mail` attribute in user entries. At least one attribute mapping must be defined.
- `user-base-dn`. Specifies a multivalued property that is used to specify the set of base DNs below which the server is to look for matching entries. If this is not present, then the server searches below all public naming contexts.

The following example uses `dsconfig` to configure the Subject Attribute to User Attribute certificate mapper:

```
$ dsconfig -D "cn=directory manager" -w password -n set-certificate-mapper-prop \
  --mapper-name "Subject Attribute to User Attribute" --advanced
```

If multiple attribute mappings are defined, then the server combines them with an AND search. For example, if two mappings are defined `cn:cn` and `e:mail`, and the server is presented with a certificate having a subject of `E=john.doe@example.com,CN=John Doe,O=Example Corp,C=US`, then it generates a search filter of `(&(cn=John Doe)(mail=john.doe@example.com))`. Any attribute for which a mapping is defined but is not contained in the certificate subject is not included in the generated search filter. All attributes that can be used in generated search filters should have corresponding indexes in all back-end databases that can be searched by this certificate mapper.

For the mapping to be successful, the generated search filter must match exactly one user in the directory (within the scope of the base DNs for the mapper). If no users match the generated criteria or if multiple users match, then the mapping fails.

Using the Subject DN to User Attribute Certificate Mapper

The Subject DN to User Attribute certificate mapper attempts to establish a mapping by searching for the subject of the provided certificate in a specified attribute in user entries. In this case, you must ensure that user entries are populated with the subjects of the certificates associated with those users. However, it is possible that this process could be automated in the future with a plug-in that automatically identifies any certificates contained in a user entry and adds the subjects of those certificates to a separate attribute.

Use `dsconfig` to set the properties of this certificate mapper:

- `subject-attribute`. This is a single-valued attribute whose value is the name of the attribute type that should contain the certificate subject in user entries. This attribute must be defined in the server schema, and it should be indexed for equality in all back ends that might be searched.
- `user-base-dn`. This is a multivalued attribute that is used to specify the set of base DNs below which the server should look for matching entries. If this is not present, then the server will search below all public naming contexts.

The following example uses `dsconfig` to configure the Subject DN to User Attribute certificate mapper:

```
$ dsconfig -D "cn=directory manager" -w password -n set-certificate-mapper-prop \
  --mapper-name "Subject DN to User Attribute" --advanced
```

Although there is no standard attribute for holding the subjects of the certificates that a user might hold, the directory server does define a custom attribute type, `ds-certificate-subject-dn`, that can be used for this purpose. This attribute can be added to user entries along with the `ds-certificate-user` auxiliary object class. This attribute is multivalued attribute, and if a user has multiple certificates, then it should contain the subjects for each of them as separate values. However, this attribute is not indexed by default, so if it is to be used, update the corresponding back ends so that they contain an equality index for this attribute.

For the mapping to be successful, the certificate mapper must match exactly one user (within the scope of the base DNs for the mapper). If no entries match or if multiple entries match, then the mapping fails.

Using the Fingerprint Certificate Mapper

The Fingerprint certificate mapper attempts to establish a mapping by searching for the MD5 or SHA1 fingerprint of the provided certificate in a specified attribute in user entries. In this case, you must ensure that user entries are populated with the certificate fingerprints (in standard

hexadecimal notation with colons separating the individual bytes, for example, `07:5A:AB:4B:E1:DD:E3:05:83:C0:FE:5F:A3:E8:1E:EB`). In the future, this process could be automated by a plug-in that automatically identifies any certificates contained in user entries and adds the fingerprints of those certificates to the appropriate attribute.

Use `dsconfig` to set the properties of this certificate mapper:

- `fingerprint-attribute`. Specifies a single-valued attribute whose value is the name of the attribute type that should contain the certificate fingerprint in user entries. This attribute must be defined in the server schema, and it should be indexed for equality in all back ends that can be searched.
- `fingerprint-algorithm`. Specifies which digest algorithm to use to calculate certificate fingerprints. The value is either `MD5` or `SHA1`.
- `user-base-dn`. Specifies a multivalued attribute that is used to specify the set of base DNs below which the server is to look for matching entries. If this property is not present, then the server searches below all public naming contexts.

The following example uses `dsconfig` to configure the Fingerprint certificate mapper:

```
$ dsconfig -D "cn=directory manager" -w password -n set-certificate-mapper-prop \
  --mapper-name "Fingerprint Mapper" --advanced
```

Although there is no standard attribute for holding certificate fingerprints, the directory server does define a custom attribute type, `ds-certificate-fingerprint`, that can be used for this purpose. This attribute can be added to user entries along with the `ds-certificate-user` auxiliary object class. This attribute is multivalued, and if a user has multiple certificates, then it should contain the fingerprints for each of them as separate values. However, this attribute type is not indexed by default in any of the server back ends, so if it is to be used, add the corresponding equality index to all appropriate back ends.

For the mapping to be successful, the certificate mapper must match exactly one user (within the scope of the base DNs for the mapper). If no entries match or if multiple entries match, then the mapping fails.

Configuring SSL and StartTLS for LDAP and JMX

After the directory server has been configured with at least one key manager provider and at least one trust manager provider enabled, you can enable SSL and StartTLS for the connection handlers.

The examples in this section use the `dsconfig` command to modify the server configuration. The `dsconfig` command accesses the server configuration over SSL via the administration connector. As such, the relevant connection options must be specified, including how the SSL certificate is trusted. These examples use the `-X` option to trust all certificates.

Configuring the LDAP and LDAPS Connection Handlers

The LDAP connection handler is responsible for managing all communication with clients using LDAP. By default, the LDAP protocol does not specify any form of security for protecting that communication, but it can be configured to use SSL or also to allow the use of the StartTLS extended operation.

The directory server configures two connection handlers that can be used for this purpose. While the LDAP connection handler entry is enabled by default and is used to perform unencrypted LDAP communication, it can also be configured to support StartTLS. For information, see "To Enable StartTLS Support" on page 62. The LDAPS connection handler entry is disabled, but the default configuration is set up for "To Enable SSL-Based Communication" on page 62.

The following sections describe how to configure LDAP and LDAPS connection handler parameters with `dsconfig`.

▼ To Enable a Connection Handler

- **Set the `enabled` property of the connection handler to `true`.**

  This example enables the LDAP connection handler.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X \
  set-connection-handler-prop --handler-name "LDAP Connection Handler" \
  --set enabled:true
```

▼ To Specify a Connection Handler's Listening Port

- **Set the `listen-port` property of the connection handler.**

  The `listen-port` property specifies the port number to use when communicating with the directory server through this connection handler. The standard port to use for unencrypted LDAP communication (or LDAP using StartTLS) is `389`, and the standard port for SSL-encrypted LDAP is `636`. However, it might be desirable or necessary to change this in some environments (for example, if the standard port is already in use, or if you are running on a UNIX system as a user without sufficient privileges to bind to a port below `1024`).

  This example sets the LDAPS connection handler's listen port to `1636`.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X \
  set-connection-handler-prop --handler-name "LDAPS Connection Handler" \
  --set listen-port:1636
```

To Specify a Connection Handler's Authorization Policy

- **Set the `ssl-client-auth-policy` property of the connection handler.**

  The `ssl-client-auth-policy` property specifies how the connection handler should behave when requesting a client certificate during the SSL or StartTLS negotiation process. If the value is `optional`, the server requests that the client present its own certificate but still accepts the connection even if the client does not provide a certificate. If the value is `required`, the server requests that the client present its own certificate and rejects any connection in which the client does not do so. If the value is `disabled`, the server does not ask the client to present its own certificate.

  This example sets the LDAPS connection handler's authorization policy to `required`.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X \
  set-connection-handler-prop --handler-name "LDAPS Connection Handler" \
  --set ssl-client-auth-policy:required
```

To Specify a Nickname for a Connection Handler's Certificate

- **Set the `ssl-cert-nickname` property of the connection handler.**

  The `ssl-cert-nickname` property specifies the nickname of the certificate that the server presents to clients during SSL or StartTLS negotiation. This property is primarily useful when multiple certificates are in the keystore and you want to specify which certificate is to be used for that listener instance.

  This example sets the nickname of the LDAP connection handler's certificate to `server-cert`.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X \
  set-connection-handler-prop --handler-name "LDAP Connection Handler" \
  --set ssl-cert-nickname:server-cert
```

To Specify a Connection Handler's Key Manager Provider

- **Set the `key-manager-provider` property of the connection handler.**

  The `key-manager-provider` property specifies which key manager provider among the available "Configuring Key Manager Providers" on page 44 that should be used by the connection handler to obtain the key material for the SSL or StartTLS negotiation.

  This example sets the LDAP connection handler's key manager provider to `JKS`. The specified manager must already be configured for the command to succeed.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X \
  set-connection-handler-prop --handler-name "LDAP Connection Handler" \
  --set key-manager-provider:JKS
```

▼ To Specify a Connection Handler's Trust Manager Provider

- **Set the `trust-manager-provider` property of the connection handler.**

  The `trust-manager-provider` property specifies which trust manager provider among the available "Configuring Trust Manager Providers" on page 51 to be used by the connection handler to decide whether to trust client certificates presented to it.

  This example sets the LDAP connection handler's trust manager to `JKS`. The specified manager must already be configured for the command to succeed.

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X \
  set-connection-handler-prop --handler-name "LDAP Connection Handler" \
  --set trust-manager-provider:JKS
```

▼ To Enable StartTLS Support

1. **Specify the appropriate values for the `key-manager-provider` and `trust-manager-provider` properties.**

2. **Set the `allow-start-tls` property to `true`, as follows:**

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X \
  set-connection-handler-prop --handler-name "LDAP Connection Handler" \
  --set allow-start-tls:true
```

Note – If SSL is enabled, the `allow-start-tls` property cannot be set.

▼ To Enable SSL-Based Communication

1. **Display the connection handler properties to ensure that the configured key manager provider and trust manager provider values are correct.**

   The following example displays the properties of the LDAPS connection handler:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X \
  get-connection-handler-prop --handler-name "LDAPS Connection Handler"
```

2. **Set the `enabled` property to `true`, as follows:**

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X \
  set-connection-handler-prop --handler-name "LDAPS Connection Handler" \
  --set enabled:true
```

Note – If SSL is enabled, non-SSL communication will not be available for that connection handler instance.

Enabling SSL in the JMX Connection Handler

The JMX connection handler can be used to communicate with clients using the JMX (Java Management Extensions) protocol. This protocol does not support the use of StartTLS to allow both encrypted and unencrypted communication over the same port, but it can be configured to accept only unencrypted JMX or only SSL-encrypted JMX communication.

The JMX connection handler provides the server's default configuration for communicating over JMX. To enable SSL for this connection handler, use `dsconfig` to set the following configuration attributes:

- `key-manager-provider`. Specifies the DN of the configuration entry for the key manager provider that is used to obtain the key material for the SSL negotiation.
- `ssl-cert-nickname`. Specifies the nickname (or alias) of the certificate that is presented to clients.
- `use-ssl`. Indicates whether the connection handler is to use SSL to communicate with clients.

The following example uses `dsconfig` to configure the JMX connection handler:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X \
  set-connection-handler-prop --handler-name "JMX Connection Handler"
```

Using SASL Authentication

The LDAP protocol definition provides two ways in which clients can authenticate to the server: LDAP simple authentication and SASL authentication.

In LDAP simple authentication, the client specifies the DN and password for the user. This is by far the most common authentication mechanism, and in most cases it is also the easiest to use. However, it has a number of limitations, including the following:

- The user is always required to provide a full DN, rather than something that could be more user-friendly like a username.
- Only password-based authentication is allowed.
- The client must provide the complete clear-text password to the server.

To address these issues, it is also possible to authenticate clients through the Simple Authentication and Security Layer (SASL), as defined in RFC 4422 (`http://www.ietf.org/rfc/rfc4422.txt`). This is a very extensible framework, and makes it possible for servers to support many different kinds of authentication.

Supported SASL Mechanisms

The directory server currently supports the following SASL mechanisms:

ANONYMOUS	This mechanism does not actually authenticate clients, but does provide a mechanism for including trace information in server logs for debugging purposes.
CRAM-MD5	This mechanism is provided for backward compatibility only. Do not configure CRAM-MD5 in a production environment. Use the DIGEST-MD5 mechanism instead, because it provides much better security.
DIGEST-MD5	This mechanism provides the ability for clients to use password-based authentication without sending the password to the server. Instead, the client only needs to provide information that proves it knows the password. This mechanism offers more options and better security than the CRAM-MD5 mechanism.
EXTERNAL	This mechanism provides the ability for clients to identify themselves based on information provided outside of the direct flow of LDAP communication. In OpenDS, this may be achieved through the use of SSL client certificates.
GSSAPI	This mechanism provides the ability for clients to authenticate to the server through their participation in a Kerberos V5 environment.
PLAIN	This mechanism uses a password based authentication, but does offer the ability to use a username rather than requiring a DN.

Support for additional SASL mechanisms can be added by implementing custom SASL mechanism handlers in the server..

Because SASL mechanisms are so extensible, the set of information that the client needs to provide to the server in order to perform the authentication varies from one mechanism to another. As such, OpenDS clients use a generic interface for users to provide this information. This is exposed through the `-o` or `--saslOption` argument, and the value for this argument should be a name-value pair. Select which SASL mechanism to use using the `mech` option, for example:

```
--saslOption mech=DIGEST-MD5
```

The other options that are available for use depend on the SASL mechanism that has been chosen, as described in the following sections.

Authorization IDs

Many of the SASL mechanisms below provide the ability to identify a user based on an authorization ID rather than a user DN. An authorization ID may be given in one of two forms:

`dn:`*dn* This is used to provide the full DN of the user to authenticate (for example, `dn:uid=john.doe,ou=People,dc=example,dc=com`). A value of `dn:` with no DN is to be treated as the anonymous user, although this form is not accepted by many of the SASL mechanisms listed below.

`u:`*username* This is used to provide the username of the user rather than the full DN (for example, `u:john.doe`).

If the `u:`*username* form is used, the mechanism that the server uses to resolve that username to the corresponding user entry is based on the identity mapping configuration within the server.

SASL Options for the ANONYMOUS Mechanism

Because the ANONYMOUS mechanism is not really used to perform authentication, no additional options are required. However, the following option can be supplied:

`trace` This option can be used to provide a trace string that is written to the server's access log. This can be useful for debugging or to identify the client, although without authentication it is not possible to rely on the validity of this value.

The following command demonstrates the use of SASL anonymous authentication:

```
$ ldapsearch --hostname server.example.com --port 1389 --saslOption mech=ANONYMOUS \
  --saslOption "trace=Example Trace String" --baseDN "" \
  --searchScope base "(objectClass=*)"
```

SASL Options for the CRAM-MD5 Mechanism

The CRAM-MD5 mechanism is used to perform password-based authentication to the server without exposing the clear-text password. It does this by providing an MD5 digest of the clear-text password combined with some randomly-generated data provided by the server, which helps prevent replay attacks.

The SASL CRAM-MD5 mechanism has one SASL option that must be provided:

`authid` This specifies the identity of the user that is authenticating to the server. It should be an authorization ID value as described above.

The password is specified using either the `--bindPassword` or `--bindPasswordFile` option, just as when using simple authentication. The following command demonstrates the use of SASL CRAM-MD5 authentication:

```
ldapsearch --hostname server.example.com --port 1389 --saslOption mech=CRAM-MD5 \
--saslOption authid=u:john.doe --baseDN "" --searchScope base "(objectClass=*)"
```

SASL Options for the DIGEST-MD5 Mechanism

The DIGEST-MD5 mechanism is similar to the CRAM-MD5 mechanism, but it is more secure because it combines random data from both the client and the server in order to help foil both replay and man-in-the-middle attacks. DIGEST-MD5 authentication also offers a number of SASL options, including the following:

`authid` Specifies the identity of the user that is authenticating to the server. This option must be provided.

`realm` This option should not be specified as a DN.

Note – Do not use the `realm` option, because the server does not use it when mapping identities.

`digest-uri` Specifies the digest URI that the client uses to communicate with the server. This is an optional parameter, but if it is provided, specify it in the form `ldap/`*serveraddress*, where *serveraddress* is the fully-qualified address of the server.

Note – Do not use the `digest-uri` option in a production environment.

`authzid` Specifies the authorization ID that should be used during the authentication process. This option can be used to indicate that the operations requested on the connection after authentication should be performed under the authority of another user.

The password is specified using either the `--bindPassword` or `--bindPasswordFile` option, just as when using simple authentication. The following command demonstrates the use of SASL DIGEST-MD5 authentication:

```
$ ldapsearch --hostname server.example.com --port 1389 --saslOption mech=DIGEST-MD5 \
  --saslOption authid=u:john.doe --saslOption realm=dc=example,dc=com --baseDN "" \
  --searchScope base "(objectClass=*)"
```

SASL Options for the EXTERNAL Mechanism

The EXTERNAL mechanism is used to perform authentication based on information that is available to the server outside of the LDAP session. At present, this is available only through SSL client authentication, in which case the information that the client's SSL certificate will be used to authenticate that client. As such, it is necessary to use SSL or StartTLS when communicating with the server, and a client certificate keystore must be available.

The EXTERNAL mechanism does not support any additional SASL options. In most cases, it can be requested using either `--saslOption mech=EXTERNAL` or `--useSASLExternal`. The following command demonstrates the use of SASL EXTERNAL authentication:

```
$ ldapsearch --hostname server.example.com --port 1636 --useSSL \
  --keyStorePath /path/to/key.store --keyStorePasswordFile /path/to/key.store.pin \
  --trustStorePath /path/to/trust.store --saslOption mech=EXTERNAL --baseDN "" \
  --searchScope base "(objectClass=*)"
```

For more information, see "Configuring SASL External Authentication" on page 68.

SASL Options for the GSSAPI Mechanism

The GSSAPI mechanism is used to perform authentication in a Kerberos V5 environment, and generally requires that the client system be configured to participate in such an environment. The options available for use with the GSSAPI mechanism include:

`authid`	Specifies the authentication ID that should be used to identify the user. This ID should be in the form of a Kerberos principal and not in the authorization ID form described previously. This option must be provided if the user has not authenticated to Kerberos before attempting to bind.
`authzid`	Specifies the authorization ID that should be used to identify the user under whose authority operations should be performed. The directory server does not yet support this capability.
`quality-of-protection`	Specifies the quality of protection to use for the communication. Currently, only the `auth` quality-of-protection value is supported by the directory server clients. The `auth-int` and `auth-conf` values are supported by the server.

If the user already has a valid Kerberos ticket on the system when attempting to use GSSAPI, the client attempts to use it so that no password is required. However, if the user does not have a valid Kerberos ticket or if it cannot be accessed for some reason, a password must be provided using either the `--bindPassword` or `--bindPasswordFile` options.

The following command demonstrates the use of SASL GSSAPI authentication for a user that already has a valid Kerberos session:

```
$ ldapsearch --hostname server.example.com --port 1389 --saslOption mech=GSSAPI \
--saslOption authid=jdoe@EXAMPLE.COM --baseDN "" --searchScope base "(objectClass=*)"
```

SASL Options for the PLAIN Mechanism

The PLAIN mechanism provides many of the same capabilities as LDAP simple authentication, although the user may be identified in the form of an authorization ID rather than requiring a full DN. The following options are available for use when using SASL PLAIN authentication:

`authid` Specifies the identity of the user that is authenticating to the server. It should be an authorization ID value as described above. This option must be provided.

`authzid` Specifies the identity of the user under whose authority operations should be performed. It should also be in the form of an authorization ID. The directory server does not yet support this capability.

The password is specified using either the `--bindPassword` or `--bindPasswordFile` option, just as when using simple authentication. The following command demonstrates the use of SASL PLAIN authentication:

```
$ ldapsearch --hostname server.example.com --port 1389 --saslOption mech=PLAIN \
--saslOption authid=u:john.doe --baseDN "" --searchScope base "(objectClass=*)"
```

Configuring SASL Authentication

This section describes the requirements for configuring directory server to use the various SASL authentication mechanisms.

Configuring SASL External Authentication

The SASL EXTERNAL mechanism is used to allow a client to authenticate itself to the directory server using information provided outside of what is strictly considered LDAP communication. The directory server currently supports authentication using a client certificate presented to the server during SSL or StartTLS negotiation, for LDAP communication only.

Configuring the LDAP Connection Handler to Allow SASL EXTERNAL Authentication

For the directory server to be able to map the client certificate to a user entry, ensure that the connection handler is configured to handle client certificates. Use the `dsconfig` to set the following LDAP connection handler properties:

- **ssl-client-auth-policy.** Specifies whether the directory server prompts the client to present its own certificate during the SSL or StartTLS negotiation process. To support SASL EXTERNAL authentication, the value must be either `optional` or `required`. If the value is `disabled`, clients are not prompted to provide a certificate and no certificate is available for authentication.
- **trust-manager-provider.** Specifies the DN of the trust manager provider used to determine whether the directory server trusts the validity of the client certificate. If the directory server does not trust the client certificate, the SSL or StartTLS negotiation fails and it is not possible for the client to request SASL EXTERNAL authentication. If the directory server trusts illegitimate client certificates, it is possible for malicious users to forge certificates and impersonate any user in the directory. In most cases, the JKS or PKCS12 trust manager provider should be used and the corresponding trust store loaded only with the issuer certificates that are used to sign client certificates.

Note – The `dsconfig` command accesses the server configuration over SSL via the administration connector. As such, the relevant connection options must be specified, including how the SSL certificate is trusted. These examples use the `-X` option to trust all certificates.

The following example uses `dsconfig` to set LDAP connection handler properties:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager"-w password -X -n \
  set-connection-handler-prop --handler-name "LDAP Connection Handler"
```

Configuring the EXTERNAL SASL Mechanism Handler

SASL EXTERNAL bind requests are processed by the SASL mechanism handler. Use the `dsconfig` command to set the following SASL mechanism handler properties:

- **java-class.** Specifies the fully-qualified name of the Java class that provides the logic for the SASL mechanism handler. For the EXTERNAL mechanism, this value is always `org.opends.server.extensions.ExternalSASLMechanismHandler`. An advanced property.
- **enabled.** Indicates whether the EXTERNAL SASL mechanism is enabled for use in the directory server. If you do not want to allow clients to use SASL EXTERNAL authentication, change its value to `false`.

- **certificate-mapper.** Specifies the DN of the configuration entry for the certificate mapper to be used to map client certificates to user entries.
- **certificate-validation-policy.** Specifies whether the directory server attempts to locate the client certificate in the user's entry after establishing a mapping. If the value is `always`, the authentication succeeds only if the mapped user's entry contains the certificate presented by the client. If the value is `ifpresent` (the default value) and the user's entry contains one or more certificates, the authentication succeeds only if one of those certificates matches the one presented by the client. If the value is `ifpresent` and the user's entry does not contain any certificates, the authentication still succeeds based on the fact that it would have been accepted by the trust manager and mapped by the certificate mapper. If the value is `never`, the server does not attempt to match the certificate to a value in the user's entry even if that entry contains one or more certificates.
- **certificate-attribute.** Specifies the name of the attribute that holds user certificates to be examined if the `certificate-validation-policy` property has a value of either `always` or `ifpresent`.

The following example uses `dsconfig` to set EXTERNAL SASL mechanism handler properties:

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager"-w password -X -n \
  set-sasl-mechanism-handler-prop --handler-name "EXTERNAL" --advanced
```

Configuring SASL DIGEST-MD5 Authentication

This section explains the access control and privilege restrictions on a user using the authorization ID keyword (`authzid`). If the user is not using the `authzid` keyword, these restrictions do not apply. Any user that binds using DIGEST-MD5 and the `authzid` keyword must fulfill the following requirements:

- The authentication ID (`authid`) must be granted access by an ACI that grants it the proxy right to the authorization ID.
- The authentication ID (`authid`) entry must contain the `proxied-auth` privilege. The following example creates a test environment and demonstrates the requirements for user authentication using the DIGEST-MD5 SASL mechanism.

The following example creates a test environment and then demonstrates the requirements for a user authentication using the DIGEST-MD5 SASL mechanism.

1. Import the following entries into the directory. These entries define an ACI and three users:
   - The entry `uid=user.0,ou=People,dc=example,dc=com` does not have the `proxied-auth` privilege but is granted proxy access by the ACI.
   - The entry `uid=user.1,ou=People,dc=example,dc=com` has the `proxied-auth` privilege but is not granted proxy access by the ACI.

- The entry `uid=user.2,ou=People,dc=example,dc=com` has the `proxied-auth` privilege and is granted proxy access by the ACI.

```
dn: ou=People,dc=example,dc=com
objectClass: top
objectClass: organizationalunit
objectClass: posixGroup
ou: People
aci: (target="ldap:///uid=proxy user,ou=People,dc=example,dc=com") \
 (targetattr="*") (version 3.0; acl "allow SASL Example"; \
 allow (proxy) userdn="ldap:///uid=user.0,ou=People,dc=example,dc=com ||
 ldap:///uid=user.2,ou=People,dc=example,dc=com";)

dn: uid=user.0,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
...
description: This is the description for user.0

dn: uid=user.1,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
...
description: This is the description for user.1
ds-privilege-name: proxied-auth

dn: uid=proxy user,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
...
description: This is the description for proxy user

dn: uid=user.2,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
...
description: This is the description for user.2
ds-privilege-name: proxied-auth
```

2. Bind using DIGEST-MD5 as `uid=user.1,ou=People,dc=example,dc=com`:

```
$ ldapsearch --port 1389 -w password --saslOption mech=DIGEST-MD5 \
  --saslOption authid=dn:uid=user.1,ou=People,dc=example,dc=com --saslOption \
  authzid=dn:uid=proxy user,ou=People,dc=example,dc=com --baseDN "" \
  --searchScope base "(objectClass=*)"
 The SASL DIGEST-MD5 bind attempt failed Result Code: 49 (Invalid Credentials)
```

The search fails because `uid=user.1,ou=People,dc=example,dc=com` is not granted the proxy right by the ACI.

3. Bind using DIGEST-MD5 as `uid=user.0,ou=People,dc=example,dc=com`:

```
$ ldapsearch --port 1389 -w password --saslOption mech=DIGEST-MD5 \
  --saslOption authid=dn:uid=user.0,ou=People,dc=example,dc=com --saslOption \
  authzid=dn:uid=proxy user,ou=People,dc=example,dc=com --baseDN "" \
  --searchScope base "(objectClass=*)"
The SASL DIGEST-MD5 bind attempt failed Result Code: 49 (Invalid Credentials)
```

The search fails because `uid=user.0,ou=People,dc=example,dc=com` does not have the `proxied-auth`property.

4. Bind using DIGEST-MD5 as `uid=user.2,ou=People,dc=example,dc=com` authid with both access control access and the `proxied-auth` privilege:

```
$ ldapsearch --port 1389 -w password --saslOption mech=DIGEST-MD5 \
  --saslOption authid=dn:uid=user.2,ou=People,dc=example,dc=com --saslOption \
  authzid=dn:uid=proxy user,ou=People,dc=example,dc=com --baseDN "" \
  --searchScope base "(objectClass=*)"
dn:
objectClass: ds-root-dse
objectClass: top
```

The search succeeds because `uid=user.2,ou=People,dc=example,dc=com` has access allowed by the ACI and the `proxied-auth` privilege.

Configuring SASL GSSAPI Authentication

This section explains the access control and privilege restrictions on a user using the authorization ID keyword (`authzid`). If the user is not using the `authzid` keyword, the restrictions do not apply.

Any user that binds using GSSAPI must fulfill the following requirements:

- The authentication ID (`authid`) must be granted access by an ACI that grants it the proxy right to the authorization ID.
- The authentication ID (`authid`) entry must contain the `proxied-auth` privilege.

The following example creates a test environment with three example entries and demonstrates the requirements for user authentication using the GSSAPI SASL mechanism. These examples require a fully configured Kerberos environment, including a valid keytab file.

1. Create three Kerberos principals in the realm `TESTLOCAL.NET`:
   - `user.0@TESTLOCAL.NET`
   - `user.1@TESTLOCAL.NET`
   - `user.2@TESTLOCAL.NET`
2. Configure the GSSAPI SASL handler to be enabled, to use the regular expression identity mapper, and to use a valid TESTLOCAL.NET keytab file.

   ```
   $ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X \
     set-sasl-mechanism-handler-prop --handler-name "GSSAPI" \
     --set enabled:true --set identity-mapper:"Regular Expression" \
     --set keytab:keytabPath
   ```

   The default value of the GSSAPI `enabled` property is `false`, so it must be set to `true`. The default value of `identity-mapper` is Regular Expression. The default value of the `keytab` property is `/etc/krb5/krb5.keytab`.
3. Import the following entries into the directory. These entries define an ACI and three users:
   - The entry `uid=user.0,ou=People,dc=example,dc=com` does not have the `proxied-auth` privilege but is granted proxy access by the ACI.
   - The entry `uid=user.1,ou=People,dc=example,dc=com` has the `proxied-auth` privilege but is not granted proxy access by the ACI.
   - The entry `uid=user.2,ou=People,dc=example,dc=com` has the `proxied-auth` privilege and is granted proxy access by the ACI.

   ```
   dn: ou=People,dc=example,dc=com
   objectClass: top
   objectClass: organizationalunit
   objectClass: posixGroup
   ou: People
   aci: (target="ldap:///uid=proxy user,ou=People,dc=example,dc=com") \
    (targetattr="*") (version 3.0; acl "allow SASL Example"; \
    allow (proxy) userdn="ldap:///uid=user.0,ou=People,dc=example,dc=com"
    || "ldap:///uid=user.2,ou=People,dc=example,dc=com";)


   dn: uid=user.0,ou=People,dc=example,dc=com
   objectClass: top
   objectClass: person
   objectClass: organizationalperson
   objectClass: inetorgperson
   uid=user.0
   ```

```
...
description: This is the description for user.0

dn: uid=user.1,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
uid=user.1
...
description: This is the description for user.1
ds-privilege-name: proxied-auth

dn: uid=user.2,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
uid=user.2
...
description: This is the description for user.2
ds-privilege-name: proxied-auth

dn: uid=proxy user,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalperson
objectClass: inetorgperson
uid=proxy user
...
description: This is the description for proxy user
```

4. Run this command to demonstrate a failing GSSAPI SASL bind using the Kerberos principal, `user.0@TESTLOCAL.NET`:

```
$ ldapsearch --port 1389 \
  --saslOption mech=GSSAPI \
  --saslOption authid=user.1@TESTLOCAL.NET \
  --saslOption authzid=dn:uid=proxy user,ou=People,dc=example,dc=com \
  --baseDN "" --searchScope base "(objectClass=*)"
The SASL DIGEST-MD5 bind attempt failed
Result Code:  49 (Invalid Credentials)
```

This search fails because `user.0@TESTLOCAL.NET` maps to `uid=user.0,ou=People,dc=example,dc=com`, which has the `proxied-auth` privilege but does not have access control permissions to `uid=proxy user,ou=People,dc=example,dc=com`.

5. Run this command to demonstrate a failing GSSAPI SASL bind using the Kerberos principal, `user.1@TESTLOCAL.NET`.

```
$ ldapsearch --port 1389 \
  --saslOption mech=GSSAPI \
  --saslOption authid=user.2@TESTLOCAL.NET \
  --saslOption authzid=dn:uid=proxy user,ou=People,dc=example,dc=com \
  --baseDN "" --searchScope base "(objectClass=*)"
The SASL DIGEST-MD5 bind attempt failed
Result Code:  49 (Invalid Credentials)
```

   This search fails because `user.1@TESTLOCAL.NET` maps to `uid=user.1,ou=People,dc=example,dc=com`, which has access control permissions to `uid=proxy user,ou=People,dc=example,dc=com` but does not have the `proxied-auth` privilege.

6. Run this command to demonstrate a successful GSSAPI SASL bind using the Kerberos principal `user.2@TESTLOCAL.NET`:

```
$ ldapsearch --port 1389 \
  --saslOption mech=GSSAPI \
  --saslOption authid=user.0@TESTLOCAL.NET \
  --saslOption authzid=dn:uid=proxy user,ou=People,dc=example,dc=com \
  --baseDN "" --searchScope base "(objectClass=*)"
dn:
objectClass: ds-root-dse
objectClass: top }}} \\ \\
```

   This search succeeds because `user.2@TESTLOCAL.NET` maps to `uid=user.2,ou=People,dc=example,dc=com`, which has both the `proxied-auth` privilege and access control permission to `id=proxy user,ou=People,dc=example,dc=com`.

Configuring Kerberos and the Sun OpenDS Standard Edition Directory Server for GSSAPI SASL Authentication

The following sections describe how to configure the directory server and Kerberos Version 5 for GSSAPI SASL authentication.

To Configure Kerberos V5 on a Host

You must configure Kerberos V5 on the host machine where your LDAP clients will run.

1. **Install Kerberos V5 according to its installation instructions.**

   Sun recommends installing the Sun Enterprise Authentication Mechanism 1.0.1 client software.

2. **Configure the Kerberos software.**

   Using the Sun Enterprise Authentication Mechanism software, configure the files under `/etc/krb5`. This configuration sets up the `kdc` server, and defines the default realm and any other configuration required by your Kerberos system.

3. **If necessary, modify the file `/etc/gss/mech` so that the first value that is listed is `kerberos_v5`.**

To Specify SASL Options for Kerberos Authentication

You must specify appropriate SASL options for the Kerberos installation.

1. **Before using a client application that is enabled with the GSSAPI mechanism, initialize the Kerberos security system with your user Principal.**

   ```
   $ kinit user-principal
   ```

   where the *user-principal* is your SASL identity, for example, `bjensen@example.com`.

2. **Specify SASL options for using Kerberos.**

   Note that in the UNIX environment, you must set the `SASL_PATH` environment variable to the correct path for the SASL libraries. For example in the Korn shell:

   ```
   $ export SASL_PATH=SASL-library
   ```

   This path assumes that the Sun OpenDS Standard Edition software is installed on the same host where the LDAP tools are invoked.

   The following example of the `ldapsearch` tool shows the use of the `-o` (lowercase letter o) option to specify SASL options for using Kerberos:

   ```
   $ ldapsearch -h www.host1.com -p 1389 -o mech=GSSAPI -o authid="bjensen@EXAMPLE.COM" \
    -o authzid="bjensen@EXAMPLE.COM" -b "dc=example,dc=com" "(givenname=Richard)"
   ```

   The `authid` can be omitted because it is present in the Kerberos cache that was initialized by the `kinit` command. If `authid` is present, `authid` and `authzid` must be identical, although the `authzid` intended for proxy operations is not used. The value of `authid` is the Principal that is used in identity mapping. The Principal must be the full Principal, including the realm.

Example Configuration of Kerberos Authentication Using GSSAPI With SASL

Configuring Kerberos for the Sun OpenDS Standard Edition directory server can be complicated. Your first point of reference should be the Kerberos documentation.

For more help, use the following example procedure to get an idea of which steps to follow. Be aware, however, that this procedure is an example. You must modify the procedure to suit your own configuration and your own environment.

Additional information about configuring and using Kerberos in the Solaris OS can be found in *System Administration Guide: Security Services*. This guide is a part of the Solaris documentation set. You can also consult the man pages.

Information about this example and the steps used are as follows:

Assumptions for This Example

This example procedure describes the process of configuring one machine to operate as a Key Distribution Center (KDC), and a second machine to run the directory server. The result of this procedure is that users can perform Kerberos authentication through GSSAPI.

It is possible to run both the KDC and the directory server on the same machine. If you choose to run both on the same machine, use the same procedure, but omit the steps for the directory server machine that have already been done for the KDC machine.

This procedure makes a number of assumptions about the environment that is used. When using the example procedure, modify the values accordingly to suit your environment. These assumptions are:

- This system has a fresh installation of the Solaris 10 software with the latest recommended patch cluster installed. Kerberos authentication to the directory server can fail if the appropriate Solaris patches are not installed.
- The machine that is running the Kerberos daemons has the fully qualified domain name of `kdc.example.com`. The machine must be configured to use DNS as a naming service. This configuration is a requirement of Kerberos. Certain operations might fail if other naming services such as `file` are used instead.
- The machine that is running the directory server has the fully qualified domain name of `directory.example.com`. This machine must also be configured to use DNS as a naming service.
- The directory server machine serves as the client system for authenticating to the directory server through Kerberos. This authentication can be performed from any system that can communicate with both the directory server and Kerberos daemons. However, all of the necessary components for this example are provided with the Sun OpenDS Standard Edition directory server, and the authentication is performed from that system.
- Users in the directory server have DNs of the form `uid=`*username*`,ou=People,dc=example,dc=com`. The corresponding Kerberos principal is *username*`@EXAMPLE.COM`. If a different naming scheme is used, a different GSSAPI identity mapping must be used.

All Machines: Edit the Kerberos Client Configuration File

The `/etc/krb5/krb5.conf` configuration file provides information that Kerberos clients require in order to communicate with the KDC.

Edit the `/etc/krb5/krb5.conf` configuration file on the KDC machine, the directory server machine, and any client machines that will authenticate to the directory server using Kerberos.

- Replace every occurrence of "`___default_realm___`" with "`EXAMPLE.COM`".
- Replace every occurrence of "`___master_kdc___`" with "`kdc.example.com`".
- Remove the lines that contain "`___slave_kdcs___`" as there will be only a single Kerberos server.
- Replace "`___domain_mapping___`" with "`.example.com = EXAMPLE.COM`" (note the initial period in `.example.com`).

The updated `/etc/krb5/krb5.conf` configuration file should look like the contents of the following example.

EXAMPLE 2 Edited Kerberos Client Configuration File `/etc/krb5/krb5.conf`

```
#pragma ident   "@(#)krb5.conf  1.2     99/07/20 SMI"
# Copyright (c) 1999, by Sun Microsystems, Inc.
# All rights reserved.
#
# krb5.conf template
# In order to complete this configuration file
# you will need to replace the __<name\>__ placeholders
# with appropriate values for your network.
#

[libdefaults]
        default_realm = EXAMPLE.COM
[realms]
        EXAMPLE.COM = {
                kdc = kdc.example.com
                admin_server = kdc.example.com
        }
[domain_realm]
        .example.com = EXAMPLE.COM
[logging]
        default = FILE:/var/krb5/kdc.log
        kdc = FILE:/var/krb5/kdc.log
        kdc_rotate = {

# How often to rotate kdc.log. Logs will get rotated no more
# often than the period, and less often if the KDC is not used
# frequently.
                period = 1d

# how many versions of kdc.log to keep around (kdc.log.0, kdc.log.1, ...)
                versions = 10
        }

[appdefaults]
        kinit = {
                renewable = true
                forwardable= true
        }
        gkadmin = {
                help_url =
 http://docs.sun.com:80/ab2/coll.384.1/SEAM/@AB2PageView/1195
        }
```

All Machines: Edit the Administration Server ACL Configuration File

Replace "___default_realm___" with "EXAMPLE.COM" in the /etc/krb5/kadm5.acl configuration file. The updated file should look like the following example.

EXAMPLE 3 Edited Administration Server ACL Configuration File

```
#
# Copyright (c) 1998-2000 by Sun Microsystems, Inc.
# All rights reserved.
#
# pragma ident   "@(#)kadm5.acl  1.1     01/03/19 SMI"
*/admin@EXAMPLE.COM *
```

KDC Machine: Edit the KDC Server Configuration File

Edit the /etc/krb5/kdc.conf file to replace "___default_realm___" with "EXAMPLE.COM". The updated file should look like the following example.

EXAMPLE 4 Edited KDC Server Configuration File /etc/krb5/kdc.conf

```
# Copyright 1998-2002 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#ident  "@(#)kdc.conf   1.2     02/02/14 SMI"

[kdcdefaults]
        kdc_ports = 88,750

[realms]
        EXAMPLE.COM = {
                profile = /etc/krb5/krb5.conf
                database_name = /var/krb5/principal
                admin_keytab = /etc/krb5/kadm5.keytab
                acl_file = /etc/krb5/kadm5.acl
                kadmind_port = 749
                max_life = 8h 0m 0s
                max_renewable_life = 7d 0h 0m 0s
                default_principal_flags = +preauth
        }
```

KDC Machine: Create the KDC Database

```
$ /usr/sbin/kdb5_util create -r EXAMPLE.COM -s
Initializing database '/var/krb5/principal' for realm 'EXAMPLE.COM',
master key name 'K/M@EXAMPLE.COM'
You will be prompted for the database Master Password.
```

```
It is important that you NOT FORGET this password.
Enter KDC database master key: password
Re-enter KDC database master key to verify: password
$
```

KDC Machine: Create an Administration Principal and Keytab

Use the following command to create an administration user with a Principal of `kws/admin@EXAMPLE.COM` and service keys that will be used by the administration daemon.

```
$ /usr/sbin/kadmin.local
kadmin.local:  add_principal kws/admin
Enter password for principal "kws/admin@EXAMPLE.COM": secret
Re-enter password for principal "kws/admin@EXAMPLE.COM": secret
Principal "kws/admin@EXAMPLE.COM" created.
kadmin.local:  ktadd -k /etc/krb5/kadm5.keytab kadmin/kdc.example.com
Entry for principal kadmin/kdc.example.com with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/kadm5.keytab.
kadmin.local:  ktadd -k /etc/krb5/kadm5.keytab changepw/kdc.example.com

Entry for principal changepw/kdc.example.com with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/kadm5.keytab.
kadmin.local:  ktadd -k /etc/krb5/kadm5.keytab kadmin/changepw
Entry for principal kadmin/changepw with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/kadm5.keytab.
kadmin.local:  quit$
```

KDC Machine: Start the Kerberos Daemons

The Kerberos daemons are managed by the Service Management Facility (SMF) framework. Run the following commands to start the KDC and administration daemons:

```
$ /etc/init.d/kdc start
$ /etc/init.d/kdc.master start
$

$ svcadm disable network/security/krb5kdc
$ svcadm enable network/security/krb5kdc
$ svcadm disable network/security/kadmin
$ svcadm enable network/security/kadmin
$
```

The KDC process appears in the process list as `/usr/lib/krb5/krb5kdc`. The administration daemon appears as `/usr/lib/krb5/kadmind`.

KDC Machine: Add Host Principals for the KDC and OpenDS Machines

Use the following sequence of commands to add host Principals to the Kerberos database for the KDC and the directory server machines. The host Principal is used by certain Kerberos utilities such as `klist`.

```
$ /usr/sbin/kadmin -p kws/admin
Enter Password: secret
kadmin:  add_principal -randkey host/kdc.example.com
Principal "host/kdc.example.com@EXAMPLE.COM" created.
kadmin:  ktadd host/kdc.example.com
Entry for principal host/kdc.example.com with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:  add_principal -randkey host/directory.example.com
Principal "host/directory.example.com@EXAMPLE.COM" created.
kadmin:  ktadd host/directory.example.com
Entry for principal host/directory.example.com with kvno 3, encryption type
 DES-CBC-CRC added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:  quit
$
```

KDC Machine: Add an LDAP Principal for the Directory Server

For the directory server to be able to validate the Kerberos tickets that are held by authenticating users, the directory server must have its own Principal. Currently, the Sun OpenDS Standard Edition directory server is hard coded to require a Principal of `ldap/`*fqdn*`@`*realm* where *fqdn* is the fully-qualified domain name of the directory server and *realm* is the Kerberos realm. The *fqdn* must match the fully qualified name that is provided when you install the directory server. In this case, the Principal for the directory server would be `ldap/directory.example.com@EXAMPLE.COM`.

Use the following sequence of commands to create an LDAP Principal for the directory server:

```
$ /usr/sbin/kadmin -p kws/admin
Enter Password: secret
kadmin: add_principal -randkey ldap/directory.example.com
Principal "ldap/directory.example.com@EXAMPLE.COM" created.
kadmin: quit
$
```

KDC Machine: Add a Test User to the KDC

To perform Kerberos authentication, the user authenticating must exist in the Kerberos database. In this example, the user has the user name `kerberos-test`, which means that the Kerberos Principal is `kerberos-test@EXAMPLE.COM`.

Create the user by using the command sequence in this example:

```
$ /usr/sbin/kadmin -p kws/admin
Enter Password: secret
kadmin:  add_principal kerberos-test
Enter password for principal "kerberos-test@EXAMPLE.COM": secret

Re-enter password for principal "kerberos-test@EXAMPLE.COM": secret

Principal "kerberos-test@EXAMPLE.COM" created.
kadmin:  quit
$
```

Directory Server Machine: Install the Directory Server

Install the directory server using at least version 1.3. Versions of the directory server earlier than 1.3 do not offer full support of GSSAPI SASL. The following table lists the installation settings that this section uses in examples.

Variable Type	Example Value
Fully qualified directory server DNS name	`directory.example.com`
Server port	`389`
Suffix	`dc=example,dc=com`
Installation directory	`/opt/opends`
OpenDS server user	`opends`
OpenDS server group	`opends`
Kerberos test principal	`kerberos-test`
OpenDS keytab path	`/opt/opends/config/opends.keytab`

Note – The fully qualified directory server DNS name must resolve to the same IP address on all of the servers, namely the OpenDS servers and the Kerberos Key Distribution Center (KDC) and client machines that expect to bind to the server using GSSAPI SASL.

Directory Server Machine: Create and Configure the Directory Server LDAP Principal

As mentioned previously, to authenticate Kerberos users through GSSAPI, the directory server must have its own Principal in the KDC. For authentication to work properly, the Principal

information must reside in a Kerberos keytab on the directory server machine. This information must be in a file that is readable by the user account under which the directory server operates.

Note – This step must be performed before the `OpenDS GSSAPI SASL mechanism handler` is configured. The handler checks to make sure the keytab file exists before it will initialize.

Create a keytab file with the correct properties by using the following command sequence:

```
$ kadmin -p kws/admin@EXAMPLE.COM
kadmin:  addprinc -randkey ldap/directory.example.com
WARNING: no policy specified for ldap/directory.example.com@EXAMPLE.COM;
  defaulting to no policy
Principal "ldap/directory.example.com@EXAMPLE.COM" created.
kadmin:  ktadd -k /opt/opends/config/opends.keytab ldap/directory.example.com
Entry for principal ldap/directory.example.com with kvno 3,
  encryption type AES-128 CTS mode
  with 96-bit SHA-1 HMAC added to keytab WRFILE:/opt/opends/config/opends.keytab.
Entry for principal ldap/directory.example.com with kvno 3,
  encryption type Triple DES cbc mode
  with HMAC/sha1 added to keytab WRFILE:/opt/opends/config/opends.keytab.
Entry for principal ldap/directory.example.com with kvno 3,
  encryption type ArcFour with HMAC/md5
  added to keytab WRFILE:/opt/opends/config/opends.keytab.
Entry for principal ldap/directory.example.com with kvno 3,
  encryption type DES cbc mode with RSA-MD5
  added to keytab WRFILE:/opt/opends/config/opends.keytab.
kadmin:  quit
```

Change the permissions and ownership on this custom keytab. Make the keytab owned by the user account used to run the directory server and readable only by that user:

```
$ chown opends:opends /opt/opends/config/opends.keytab
$ chmod 600 /opt/opends/config/opends.keytab
```

Finally, to allow these changes to take effect, use one of the following ways to stop and restart the directory server:

- Click the Restart button on the Control Panel.
- Run the `stop-ds -r` command.

Directory Server Machine: Configure the Directory Server to Enable GSSAPI

This step shows examples of managing the `OpenDS GSSAPI SASL mechanism handler` on the directory server host `directory.example.com`.

Use the `dsconfig` command as shown in the following example to enable the GSSAPI SASL mechanism handler on the directory server host `directory.example.com` and configure it to use the `/opt/opends/config/opends.keytab`.

```
$ dsconfig -X -n -p 4444 -h directory.example.com \
  -D "cn=directory manager" -w password
  set-sasl-mechanism-handler-prop \
  --handler-name GSSAPI \
  --set enabled:true \
  --set keytab:/opt/opends/config/opends.keytab \
  --set server-fqdn:directory.example.com
```

The last line in this command sets the GSSAPI SASL mechanism property `server-fqdn` to `directory.example.com`. This is an optional parameter, which can be left out only if it is assured that a hostname lookup on the directory server host returns the exact hostname that was used in creating the LDAP principal. Setting this property explicitly assures that the two names are the same (in this example, `directory.example.com`).

Confirm that the configuration is correct by examining the properties of the GSSAPI SASL mechanism handler on the directory server host `directory.example.com`.

```
$ dsconfig -X -n -p 4444 -h directory.example.com \
  -D "cn=directory manager" -w password \
  get-sasl-mechanism-handler-prop \
  --handler-name GSSAPI
Property              : Value(s)
----------------------:----------------------
enabled               : true
identity-mapper       : Regular Expression
kdc-address           : -
keytab                : /opt/opends/config/opends.keytab
principal-name        : -
quality-of-protection : none
realm                 : -
server-fqdn           : directory.example.com
```

If necessary for troubleshooting, you can use `dsconfig` to list the status of all the SASL mechanism handlers on the directory server host `directory.example.com`.

```
$ dsconfig -X -n -p 4444 -h directory.example.com \
  -D "cn=directory manager" -w password \
  list-sasl-mechanism-handlers
SASL Mechanism Handler : Type       : enabled
-----------------------:------------:--------
ANONYMOUS              : anonymous  : false
CRAM-MD5               : cram-md5   : true
DIGEST-MD5             : digest-md5 : true
EXTERNAL               : external   : true
```

```
GSSAPI                : gssapi     : true
PLAIN                 : plain      : true
```

If necessary, you can use `dsconfig` to disable the GSSAPI SASL mechanism handler on the directory server host `directory.example.com`.

```
$ dsconfig -X -n -p 4444 -h directory.example.com \
  -D "cn=directory manager" -w password \
  set-sasl-mechanism-handler-prop \
  --handler-name GSSAPI \
  --set enabled:false
```

Directory Server Machine: Add a Test User to the Directory Server

To authenticate a Kerberos user to the directory server, there must be a directory entry for the user that corresponds to the Kerberos Principal for that user.

In a previous step, a test user was added to the Kerberos database with a Principal of `kerberos-test@EXAMPLE.COM`. Because of the identity mapping configuration added to the directory, the corresponding directory entry for that user must have a DN of `uid=kerberos-test,ou=People,dc=example,dc=com`.

Before you can add the user to the directory, you must create the file `testuser.ldif` with the following contents.

EXAMPLE 5 New `testuser.ldif` File

```
dn: uid=kerberos-test,ou=People,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: kerberos-test
givenName: Kerberos
sn: Test
cn: Kerberos Test
description: An account for testing Kerberos authentication through GSSAPI
```

Next, use `ldapmodify` to add this entry to the server:

```
$ ldapmodify -D "cn=Directory Manager" -w - -f testuser.ldif
adding new entry uid=kerberos-test,ou=People,dc=example,dc=com
$
```

Directory Server Machine: Obtain a Kerberos Ticket as the Test User

The test user exists in the Kerberos database, the directory server, and the KDC. Therefore, it is now possible to authenticate as the test user to the directory server over Kerberos through GSSAPI.

First, use the `kinit` command to get a Kerberos ticket for the user, as shown in the following example:

```
$ kinit kerberos-test
Password for kerberos-test@EXAMPLE.COM: secret
$
```

Then, use the `klist` command to view information about this ticket:

```
$ klist
Kerberos 5 ticket cache: 'API:6'
Default principal: kerberos-test@EXAMPLE.COM
Valid Starting     Expires            Service Principal
03/23/09 12:35:05  03/23/09 20:35:05  krbtgt/EXAMPLE.COM@EXAMPLE.COM
renew until 03/30/09 12:34:15
$
```

Client Machine: Authenticate to the Directory Server Through GSSAPI

The final step is to authenticate to the directory server by using GSSAPI. The `ldapsearch` utility provided with The directory server provides support for SASL authentication, including GSSAPI, DIGEST-MD5, and EXTERNAL mechanisms. However, to bind by using GSSAPI you must provide the client with the path to the SASL library. Provide the path by setting the `SASL_PATH` environment variable to the `lib/sasl` directory:

```
$ SASL_PATH=SASL-library
$ export SASL_PATH
$
```

To actually perform a Kerberos-based authentication to the directory server using `ldapsearch`, you must include the `-o mech=GSSAPI` and `-o authzid=`*principal* arguments.

You must also specify the fully qualified host name, shown here as `-h directory.example.com`, which must match the value of the `nsslapd-localhost` attribute on `cn=config` for the server. This use of the `-h` option is needed because the GSSAPI authentication process requires the host name provided by the client to match the host name provided by the server.

The following example retrieves the `dc=example,dc=com` entry while authenticated as the Kerberos test user account created previously:

```
$ ldapsearch -h directory.example.com -p 389 -o mech=GSSAPI \
 -o authzid="kerberos-test@EXAMPLE.COM" -b "dc=example,dc=com" -s base "(objectClass=*)"
version: 1
dn: dc=example,dc=com
dc: example
objectClass: top
objectClass: domain
$
```

Check the directory server access log to confirm that the authentication was processed as expected:

```
$ tail -12 /local/ds/logs/access

[24/Jul/2004:00:30:47 -0500] conn=0 op=-1 msgId=-1 - fd=23 slot=23 LDAP
        connection from 1.1.1.8 to 1.1.1.8
[24/Jul/2004:00:30:47 -0500] conn=0 op=0 msgId=1 - BIND dn="" method=sasl
     version=3 mech=GSSAPI
[24/Jul/2004:00:30:47 -0500] conn=0 op=0 msgId=1 - RESULT err=14 tag=97
     nentries=0 etime=0, SASL bind in progress
[24/Jul/2004:00:30:47 -0500] conn=0 op=1 msgId=2 - BIND dn="" method=sasl
     version=3 mech=GSSAPI
[24/Jul/2004:00:30:47 -0500] conn=0 op=1 msgId=2 - RESULT err=14 tag=97
     nentries=0 etime=0, SASL bind in progress
[24/Jul/2004:00:30:47 -0500] conn=0 op=2 msgId=3 - BIND dn="" method=sasl
     version=3 mech=GSSAPI
[24/Jul/2004:00:30:47 -0500] conn=0 op=2 msgId=3 - RESULT err=0 tag=97
     nentries=0 etime=0 dn="uid=kerberos-test,ou=people,dc=example,dc=com"
[24/Jul/2004:00:30:47 -0500] conn=0 op=3 msgId=4 - SRCH base="dc=example,dc=com"
     scope=0 filter="(objectClass=*)" attrs=ALL
[24/Jul/2004:00:30:47 -0500] conn=0 op=3 msgId=4 - RESULT err=0 tag=101 nentries=1
     etime=0
[24/Jul/2004:00:30:47 -0500] conn=0 op=4 msgId=5 - UNBIND
[24/Jul/2004:00:30:47 -0500] conn=0 op=4 msgId=-1 - closing - U1
[24/Jul/2004:00:30:48 -0500] conn=0 op=-1 msgId=-1 - closed.
$
```

This example shows that the bind is a three-step process. The first two steps return LDAP result `14` (SASL bind in progress), and the third step shows that the bind was successful. The `method=sasl` and `mech=GSSAPI` tags show that the bind used the GSSAPI SASL mechanism. The `dn="uid=kerberos-test,ou=people,dc=example,dc=com"` at the end of the successful bind response shows that the bind was performed as the appropriate user.

Troubleshooting Kerberos Configuration

If the Kerberos installation does not perform as expected, check the following conditions:

- Perform a successful `kinit` using the test principal from the directory server machine to make sure that the directory server can authenticate to the Kerberos KDC.
- Perform a successful `kinit` using the test principal from the client machines to make sure that the client machines can authenticate to the Kerberos KDC.
- Make sure that the directory server's keytab file exists and is readable by the directory server. That is, make sure that the keytab file's ownership and permission settings are correct.
- Make sure that the LDAP principal name in the OpenDS keytab file matches the hostname that the directory server used when it was configured. The following example shows a configuration that fails:

  1. Configure GSSAPI as shown below. The value specified for the `server-fqdn` attribute, `bad.example.com`, does not match the value used in creating the keytab, `directory.example.com`.

     ```
     $ dsconfig -X -n -p 4444 -h directory.example.com \
       -D "cn=directory manager" -w password \
       set-sasl-mechanism-handler-prop \
       --handler-name GSSAPI \
       --set enabled:true \
       --set keytab:/opt/opends/config/opends.keytab \
       --set server-fqdn:bad.example.com
     ```

  2. From a client, attempt an ldapsearch authenticating using GSSAPI.

     ```
     $ ldapsearch -h directory.example.com \
       -o mech=GSSAPI -o authid=kerberos-test@EXAMPLE.COM \
       --searchScope base \
       -b "uid=kerberos-test,ou=people,dc=example,dc=com" "(objectclass=*)"
     An error occurred while attempting to perform GSSAPI authentication to
       the Directory Server: \
     PrivilegedActionException(AccessController.java:-2)
     Result Code:  82 (Local Error)
     ```

     The search fails as expected.

  3. To determine the cause of the search failure, inspect the directory server's access log:

     ```
     $ tail opends/logs/access
     [23/Mar/2009:13:12:59 -0500] CONNECT conn=14 from=129.150.33.77:65076
       to=192.168.0.199:1389 protocol=LDAP
     [23/Mar/2009:13:13:00 -0500] BIND REQ conn=14 op=0 msgID=1
       type=SASL mechanism=GSSAPI dn=""
     [23/Mar/2009:13:13:00 -0500] BIND RES conn=14 op=0 msgID=1
       result=49 authFailureID=1310915   authFailureReason="An unexpected error
       occurred while trying to create an GSSAPI context:
       major code (13) No valid credentials provided,
       minor code (-1)   Failed to find any Kerberos Key" etime=253
     [23/Mar/2009:13:13:00 -0500] DISCONNECT conn=14 reason="Client Disconnect"
     ```

The message in the minor code of the last record in the access log shows that the directory server could not find a match in the keytab file.

4. To fix the situation, disable the handler and then re-enable it with the correct information, as shown in the following example.

```
$ dsconfig -X -n -p 4444 -h directory.example.com \
  -D "cn=directory manager" -w password \
  set-sasl-mechanism-handler-prop \
  --handler-name GSSAPI \
  --set enabled:false
$ dsconfig -X -n -p 4444 -h directory.example.com
  -D "cn=directory manager" -w password \
  set-sasl-mechanism-handler-prop \
  --handler-name GSSAPI \
  --set enabled:true \
  --set keytab:/opt/opends/config/opends.keytab \
  --set server-fqdn:directory.example.com
$ ldapsearch -h directory.example.com \
  -o mech=GSSAPI \
  -o authid=kerberos-test@EXAMPLE.COM \
  --searchScope base \
  -b "uid=kerberos-test,ou=people,dc=example,dc=com" "(objectclass=*)"
dn: uid=kerberos-test,ou=People,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: kerberos-test
givenName: Kerberos
sn: Test
cn: Kerberos Test
description: An account for testing Kerberos authentication through GSSAPI
```

Testing SSL, StartTLS, and SASL Authentication With ldapsearch

The ldapsearch utility included with the directory server is useful for testing that the server is properly configured to support SSL and StartTLS. This utility includes a number of options that are well-suited for testing in a number of different scenarios. This section describes how to use ldapsearch to test SSL and StartTLS communication, and SASL EXTERNAL authentication. The same process can be used with many of the other client tools provided with the directory server, including ldapmodify, ldapcompare, and ldapdelete.

ldapsearch Command Line Arguments Applicable To Security

The following command-line arguments are of particular interest when using the `ldapsearch` tool to communicate via SSL or StartTLS:

- `-h` *address* or `--hostname` *address* Specifies the address of the directory server to which you want to connect. If no value is specified, the IPv4 loopback address (`127.0.0.1`) is used.
- `-p` *port* or `--port` *port* Specifies the port number on which the directory server is listening for connections. If no value is specified, the standard unencrypted LDAP port (`389`) is used.
- `-Z` or `--useSSL` Indicates that the client should use SSL to secure communication with the directory server. If this option is used, the value specified for the port argument must be one on which the server is listening for SSL-based connections. The default LDAPS port is `636`.
- `-q` or `--startTLS` Indicates that the client should use the StartTLS extended operation to secure communication with the directory server. If this option is used, the value specified for the port argument must be the one on which the server is listening for clear-text LDAP connections. Note that the port argument is not required if the server is listening on the default LDAP port (`389`).
- `-r` or `--useSASLExternal` Indicates that the client should use SASL EXTERNAL authentication to authenticate to the directory server. If this option is used, you must also provide a keystore path.
- `-X` or `--trustAll` Indicates that the client should blindly trust any certificate that the directory server presents. This option should not be used in conjunction with the argument used to specify the trust store path.
- `-K` *path* or `--keyStorePath` *path* Specifies the path to the keystore that should be used if the client is to present a certificate to the directory server (for example, when using SASL EXTERNAL authentication). This should be the path to a JKS keystore.
- `-W` *password* or `--keyStorePassword` *password* Specifies the PIN required to access the contents of the key tore. This should not be used in conjunction with the keystore password file argument.
- `--keyStorePasswordFile` *path* Specifies the path to a file containing the PIN required to access the contents of the keystore. This should not be used in conjunction with the keystore password argument.
- `-N` *nickname* or `--certNickname` *nickname* Specifies the nickname, or alias, of the certificate that the client should present to the directory server. The keystore path argument must also be provided. If no nickname is given, then the client will pick the first acceptable client certificate that it finds in the keystore.

- `-P` *path* or `--trustStorePath` *path* Specifies the path to the JKS trust store file that the client should use when determining whether to trust the certificate presented by the directory server. If this argument is not given and the `trustAll` option is not given, then any certificate presented to the client will be displayed and the user will be prompted about whether to trust it.
- `--trustStorePassword` *password* Specifies the password needed to access the trust store contents. In most cases, no trust store password is required. This should not be used in conjunction with the trust store password file option.
- `--trustStorePasswordFile` *path* Specifies the path to a file containing the password needed to access the trust store contents. In most cases, no trust store password is required. This should not be used in conjunction with the trust store password option.
- `-E` or `--reportAuthzID` Indicates that the directory server should include the authorization identity of the authenticated user in the bind response. This is useful when performing SASL authentication to determine the user to which the client certificate (or other form of SASL credentials if a mechanism other than EXTERNAL was used) was mapped.

Testing SSL

The following demonstrates the use of `ldapsearch` to communicate with a directory server using LDAP over SSL:

```
$ ldapsearch --hostname directory.example.com --port 1636 \
--useSSL --baseDN "" --searchScope base "(objectClass=*)"
```

In this case, no trust store was specified, and the `--trustAll` argument was also not given. Therefore, when the server presents its certificate to the client, the user will be prompted about whether that certificate should be trusted. The entire sequence might look something like:

```
$ ldapsearch --hostname directory.example.com --port 1636 \
--useSSL --baseDN "" --searchScope base "(objectClass=*)"

The server is using the following certificate:
Subject DN: CN=directory.example.com, O=Example Corp, C=US
Issuer DN: CN=directory.example.com, O=Example Corp, C=US
Validity: Fri Mar 02 16:48:17 CST 2007 through Thu might 31 17:48:17 CDT 2007
Do you want to trust this certificate and continue connecting to the server?
Please enter "yes" or "no":
dn:
objectClass: ds-rootDSE
objectClass: top
```

If the client simply wants to always trust any certificate that the server presents without being prompted, then the `--trustAll` argument might be provided. For example:

```
$ ldapsearch --hostname directory.example.com --port 1636 \
--useSSL --trustAll --baseDN "" --searchScope base \
"(objectClass=*)"
```

If the client has a trust store and wants to use that to determine whether to trust the server certificate, then the `--trustStorePath` argument might also be given. For example:

```
$ ldapsearch --hostname directory.example.com --port 1636 \
--useSSL --trustStorePath client.truststore --baseDN "" \
--searchScope base "(objectClass=*)"
```

Testing StartTLS

The process for using StartTLS with the `ldapsearch` utility is almost identical to the process for using SSL. The only differences are that you should use the port on which the server is listening for unencrypted LDAP requests and that you should indicate that StartTLS should be used instead of SSL (that is, use `--useStartTLS` instead of `--useSSL`). The following example is the equivalent of the first example given for using SSL with `ldapsearch` except that it uses StartTLS to secure the communication:

```
$ ldapsearch -h directory.example.com --port 1389 \
--useStartTLS --baseDN "" --searchScope base "(objectClass=*)"
```

This applies to all of the other examples given. Simply change the port number from the LDAPS port to the LDAP port, and replace the `--useSSL` option with `--useStartTLS`.

Managing Directory Data

The topics included in this section describe how to add, modify, remove, and search data in the directory server. These topics also describe how to make searches more efficient, by indexing data, how to ensure that entries are unique, and how to use advanced data features such as virtual attributes.

This section includes the following topics:

Importing and Exporting Data

The directory server provides several mechanisms to move data into and out of a specific back end. This chapter outlines the various options and then describes the import and export mechanisms in more detail.

Populating a Stand-Alone Directory Server With Data

To populate a stand-alone directory server with data, use one of the following methods:

- Import the data from an LDAP Data Interchange Format (LDIF) file while you are setting up the server, either by using the `setup` utility in GUI mode or by using the `setup` utility in interactive command-line mode. This is the most convenient method of initializing a stand-alone server or the first server in a replicated topology.
- Start with an empty suffix and add entries by using the `ldapmodify` command, for example:

```
$ ldapmodify -h hostname -p 1389 -D "cn=Directory Manager" -w password \
  -a -f /usr/local/add_entry.ldif
```

- Import data from an LDIF file, using the `import-ldif` command. For example:

  ```
  $ import-ldif -b dc=example,dc=com -n userRoot -l /var/tmp/Example.ldif
  ```

  This method is much more efficient for the addition of bulk entries. The `import-ldif` command imports data from an LDIF file either by replacing any existing data in the suffix or by appending data to a base DN. Similarly, the `export-ldif` command exports entries from a database to an LDIF file, which can then be imported to another server. Both tools support file compression, SASL extension, and client/server authentication using SSL and startTLS.

- Copy the binary database from another server. This method is also called *binary copy*.

  ```
  $ cp instance-path/db/example.db destination-path/db
  ```

- Restore the database from a backup using the `restore` command, for example:

  ```
  $ restore -d /home/backup/userRoot
  ```

Note – Performing a binary database copy or restoring a database from a backup requires the source server and the destination server to have the same database back-end structures and indexes.

Importing Data Using `import-ldif`

The `import-ldif` command is used to populate a directory server back end with data read from an LDIF file or with data generated based on a "Creating MakeLDIF Template Files" on page 111. In most cases, `import-ldif` is significantly faster than adding entries using `ldapmodify`.

The `import-ldif` command supports both LDIF files and compressed files (`.zip`).

Note –

- A complete import to an entire Oracle Berkeley DB Java Edition (JE) back end will have better performance than a partial import to a branch of the JE back end. All imported LDIF files must use UTF-8 character-set encoding.
- Importing *suffixes* is a resource-intensive operation. If you import LDIF files that include a large number of suffixes, your system might have insufficient heap to complete the import operation. Before importing such LDIF files, you should therefore increase the heap as much as possible. For more information, see "Tuning Performance" on page 363 and "Improving Performance When Importing Large Data Sets" on page 367.

You do not need root privileges to import an LDIF file, but you must authenticate as a user with root permissions, such as `cn=Directory Manager`.

`import-ldif` Operation Modes

The `import-ldif` command has two modes of operation: online and offline.

- **Online mode**. In online mode, `import-ldif` contacts a running directory server instance and registers an import task. The command accesses the task back end over SSL via the administration connector. For more information, see "Managing Administration Traffic to the Server" on page 5. Online mode runs automatically when any connection options (such as `--hostname`, `--port`, `--bindDN`, and `--bindPassword`) are specified.

  In general, if you expect to do online imports, you should increase the heap when you start the server. For more information, see "Tuning Performance" on page 363.
- **Offline mode**. When no connection options are specified, the command runs in offline mode. In offline mode, `import-ldif` accesses the database directly rather than through a directory server instance. In this case, the directory server must be stopped.

▼ To Import Data in Offline Mode

This procedure imports a back-end database with new entries specified in an import LDIF file. The command runs in *offline* mode, which requires the server to be shut down prior to import.

1 Stop the server if it is running.

```
$ stop-ds
```

2 Import the LDIF file, as shown in the following example:

```
$ import-ldif -b dc=example,dc=com -n userRoot -l Example.ldif
```

This command specifies the base DN for the branch of the data that should be included in the import (`-b`), the back-end ID into which the data is imported (`-n`), and the LDIF file used for the import (`-l`).

To Replace Existing Data During an Offline Import

The following procedure replaces an existing back-end with new entries specified in an import file.

1. **Stop the server if it is running.**

```
$ stop-ds
```

2. **Import the LDIF file, replacing the existing data. For example:**

```
$ import-ldif --includeBranch dc=example,dc=com --backendID userRoot \
  --replaceExisting --ldifFile Example.ldif
```

To Append Imported Data to Existing Data

The following procedure appends the entries in an import file to the existing entries in the back end.

1. **Stop the server if it is running.**

```
$ stop-ds
```

2. **Import the LDIF file, appending the new data to the existing data. For example:**

```
$ import-ldif --backendID userRoot --append --ldifFile new.ldif
```

To Import Fractional Files

The `import-ldif` command provides options to import a portion of an import file by specifying the base DN to include or exclude during the process.

This example imports all entries below the base DN, `dc=example,dc=com`, and excludes all entries below `ou=People,dc=example,dc=com`.

1. **Stop the server if it is running.**

```
$ stop-ds
```

2. **Import a portion of the LDIF file. For example:**

```
$ import-ldif --includeBranch dc=example,dc=com \
  --excludeBranch ou=People,dc=example,dc=com --backendID userRoot --replaceExisting \
  --ldifFile Example.ldif
```

To Import Fractional Files by Using Filters

The `import-ldif` command provides options to import part of an import file by using filters for data inclusion or exclusion. Make sure that you fully understand how this mechanism works before you use it.

In this example, the contents of an LDIF file are imported, except those entries that match the search filter `l=Auckland` (that is, `location=Auckland`).

Note – The `--includeFilter` option works in a similar manner to `--excludeFilter`, except that it includes all entries that match the search filter during import.

1 **Stop the server if it is running.**

```
$ stop-ds
```

2 **Import a portion of the file by using an exclude filter. For example:**

```
$ import-ldif --excludeFilter "(l=Auckland)" --backendID userRoot \
  --replaceExisting --ldifFile Example.ldif
```

▼ To Include or Exclude Attributes During Import

The `import-ldif` command provides options to include and exclude attributes during import by using the `--includeAttribute` and `--excludeAttribute` options, respectively. Make sure that you fully understand how this mechanism works before you use it.

1 **Stop the server if it is running.**

```
$ stop-ds
```

2 **(Optional) View the entries of the import file before you start the import.**

The directory server provides useful utilities to search, modify, compare, or delete import files without connecting to the server. You can use the `ldifsearch` command to display an entry in your import file. For example, to display the entry for Sam Carter, use the following command:

```
$ ldifsearch -b dc=example,dc=com --ldifFile Example.ldif "(cn=Sam Carter)"
dn: uid=scarter,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: top
givenname: Sam
uid: scarter
cn: Sam Carter
telephonenumber: +1 408 555 4798
sn: Carter
userpassword: sprain
roomnumber: 4612
mail: scarter@example.com
l: Sunnyvale
ou: Accounting
ou: People
facsimiletelephonenumber: +1 408 555 9751
```

In this entry, notice the presence of the roomnumber attribute below the telephonenumber attribute.

3 **Import the file, excluding the roomnumber attribute for all entries.**

```
$ import-ldif --excludeAttribute "roomnumber" --backendID userRoot \
  --replaceExisting --ldifFile Example.ldif
```

4 **Start the server.**

```
$ start-ds
```

5 **Perform an ldapsearch to verify the import.**

The following example shows that the roomnumber attribute is now absent from Sam Carter's entry.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com --bindDN "cn=Directory Manager" \
  --bindPassword password "(cn=Sam Carter)"
dn: uid=scarter,ou=People,dc=example,dc=com \
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
givenName: Sam
uid: scarter
cn: Sam Carter
sn: Carter
telephoneNumber: +1 408 555 4798
ou: Accounting
ou: People
l: Sunnyvale
mail: scarter@example.com
facsimileTelephoneNumber: +1 408 555 9751
```

▼ To Import a Compressed LDIF File

The import-ldif utility supports compressed LDIF files.

1 **Stop the server if it is running.**

```
$ stop-ds
```

2 **Import the compressed LDIF file.**

```
$ import-ldif --includeBranch dc=example,dc=com
  --excludeBranch "ou=People,dc=example,dc=com" --ldifFile Example.ldif \
  --backendID userRoot --replaceExisting --isCompressed
```

To Record Rejected or Skipped Entries During Import

The `import-ldif` command provides a means to write to an output file for any entries that are rejected or skipped during the import process. This enables easy debugging of an LDIF file. Rejected entries occur when the directory server rejects the added entries due to schema violations. Skipped entries occur when entries cannot be placed under the specified base DN.

1. **Stop the server if it is running.**

   ```
   $ stop-ds
   ```

2. **Import the file, using the `--rejectFile` and `--skipFile` options.**

   You can also use the `--overWrite` option to replace any previous items in the two files. Without the option, the directory server appends new rejected and skipped entries to the existing files.

   ```
   $ import-ldif --backendID userRoot --append --ldifFile new.ldif
     --overwrite --rejectFile rejected.ldif --skipFile skipped.ldif
   ```

3. **(Optional) View the contents of the `rejectFile` and `skipFile` to determine which entries were rejected or skipped during the import. For example:**

   ```
   $ more rejected.ldif
   # Entry ou=Contractors,dc=example,dc=com read from LDIF starting at line 1
   is not valid because it violates the server's schema configuration:
   Entry ou=Contractors,dc=example,dc=com violates the Directory Server schema
   configuration because it includes attribute changeType which is not allowed.
   changetype: add objectclasses defined in that entry objectclass: top
   objectclass: organizationalUnit ou: Contractors ou: Product Testing
   ou: Product Dev ou: Accounting ...

   $ more skipped.ldif
   # Skipping entry ou=People,dc=example,dc=com because the DN is not one that should be
     included based on the include and exclude branches objectclass: top
     objectclass: organizationalunit ou: People
     aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr ="userpassword ||
     telephonenumber || facsimiletelephonenumber")(version 3.0;acl "Allow self entry
     modification"; allow (write)(userdn = "ldap:///self");)
     aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr h3.="cn || sn ||
     uid") (targetfilter ="(ou=Accounting)")(version 3.0;acl "Accounting Managers Group
     Permissions"; allow (write)
     (groupdn = "ldap:///cn=Accounting Managers,ou=groups,dc=example,dc=com");)
     aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr h3.="cn || sn ||
     uid") (targetfilter ="(ou=Human Resources)")(version 3.0;acl "HR Group Permissions";
     allow write)(groupdn = "ldap:///cn=HR Managers,ou=groups,dc=example,dc=com");) aci:
     (target ="ldap:///ou=People,dc=example,dc=com")(targetattr h3.="cn ||sn || uid")
     (targetfilter ="(ou=Product Testing)")(version 3.0;acl "QA Group Permissions"; allow
     (write)(groupdn = "ldap:///cn=QA Managers,ou=groups,dc=example,dc=com");)
     aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr h3.="cn || sn ||
     uid") (targetfilter ="(ou=Product Development)")(version 3.0;acl "Engineering Group
   ```

```
Permissions"; allow (write)(groupdn =
"ldap:///cn=PD Managers,ou=groups,dc=example,dc=com");) ...
```

To Import Data From a MakeLDIF Template

The directory server includes the Java utility, makeLDIF, that can be used to generate sample data for import. The makeLDIF utility requires a template file. You can create your own template file, or you can use the template file located in *install-dir*/config/MakeLDIF/example.template, editing it as required. For more information, see "Creating MakeLDIF Template Files" on page 111.

1. **Stop the server if it is running.**

   ```
   $ stop-ds
   ```

2. **Import the data, using a template file.**

   The sample template generates 10,003 sample entries in the specified back end.

   ```
   $ import-ldif --backendID userRoot --templateFile example.template --randomSeed 0
   ```

See Also "make-ldif" in *Sun OpenDS Standard Edition 2.0 Command-Line Usage Guide*

To Run an Import in Online Mode

The import-ldif utility can also be run with the server online. In online mode, the command accesses the task back end over SSL via the administration connector. To run the command in online mode you must specify the relevant connection options, including how the SSL certificate will be trusted. This example uses the -X option to trust all certificates. For more information, see "Managing Administration Traffic to the Server" on page 5.

- **Run the import-ldif command with the appropriate connection options.**

  ```
  $ import-ldif -h localhost -port 4444 -D "cn=Directory Manager" -w password -X \
    -l /ldif-files/example.ldif
  ```

To Schedule an Import

The import-ldif utility provides a --start option for scheduling the import at some future date. You can view this scheduled task by using the manage-tasks utility. The command accesses the task back end over SSL via the administration connector. To schedule an import task, you must specify the relevant connection options, including how the SSL certificate will be trusted. This example uses the -X option to trust all certificates. For more information, see

- **Run the import-ldif command with the --start option.**

  ```
  $ import-ldif -h localhost -port 4444 -D "cn=Directory Manager" -w password -X \
    -l /ldif-files/example.ldif --start 20080124121500
  ```

See Also "Configuring Commands As Tasks" on page 23

Exporting Data Using export-ldif

The export-ldif command is used to export data from a directory server back end. The command is useful for the following tasks:

- Backing up directory data
- Exporting data to another application
- Repopulating a database after a change to the directory topology
- Reinitializing master servers in a replicated topology

Note – The export-ldif command cannot be used to export data from the following back ends: monitor, ads-truststore, backup, and config-file-handler.

---

export-ldif Operation Modes

The export-ldif command has two modes of operation: online and offline.

- **Online mode**. In online mode, export-ldif contacts a running directory server instance and registers an export task. This mode runs automatically when the LDAP connection options (--hostname, --port, --bindDN, and --bindPassword) are used. The command accesses the task back end over SSL via the administration connector. For more information, see "Managing Administration Traffic to the Server" on page 5.
- **Offline mode**. When no connection options are specified, the command runs in offline mode. In offline mode, export-ldif accesses the database directly rather than through a directory server instance. In this case, the directory server must be stopped.

▼ To Export Data to LDIF

1 **Stop the server if it is running.**

```
$ stop-ds
```

2 **Export the back end to a specified LDIF file.**

```
$ export-ldif --includeBranch "dc=example,dc=com" --backendID userRoot \
  --ldifFile example.ldif
```

▼ To Export Partial Data

The export-ldif command provides options to export a part of a back end by specifying the base DN and its children for inclusion or exclusion during processing.

1 **Stop the server if it is running.**

```
$ stop-ds
```

2 Export a portion of the back end.

In this example, only the entries under ou=People,dc=example,dc=com are exported.

```
$ export-ldif --includeBranch ou=People,dc=example,dc=com --backendID userRoot \
  --ldifFile example-people.ldif
```

3 (Optional) Use the ldifsearch command to verify the exported file.

The ldifsearch command verifies entries in an LDIF file without connecting to the directory server. You can use it in a manner similar to the ldapsearch command. For example:

```
$ ldifsearch -b dc=example,dc=com --ldifFile export.ldif "(objectclass=*)"
dn: ou=People,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: People
dn: uid=scarter,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
givenName: Sam
uid: scarter
cn: Sam Carter
sn: Carter
telephoneNumber: +1 408 555 4798
userPassword: {SSHA}Ocpp2P4sImz2MziL69AUG9+khdIhFpmU4B5mvA==
roomNumber: 4612
ou: Accounting
ou: People
l: Sunnyvale
mail: scarter@example.com
facsimileTelephoneNumber: +1 408 555 9751 ...
```

▼ To Export Part of a Back End by Using Filters

The export-ldif command provides options to export part of a back end by using a search filter. The directory server includes or excludes all entries that match the filter. Make sure that you fully understand how this mechanism works before you use it.

In this example, only those entries that match the search filter l=Cupertino (that is, location=Cupertino) are exported. The --excludeFilter option works in a similar manner to --includeFilter, except that it excludes all entries that match the filter during export.

1 Stop the server if it is running.

```
$ stop-ds
```

2. **Export a portion of the back end by using the `--includeFilter` option.**

```
$ export-ldif --includeFilter "(l=Cupertino)" --backendID userRoot \
  --ldifFile export.ldif
```

To Include or Exclude Attributes During Export

The `export-ldif` utility provides options to include and exclude attributes during export by using the `--includeAttribute` and `--excludeAttribute` options, respectively. Make sure that you fully understand how this mechanism works before you use it.

1. **(Optional) With the server running, view a sample entry, by using the `ldapsearch` command. For example:**

```
$ ldapsearch --baseDN dc=example,dc=com "(cn=Sam Carter)"
dn: uid=scarter,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: top
givenname: Sam
uid: scarter
cn: Sam Carter
telephonenumber: +1 408 555 4798
sn: Carter
userpassword: sprain
roomnumber: 4612
mail: scarter@example.com
l: Sunnyvale
ou: Accounting
ou: People
facsimiletelephonenumber: +1 408 555 9751
```

2. **Stop the server.**

```
$ stop-ds
```

3. **Export the back end, using the `--includeAttribute` option to specify the attributes that should be included in the export.**

   You can use the `--includeAttribute` option multiple times for each attribute that should be included. In this example, only the top level attributes are exported.

```
$ export-ldif --backendID userRoot --includeAttribute dn --includeAttribute dc \
  --includeAttibute cn --includeAttribute sn --includeAttribute givenname \
  --includeAttribute objectclass --includeAttribute ou --includeAttribute uid \
  --ldifFile export.ldif
```

4 **(Optional) Use the `ldifsearch` command to verify the export file.**

If an error occurs, the server continues processing the command.

```
$ ldifsearch --baseDN dc=example,dc=com --ldifFile export.ldif "(objectclass=*)"
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
dn: ou=Groups,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: Groups
dn: cn=Directory Administrators,ou=Groups,dc=example,dc=com
objectClass: groupofuniquenames
objectClass: top
cn: Directory Administrators
ou: Groups
dn: ou=People,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: People ...
```

▼ To Export to LDIF and Then Compress the File

The `export-ldif` command allows you to compress the output LDIF file.

1 **Stop the server if it is running.**

```
$ stop-ds
```

2 **Export to LDIF and then compress the file.**

```
$ export-ldif --backendID userRoot --ldifFile export.ldif --compress
```

▼ To Run an Export in Online Mode

The `export-ldif` command can also be run with the server online. In online mode, the command accesses the task back end over SSL via the administration connector. For more information, see "Managing Administration Traffic to the Server" on page 5. To run the command in online mode you must specify the relevant connection options, including how the SSL certificate will be trusted. This example uses the `-X` option to trust all certificates.

- **Run the `export-ldif` command with the LDAP connection options. For example:**

```
$ export-ldif -h localhost -p 4444 -D "cn=Directory Manager" -w password -X \
  --includeBranch "dc=example,dc=com" --backendID userRoot --ldifFile export.ldif
```

▼ To Schedule an Export

The `export-ldif` utility provides a `--start` option for scheduling the export at some future date. You can view this scheduled task by using the `manage-tasks` utility. The command accesses the task back end over SSL via the administration connector. For more information, see "Managing Administration Traffic to the Server" on page 5. To schedule an export task, you must specify the relevant connection options, including how the SSL certificate will be trusted. This example uses the `-X` option to trust all certificates.

The server must be running to schedule an export.

- **Run the `export-ldif` command with the `--start` option and the LDAP connection parameters.**

  The `--start` option takes as its value a date and time in the format `yyyymmddhhmmss`. For example:

```
$ export-ldif -h localhost -p 4444 -D "cn=Directory Manager" -w password -X \
  --includeBranch "dc=example,dc=com" --backendID userRoot \
  --ldifFile export.ldif --start 20080124121500
```

Importing and Exporting Entries With the Control Panel

You can use the Control Panel to import and export entries, as described in the following sections.

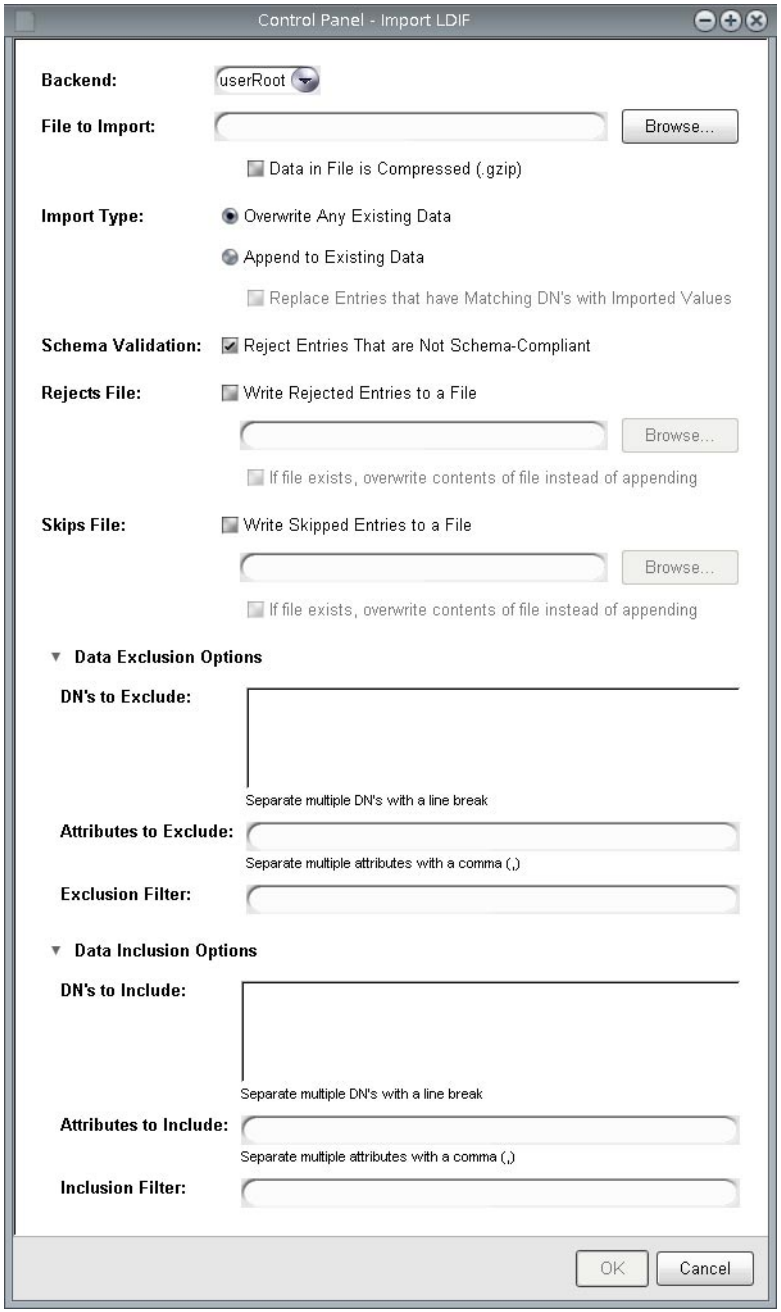
▼ To Import Entries With the Control Panel

This procedure shows how to use the Control Panel to import entries from an LDIF file.

1. **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2. **Click the Import LDIF link under the Directory Data menu on the left side of the Control Panel window.**

   The Import LDIF window appears, displaying fields for specifying the LDIF to import and how to manage the import operation.



3 Enter values in the fields to specify the LDIF and the import operation.

4 **Click the OK button.**

The Import LDIF window displays the success or failure of the operation.

5 **When the operation is complete, click the Close button to close the Import LDIF window.**

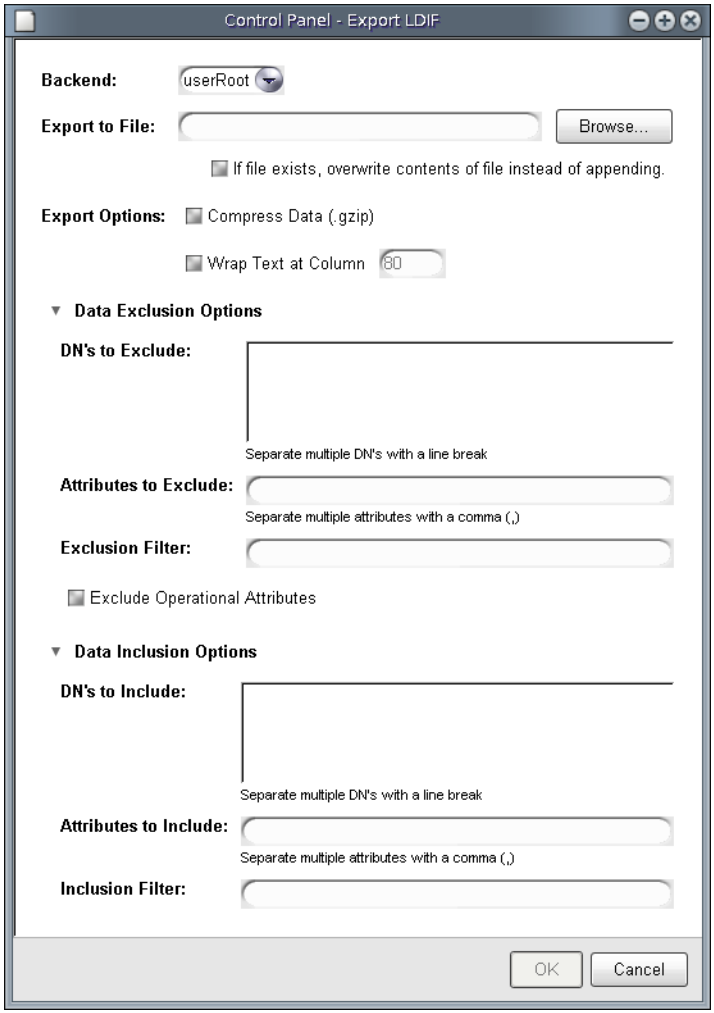
▼ To Export Entries to an LDIF File With the Control Panel

This procedure shows how to use the Control Panel to export entries to an LDIF file.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Click the Export LDIF link under Directory Data menu on the left side of the Control Panel window.**

The Export LDIF window appears, displaying fields for specifying the LDIF to export and how to manage the import operation.



3. **Enter values in the fields to specify the LDIF and the export operation.**

4. **Click OK.**

   The Export LDIF window displays the success or failure of the operation.

5. **When the operation is complete, click the Close button to close the Export LDIF window.**

Creating MakeLDIF Template Files

The `make-ldif` command can use template files to define the way in which LDIF files are to be generated. This approach allows for flexibility without the need to alter any code to produce the desired result. The topics in this section describe how to use the `make-ldif` command to create customized LDIF files.

The Template File Format

Template files can contain up to four sections, that must be provided in the following order:

1. "Custom Tag Includes" on page 111
2. "Global Replacement Variables" on page 111
3. "Branch Definitions" on page 112
4. "Template Definitions" on page 113

Custom Tag Includes

Custom tag includes provide a mechanism for loading custom tags and making them available for use when processing `make-ldif` templates. This should be done using the `include` directive, as follows:

```
include com.example.opends.makeldif.MyCustomTag
```

The specified class must be in the class path, and it must be a subclass of the `org.opends.server.tools.makeldif.Tag` class. For information about developing custom tags, see "Defining Custom Tags" on page 124.

All of the standard replacement tags that are provided with `make-ldif` are automatically available for use and therefore do not require an explicit `include` directive.

Global Replacement Variables

The first section that should be present in the template file is the section that defines the global replacement variables. Global replacement variables are used to define strings of text that can be referenced later in the template file and are automatically replaced as each line is read into memory (much like a C preprocessor replaces macros in code with their defined values). For example, the following replacement variable definition creates a global replacement variable named `suffix` with a value of `dc=example,dc=com`:

```
define suffix=dc=example,dc=com
```

When a global replacement variable is defined, any case in which that variable name appears in square brackets (for example, `[suffix]`), causes the token to be replaced with the value that has been defined for that replacement variable.

When all the replacement variable definitions have been read (as signified by the first blank line following one or more replacement variable definitions), all remaining lines that are read from the template file are processed on a line-by-line basis. Any occurrences of a replacement variable name in square brackets are replaced with the value of that variable. Because that replacement is done as the template file is read into memory, replacement variables can occur in any point, including branch and template definitions, and even inside tags.

If there are global replacement variables defined in the template file, they must appear at the top of the file and there should not be any spaces between them. However, replacement variables are not required. If there are no replacement variables, the template file must start with the branch definitions.

Branch Definitions

Branch definitions are used in make-ldif template files to define the basic structure to use for the generated LDIF. They specify the entry or entries that should appear at the top of the hierarchy, and the number and types of entries that should appear below them.

The most basic form of a branch definition is as follows:

```
branch: dc=example,dc=com
```

This example specifies that the following entry is to be created with a DN of `dc=example,dc=com`:

```
dn: dc=example,dc=com
objectClass: top
objectClass: domain
dc: example
```

The basic structure of the entry is defined by the RDN attribute of `dc` specified in the DN of the branch definition. The make-ldif command automatically associates the `dc` RDN attribute with the `domain` object class. The make-ldif command has similar definitions for other common RDN attributes in branch entries:

`o` Creates an entry with the `organization` object class.

`ou` Creates an entry with the `organizationalUnit` object class.

`c` Creates an entry with the `country` object class.

You can also use any other kind of RDN attribute for a branch entry. For branch entries with an RDN attribute other than the ones specified above, the entry is created with the `untypedObject` and `extensibleObject` object classes.

The branch definition provided above does not cause any additional entries to be created below that branch entry. To do this, you must specify one or more `subordinateTemplate` lines. For example:

```
branch: ou=People,dc=example,dc=com
subordinateTemplate: person:100
```

This causes the `ou=People,dc=example,dc=com` entry to be created, and then 1000 other entries created below it modeled after the `person` template. The `person` template should be defined later in the template file. For more information, see "Template Definitions" on page 113.

Note – Branch entries are not limited to just one `subordinateTemplate` definition. You can specify multiple `subordinateTemplate` definitions by including them on separate lines of the branch definition. The following example creates 1000 entries based on the `person` template and an additional 100 entries based on the `certificatePerson` template:

```
branch: ou=People,dc=example,dc=com
subordinateTemplate: person:10000
subordinateTemplate: certificatePerson:100
```

In all of the examples described previously, the branch entries themselves contain only the DN, the RDN attribute, and the object classes associated with the RDN attribute. You can include any other attributes in the branch entry by including them in the branch definition in the template file. For example, the branch definition:

```
branch: dc=example,dc=com
description: This is the description for dc=example,dc=com
```

creates the entry:

```
dn: dc=example,dc=com
objectClass: top
objectClass: domain
dc: example
description: This is the description for dc=example,dc=com
```

This additional text can be static, can contain any defined global replacement variables, or can contain a subset of the replacement tags that can be used in template definitions. For an overview of the tags available and information about which tags can be used in branch definitions, see "Standard Replacement Tags" on page 115.

Template Definitions

The heart of the `make-ldif` template file structure is the set of template definitions. Templates define the structure of the entries that are generated. They specify the set of attributes that should be included in the entries and the types of values that those attributes should contain. The specification of values is handled through tags that are parsed by `make-ldif` and replaced with the appropriate values for those tags.

A sample template definition might look as follows:

```
template: person
rdnAttr: uid
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
givenName: <first>
sn: <last>
cn: {givenName} {sn}
initials: {givenName:1}<random:chars:ABCDEFGHIJKLMNOPQRSTUVWXYZ:1>{sn:1}
employeeNumber: <sequential:0>
uid: user.{employeeNumber}
mail: {uid}@[maildomain]
userPassword: password
telephoneNumber: <random:telephone>
homePhone: <random:telephone>
pager: <random:telephone>
mobile: <random:telephone>
street: <random:numeric:5> <file:streets> Street
l: <file:cities>
st: <file:states>
postalCode: <random:numeric:5>
postalAddress: {cn}${street}${l}, {st}  {postalCode}
description: This is the description for {cn}.
```

This example illustrates some of the flexibility that `make-ldif` provides when generating LDIF data. The tags that can be included in a template definition are described in the topics that follow (see "Standard Replacement Tags" on page 115 and "Attribute Value Reference Tags" on page 123).

At the top of the template definition are two lines that provide information about the template itself and are not included in the entries created from this template. The first line specifies the name of the template. This is the name that is referenced in the `subordinateTemplate` lines of the branch definition. The second line specifies the name of the attribute that should be used as the RDN attribute for the entry. The RDN attribute must be assigned a value in the body of the template definition, and the way in which the value is assigned must ensure that the value will be unique among all other entries created with the same template below the same parent.

Note – It is possible to specify multivalued RDNs by separating the attribute names with a plus sign, as shown in the following example:

```
rdnAttr: uid+employeeNumber
```

If multivalued RDNs are used, all of the RDN attributes must be defined values in the template body and the combination of the RDN values for each entry must be unique. However, it is possible for one or more of the attributes in the RDN to be non-unique as long as the combination is never duplicated.

In addition to the `template` and `rdnAttr` lines, you can include one or more `subordinateTemplate` lines. This enables you to include dynamically-generated entries below other entries that have been dynamically generated (for example, if each user entry has one or more entries below it), and to allow for complex hierarchies. Although there is no limit placed on this level of nesting, you must ensure that no recursive loops are created by having a `subordinateTemplate` that either directly or indirectly will create additional entries using the same template.

Template definitions also support the concept of inheritance through the use of the `extends` keyword. For example, entries generated from the following template definition include all of the attributes defined in the `person` template as well as `userCertificate;binary` with the specified format:

```
template: certificatePerson
rdnAttr: uid
extends: person
userCertificate;binary:: <random:base64:1000>
```

Multiple inheritance is allowed (by including multiple lines with the `extends` keyword), but as with the `subordinateTemplate` keyword it is important not to create a recursive loop in which a template file could either directly or indirectly inherit from itself.

make-ldif Template File Tags

To ensure that `make-ldif` can generate LDIF files that can be used to simulate a wide variety of deployments, a large number of tags have been defined for use in templates. This section describes the standard set of tags that can be used in a `make-ldif` template file. You can also create custom tags, as described in "Defining Custom Tags" on page 124.

Standard Replacement Tags

The `make-ldif` standard replacement tags are special elements that are enclosed in angle brackets (beginning with a less-than sign (<) and ending with a greater-than sign (>) that are dynamically replaced with generated values. Some standard replacement tags do not require any arguments (for example, `<first>`). Others do take arguments, in which case the tag name comes first followed by a colon and the argument list with a colon between each argument (for example, `<random:numeric:5>`). The tag name is treated in a case-insensitive manner, although the arguments are generally case sensitive.

The following types of standard replacement tags are currently included as part of `make-ldif`:

The DN tag

The DN standard replacement tag is replaced with the DN of the current entry. If that DN is not yet available (for example, because the RDN attribute has not yet been assigned a value in the entry being generated), it is replaced with an empty string. In general, you should ensure that all RDN attributes are assigned values earlier in the template before this tag is used.

The DN tag can be used without any arguments (for example, `<DN>`), in which case it is replaced with the full DN of the entry. The tag can also take a single integer argument, which specifies the maximum number of components to include in the output. For example, the tag `<DN:1>` will only include the left most DN component (often called the RDN) for the entry. So if the entry being generated will have a DN of `uid=john.doe,ou=People,dc=example,dc=com`, the tag `<DN:1>` will be replaced with `uid=john.doe`. If the argument value is negative rather than positive, then it takes the absolute value of the given argument value and takes that number of components from the end of the DN. For example, using a DN of `uid=john.doe,ou=People,dc=example,dc=com` the tag `<DN:-1>` is replaced with `dc=com`.

This tag can be used in both branch and template definitions.

The File tag

The File standard replacement tag is replaced with a line from a specified file. It requires either one or two arguments. The first argument is the path to the data file, and can be either an absolute path or the name of a file (with no path information) that is contained in the `config/MakeLDIF` directory. If there is a second argument, it must have a value of either `sequential` or `random`, which indicates whether the lines in the file should be taken in sequential order or chosen at random. If the second argument is not provided, the values are selected at random. For example, the tags `<file:cities>` and `<file:cities:random>` both cause the tag to be replaced with a randomly-selected line from the `cities` file, but the tag `<file:cities:sequential>` causes the city names to be taken in sequential order. If sequential ordering is used and all values are exhausted, it will wrap back around to the first line of the file.

The `make-ldif` command includes a number of standard data files that can be used in generated data. These files are included in the `config/MakeLDIF` directory and therefore only the filename is required. The files include:

`cities`	Contains a list of common city names
`first.names`	Contains a list of common first names

`last.names`	Contains a list of common last names
`states`	Contains a list of all two-character US state abbreviations
`streets`	Contains a list of common street names

This tag can be used in both branch and template definitions.

The First tag

The First standard replacement tag is replaced with a first name taken from the `config/MakeLDIF/first.names` file. Note that there is a special relationship between the `<first>` and `<last>` tags such that the combination of the first and last names is always unique. When every possible combination from the first and last name files has been exhausted, make-ldif appends an integer value onto the last name to ensure that the value always remains unique.

The `<first>` tag does not take any arguments. It can be used only in template definitions. It is not allowed for use in branch definitions.

The GUID tag

The GUID standard replacement tag is replaced with a randomly generated GUID (globally-unique identifier) value. All GUID values generated are guaranteed to be unique. The values generated consist of 32 hexadecimal digits in dash-delimited groups of 8, 4, 4, 4, and 12 digits, respectively (for example, `12345678-90ab-cdef-1234-567890abcdef`).

The `<guid>` tag does not take any arguments. It can be used in both branch and template definitions.

The IfAbsent tag

The IfAbsent standard replacement tag does not generate any value of its own, and is therefore always be replaced with an empty string. However, its value is that it can prevent an attribute from appearing in the entry altogether based on whether a specified attribute or attribute value exists.

For example, consider the following template:

```
template: example
rdnAttr: cn
objectClass: top
objectClass: untypedObject
objectClass: extensibleObject
cn: <guid>
displayName: <presence:50>{cn}
description: <ifabsent:displayName>{cn}
```

In this case, the `description` attribute is only included in the generated entry if the `displayName` attribute is not included (that is, the resulting entry will contain either `displayName` or `description` but not both).

The `IfAbsent` tag requires either one or two arguments. The first argument is the name of the target attribute. If there is a second argument, it specifies a particular value for the target attribute. If a value is provided, the `IfAbsent` tag takes action if that value is included in the generated entry.

This tag can be used in both branch and template definitions.

The IfPresent tag

The IfPresent standard replacement tag does not generate any value of its own, and is therefore always replaced with an empty string. However, its value is that it can prevent an attribute from appearing in the entry altogether based on whether a specified attribute or attribute value exists.

For example, consider the following template:

```
template: example
rdnAttr: cn
objectClass: top
objectClass: untypedObject
objectClass: extensibleObject
cn: <guid>
displayName: <presence:50>{cn}
description: <ifpresent:displayName>{cn}
```

In this case, the `description` attribute will only be included in the generated entry if the `displayName` attribute is also included (that is, the resulting entry will either contain neither attribute or it will contain both attributes).

The IfPresent tag requires either one or two arguments. The first argument is the name of the target attribute. If there is a second argument, it specifies a particular value for the target attribute. If a value is provided, the IfPresent tag will only take action if that value is included in the generated entry.

This tag can be used in both branch and template definitions.

The Last tag

The Last standard replacement tag is replaced with a last name taken from the `config/MakeLDIF/last.names` file. Note that there is a special relationship between the `<first>`and `<last>`tags such that the combination of the first and last names will always be unique. When every possible combination from the first and last name file has been

exhausted, make-ldif will append an integer value onto the last name to ensure that the value always remains unique.

The `<last>`tag does not take any arguments. It can only be used in template definitions. It is not allowed for use in branch definitions.

The List tag

The List standard replacement tag is replaced with a string selected from a provided list of values. The values to use should be provided as arguments to the List tag (at least one argument must be provided). Optionally, each value can be followed with a semicolon and an integer value that specifies the relative weight for that value. If a value does not include a weight, the weight for that item is assumed to be one. The weight is used to control how frequently the associated value is chosen compared with all of the other values in the list.

For example, to select from a list of the colors red, green, and blue in which all listed colors have equal weights, you can use:

```
<list:red:green:blue>
```

If the color red is to appear twice as frequently as either of the other colors, you can use:

```
<list:red;2:green;1:blue;1>
```

Note that in this case, the `;1` following the green and blue elements are not technically needed since the weight of any item that does not explicitly include a weight is one, but it is provided in the example above for clarity.

This tag can be used in both branch and template definitions.

The ParentDN tag

The ParentDN standard replacement tag is replaced with the DN of the parent entry of the entry being generated. This should always be available.

This tag does not take any arguments. It can only be used in template definitions. It cannot be used in branch definitions.

The Presence tag

The Presence standard replacement tag does not generate any value of its own, and is therefore always replaced with an empty string. However, its value is that it can be used to cause the associated attribute to appear in the entry a specified percentage of the time.

For example, consider the following template:

```
template: example
rdnAttr: cn
objectClass: top
objectClass: untypedObject
objectClass: extensibleObject
cn: <guid>
displayName: <presence:50>{cn}
```

In this case, the displayName attribute will only be present in about 50% of the entries generated.

The Presence tag requires exactly one argument, which is an integer value between 0 and 100, indicating the percentage of entries that should have the associated attribute.

This tag can be used in both branch and template definitions.

The Random tag

The Random standard replacement tag is replaced with a randomly-generated value. A number of different types of values can be generated. This tag accepts a variable number of arguments, but the first argument always specifies the type of value to generate. That type may be one of the following values:

`alpha` This causes the tag to be replaced with a specified number of lowercase ASCII alphabetic characters (that is, the character set `abcdefghijklmnopqrstuvwxyz`). This requires exactly one more argument, which is an integer specifying the number of characters to include in the generated value. For example, `<random:alpha:5>` generates a string of five randomly-selected alphabetic characters.

`numeric` This causes the tag to be replaced with one or more numeric digits. There can be either one or two additional arguments. If there is one additional argument, it specifies the number of numeric digits to include in the value (for example, `<random:numeric:5>` will generate a string of five numeric digits). If there are two additional arguments, they will specify the upper and lower bounds for a randomly-generated number (for example, `<random:numeric:5:10>` will generate a random integer between 5 and 10, inclusive).

`alphanumeric` This causes the tag to be replaced with a specified number of lowercase ASCII alphabetic characters

	(that is, the character set `abcdefghijklmnopqrstuvwxyz`) and/or numeric digits (that is, the character set `0123456789`). This requires exactly one more argument, which is an integer specifying the number of characters to include in the generated value. For example, `<random:alphanumeric:5>` will generate a string of five randomly-selected alphanumeric characters.
`chars`	This causes the tag to be replaced with characters from a user-defined character set. This can take either two or three additional arguments. The first additional argument is the characters for the user-defined character set. If there is a single argument after the character set, it specifies the number of characters to take from that set (for example, `<random:chars:abcd:3>` will cause three characters to be chosen in which each of those characters is either a, b, c, or d). If there are two arguments after the character set, they must be integer values and the number of characters generated will be an integer between this range (for example, `<random:chars:abcd:3:5>` will cause between 3 and 5 characters to be included in the value, where each character is either a, b, c, or d).
`hex`	This causes the tag to be replaced with a specified number of hexadecimal characters (that is, the character set `0123456789abcdef`). This requires exactly one more argument, which is an integer specifying the number of characters to include in the generated value. For example, `<random:hex:5>` will generate a string of five randomly-selected hexadecimal characters.
`base64`	This causes the tag to be replaced with a specified number of characters allowed in the base64 character set (`ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvw` This requires exactly one more argument, which is an integer specifying the number of characters to include in the generated value. For example, `<random:base64:5>` will generate a string of five randomly-selected hexadecimal characters.

`month`	This causes the tag to be replaced with the name of a month of the year. If there are no additional arguments, the full name of the month is included (for example, `<random:month>` might return a value of `October`). If there is a single additional argument, it must be an integer value that specifies the maximum number of characters to include from the name of the month (for example, `<random:month:3>` might generate a value of `Oct`).
`telephone`	This causes the tag to be replaced with a randomly-generated telephone number in the format 123-456-7890. It does not take any additional arguments (that is, it should always be used like `<random:telephone>`).

This tag can be used in both branch and template definitions.

The RDN tag

The RDN standard replacement tag is replaced with the RDN (that is, the leftmost DN component) of the current entry. If the RDN is not yet available (for example, because the RDN attribute has not yet been assigned a value in the entry being generated), it will be replaced with an empty string. In general, you should ensure that all RDN attributes are assigned values earlier in the template before this tag is used. The behavior of this tag is identical to that of the DN tag when used with a single argument whose value is one (that is, `<dn:1>`).

The RDN tag does not take any arguments. It can be used in both branch and template definitions.

The Sequential tag

The Sequential standard replacement tag is replaced with an integer value. Each entry is given a sequentially-incrementing value (for example, the first entry is given a value of zero, the next entry a value of one, and so on).

This tag can take zero, one, or two arguments:

- If there are no arguments (that is, the tag is `<sequential>`), the first value will be zero, and the value will be reset to zero for each new branch.
- If there is a single argument, it must be an integer that specifies the initial value to use (for example, a tag of `<sequential:1000>` will start generating values at 1000 instead of 0). The value will be reset to the specified initial value for each new branch.

- If there are two arguments, the first must be an integer that specifies the initial value, and the second should be a Boolean value of either `true` or `false` indicating whether to reset the counter each time a new branch is started.

This tag can be used in both branch and template definitions.

The _DN tag

The _DN (note the leading underscore character) standard replacement tag is replaced with the DN of the entry being generated, but with an underscore used instead of a comma between DN components. Apart from using underscores instead of commas, this works exactly like the DN tag. As such, it can also take an optional integer argument that specifies the number of components from the left (or from the right if the value is negative) should be included.

This tag can be used in both branch and template definitions.

The _ParentDN tag

The _ParentDN (note the leading underscore character) standard replacement tag is replaced with the DN of the parent entry of the entry being generated, but with an underscore used instead of a comma between DN components. This should always be available.

This tag does not take any arguments. It can only be used in template definitions. It cannot be used in branch definitions.

Attribute Value Reference Tags

Attribute value reference tags can be used to replace the tag with the value of a specified attribute from the same entry. They are used by enclosing the name of the desired attribute in curly braces. For example, `{cn}` will be replaced with the value of the `cn` attribute, if it has already been given a value in the target entry. If the target attribute has not yet been given a value in the entry, the tag will be replaced with an empty string.

For example, consider the following excerpt from a template:

```
givenName: <first>
sn: <last>
uid: {givenName}.{sn}
cn: {givenName} {sn}
mail: {uid}@example.com
```

If the value chosen for the first name is `John`and the last name is `Doe`, then the resulting LDIF output would be:

```
givenName: John
sn: Doe
```

```
uid: John.Doe
cn: John Doe
mail: John.Doe@example.com
```

It is also possible to place a colon after the name of the attribute followed by a positive integer value specifying the maximum number of characters to include from the target attribute. For example, the template excerpt:

```
givenName: <first>
sn: <last>
initials: {givenName:1}{sn:1}
```

would cause the following LDIF to be generated:

```
givenName: John
sn: Doe
initials: JD
```

If the specified length is longer than the value of the named attribute, the entire value is used with no padding added. Otherwise, the specified number of characters are taken from the value.

Tag Evaluation Order

All tags in the `make-ldif` syntax are currently given equal priority. As such, they are evaluated in the order that they appear in the template definition, from top to bottom, and from left to right within a given line. It is not possible to embed one tag within another.

Defining Custom Tags

The make-ldif utility has been designed in an extensible manner so that new tags can be defined and used in template files.

All tags must be subclasses of the `org.opends.server.tools.makeldif.Tag` abstract class. Custom tag definitions must include the following methods:

`public String getName()`
: This retrieves the name that should be used to reference the tag. The value that it returns must be unique among all other tags in use by the server.

`public boolean allowedInBranch()`
: This indicates whether the tag will be allowed in branch definitions. If it returns a value of `true`, then the tag may be used in both branch and template definitions. If it returns a value of `false`, then the tag may be used in template definitions but not branch definitions.

`public void initializeForBranch(TemplateFile templateFile, Branch branch, String[] arguments, int lineNumber, List<String> warnings)`
This performs any initialization that may be required if the tag is to be used in a branch definition. This does not need to be implemented if `allowedInBranch()` returns `false`.

`public void initializeForTemplate(TemplateFile templateFile, Template template, String[] arguments, int lineNumber, List<String> warnings)`
This performs any initialization that may be required of the tag is to be used in a template definition.

`public void initializeForParent(TemplateEntry parentEntry)`
This performs any initialization that may be required before starting to generate entries below a new parent. This does not need to be implemented if no special initialization is required.

`public TagResult generateValue(TemplateEntry templateEntry, TemplateValue templateValue)`
This generates the value that will be used to replace the associated tag when generating entries.

All of the tags available in `make-ldif` are included in the `org.opends.server.tools.makeldif` package. They may be used for reference to understand what is involved in implementing a custom tag.

Note – If you define a custom tag, ensure that it is available for use in any template file that might need it. This is done using the `include` statement, that should appear at the top of the template file. For more information, see "Custom Tag Includes" on page 111.

Backing Up and Restoring Data

The directory server provides an extensible framework that supports a variety of repository types. The directory server uses the Berkeley DB Java Edition (JE) as its primary back end. The JE back end provides some advantages over other databases as it provides a high-performance, scalable transactional B-tree database with full support for ACID semantics for small to very large data sets. It can also store its entries in encoded form and provide indexes for fast, efficient data retrieval.

Overview of the Backup and Restore Process

To maintain the directory data on the JE back end, the directory server provides efficient backup and restore utilities that support full and incremental backups. A *full backup* saves the directory data files in the environment as a compressed archive file. An *incremental backup* saves and compresses just those files that have been written since the previous backup, together

with a list of names of files that are unchanged since the previous backup. The directory server stores its backup information in a *backup back end* for easy restores.

Directory server backups also can be made on the local disks or on remote disks, for example, on network-attached storage (NAS). If you run a backup locally, you should then copy and store the backup on a different machine or file system for security purposes.

Before you start backing up and restoring data, consider the following:

- You must design a workable backup and restore strategy for your directory services system. For example, you can run an incremental backup daily and perform a full backup at least once a week. Test your backup process and your ability to restore regularly. For data restores, many companies restore a directory server from a replicated server, which ensures that the most update copy of the directory data is used. Backup tapes are still needed if the directory data is damaged (for example, missing entries) and the corrupted data has been replicated to other servers.
- Ensure that you have a disaster recovery plan in place. Disaster recovery is necessary when catastrophic events, data corruption, or data tampering occurs. Companies devise their own plans or out source the work to third party specialists. See "Backing Up for Disaster Recovery" on page 130 for more information.
- Ensure that you have a place to store your back ups. Store the archived data, configuration directory, schema subdirectory, and installation directory used for your server together in a single location. All these items are required when you restore the server.

Backing Up Data

The directory server provides an efficient command-line utility (`backup`) to back up databases. The `backup` command can be run immediately or scheduled as a task. If the backup is scheduled, the command contacts the server over SSL, using the administration connector, and registers a backup task. If no connection options are specified, the command runs immediately.

The following procedures show the use of the `backup` command in various backup scenarios.

▼ To Back Up All Back Ends

You can back up all back ends end by using the `--backUpAll` option.

- **Run the backup command with the `--backUpAll` option.**

  The following command is run on a standalone directory server and specifies that all databases should be backed up, compresses the backup file, and saves the file to a specified location.

```
$ backup --backUpAll --compress --backupDirectory /tmp/backup
```

The backup directory contains subdirectories for each back end:

```
$ ls /tmp/backup
./ ../ config/ schema/ tasks/ userRoot/
```

The `backup` utility writes the backup to the specified directory and creates a `backup.info` file that provides details about the backup. The directory server assigns a backup ID based on the current date and time. To create your own ID, use the `--backupID` option:

```
$ ls /tmp/backup/config
./ backup.info
../ config-backup-20070827153501Z
```

The `backup.info` file contains detailed information about the current backup.

```
$ more /tmp/backup/config/backup.info
backend_dn=ds-cfg-backend-id=config,cn=Backends,cn=config

backup_id=20070827153501Z
backup_date=20070827153511Z
incremental=false
compressed=true
encrypted=false
property.archive_file=config-backup-20070827153501Z
```

▼ To Back Up All Back Ends with Encryption and Signed Hashes

The `backup` utility provides encryption and signed hash support for secure backups. The use of the encryption and signed hash options requires a connection to an online server instance, so the appropriate connection options must be specified.

- **Run the backup command .**

  The following command backs up all back ends, compresses them, generates a hash, signs the hash, and encrypts the data.

  ```
  $ backup -h localhost -p 4444 -D "cn=directory manager" -w password --backUpAll -X \
    --compress --hash --signHash --encrypt --backupID 123 --backupDirectory /tmp/backup
  ```

▼ To Perform an Incremental Backup on All Back Ends

Incremental backups save only those changes that have occurred since the last backup (full or incremental). The main advantage of an incremental backup is the faster time to back up a system when compared to that of full backups. The disadvantage of an incremental backup is that each incremental backup must be restored, which requires more time and care than that of a full restore.

- **Run the backup command with the `--incremental` option.**

  ```
  $ backup --backUpAll --incremental --compress --backupDirectory /tmp/backup
  ```

To Back Up a Specific Back End

You can back up a single back end by using the `--backendID` option, which specifies the back end to save.

Note – If you back up a single back end and replication is configured, any changes made to that back end are stored in the change log on the replication server. When you restore that back end, the replication server detects that the back end is not up to date and replays the changes made after the backup. This behavior occurs even if there is only one directory server in the replicated topology, because the changes are stored on the replication server.

If you do not want this behavior, back up all back ends in a replicated environment. This ensures that the data, and the replication server are backed up. In this case when a restore is done, the directory server and the replication server are restored to their state before the back up, and no memory of subsequent changes remains.

---

1. **(Optional) List the back ends that are configured on the server, by running the `list-backends` command. For example:**

```
$ list-backends

Backend ID     Base DN
-------------- -----------------
adminRoot      cn=admin data
ads-truststore cn=trust-store
backup         cn=backups
config         cn=config
monitor        cn=monitor
schema         cn=schema
tasks          cn=tasks
userRoot       dc=example,dc=com
```

2. **Run the backup command with the `--backendID` option.**

   For example, to back up the `userRoot` back end, run the following command:

```
$ backup --backendID userRoot --backupDirectory /tmp/backup
```

To Perform an Incremental Backup on a Specific Back End

1. **(Optional) List the back ends that are configured on the server, by running the `list-backends` command. For example:**

```
$ list-backends

Backend ID     Base DN
-------------- -----------------
adminRoot      cn=admin data
```

```
ads-truststore cn=trust-store
backup         cn=backups
config         cn=config
monitor        cn=monitor
schema         cn=schema
tasks          cn=tasks
userRoot       dc=example,dc=com
```

2 **Run the backup command with the `--incremental` option.**

```
$ backup --incremental --backendID userRoot --backupDirectory /tmp/backup
```

▼ To Schedule a Backup as a Task

The directory server provides a task back end for processing administrative tasks, such as backups and restores. You can specify the start time for a backup or restore by using the `-t` or `--start` option. If one of these options is provided, the utility exits immediately after scheduling the task. To schedule a task for immediate execution and have the utility exit immediately after scheduling the task, specify `0` as the value for the start time. If the `-t` or `--start` option is omitted, the utility schedules the task for immediate execution and tracks the task's progress, printing log messages as they are available and exiting when the task has completed.

Access to the task back end is provided over SSL via the administration connector. If you schedule the backup as a task, you must therefore specify how the SSL certificate will be trusted. This example schedules a backup for execution at a future time. The `-X` option specifies that all certificates presented by the server are trusted. For more information, see "Managing Administration Traffic to the Server" on page 5.

1 **Run the `backup` command with the following options:**

```
$ backup --port 4444 --bindDN "cn=Directory Manager" --bindPassword password -X \
  --backUpAll --backupDirectory /tmp/backups --start 20080601121500 \
  --completionNotify admin@example.com --errorNotify admin@example.com
```

2 **(Optional) View information about the scheduled task by using the `manage-tasks` command. For example:**

```
$ manage-tasks --port 4444 --bindDN "cn=Directory Manager" --bindPassword password -X \
  --info 2008040210324704 --no-prompt
```

Backing Up the Server Configuration

All configuration settings for a directory server instance are stored in the `config.ldif` file, which is located in the `config` directory. The directory server automatically saves the `config.ldif` file to ensure that changes are properly accounted for in the configuration. The file is saved at two specific times:

- **At startup**. If the current configuration does not match the archived configuration, the server saves the `config.ldif` file.
- **At modification time**. Whenever a directory administrator makes changes to the configuration by using the `dsconfig` utility with the server online, the directory server saves the `config.ldif` file prior to the change.

You can access archived configuration files from the *install-dir*`/config/archived-configs` directory. This directory lists each saved configuration file, compresses it as a `.gz` file, and saves the configuration as `config-`*timestamp*`.gz`. For example, you can see archived `config.ldif` files as follows:

```
$ ls config/archived-configs
09/02/2007 03:43 PM 9,045 config-20070819055359Z.gz
```

Backing Up for Disaster Recovery

Directory and system administrators should have a disaster recovery plan in place in the event of a natural, human-induced, or catastrophic disaster. If your directory service is distributed over multiple individual servers, back up all the servers individually or back up all the directory data from a central location.

Alternatively, consider replication as a backup and restore strategy. Replication provides faster restores and more update data from another replicated server. For more information, see "Restoring Replicated Directory Servers" on page 133.

▼ To Back Up the Directory Server For Disaster Recovery

1. **Make a backup of all back ends by using the `--backUpAll` option, for example:**

   ```
   $ backup --backUpAll --backupDirectory /tmp/backup
   ```

2. **Copy the configuration directory, *install-dir*`/config`.**

   Make sure that the `schema` subdirectory is present within the *install-dir*`/config` directory.

3. **Copy the files in *install-dir*`/logs`.**

4. **Make a copy of the installation directory.**

5. **Store the archived data, configuration directory, schema subdirectory, log files and installation directory together in a single location.**

   All items are required when restoring the server.

Restoring Data

You can restore data by using the `restore` utility. The `restore` utility allows you to restore only one back end at a time. The directory server must be stopped prior to a restore, unless you are scheduling a restore task, or you are restoring data that has been signed or hashed.

To Restore a Back End

1. **Stop the server, if it is running.**
2. **(Optional) Display the backup information by running the `restore` command with the `--listBackups` option. For example:**

```
$ restore --listBackups --backupDirectory backup/userRoot
Backup ID: 20080827153501Z
Backup Date: 27/Aug/2008:10:35:11 -0500
Is Incremental: false
Is Compressed: true
Is Encrypted: false
Has Unsigned Hash: false
Has Signed Hash: false
Dependent Upon: none
```

3. **Restore the back end.**

```
$ restore --backupDirectory backup/userRoot
```

4. **Repeat the restore for the other back ends.**

To Restore a Back End From Incremental Backups

Typically, system administrators run a weekly full backup with daily incremental backups. Be aware that it takes longer to restore your system from incremental backups.

1. **Restore the last full backup on your system by using the `restore` command.**

   Each back end must be restored individually.

2. **Restore each incremental backup by using the `restore` command.**

   Restore each incremental backup starting from the last full backup.

To Schedule a Restore as a Task

The directory server provides a task back end for processing administrative tasks, such as backups and restores. You can specify the start time for a restore by using the `-t` or `--start` option. If one of these options is provided, the utility exits immediately after scheduling the task. To schedule a task for immediate execution and have the utility exit immediately after

scheduling the task, specify `0` as the value for the start time. If the `-t` or `--start` option is omitted, the utility schedules the task for immediate execution and tracks the task's progress, printing log messages as they are available and exiting when the task has completed.

Access to the task back end is provided over SSL, using the administration connector. If you schedule the restore as a task, you must therefore specify how the SSL certificate will be trusted.

1. **Ensure that the server is stopped prior to the scheduled restore time.**

2. **Schedule the restore by using the `-t` or `--start` option of the `restore` command.**

   The following command restores the `userRoot` back end at a scheduled start time by using the `--start` option. The restore sends a completion and error notification to `admin@example.com`. The `-X` option specifies that all certificates presented by the server are trusted.

   ```
   $ restore -p 4444 -D "cn=Directory Manager" -w password -X \
     -d /backup/userRoot --start 20080125121500 --completionNotify admin@example.com \
     --errorNotify admin@example.com
   ```

3. **(Optional) You can view this scheduled task by using the `manage-tasks` utility.**

   For more information, see "Configuring Commands As Tasks" on page 23.

▼ To Restore the Configuration File

You might need to restore the configuration file to transfer the configuration to another server, for disaster recovery purposes, or for other events. In general, if a server is online, the current configuration file is equivalent to the latest archived configuration file. However, you can choose to restore the `config.ldif` file from a previous date.

1. **Stop the server if it is running.**

2. **Locate the required configuration file on the system. For example:**

   ```
   $ ls install-dir/config/archived-configs
   ./
   ../
   config-20070817192057Z.gz
   config-20070827153200Z.gz
   config-20070817192052Z.gz
   config-20070827153214Z-2.gz
   ```

3. **Manually decompress the archived configuration file, using a decompression utility such as `gunzip`.**

4. **Copy the file to the `config` directory, replacing the current `config.ldif` file.**

   ```
   $ cp config-20070817182052Z install-dir/config/config.ldif
   ```

▼ To Restore a Directory Server During Disaster Recovery

1. **Install the same version of the directory server that was previously installed on the host.**
2. **Create a server instance by using the `setup` command.**
3. **Copy the saved `config` directory to *install-dir*`/config`.**

   The `config.ldif` file should reside in this directory. The saved schema subdirectory should be located in *install-dir*`/config/schema`.
4. **Check that the configuration for the restored server is correct.**
5. **Restore the individual back ends by using the `restore` command.**

Restoring Replicated Directory Servers

Performing binary restores in replicated environments requires special care depending on your replicated topology. If possible, update your back end by using the replication mechanisms in your system instead of restoring it from a backup. Replication has distinct advantages over traditional tape backups. Data restores are much faster than tape restores, and the data is more up to date. However, tapes are still needed in the event that the replicated data is corrupt and has been propagated to other servers.

When restoring a replicated server, ensure that the configuration file *install-dir*`/config/config.ldif` is the same as when the backup was made. Restore the `config.ldif` file prior to restoring the server back ends.

You cannot restore an old backup to a master server because it might be out of date. Rather allow the replication mechanism to bring a master up to date with the other master servers by setting that master to read-only. When the master has been synchronized, you can reset it to read-write.

If you need to restore a replicated server, reinitialize the server from one of the other replicated servers by importing an LDIF file.

For very large databases (millions of entries), make a binary copy of one server and restore it on the other replicated server.

If you have a fairly recent backup (one that is not older than the maximum age of the change log contents on any of the other replicated servers), you can use that version to restore your data. When the old backup is restored, the other servers will update that server with recent updates made since the backup was saved.

Backing Up and Restoring Directory Data With the Control Panel

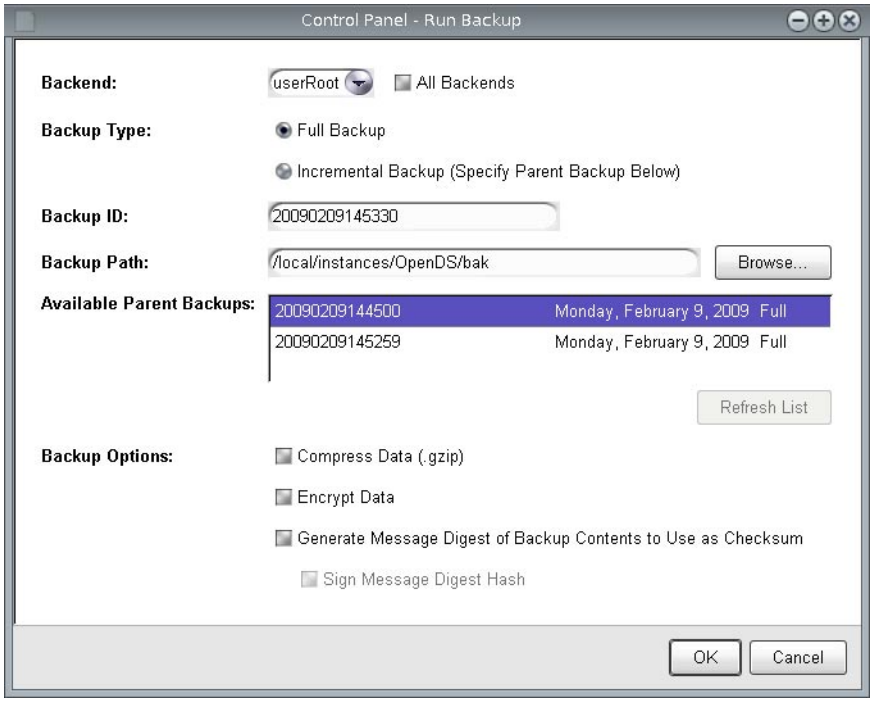
The following procedures describe how to use the Control Panel to back up and restore directory data.

▼ To Back Up Data With the Control Panel

This procedure shows how to use the Control Panel to back up directory data.

1. **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2. **Click the Backup link under Directory Data menu on the left side of the Control Panel window.**
   The Run Backup window appears, displaying fields for specifying how to perform the backup.



3. **Specify the settings for the backup in the fields in this window**

4. **Click the OK button.**
   The Run Backup window displays the progress of the backup operation.

5 **Click the Close button to close the Run Backup window.**

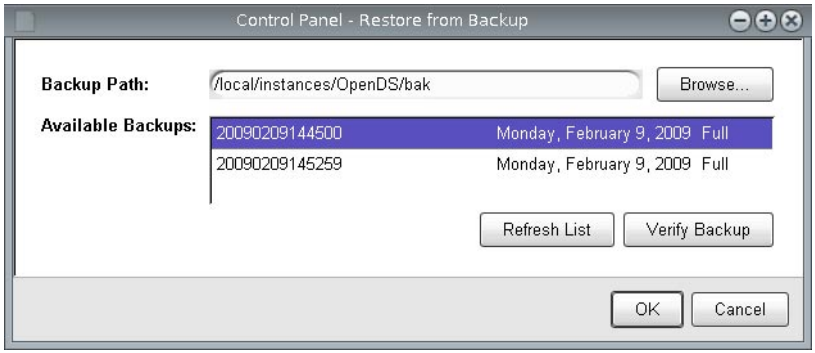
▼ To Restore Data With the Control Panel

This procedure shows how to use the Control Panel to restore directory data that has been backed up previously.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Click the Restore link under Directory Data menu on the left side of the Control Panel window.**

The Restore from Backup window appears, displaying fields for specifying how to perform the backup.



3 **Specify the backups that you want to restore in the fields of the Restore from Backup window.**

You can verify the integrity of the specified backup by clicking the Verify Backup button.

4 **Click the OK button.**

The Restore from Backup window displays the progress of the backup operation.

5 **Click the Close button to close the Restore from Backup window.**

Searching Directory Data

The directory server provides a suite of LDAPv3-compliant command-line tools, including a sophisticated look-up operation in the form of a search function and filters. This section explains how to use the `ldapsearch` command-line utility to locate entries in the directory.

Overview of the ldapsearch Command

The ldapsearch command allows you to enter a search request where you specify the host name, port, bind DN and password plus search criteria to locate entries in the directory. When an LDAP client makes a search request to the directory server, it opens a connection to the directory server over TCP/IP. The client then performs a *bind* operation to the directory server by attempting to match a given entry, which effectively authenticates the client. Most users have the option to bind as a particular user, such as a Directory Administrator or themselves, or to not bind as any user, in which case the directory server assumes that the user is bound as an *anonymous* user.

Because all access to directory data is based on how a connection is bound, the directory server checks the client's privileges to see if the client can run a particular search operation. After the directory server checks the user's access rights, the client passes a search request consisting of a set of search criteria and options to the directory server.

The directory server searches all entries that match the search criteria and options. It then returns the entries, the DN, and all attributes for each entry, in the form of LDIF text to standard output. If an error occurs, the directory server displays an error message indicating the error. Finally, the client closes the connection when the search operation has completed.

ldapsearch Location and Format

The ldapsearch utility is found in the following location:

```
(UNIX, Linux) install-dir/bin
(Windows) install-dir\bat
```

The utility has the following format:

```
ldapsearch optional-options search-filter optional-list-of-attributes
```

where:

- *optional-options* are command-line options that must appear before the search filter.
- *search-filter* is an LDAP search filter either specified on the command-line or in a file.
- *optional-list-of-attributes* is a list of attributes separated by a space. The list of attributes must appear after the search filter.

Common ldapsearch Options

The ldapsearch command has many options to search entries in the directory. Options are allowed in either their short form (for example, `-b` *baseDN*) or their long form (for example, `--baseDN`). The most common command options to use with ldapsearch are as follows:

`-h, --hostname` *address*	Specifies the host name or IP address of the directory server on which the search should be run. It can be an IP address or a resolvable name. If this is not provided, a default value of `localhost` is used.
`-p, --port` *port*	Specifies the directory server port. It should be an integer value between `1` and `65535`, inclusive. If this is not provided, a default port of `389` is used.
`-b, --baseDN` *baseDN*	Specifies the base DN to use for the search operation. If a file containing multiple filters is provided using the `--filename` option, this base DN is used for all of the searches. This is a required option.
`-s, --searchScope` *scope*	Sets the scope for the search operation. Its value must be one of the following: ■ `base`. Searches only the entry specified by the `--baseDN` or `-b` option. ■ `one`. Searches only the entry specified by the `--baseDN` or `-b` option and its immediate children. ■ `sub` or `subordinate`. Searches the entire subtree whose base is the entry specified by the `--baseDN` or `-b` option. This is the default option when no `--searchScope` option is provided.
`-D, --bindDN` *bindDN*	Specifies the DN to use when binding to the directory server through simple authentication. This option is not required when using SASL authentication or anonymous binding.
`-w, --bindPassword` *bindPassword*	Specifies the password to use when binding to the directory server. This option is used for simple authentication, as well as for password-based SASL mechanisms like CRAM-MD5, DIGEST-MD5, and PLAIN. It is not required if anonymous binding is used. This option must not be used in conjunction with the `--bindPasswordFile` option. To prompt for the password, type `-w -`.
`-l, --timeLimit` *numSeconds*	Sets the maximum length of time in seconds that the directory server should spend processing any search request. If this is not provided, no time limit is imposed by the client. Note that the directory server may enforce a lower time limit than the one requested by the client.

`-z, --sizeLimit` *numEntries*	Sets the maximum number of matching entries that the directory server should return to the client. If this is not provided, no maximum size is imposed by the client. Note that the directory server may enforce a lower size limit than the one requested by the client.
`-S, --sortOrder` *sortOrder*	Sorts the results before returning them to the client. The sort order is a comma-delimited list of sort keys, where each sort key consists of the following elements: ■ `+/- (plus or minus sign)`. Indicates that the sort should be in ascending (+) or descending (-) order. If this value is omitted, the sort uses ascending order by default. ■ `Attribute name`. The name of the attribute to sort the data. This element is required. ■ `Name or OID Matching Rule`. An optional colon followed by the name or OID of the matching rule used to perform the sort. If this is not provided, the default ordering matching rule for the specified attribute type is used. For example, the sort order string `sn,givenName` sorts the entries in ascending order first by `sn` and then by `givenName`. Alternately, using `-modifyTimestamp`, the directory server sorts the `modifyTimestamp` attributes with the most recent values first.

Understanding Search Criteria

The `ldapsearch` command requires three sets of information to specify where and what to search in the directory information tree:

- **Base DN**. By specifying the base DN, you are defining the topmost distinguished name (DN) or starting point in the directory to conduct the search. All searches begin at or below the base DN, depending on the scope, and move down the tree, never upwards. Examples of base DNs are: `dc=example,dc=com` and `ou=People,dc=example,dc=com`.
- **Scope**. The scope determines which set of entries at or below the base DN should be evaluated by the search filter. The search scope and base DN together indicate "where" to look for entries in the directory.
- **Search filter**. The search filter specifies the conditions that the entries must meet to be returned to the client.

Specifying Filter Types and Operators

The directory server provides seven types of search filters, defined in the LDAP protocol. With each search filter type, you use operators that test the relationships between two entities, *attribute* and *value*.

The following table shows how search filters are used to return specific entries in a search query.

Search Filter	Operator	Description
Presence	`attr=*`	Return all entries that have any value associated with the specified attribute. The filter uses the wildcard character to denote zero or more characters in the string. For example, the following filter is common and returns all entries that have an object class with any value, which every entry has: `(objectclass=\*)`. **Note –** the LDAP protocol specifies that filters should have the form "(filter)", which includes parentheses surrounded by quotation marks. Although most directory servers accept filters without the parentheses and quotation marks, it is good practice to include them.
Equality	`attr=value`	Return entries containing attributes equal to a specified value. For example: `(sn=Bergin)` returns all entries that have a surname (`sn`) attribute with the value of `Bergin`. **Note –** The `sn` value is case insensitive, so entries associated with `sn=bergin` or `sn=Bergin` will be returned.

Search Filter	Operator	Description
Substring	`attr=<initial-string><any substring><final-string>`	Return entries with attributes containing a specified substring or partial substring. The filter uses the wildcard character to denote zero or more characters in the string. ■ Run an initial substring search that looks for all attribute values that have the characters `Ber` at the start of the string: `(sn=Ber\*)` ■ Specify the middle substring of an attribute value. For example: `(sn=\*erg\*)` ■ Specify the end of a substring of an attribute value. For example: `(sn=\*gin)`. Or you can specify some combination of substrings ■ Specify the initial and middle substring: `(sn=ber\*gi\*)` ■ Specify the initial and ending substrings: `(sn=be\*in)` ■ Specify the middle and end substrings: `(sn=\*er\*in)` **Note –** Substring filters do not use true wild cards such as in system listings or regular expressions. Thus, the following filter would be invalid because of too many criteria: `(sn=\*B\*rg\*n)`.
Greater than or equal to	`attr>=value`	Return entries containing attributes that are greater than or equal to the specified value. For example, `(sn>=Bergin)` returns all entries that have an attribute greater than or equal to the value, `Bergin`, based on the matching rules for attributes (see "Understanding Matching Rules" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*).
Less than or equal to	`attr<=value`	Return entries containing attributes that are less than or equal to the specified value. For example, `(sn&lt;=Bergin)` returns all entries that have an attribute less than or equal to the value, `Bergin`, based on the matching rules for attributes.

Search Filter	Operator	Description
Approximate	`attr~=value`	Return entries containing the specified attribute with a value that is approximately equal to the value specified in the search filter. For example: `(sn~=Bergan)` could return the entry associated with `(sn=Bergin)` or `(sn=Bergan)`. The Approximate search filter works only with English language strings. It does not work with non-ASCII-based strings, such as Ja or Zn.

Extensible match	`attr= attr [":dn"] [":" matchingrule] ":=" value`
	`Or:`
	`[:dn] ":" matchingrule ":=" value`
		Return the results entries when an attribute equals the value with the specified matching rule. LDAP version 3 enables you to build match operators and rules for a particular attribute. Matching rules define how to compare attribute values with a particular syntax. In other words, an extensible search filter enables you to add a matching rule to a search filter. For example, the following search filter compares entries containing the surname attribute with value equal to "`Jensen`" by using the matching rule designated by OID 2.5.13.5: `(sn:2.5.13.5:=Jensen)`. Another example illustrates the use of the "`:dn`" notation to indicate that the OID `2.5.13.5` should be used when making comparisons, and that the attributes of an entry should be considered part of the entry when evaluating the match:`(sn:dn:2.5.13.5:=Jensen)`

Using Compound Search Filters

Multiple search filter components can be combined and evaluated by using the operator:

```
(Boolean-Operator(filter)(filter)(filter))
```

Boolean operators can be combined and nested together to form complex expressions:

```
(Boolean-Operator(filter)(Boolean-operator(filter)(filter)))
```

The following table describes the Boolean operators.

Search Filter	Operator	Description
AND	`(&(filter)(filter))`	All specified filters must be true for the statement to be true. For example, `(&(sn=Carter)(l=Cupertino))` returns all entries that have the surname attribute equal to "Carter" and the location attribute equal to `Cupertino` if any.
OR	`(|(filter)(filter))`	At least one specified filter must be true for the statement to be true. For example, `(|(sn=Carter)(l=Cupertino))` returns all entries that have the surname attribute equal to "Carter" or the location attribute equal to `Cupertino` if any.
NOT	`(!(filter)(filter))`	The specified filter must not be true for the statement to be true. For example, `(!(sn=Bergin))` returns all entries that do *not* have a surname attribute equal to the string `Smith`. The filter also returns all entries that do not have the `sn` attribute.

Using UTF-8 Encoding in Search Filters

UTF8 is a byte-order, variable-length character code for Unicode and a subset of ASCII. You use UTF-8 for multiple-language support by replacing each character of a non 7-bit ASCII character with a byte of a UTF-8 encoding. Typically, you must escape the UTF-8 encoding with a backslash.

For example, the character é has a UTF-8 representation of `c3a9` and è has a UTF-8 representation `c3a8`. A UTF-8 encoding is represented with an escaped backslash. So, é is represented as `\\c3\\a9` and è is represented as `\\c3\\a8`. To represent `cn=Hélène Laurent`, you would use the following encoding:

```
(cn=H\\c3\\a9l\\c3\\a8ne Laurent)
```

Using Special Characters in Search Filters

You must specify special characters (for example, a space, backslash, asterisk, comma, period, or others) by using the escape backslash.

- Asterisk. Represent an asterisk (`*`) as `\\2a`. For example, `Five*Star` would be represented as `"(cn=Five\\2aStar)"`.
- Backslash. Represent a backslash (`\`) as `\\5c`. For example, `c:\\file` would be represented as `"(cn=c:\\5c\\5cfile)"`.
- Parentheses. Represent parentheses `( )` as `\\28` and `\\29`, respectively. For example, `John Doe (II)` would be represented as `"(cn=John Doe \\28II\\29)"`.
- Null. Represent null as `\\00`. For example, 0001 would be represented as `"(bin=\\00\\00\\00\\01)"`.
- Comma. Represent a comma (`,`) by escaping it as `\\,`. For example, `"(cn=Mkt\\,Peru,dc=example,dc=com)"`.

- Space. Generally, use quotation marks around strings that contain a space. For example, `(cn="HR Managers,ou=Groups,dc=example,dc=com")`.

ldapsearch Examples

The following examples show the use of the `ldapsearch` command with various search options. These examples all assume that your current working directory is *install-dir*`/bin` (*install-dir*`\bat` on Windows systems).

The following points pertain to all the examples in this section:

- If the example does not specify a scope (with the `--searchScope` or `-s` option), `ldapsearch` assumes that the scope is `subordinate` or `sub`, which returns the full subtree of the base DN.
- If no attributes are specified, the command returns all attributes and their values.
- If no `--bindDN` and `--bindPassword` are specified, the search uses an anonymous bind.
- If no `--hostname` is specified, the default (`localhost`) is used.

Note – Many UNIX and Linux operating systems provide an installed version of common LDAP-client tools, such as `ldapsearch`, `ldapmodify`, and `ldapdelete` in the `/usr/bin` directory. You should use the `ldapsearch` provided with the directory server to search the directory server. You can check which version of `ldapsearch` you are using by typing the following command:

```
$ which ldapsearch
```

If you are using the `ldapsearch` in `/usr/bin`, put *install-dir*`/bin` at the beginning of your `$PATH`.

▼ To Return All Entries

You can return all entries below a specified branch DN using the presence search filter `(objectclass=*)`. The search filter looks for all entries that have one or more object classes with any value. Because all entries have several object class definitions, the filter guarantees that all entries will be returned.

- **Run the `ldapsearch` command with the filter `(objectclass=*)`.**

```
$ ldapsearch --hostname localhost --port 1389 --baseDN "dc=example,dc=com" \
  "(objectclass=*)"
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example

dn: ou=Groups,dc=example,dc=com
```

```
objectClass: organizationalunit
objectClass: top
ou: Groups

dn: cn=Directory Administrators,ou=Groups,dc=example,dc=com
objectClass: groupofuniquenames
objectClass: top
ou: Groups
cn: Directory Administrators
uniquemember: uid=kvaughan, ou=People, dc=example,dc=com
uniquemember: uid=rdaugherty, ou=People, dc=example,dc=com
uniquemember: uid=hmiller, ou=People, dc=example,dc=com
...
```

To Search For a Specific User

You can use an equality filter to locate a specific user in the directory. This example locates an employee with the common name of "Frank Albers".

- **Run the `ldapsearch` command with the filter `"(cn=Frank Albers)"`.**

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com  "(cn=Frank Albers)"

dn: uid=falbers,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
givenName: Frank
uid: falbers
cn: Frank Albers
sn: Albers
telephoneNumber: +1 408 555 3094
userPassword: {SSHA}nDTQJ9DDiMUrBwR0WNKq0tgS4iB2A9QJFgpZiA==
roomNumber: 1439
ou: Accounting
ou: People
l: Sunnyvale
mail: falbers@example.com
facsimileTelephoneNumber: +1 408 555 9751
```

To Search for Specific User Attributes

You can use an equality filter to locate an entry's attribute(s) in the directory. Specify one or more attributes by placing them after the search filter. This example locates the `telephoneNumber` and `mail` attributes from the user entry for `Frank Albers`.

- **Run the `ldapsearch` command with the filter `"(cn=Frank Albers)"` and the corresponding attributes.**

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com \
 "(cn=Frank Albers)" telephoneNumber mail
dn: uid=falbers,ou=People,dc=example,dc=com
telephoneNumber: +1 408 555 3094
mail: falbers@example.com
```

To Perform a Search With Base Scope

Together with the search base DN, the scope determines what part of the directory information tree (DIT) is examined. A base scope examines only the level specified by the base DN (and none of its child entries). You specify a base scope by using the `--searchScope base` option or its short form equivalent `-s base`.

- **Run the `ldapsearch` command with the `--searchScope base` option.**

```
$ ldapsearch --hostname localhost --port 1389 --baseDN "dc=example,dc=com" \
  --searchScope base "(objectclass=*)"
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
```

To Perform a Search With One-Level Scope

A one-level scope examines only the level immediately below the base DN. You specify a one-level scope by using the `--searchScope one` option or its short form equivalent `-s one`. This example displays the entries immediately below the base DN.

- **Run the `ldapsearch` command with the `--searchScope one` option.**

```
$ ldapsearch --hostname localhost --port 1389 --baseDN "dc=example,dc=com" \
  --searchScope one "(objectclass=*)"
dn: ou=Groups,dc=example,dc=com
objectClass: top
objectClass: organizationalunit
ou: Groups
dn: ou=People,dc=example,dc=com
objectClass: top
objectClass: organizationalunit
ou: People
dn: ou=Special Users,dc=example,dc=com
objectClass: top
objectClass: organizationalUnit
ou: Special Users
description: Special Administrative Accounts
dn: ou=Company Servers,dc=example,dc=com
```

```
objectClass: top
objectClass: organizationalUnit
ou: Company Servers
description: Standard branch for Company Server registration
```

To Perform a Search With Subtree Scope

The subtree scope examines the subtree below the base DN and includes the base DN level. You specify a subtree scope using the `--searchScope sub` option, or its short form equivalent `-s sub`. If you do not specify the `--searchScope`, `ldapsearch` assumes a subtree scope.

- **Run the `ldapsearch` command with the `--searchScope sub` option.**

```
$ ldapsearch --hostname localhost --port 1389 \
  --baseDN "cn=Directory Administrators,ou=Groups,dc=example,dc=com" \
  --searchScope sub "(objectclass=*)"
dn: cn=HR Managers,ou=groups,dc=example,dc=com
objectClass: groupOfUniqueNames
objectClass: top
ou: groups
description: People who can manage HR entries
cn: HR Managers
uniqueMember: uid=kvaughan, ou=People, dc=example,dc=com
uniqueMember: uid=cschmith, ou=People, dc=example,dc=com
```

To Return Attribute Names Only

The `ldapsearch` command provides a convenient option to check if an attribute is present in the directory. Use the `--typesOnly` option or its short form equivalent `-A` to instruct the directory server to display the attribute names but not their values.

- **Run the `ldapsearch` command with the `--typesOnly` option.**

```
$ ldapsearch --hostname localhost --port 1389 \
  --baseDN "dc=example,dc=com" --typesOnly "(objectclass=*)"
dn: dc=example,dc=com
objectClass
dc
dn: ou=Groups,dc=example,dc=com
objectClass
ou ...
```

To Return User Attributes Only

You can use `ldapsearch` to return only user attributes for entries that match the search filter, by including an asterisk `*`. User attributes (as opposed to operational attributes) store user information in the directory. If you do not specify the asterisk, the user attributes are returned by default. You must escape the asterisk appropriately for your shell.

- **Run the `ldapsearch` command, specifying `'*'` after the search filter.**

```
$ ldapsearch --hostname localhost --port 1389 --baseDN "dc=example,dc=com" \
  "(objectclass=*)" '*'
dn: cn=Aggie Aguirre,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetorgperson
objectClass: organizationalperson
objectClass: top
postalAddress: Aggie Aguirre$15172 Jackson Street$Salt Lake City, MI  49843
postalCode: 49843
uid: user.99
description: This is the description for Aggie Aguirre.
employeeNumber: 99
initials: AGA
givenName: Aggie
pager: +1 514 297 1830
mobile: +1 030 300 0720
cn: Aggie Aguirre
telephoneNumber: +1 730 027 2062
sn: Aguirre
street: 15172 Jackson Street
homePhone: +1 229 128 3072
mail: user.99@maildomain.net
l: Salt Lake City
st: MI
```

▼ To Return Base DNs Only

You can use `ldapsearch` to return only the base DNs for entries that match the search filter by including a `1.1` string after the search filter.

- **Run the `ldapsearch` command, specifying `1.1` after the search filter.**

```
$ ldapsearch --hostname localhost --port 1389 --baseDN "dc=example,dc=com" \
  "(objectclass=*)" 1.1
version: 1
dn: cn=Richard Arnold,ou=people,dc=example,dc=com

dn: cn=Kevin Booysen,ou=people,dc=example,dc=com

dn: cn=Steven Morris,ou=people,dc=example,dc=com

dn: cn=Leila Shakir,ou=people,dc=example,dc=com

dn: cn=Emily Smith,ou=people,dc=example,dc=com
...
```

To Search For Specific Object Classes

You can search all entries where the attributes are referenced by a specific object class by prepending a @ character to the object class name. For example, to view all entries that have an object class of groupOfUniqueNames, include @groupOfUniqueNames after the search filter.

- **Run the ldapsearch command, specifying @ and the object class after the search filter.**

```
$ ldapsearch --hostname localhost --port 1389 \
  --baseDN "ou=Groups,dc=example,dc=com" "(objectclass=*)" @groupOfUniqueNames
dn: ou=Groups,dc=example,dc=com
ou: Groups
objectClass: organizationalunit
objectClass: top
dn: cn=Directory Administrators,ou=Groups,dc=example,dc=com
ou: Groups
objectClass: groupofuniquenames
objectClass: top
cn: Directory Administrators
uniqueMember: uid=kvaughan, ou=People, dc=example,dc=com
uniqueMember: uid=rdaugherty, ou=People, dc=example,dc=com
uniqueMember: uid=hmiller, ou=People, dc=example,dc=com ...
```

To Return a Count of All Entries in the Directory

The ldapsearch command provides the --countentries to return the total number of entries in the directory. The directory server returns all entries that match the search filter and displays the total number on the last line. This example determines the number of employee entries whose location is Cincinnati.

- **Run the ldapsearch command with the --countentries option.**

```
$ ldapsearch --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password --baseDN dc=example,dc=com --countentries "l=Cincinnati"
dn: cn=Adi Adamski,ou=People,dc=example,dc=com
...
l: Cincinnati
st: OH

dn: Aggi Aguinsky,ou=People,dc=example,dc=com
objectClass: person
...
l: Cincinnati
st: OH

# Total number of matching entries: 2
```

To Perform a Search With a Compound Filter

Compound search filters involve multiple tests using the boolean operators AND (&), OR (|), or NOT (!). You can combine and nest boolean operators and filters together to form complex expressions. The following example searches for all entries for employees named `Jensen` who work in `Cupertino`. The command returns two results.

- **Run the `ldapsearch` command with a compound search filter.**

```
$ ldapsearch --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password --baseDN dc=example,dc=com "(&(sn=jensen)(l=Cupertino))"
dn: uid=bjensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Product Development
ou: People
sn: Jensen
...
l: Cupertino
st: CA

dn: uid=rjensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Accounting
ou: People
sn: Jensen
...
l: Cupertino
st: CA
```

To Perform a Search Using a Filter File

You can place complex or multiple filters in a file by using the `--filename` option. If the file contains multiple filters, the file should be structured with one filter per line. Searches are performed using the same connection to the directory server in the order in which they appear in the filter file. If the `--filename` option is used, any trailing options are treated as separate attributes. Otherwise, the first trailing option must be the search filter.

This example searches all entries for employees named `Jensen` who work in `Cupertino` and who do not work in the Accounting department.

1. **Create the filter file.**

   For this example, create a file called `myfilter.txt` with the following content: `(&(sn=jensen)(l=Cupertino)(!(ou=Accounting)))`

2. **Run the `ldapsearch` command, specifying the file name as a filter.**

```
$ ldapsearch --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
--bindPassword password --baseDN dc=example,dc=com --filename myfilter.txt
dn: uid=bjensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Product Development
ou: People
sn: Jensen
l: Cupertino
cn: Barbara Jensen
cn: Babs Jensen
telephoneNumber: +1 408 555 1862
givenName: Barbara
uid: bjensen
mail: bjensen@example.com
```

▼ To Limit the Number of Entries Returned in a Search

You can limit the number of entries that are returned by using the `-z` or `--sizeLimit` option. If the number of entries exceeds the number that is specified, the search returns the specified number of entries, then returns an error stating that the size limit was exceeded. The following example requests a maximum of 5 entries.

- **Run the `ldapsearch` command with the `--sizeLimit` option.**

```
$ ldapsearch --hostname localhost --port 1389 -b "dc=example,dc=com" \
  --sizeLimit 5 "objectclass=*" 1.1
dn: dc=example,dc=com

dn: ou=People,dc=example,dc=com

dn: uid=user.0,ou=People,dc=example,dc=com

dn: uid=user.1,ou=People,dc=example,dc=com

dn: uid=user.2,ou=People,dc=example,dc=com

SEARCH operation failed
Result Code:  4 (Size Limit Exceeded)
Additional Information:  This search operation has sent the maximum of 5 entries
 to the client
```

Using Advanced Search Features

The directory server supports LDAPv3-compliant search functionality by using the `ldapsearch` command. You can use special attributes, security options, and LDAP controls with the search process, based on your system configuration. For additional information, see "Searching Directory Data" on page 135, "Using a Properties File With Directory Server Commands" in *Command Line Usage for Sun Virtual Directory Proxy 1.0*, and "ldapsearch" in *Sun OpenDS Standard Edition 2.0 Command-Line Usage Guide*.

Searching for Special Entries and Attributes

This section describes how to search for operational attributes and how to search the Root DSE entry.

▼ To Search for Operational Attributes

Operational attributes are used for storing information needed for processing by the directory server itself or for holding any other data maintained by the directory server that was not explicitly provided by clients. Operational attributes are not included in entries returned from search operations unless they are explicitly included in the list of search attributes. You can request the directory server to return operational attributes by adding + (the plus sign) in your `ldapsearch` command.

- **Run the `ldapsearch` command with the + character.**

  You must escape the character using a means appropriate to your shell.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" \
  -w password -b "dc=example,dc=com" "(objectclass=*)" "+"
...
dn: cn=PD Managers,ou=groups,dc=example,dc=com
numSubordinates: 0
hasSubordinates: false
subschemaSubentry: cn=schema
entryDN: cn=pd managers,ou=groups,dc=example,dc=com
entryUUID: 38666d52-7a53-332e-902f-e34dd4aaa7a0
...
```

▼ To Search the Root DSE Entry

The Root DSE is a special entry that provides information about the server's name, version, naming contexts, and supported features. Because many of the attributes are operational, you must specify + (the plus sign) to display the attributes of the Root DSE entry.

- **Run the `ldapsearch` command with a baseDN of "".**

  Specify the scope as `base` and include the + character to display operational attributes.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" \
  -w password -b "" --searchScope base "(objectclass=*)" "+"
dn:
supportedExtension: 1.3.6.1.4.1.4203.1.11.3
supportedExtension: 1.3.6.1.4.1.4203.1.11.1
supportedExtension: 1.3.6.1.4.1.26027.1.6.2
supportedExtension: 1.3.6.1.4.1.26027.1.6.1
supportedExtension: 1.3.6.1.1.8
supportedExtension: 1.3.6.1.4.1.1466.20037
...
```

To Search for ACI Attributes

The directory server stores access control instructions (ACIs) as one or more values of the `aci` attribute on an entry to allow or deny access to the directory database. The `aci` attribute is a multi-valued operational attribute that can be read and modified by directory users and that should itself be protected by ACIs. Administrative users are usually given full access to the `aci` attribute and can view its values by running an `ldapsearch` command.

- **Run the `ldapsearch` command as follows:**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" \
  -w password -b dc=example,dc=com --searchScope base "(aci=*)" aci
dn: dc=example,dc=com
aci: (target ="ldap:///dc=example,dc=com")(targetattr h3.="userPassword")
  (version 3.0;acl "Anonymous read-search access";allow (read, search, compare)
  (userdn = "ldap:///anyone");)
aci: (target="ldap:///dc=example,dc=com") (targetattr = "*")
  (version 3.0; acl "allow all Admin group"; allow(all)
  groupdn = "ldap:///cn=Directory Administrators,ou=Groups,dc=example,dc=com";)
```

To Search the Schema Entry

The directory server holds schema information in the schema entry (`cn=schema`) for the object classes and attributes defined on your instance.

- **Run the `ldapsearch` command on the `cn=schema` base DN.**

  Because the attributes in the schema are operational attributes, you must include "+" at the end of your search.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" \
  -w password -b cn=schema --searchScope base "(objectclass=*)" "+"
dn: cn=schema
nameForms: ( 1.3.6.1.1.10.15.1 NAME 'uddiBusinessEntityNameForm' OC uddiBusiness
 Entity MUST ( uddiBusinessKey ) X-ORIGIN 'RFC 4403' )
```

```
nameForms: ( 1.3.6.1.1.10.15.2 NAME 'uddiContactNameForm' OC uddiContact MUST
 (uddiUUID ) X-ORIGIN 'RFC 4403' )
nameForms: ( 1.3.6.1.1.10.15.3 NAME 'uddiAddressNameForm' OC uddiAddress MUST
 (uddiUUID ) X-ORIGIN 'RFC 4403' )
...
attributeTypes: ( 1.3.6.1.1.1.1.12 NAME 'memberUid' EQUALITY caseExactIA5Match
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 X-ORIGIN 'draft-howard-rfc2307bis' )
attributeTypes: ( 1.3.6.1.1.1.1.13 NAME 'memberNisNetgroup' EQUALITY caseExactIA
 5Match SUBSTR caseExactIA5SubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
 X-ORIGIN 'draft-howard-rfc2307bis' )
attributeTypes: ( 1.3.6.1.1.1.1.14 NAME 'nisNetgroupTriple' DESC 'Netgroup
 triple' EQUALITY caseIgnoreIA5Match SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 X-ORIGIN
 'draft-howard-rfc2307bis' )
...
```

To Search the Configuration Entry

The directory server stores its configuration under the `cn=config` entry. Direct access to this entry over LDAP is not advised. The configuration is accessible and modifiable by using the `dsconfig` command. `dsconfig` connects to the directory server over SSL via the administration connector. For more information, see "Managing Administration Traffic to the Server" on page 5.

- **To search the configuration entry using `dsconfig` in interactive mode, run the command as follows:**

  ```
  $ dsconfig -h localhost -p 1389 -D "cn=Directory Manager" -w password
  ```

  For more information about accessing the server configuration by using `dsconfig`, see "Configuring the Directory Server With `dsconfig`" on page 7.

To Search the Monitoring Entry

The directory server monitor entry `cn=monitor` provides statistical information about the server performance, state, and version. You can access this information by using the `ldapsearch` command.

Although you can access `cn=monitor` using any configured LDAP connection handler, it is recommended that you use the administration connector for all access to administrative suffixes. Using the administration connector ensures that monitoring data is not polluted and that server administration takes precedence over user traffic. To use the administration connector, specify the administration port, and include the `--useSSL` option. For more information, see "Managing Administration Traffic to the Server" on page 5.

- **Run the `ldapsearch` command on the base DN `cn=monitor`.**

  ```
  $ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" \
    -w password -b cn=monitor "(objectclass=*)"
  dn: cn=monitor
  ```

```
objectClass: top
objectClass: extensibleObject
objectClass: ds-monitor-entry
currentTime: 20081103161832Z
startTime: 20081103132044Z
productName: OpenDS Directory Server
...
```

Searching Over SSL

If you have configured the directory server to accept SSL connections by using a self-signed certificate or certificate, you can search using client authentication. The following procedures show how to search the directory over SSL using various authentication mechanisms.

▼ To Search Over SSL With Blind Trust

You can configure the client to automatically trust any certificate that the server presents to it. However, this method is not secure and is vulnerable to man-in-the-middle attacks. Generally, you should use this type of authentication for testing purposes only.

- **Run the `ldapsearch` command with the `--trustAll` option.**

  The following command searches the Root DSE.

  ```
  $ ldapsearch -h localhost -p 1636 --useSSL --trustAll -b "" \
    --searchScope base "(objectClass=*)"
  ```

▼ To Search Over SSL Using a Trust Store

You can configure the client to use a certificate trust store, which contains information about the certificates it can trust. The client can check any server certificate to those listed in its trust store. If the client finds a match, a secure communication can take place with the server. If no match is found, the server cannot be trusted. You must ensure that the presented certificate is valid and add it to the trust store, which then allows secure communication.

- **Run the `ldapsearch` command with the `--trustStorePath` option.**

  The following command searches the Root DSE using a trust store.

  ```
  $ ldapsearch -h localhost -p 1636 --useSSL \
    --trustStorePath /home/scarter/security/cert.db -b "" \
    --searchScope base "(objectClass=*)"
  ```

▼ To Search Over SSL With No Trust Store

If no trust store is specified, you are prompted as to whether the certificate that was presented to the client should be trusted.

- **Run the `ldapsearch` command without the `--trustStorePath` option.**

  The following command searches the Root DSE without using a trust store.

```
$ ldapsearch -h localhost -p 1636 --useSSL -b "" \
  --searchScope base "(objectclass=*)"
The server is using the following certificate:
Subject DN: CN=example.com, O=Example Corp, C=US
Issuer DN: CN=example.com, O=Example Corp, C=US
Validity: Fri Mar 02 16:48:17 CST 2007 through Thu May 31 17:48:17 CDT 2007
Do you wish to trust this certificate and continue connecting to the server?
Please enter "yes" or "no": yes

dn: objectClass: ds-rootDSE
objectClass: top
```

To Search Over SSL Using a Keystore

If the client is required to present its own certificate to the directory server, that client must know which certificate keystore to use. The client can determine the certificate keystore by specifying the `--keyStorePath` option with either the `--keyStorePassword` or `--keyStorePasswordFile`. This scenario typically occurs when the client performs a SASL EXTERNAL authentication or if the server always requires the client to present its own certificates.

- **Run the `ldapsearch` command with the `--keyStore...` options.**

  The following command searches the Root DSE using a trust store and a key store.

```
$ ldapsearch -h localhost -p 1636 --useSSL \
  --keyStorePath /home/scarter/security/key.db \
  --keyStorePasswordFile /home/keystore.pin \
  --trustStorePath /home/scarter/security/cert.db --useSASLExternal -b "" \
  --searchScope base "(objectClass=*)"
```

To Search Using StartTLS

The process for using StartTLS with the `ldapsearch` utility is very similar to the process for using SSL. However, you must do the following:

- Use the port on which the server is listening for *unencrypted* LDAP requests
- Indicate that StartTLS should be used instead of SSL (that is, use the `--startTLS` option instead of the `--useSSL` option).

- **Run the `ldapsearch` command with the `--startTLS` option.**

  The following command searches the Root DSE using startTLS.

```
$ ldapsearch -h localhost -p 1389 --startTLS \
  -b "" --searchScope base "(objectClass=*)"
```

To Search Using SASL With DIGEST-MD5 Client Authentication

The directory server supports a number of Simple Authentication and Security Layer (SASL) mechanisms. DIGEST-MD5 is one form of SASL authentication to the server that does not expose the clear-text password.

- **Run the `ldapsearch` command with the appropriate `--saslOption` options.**

  The `authid` option specifies the identity of the user that is authenticating to the server. The option can be in the form of a `dn` (for example, `dn:uid=scarter,dc=example,dc-com`) or a user name (for example, `authid=u:sam.carter`). The attribute can be used to indicate that the search operation should be performed under the authority of another user after authentication. The `realm` specifies the fully qualified name of the server host machine and is optional.

  This example searches the Root DSE.

```
$ ldapsearch -h localhost -p 1636 --useSSL \
  --trustStorePath /home/cert.db --certNickName "my-cert" -w - \
  --saslOption mech=DIGEST-MD5 --saslOption realm="example.com" \
  --saslOption authid="dn:uid=scarter,dc=example,dc=com" -b "" "(objectclass=*)"
```

To Search Using SASL With the GSSAPI Mechanism

The GSSAPI mechanism performs authentication in a Kerberos environment and requires that the client system be configured to participate in such an environment.

- **Run the `ldapsearch` command to search as a user who already has a valid Kerberos session.**

  The `authid` attribute specifies the authentication ID that should be used to identify the user.

  This example searches the Root DSE.

```
$ ldapsearch -h localhost -p 1389 \
  --saslOption mech=GSSAPI --saslOption authid="dn:uid=scarter,dc=example,dc=com" \
  --searchScope "" -b "" "(objectclass=*)"
```

To Search Using SASL With the PLAIN Mechanism

The PLAIN mechanism performs authentication in a manner similar to LDAP simple authentication except that the user is identified in the form of an authorization ID rather than a full DN.

- **Run the `ldapsearch` command to search as a user who already has a valid Kerberos session.**

  The `authid` attribute specifies the authentication ID that should be used to identify the user.

  This example searches the Root DSE.

```
$ ldapsearch -h localhost -p 1389 \
  --saslOption mech=PLAIN --saslOption authid="dn:uid=scarter,dc=example,dc=com" \
  --searchScope "" -b "" "(objectclass=*)"
```

Searching Using Controls

LDAP controls extend the functionality of LDAP commands, such as `ldapsearch`, to carry out additional operations on top of the search. Each control is defined as an object identifier (OID) that uniquely identifies the control, a criticality flag, and any associated values. If the client sets the criticality flag when sending the control to the directory server, the directory server must either perform the operation with the control or not process it. If the flag is not set by the client, the directory server is free to ignore the control if it cannot process it.

You can use multiple controls in a single operation, such as the virtual list view with server-side sorting. The virtual list view control requires additional explanation and is therefore described in its own section, following this one.

▼ To View the Available Controls

You can view the current list of controls for your directory server by searching the Root DSE entry for the `supportedControl` attribute.

- **Run the `ldapsearch` command on the Root DSE entry.**

```
$ ldapsearch -h localhost -p 1389 -b "" --searchScope base \
  "(objectclass=*)" supportedControl
dn:
supportedControl: 1.2.826.0.1.3344810.2.3
supportedControl: 1.2.840.113556.1.4.319
supportedControl: 1.2.840.113556.1.4.473
supportedControl: 1.2.840.113556.1.4.805
supportedControl: 1.3.6.1.1.12
supportedControl: 1.3.6.1.1.13.1
supportedControl: 1.3.6.1.1.13.2
supportedControl: 1.3.6.1.4.1.26027.1.5.2
supportedControl: 1.3.6.1.4.1.42.2.27.8.5.1
supportedControl: 1.3.6.1.4.1.42.2.27.9.5.2
supportedControl: 1.3.6.1.4.1.42.2.27.9.5.8
supportedControl: 1.3.6.1.4.1.4203.1.10.2
supportedControl: 1.3.6.1.4.1.7628.5.101.1
supportedControl: 2.16.840.1.113730.3.4.12
supportedControl: 2.16.840.1.113730.3.4.16
supportedControl: 2.16.840.1.113730.3.4.17
supportedControl: 2.16.840.1.113730.3.4.18
supportedControl: 2.16.840.1.113730.3.4.19
supportedControl: 2.16.840.1.113730.3.4.2
supportedControl: 2.16.840.1.113730.3.4.3
supportedControl: 2.16.840.1.113730.3.4.9
```

The controls are returned as a list of OIDs. See the following table for a description of the control that corresponds to each OID. Note that not all of these controls can be used with the `ldapsearch` command.

OID	Control
1.2.826.0.1.3344810.2.3	Matched Values Control
1.2.840.113556.1.4.319	Simple Paged Results Control
1.2.840.113556.1.4.473	Server-Side Sort Control
1.2.840.113556.1.4.805	Subtree Delete Control
1.3.6.1.1.12	LDAP Assertion Control
1.3.6.1.1.13.1	LDAP Pre-Read Control
1.3.6.1.1.13.2	LDAP Post-Read Control
1.3.6.1.4.1.26027.1.5.2	Replication Repair Control
1.3.6.1.4.1.42.2.27.8.5.1	Password Policy control
1.3.6.1.4.1.42.2.27.9.5.2	Get Effective Rights Control
1.3.6.1.4.1.42.2.27.9.5.8	Account Usability Request Control
1.3.6.1.4.1.4203.1.10.2	LDAP No-Op Control
1.3.6.1.4.1.7628.5.101.1	LDAP Subentry Request Control
2.16.840.1.113730.3.4.12	Proxied Authorization v1 Control
2.16.840.1.113730.3.4.16	Authorization Identity Control
2.16.840.1.113730.3.4.17	Real Attributes Only Control
2.16.840.1.113730.3.4.18	Proxied Authorization v2 Control
2.16.840.1.113730.3.4.19	Virtual Attributes Only Control
2.16.840.1.113730.3.4.2	Manage DSA IT Control
2.16.840.1.113730.3.4.3	Persistent Search Control
2.16.840.1.113730.3.4.9	Virtual List View Control

▼ To Search Using the Account Usability Request Control

The Account Usability Request Control determines if a user account can be used to authenticate to a server. If the user account is available, the control adds a message before any entry about whether the account is usable.

You can specify the Account Usability Request Control with `ldapsearch` in the following ways:

- **OID.** Use the `--control` or `-J` option with the Account Usability Request Control OID: `1.3.6.1.4.1.42.2.27.9.5.8` with no value.

- **Named constant.** Use a named constant, `accountusable` or `accountusability`, with the `--control` or `-J` option, instead of using the Account Usability Request Control OID. For example, use `-J accountusable` or `-J accountusability` with the `ldapsearch` command.

- **Use the `ldapsearch` command with the `--control` option or its short form `-J`.**

```
$ ldapsearch -h localhost -p 1389 -b "dc=example,dc=com" \
  --searchScope sub -J "accountusability:true" "(objectclass=*)"
# Account Usability Response Control
# The account is usable
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
...
```

To Search Using the Authorization Identity Request Control

The Authorization Identity Request Control allows the client to obtain the authorization identity for the client connection during the LDAP bind request. The authorization ID returned by the server is displayed to the client as soon as authentication has completed. The line containing the authorization ID is prefixed with a # character, making it a comment if the output is to be interpreted as an LDIF.

You can specify the Authorization Identity Request Control with `ldapsearch` in a number of ways:

- **OID.** Use the `--control` or `-J` option with the Authorization Identity Request Control OID: `2.16.840.1.113730.3.4.16` with no value.
- **Named constant.** Use a named constant, `authzid` or `authorizationidentity` with the `-control` or `-J` option instead of using the Authorization Identity Request Control OID. For example, use `-J authzid` or `-J authorizationidentity` with the `ldapsearch` command.

- **Use the `ldapsearch` command with the `--reportAuthzID` option.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" \
  -w password -b dc=example,dc=com --searchScope base \
  --reportAuthzID "(objectclass=*)"
# Bound with authorization ID dn:cn=Directory Manager,cn=Root DNs,cn=config
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
```

To Search Using the Get Effective Rights Control

The Get Effective Rights Control enables you to evaluate existing or new ACIs and to see the effective rights that they grant for a user on a specified entry.

The response to this control is to return the effective rights information about the entries and attributes in the search results. This extra information includes read and write permissions for each entry and for each attribute in each entry. The permissions can be requested for the bind DN used for the search or for an arbitrary DN, allowing administrators to test the permissions of directory users.

The `ldapsearch` command provides two ways to use the Get Effective Rights Control:

- Use `-J effectiverights` or the OID `-J "1.3.6.1.4.1.42.2.27.9.5.2"`. The request only takes an authorization ID (`authzid`). If you specify a NULL value for the authorization ID (`authzid`), the bind user is used as the `authzid`.
- Use `-g dn:"`*dn*`"`. The command option shows the effective rights of the user binding with the given DN. You can use this option together with the `-e` option to include the effective rights on the named attributes. You can use the option to determine if a user has permission to add an attribute that does not currently exist in an entry.

Note – You cannot use the `-g` option with the `-J` option.

To view effective rights, you should specify the virtual attributes `aclRights` and `aclRightsInfo`, which are generated by the server in response to the effective rights request. Thus, you should not use these attributes in search commands of any kind.

1 Use the `ldapsearch` command to display the effective rights of all users.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com -J effectiverights "(objectclass=*)" aclRights

dn: dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: ou=Groups, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

dn: cn=HR Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0

...
```

2 **Use the `ldapsearch` command to display the effective rights of a specific user.**

This example uses the `--getEffectiveRightsAuthzid` option. You can also use the `--control` or `-J` option, such as `-J geteffectiverights`.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com \
  --getEffectiveRightsAuthzid "dn:uid=scarter,ou=People,dc=example,dc=com" \
  "(uid=scarter)" aclRights
dn: uid=scarter,ou=People,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:1,proxy:0
```

3 **Use the `ldapsearch` command to display effective rights information for a specific user.**

The `aclRightsInfo` attribute provides more detailed logging information that explains how effective rights are granted or denied.

```
ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com \
  --getEffectiveRightsAuthzid "dn:uid=scarter,ou=People,dc=example,dc=com"\
  "(uid=scarter)" aclRightsInfo

dn: uid=scarter,ou=People,dc=example,dc=com
aclRightsInfo;logs;entryLevel;add: acl_summary(main): access not allowed(add) on
entry/attr(uid=scarter,ou=People,dc=example,dc=com, NULL) to
 (uid=scarter,ou=People,dc=example,dc=com)
(not proxied) ( reason: no acis matched the subject )
aclRightsInfo;logs;entryLevel;proxy: acl_summary(main): access not allowed(proxy ) on
entry/attr(uid=scarter,ou=People,dc=example,dc=com, NULL) to
 (uid=scarter, ou=People,dc=example,dc=com)
(not proxied) ( reason: no acis matched the subject )
aclRightsInfo;logs;entryLevel;write: acl_summary(main): access allowed(write) on
entry/attr(uid=scarter,ou=People,dc=example,dc=com, NULL) to
 (uid=scarter,ou=People,dc=example,dc=com)
(not proxied) ( reason: evaluated allow , deciding_aci : Allow self entry modification)
aclRightsInfo;logs;entryLevel;read: acl_summary(main): access allowed(read) on
entry/attr(uid=scarter,ou=People,dc=example,dc=com, NULL) to
 (uid=scarter,ou=People,dc=example,dc=com)
(not proxied) ( reason: evaluated allow , deciding_aci: Anonymous extended
 operation access)
aclRightsInfo;logs;entryLevel;delete: acl_summary(main): access not allowed(delete) on
entry/attr(uid=scarter,ou=People,dc=example,dc=com, NULL) to
 (uid=scarter,ou=People,dc=example,dc=com)
(not proxied) ( reason: no acis matched the subject )
```

▼ To Search Using the LDAP Assertion Control

The LDAP Assertion Control allows you to specify a condition that must evaluate to true for the searching operation to process. The value of the control should be in the form of an LDAP search filter. The server tests the base object before searching for entries that match the search scope and filter. If the assertion fails, no entries are returned.

This example determines first if the assertion is met, and returns the entry if it matches the search filter.

- **Run the `ldapsearch` command with the `--assertionFilter` option using the assertion `(objectclass=top)`.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b "cn=HR Managers,ou=Groups,dc=example,dc=com" \ -s sub \
  --assertionFilter "(objectclass=top)" "(objectclass=*)"
dn: cn=HR Managers,ou=groups,dc=example,dc=com
objectClass: groupOfUniqueNames
objectClass: top
ou: groups
description: People who can manage HR entries
uniqueMember: uid=kvaughan, ou=People, dc=example,dc=com
uniqueMember: uid=cschmith, ou=People, dc=example,dc=com
cn: HR Managers
```

▼ To Search Using the LDAP Subentry Control

The LDAP Subentry Control allows the client to request that the server return only entries with the `ldapSubEntry` object class during a search operation. LDAP subentries are *operational objects*, similar to operational attributes, that are returned only if explicitly requested. Typically, you can use the control when searching the schema.

You can specify the LDAP Subentry Control with `ldapsearch` with base-level scope (that is, `--searchScope base`) in a number of ways:

- **Equality filters**. Use the equality filter, `(objectclass=ldapSubentry)`.
- **OID**. Use the `--control` or `-J` option with the LDAP Subentry Control OID: `1.3.6.1.4.1.7628.5.101.1` with no value.
- **Named constant**. Use the named constant, `subentries`, with the `--control` or `-J` option instead of using the LDAP Subentry Control OID. For example, use `-J subentries` with the `ldapsearch` command.

- **Run the `ldapsearch` command with the `-J subentries` option.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b "cn=schema" -s base -J subentries "(objectclass=*)"
```

▼ To Search Using the Manage DSA IT Control

The Manage DSA IT Control allows the client to request that the server treat smart referrals as regular entries during the search. A *smart referral* is an entry that references another server or location in the directory information tree `DIT` and contains the `referral` object class with one or more attributes containing the LDAP URLs that specify the referral.

You can specify the Manage DSA IT Control with `ldapsearch` in a number of ways:

- **OID**. Use the `--control` or `-J` option with the Manage DSA IT Control OID: `2.16.840.1.113730.3.4.2` with no value.
- **Named constant**. Use the named constant, `managedsait` with the `--control` or `-J` option instead of the Manage DSA IT Control OID. For example, use `-J managedsait` with the `ldapsearch` command.

- **Run the `ldapsearch` command with the `-J` option.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com -J managedsait "(uid=president)" ref
dn: uid=president,ou=People,dc=example,dc=com
ref: ldap://example.com:389/dc=example,dc=com??sub?(uid=bjensen)
```

Note – Without the `-J managedsait` argument, the command returns the referred entry.

To Search Using the Matched Values Filter Control

The Matched Values Filter Control allows clients to request a subset of attribute values from an entry that evaluate to TRUE. This control allows the user to selectively read a subset of attribute values without retrieving all values, and then scan for the desired set locally.

- **Run the `ldapsearch` command with the `--matchedValuesFilter` option.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b ou=groups,dc=example,dc=com --matchedValuesFilter "(uniquemember=uid=kvaughan*)"
  "(objectclass=*)"
dn: ou=Groups,dc=example,dc=com
dn: cn=Directory Administrators,ou=Groups,dc=example,dc=com
uniqueMember: uid=kvaughan, ou=People, dc=example,dc=com
dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
dn: cn=HR Managers,ou=groups,dc=example,dc=com
uniqueMember: uid=kvaughan, ou=People, dc=example,dc=com
dn: cn=QA Managers,ou=groups,dc=example,dc=com
dn: cn=PD Managers,ou=groups,dc=example,dc=com
```

To Search Using the Password Policy Control

The Password Policy Control allows a client to request information about the current password policy information for a user entry.

You can specify the Password Policy Control with `ldapsearch` in a number of ways:

- **OID**. Use the `--control` or `-J` option with the Password Policy Control OID: `1.3.6.1.4.1.42.2.27.8.5.1` with no value.
- **Named constant**. Use the named constants, `pwpolicy` or `passwordpolicy` with the `--control` or `-J` option instead of the Password Policy Control OID. For example, use `-J pwpolicy` or `-J passwordpolicy` with `ldapsearch`.

- **Option**. Use the `--usePasswordPolicyControl` option.

Note – The `-J` or `--control` option is used to specify which controls to use in a *search* request. The `--usePasswordPolicyControl` option is used for *bind* requests.

- **Run the `ldapsearch` command with the `--usePasswordPolicyControl` option.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com -s base --usePasswordPolicyControl "(objectclass=*)"
```

▼ To Search Using the Persistent Search Control

The Persistent Search Control allows a client to receive notification when entries in the directory are changed by an add, delete, or modify operation. When a change occurs, the server sends the updated entry to the client if the entry matches the search criteria that was used by the Entry Change Notification Control.

The `ldapsearch` command provides an option to run a persistent search (`-C`) that keeps the connection open and displays the entries that match the scope and filter whenever any changes (add, delete, modify, or all) occur. You can quit the search by pressing `Control-C`.

The value for this argument must be in the form:
`ps[[:''changetype''[[:''changesonly''[[:''entrychangecontrols'']]]`

The elements of this value include the following:

- `ps` — Required operator.
- `changetype` — Indicates the types of changes for which the client wants to receive notification. This element can be any of `add`, `del`, `mod`, or `moddn`, or it can be `all` to register for all change types. It can also be a comma-separated list to register for multiple specific change types. If this element is not provided, it defaults to including `all` change types.
- `changesonly` — If `True`, the client should only be notified of changes that occur to matching entries after the search is registered. If `False`, the server should also send all existing entries in the server that match the provided search criteria. If this element is not provided, then it will default to only returning entries for updates that have occurred since the search was registered.
- `entrychangecontrols` — If `True`, the server should include the Entry Change Notification Control in entries sent to the client as a result of changes. If `False`, the Entry Change Notification Control should not be included. If this element is not provided, then it will default to including the Entry Change Notification Controls.

1. **Run the `ldapsearch` command as follows:**

```
$ ldapsearch -h localhost -p 1389 -D "cn=admin,cn=Administrators,cn=config" \
  -w password -b dc=example,dc=com --persistentSearch ps:add:true:true \
  "(objectclass=*)"
```

Note – When you use this command, the server waits for any changes made using `add`, `delete`, `modify` or `all` to return values.

2 **Open another terminal window , and use `ldapmodify` to add a new entry.**

```
$ ldapmodify -h localhost -p 1389 -b dc=example,dc=com \
  --defaultAdd --filename new_add.ldif
Processing ADD request for uid=Marcia Garza,ou=People,dc=example,dc=com
ADD operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

3 **The original terminal window shows the change.**

To end the session, press Control-Z (Unix/Linux) or Control-C (Windows).

```
# Persistent search change type:  add
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: top
givenName: Marcia
uid: mgarza
uid: Marcia Garza
cn: Marcia Garza
sn: Garza
userpassword: {SSHA}SNfL1RUm5uvTnLK+G0K3oz+Peb1i5/+YsylfBg==
roomnumber: 5484
l: Santa Clara
ou: Accounting
ou: People
mail: mgarza@example.com
```

4 **To terminate the session, press Control-D (Unix/Linux) or Control-C (Windows), and then type `Y` to quit.**

```
Terminate batch job (Y/N)?
```

▼ To Search Using the Proxied Authorization Control

The Proxied Authorization Control allows a client to impersonate another entry for a specific operation. This control can be useful in trusted applications that need to perform on behalf of many different users, so that the application does not need to re-authenticate for each operation.

- **Run the `ldapsearch` command.**

  Here, `clientApp` must have the appropriate ACI permissions within the subtree to use the Proxied Authorization Control. If not granted, LDAP error 50 `insufficient access rights` will be returned to the client.

```
$ ldapsearch -h localhost -p 1389 \
  -D "uid=clientApp,ou=Applications,dc=example,dc=com" -w password \
  -s sub -b dc=example,dc=com \
  --proxyAs "dn:uid=acctgAdmin,ou=Administrators,ou=People,dc=example,dc=com" \
  "(uid=kvaughan)" mail
```

To Search Using the Server-Side Sort Control

The Server-Side Sort Control allows the client to request that the server sort the search results before sending them to the client. This is convenient when the server has indexes that can satisfy the sort order requested by the client faster than the client can.

You can sort the number of entries returned by using the `--sortorder` option. If you do not specify + (a plus sign) for ascending or - (a minus sign) for descending, then the default option is to sort in ascending order.

1. **Use the `ldapsearch` command to search all entries and to display the results in ascending order.**

   Use the `--sortorder` option sorted on the attributes `sn` and `givenName`.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --s sub -b dc=example,dc=com --sortorder sn,givenName "(objectclass)"
dn: uid=dakers,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
...<search results>...
```

2. **Use the `ldapsearch` command to search all entries and display the results in descending order.**

   Use the `--sortorder` option sorted on the attribute `sn`.

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -s sub -b dc=example,dc=com --sortorder -sn "(objectclass)"
dn: uid=pworrell,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
...<search results>...
```

To Search Using the Simple Paged Results Control

The Simple Paged Results Control allows a search operation to return only a subset of the results at a time. It can be used to iterate through the search results a page at a time. It is similar to the Virtual List View Control with the exception that it does not require the results to be sorted and can only be used to iterate sequentially through the search results.

- **Use the `ldapsearch` command with the `--simplePageSize` option.**

  The following command also uses the `--countEntries` option to mark each page.

```
$ ldapsearch --hostname localhost --port 1389 \
 --bindDN "cn=Directory Manager" --bindPassword password \
 --searchScope sub --baseDN dc=example,dc=com \
 --simplePageSize 2 --countEntries "(objectclass=*)"

dn: ou=Groups,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: Groups

dn: ou=People,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: People

# Total number of matching entries: 2

dn: ou=Special Users,dc=example,dc=com
objectClass: organizationalUnit
objectClass: top
description: Special Administrative Accounts
ou: Special Users

dn: ou=Company Servers,dc=example,dc=com
objectClass: organizationalUnit
objectClass: top
description: Standard branch for Company Server registration
ou: Company Servers

# Total number of matching entries: 2

dn: ou=Contractors,dc=example,dc=com
objectClass: organizationalUnit
objectClass: top
ou: Contractors
ou: Product Testing
ou: Product Development
ou: Accounting

# Total number of matching entries: 1
```

Searching Using the Virtual List View Control

The Virtual List View Control allows a client to request that the server send search results in small, manageable chunks within a specific range of entries. It also allows a client to move forward and backward through the results of a search operation if configured with a GUI browser or application, or jump directly to a particular entry.

Note – The Virtual List View Control requires that the returned entries be sorted.

Together with the `--virtualListView` option or its short form `-G`, specify the following arguments:

- **before**. Specify the number of entries before the target to include in the results.
- **after**. Specify the number of entries after the target to include in the results.
- **index**. Specify the offset of the target entry within the result set. An index of 1 always means the first entry. If `index` and `content_count` are equal, the last entry is selected.
- **count**. Specify the expected size of the result set.
  - **count=0**. The target entry is the entry at the specified *index* position, starting from 1 and relative to the entire list of sorted results. Use this argument if the client does not know the size of the result set.
  - **count=1**. The target entry is the first entry in the list of sorted results.
  - **count>1**. The target entry is the first entry in the portion of the list represented by the fraction *index/count*. To target the last result in the list, use an *index* argument greater than the *count* argument. Client applications can use interfaces that allow users to move around a long list by using a scroll bar. For example, for an index of 33 and a count of 100, the application can jump 33 percent of the way into the list.

The arguments (0:4:1:0) indicate that you want to show 0 entries before and 4 entries after the target entry at index 1. If the client does not know the size of the set, the count is 0.

▼ To Search Using the Virtual List View Control

The sort order option (`-S`) must be used with the Virtual List View control. This example uses the Virtual List View Control options to specify the following:

- **Before=0*.** Specifies that 0 entries before the target should be displayed.
- **After=2*.** Specifies that 2 entries after the target should be displayed.
- **Index=1*.** Specifies that the offset of the target entry within the result set be returned.
- **Count=0*.** Specifies that target entry at the index position be returned, which is the first entry.

Thus, the server returns the first entry plus two entries after the target sorted in ascending order by the `givenName` attribute.

- **Use the `ldapsearch` command with the `--virtualListView` option.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w bindPassword \
  -b dc=example,dc=com --searchScope sub --sortOrder givenName \
  --virtualListView "0:2:1:0" "(objectclass=*)"

dn: uid=awhite,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
givenName: Alan
uid: awhite
cn: Alan White
sn: White
...

dn: uid=aworrell,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
givenName: Alan
uid: aworrell
cn: Alan Worrell
sn: Worrell
...

dn: uid=alutz,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
givenName: Alexander
uid: alutz
cn: Alexander Lutz
sn: Lutz
...

# VLV Target Offset: 1
# VLV Content Count: 172
```

▼ To Search Using Virtual List View With a Specific Target

The sort order (`-S`) option must also be used with Virtual List View. The example command uses the Virtual List View Control options to specify the following:

- **Before=0*.** Specifies that 0 entries before the target should be displayed.
- **After=4*.** Specifies that 4 entries after the target should be displayed.
- **Index=jensen*.** Specifies that the string `jensen` within the result set be returned.
- **Count=not specified*.** Use the default `count=0`, which is the first entry.

Thus, the server returns the first `sn` attribute that matches `jensen` plus four `sn` attributes after the target sorted in ascending order by the `sn` attribute.

- **Use the `ldapsearch` command with the `--virtualListView` option.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com --searchScope sub --sortOrder sn \
  --virtualListView "0:4:jensen" "(objectclass=*)" sn

dn: uid=kjensen,ou=People,dc=example,dc=com
sn: Jensen

dn: uid=bjensen,ou=People,dc=example,dc=com
sn: Jensen

dn: uid=gjensen,ou=People,dc=example,dc=com
sn: Jensen

dn: uid=jjensen,ou=People,dc=example,dc=com
sn: Jensen

dn: uid=ajensen,ou=People,dc=example,dc=com
sn: Jensen

# VLV Target Offset: 56
# VLV Content Count: 172
```

To Search Using Virtual List View With a Known Total

The sort order (`-s`) option must also be used with Virtual List View. The example command uses the Virtual List View Control options to specify the following:

- **Before=0*.** Specifies that 0 entries before the target should be displayed.
- **After=2*.** Specifies that 2 entries after the target should be displayed.
- **Index=57*.** Specifies that the index of 57 within the result set should be returned. This is roughly one-third of the list.
- **Count=172*.** Use the total count.

Thus, the server returns the first `sn` attribute that is one-third within the list, plus two `sn` attributes sorted in ascending order by the `sn` attribute.

- **Use the `ldapsearch` command with the `--virtualListView` option.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com -s sub --sortOrder sn \
  --virtualListView "0:2:57:172" "(objectclass=*)" sn

dn: uid=bjensen,ou=People,dc=example,dc=com
sn: Jensen

dn: uid=gjensen,ou=People,dc=example,dc=com
sn: Jensen

dn: uid=jjensen,ou=People,dc=example,dc=com
sn: Jensen

# VLV Target Offset: 57
# VLV Content Count: 172
```

Searching in Verbose Mode and With a Properties File

This section describes how to search in verbose mode and how to search by using a properties file.

▼ To Search in Verbose Mode

Verbose mode displays the processing information that is transmitted between client and server. This mode is convenient for debugging purposes.

- **Use the `ldapsearch` command as follows:**

```
$ ldapsearch -h localhost -port 1389 -D "cn=Directory Manager" -w password \
   -b dc=example,dc=com -s base --verbose "(objectclass=*)"
LDAP: C>S 01:43:46.140 (0ms) LDAPMessage(msgID=1, protocolOp=BindRequest
  (version =3, dn=cn=Directory Manager, password=opends))
ASN1: C>S 01:43:46.140 (0ms) ASN.1 Sequence
  BER Type: 30
  Decoded Values:
  ASN1Integer(type=02, value=1)
  ASN1Sequence(type=60, values={ ASN1Integer(type=02, value=3),
    cn=Directory Manager, opends })
  Value:
  02 01 01 60 23 02 01 03   04 14 63 6E 3D 64 69 72     '#  cn=directory
  65 63 74 6F 72 79 20 6D   61 6E 61 67 65 72 80 08          manager
  70 61 73 73 77 6F 72 64                                    password
...
```

▼ To Search Using a Properties File

The directory server supports the use of a properties file that holds default argument values used with the `ldapsearch` command. The properties file is convenient when working in different configuration environments, especially in scripted or embedded applications. For more information, see "Using a Properties File With Directory Server Commands" in *Command Line Usage for Sun Virtual Directory Proxy 1.0*.

1 Create a properties file in any text editor, with the following content:

```
hostname=localhost
port=1389
bindDN=cn=Directory Manager
bindPassword=password
baseDN=dc=example,dc=com
searchScope=sub
sortOrder=givenName
virtualListView=0:2:1:0
```

2 Save the file as `tools.properties`.

3 Use the `ldapsearch` with the `--propertiesFilePath` option.

```
$ ldapsearch --propertiesFilePath tools.properties "(objectclass=*)"
```

Searching Internationalized Entries

Sun OpenDS Standard Edition supports collation rules that match entries and can be used with the server-side sorting control to sort search results. The collation rule is specified in the search filter as a matching rule, delimited by colons, as shown here:

locale.*matchingRule*

where:

- *locale* is specified in one of the following ways
  - Locale OID
  - Locale character suffix (such as `ar`, `en`, or `fr-CA`).

    See "Supported Collation Rules" on page 175 at the end of this section for a list of supported locales, their OIDs, and tags.
- *matchingRule* can specified as either a numeric suffix or a character suffix appended to the *locale*, as listed in Table 1.

Note – If the locale is specified by its OID, then the matching rule must be specified by its numeric suffix. In this case, the matching rule cannot be specified by the character suffix.

TABLE 1 Matching Rule Suffixes

Matching Rule	Numeric Suffix	Character Suffix
Less than	.1	.lt
Less than or equal to	.2	.lte
Equality	.3	.eq (default)
Greater than or equal to	.4	.gte
Greater than	.5	.gt
Substring	.6	.sub

`Equality` is the default matching rule. That is, when no matching rule suffix is specified, the collation rule uses `equality` matching rule. The two following examples are equivalent and specify the English collation rule and the equality matching rule, but the second example specifies the equality matching rule explicitly with the `.eq` suffix:

```
"cn:en:=sanchez"
"cn:en.eq:=sanchez"
```

The next example shows the same search filter, but specified using the locale's character suffix and the matching rule's numeric code:

```
"cn:en.3:=sanchez"
```

The following example shows the same search filter specified using the locale OID and the matching rule numeric suffix:

```
"cn:1.3.6.1.4.1.42.2.27.9.4.34.1.3:=sanchez"
```

The following examples specify the same search filter but with a Spanish collation rule.

```
"cn:es.eq:=sanchez"
"cn:1.3.6.1.4.1.42.2.27.9.4.49.1.3:=sanchez"
"cn:es.3:=sanchez"
```

The following examples specify a similar search filter that uses a greater-than matching rule with the Spanish collation rule.

```
"cn:es.gt:=sanchez"
"cn:1.3.6.1.4.1.42.2.27.9.4.49.1.5:=sanchez"
"cn:es.5:=sanchez"
```

Examples

EXAMPLE 6 Equality Search

The following search uses a filter with the en (en-US) locale OID to perform an equality search to return any entry with a cn value of sanchez:

```
$ ldapsearch -D "cn=directory manager" -w password -b "o=test" \
  "cn:1.3.6.1.4.1.42.2.27.9.4.34.1:=sanchez"
```

The following filters return the same results:

- "cn:en:=sanchez"
- "cn:en.3:=sanchez"
- "cn:en.eq:=sanchez"
- "cn:1.3.6.1.4.1.42.2.27.9.4.34.1.3:=sanchez"

EXAMPLE 7 Less-Than Search

The following search uses a filter with the es (es-ES) locale and performs a less-than search and returns the entry with a departmentnumbervalue of abc119:

```
$ ldapsearch -D "cn=directory manager" -w password -b "o=test" \
  "departmentnumber:1.3.6.1.4.1.42.2.27.9.4.49.1.1:=abc120"
```

The following filters return the same results:

- "departmentnumber:es.1:=abc120"
- "departmentnumber:es.lt:=abc120"

EXAMPLE 8 Less-Than-or-Equal-To Search

The following search uses a filter with the es (es-ES) locale and performs a less-than-or-equal-to search that returns the entry with a departmentnumbervalue of abc119:

```
$ ldapsearch -D "cn=directory manager" -w password -b "o=test" \
  "departmentnumber:1.3.6.1.4.1.42.2.27.9.4.49.1.2:=abc119"
```

The following filters return the same results:

- "departmentnumber:es.2:=abc119"
- "departmentnumber:es.lte:=abc119"

EXAMPLE 9 Greater-Than-or-Equal-To Search

The following search uses a filter with the `fr` (`fr-FR`) locale and performs a greater-than-or-equal-To search that returns an entry with a `departmentnumber` value of `abc119`

```
$ ldapsearch -D "cn=directory manager" -w password -b "o=test" \
  "departmentnumber:fr.4:=abc119"
```

The following filters return the same results:

- `"departmentnumber:1.3.6.1.4.1.42.2.27.9.4.76.1.4:=abc119"`
- `"departmentnumber:fr.gte:=abc119"`

EXAMPLE 10 Greater-Than Search

The following search uses a filter with the `fr` (`fr-FR`) locale and performs a greater-than search:

```
$ ldapsearch -D "cn=directory manager" -w password -b "o=test" \
  "departmentnumber:fr.5:=abc119"
```

The above search should *not* return an entry with a `departmentnumber` value of `abc119`.

The following filters return the same results:

- `"departmentnumber:1.3.6.1.4.1.42.2.27.9.4.76.1.5:=abc119"`
- `"departmentnumber:fr.gt:=abc119"`

EXAMPLE 11 Substring Search

The following search uses a filter with the `en` (`en-US`) locale and performs a substring search that returns an entry with an `sn` value of "Quebec":

```
$ ldapsearch -D "cn=directory manager" -w password -b "o=test" \
  "sn:en.6:=*u*bec"
```

The following filters return the same results:

- `"sn:1.3.6.1.4.1.42.2.27.9.4.34.1.6:=*u*bec"`
- `"sn:en.sub:=*u*bec"`

Supported Collation Rules

The following table lists the internationalization locales supported by Sun OpenDS Standard Edition and the Sun JVM, alphabetized by character suffix.

TABLE 2 Supported Collation Rules

Locale	Character Suffix	OID
Arabic	ar	1.3.6.1.4.1.42.2.27.9.4.3.1
Arabic United Arab Emirates	ar-AE	1.3.6.1.4.1.42.2.27.9.4.4.1
Arabic Bahrain	ar-BH	1.3.6.1.4.1.42.2.27.9.4.5.1
Arabic Algeria	ar-DZ	1.3.6.1.4.1.42.2.27.9.4.6.1
Arabic Egypt	ar-EG	1.3.6.1.4.1.42.2.27.9.4.7.1
Arabic India	ar-IQ	1.3.6.1.4.1.42.2.27.9.4.9.1
Arabic Jordanar	ar-JO	1.3.6.1.4.1.42.2.27.9.4.10.1
Arabic Kuwait	ar-KW	1.3.6.1.4.1.42.2.27.9.4.11.1
Arabic Lebanon	ar-LB	1.3.6.1.4.1.42.2.27.9.4.12.1
Arabic Lybia	ar-LY	1.3.6.1.4.1.42.2.27.9.4.13.1
Arabic Morocco	ar-MA	1.3.6.1.4.1.42.2.27.9.4.14.1
Arabic Oman	ar-OM	1.3.6.1.4.1.42.2.27.9.4.15.1
Arabic Qatar	ar-QA	1.3.6.1.4.1.42.2.27.9.4.16.1
Arabic Saudi Arabia	ar-SA	1.3.6.1.4.1.42.2.27.9.4.17.1
Arabic Sudan	ar-SD	1.3.6.1.4.1.42.2.27.9.4.18.1
Arabic Syria	ar-SY	1.3.6.1.4.1.42.2.27.9.4.19.1
Arabic Tunisia	ar-TN	1.3.6.1.4.1.42.2.27.9.4.20.1
Arabic Yemen	ar-YE	1.3.6.1.4.1.42.2.27.9.4.21.1
Byelorussian	be	1.3.6.1.4.1.42.2.27.9.4.22.1
Bulgaria	bg	1.3.6.1.4.1.42.2.27.9.4.23.1
Catalan	ca	1.3.6.1.4.1.42.2.27.9.4.25.1
Czech	cs	1.3.6.1.4.1.42.2.27.9.4.26.1
Danish	da	1.3.6.1.4.1.42.2.27.9.4.27.1
German	de	1.3.6.1.4.1.142.2.27.9.4.28.1
German Germany	de-DE	1.3.6.1.4.1.142.2.27.9.4.28.1
German Austria	de-AT	1.3.6.1.4.1.42.2.27.9.4.29.1
German Swiss	de-CH	1.3.6.1.4.1.42.2.27.9.4.31.1

TABLE 2 Supported Collation Rules *(Continued)*

Locale	Character Suffix	OID
German Luxembourg	de-LU	1.3.6.1.4.1.42.2.27.9.4.32.1
Greek	el	1.3.6.1.4.1.42.2.27.9.4.33.1
English	en	1.3.6.1.4.1.42.2.27.9.4.34.1
English US	en-US	1.3.6.1.4.1.42.2.27.9.4.34.1
English Australia	en-AU	1.3.6.1.4.1.42.2.27.9.4.35.1
English Canada	en-CA	1.3.6.1.4.1.42.2.27.9.4.36.1
English Great Britain	en-GB	1.3.6.1.4.1.42.2.27.9.4.37.1
English Ireland	en-IE	1.3.6.1.4.1.42.2.27.9.4.39.1
English India	en-IN	1.3.6.1.4.1.42.2.27.9.4.40.1
English New Zealand	en-NZ	1.3.6.1.4.1.42.2.27.9.4.42.1
English South Africa	en-ZA	1.3.6.1.4.1.42.2.27.9.4.46.1
Spanish	es	1.3.6.1.4.1.42.2.27.9.4.49.1
Spanish Spain	es-ES	1.3.6.1.4.1.42.2.27.9.4.49.1
Spanish Argentina	es-AR	1.3.6.1.4.1.42.2.27.9.4.50.1
Spanish Bolivia	es-BO	1.3.6.1.4.1.42.2.27.9.4.51.1
Spanish Chile	es-CL	1.3.6.1.4.1.42.2.27.9.4.52.1
Spanish Colombia	es-CO	1.3.6.1.4.1.42.2.27.9.4.53.1
Spanish Costa Rica	es-CR	1.3.6.1.4.1.42.2.27.9.4.54.1
Spanish Dominican Republic	es-DO	1.3.6.1.4.1.42.2.27.9.4.55.1
Spanish Ecuador	es-EC	1.3.6.1.4.1.42.2.27.9.4.56.1
Spanish Guatemala	es-GT	1.3.6.1.4.1.42.2.27.9.4.57.1
Spanish Honduras	es-HN	1.3.6.1.4.1.42.2.27.9.4.58.1
Spanish Mexico	es-MX	1.3.6.1.4.1.42.2.27.9.4.59.1
Spanish Nicaragua	es-NI	1.3.6.1.4.1.42.2.27.9.4.60.1
Spanish Panama	es-PA	1.3.6.1.4.1.42.2.27.9.4.61.1
Spanish Peru	es-PE	1.3.6.1.4.1.42.2.27.9.4.62.1
Spanish Puerto Rico	es-PR	1.3.6.1.4.1.42.2.27.9.4.63.1
Spanish Paraguay	es-PY	1.3.6.1.4.1.42.2.27.9.4.64.1

TABLE 2 Supported Collation Rules *(Continued)*

Locale	Character Suffix	OID
Spanish Salvador	es-SV	1.3.6.1.4.1.42.2.27.9.4.65.1
Spanish Uraguay	es-UY	1.3.6.1.4.1.42.2.27.9.4.67.1
Spanish Venezuela	es-VE	1.3.6.1.4.1.42.2.27.9.4.68.1
Estonian	et	1.3.6.1.4.1.42.2.27.9.4.69.1
Finnish	fi	1.3.6.1.4.1.42.2.27.9.4.74.1
French	fr	1.3.6.1.4.1.42.2.27.9.4.76.1
French	fr-FR	1.3.6.1.4.1.42.2.27.9.4.76.1
French	fr-BE	1.3.6.1.4.1.42.2.27.9.4.77.1
French	fr-CA	1.3.6.1.4.1.42.2.27.9.4.78.1
French	fr-CH	1.3.6.1.4.1.42.2.27.9.4.79.1
French	fr-LU	1.3.6.1.4.1.42.2.27.9.4.80.1
Hebrew	he	1.3.6.1.4.1.42.2.27.9.4.85.1
Croatian	hr	1.3.6.1.4.1.42.2.27.9.4.87.1
Hungarian	hu	1.3.6.1.4.1.42.2.27.9.4.88.1
Icelandic	is	1.3.6.1.4.1.42.2.27.9.4.91.1
Italian	it	1.3.6.1.4.1.42.2.27.9.4.92.1
Italian-Swiss	it-CH	1.3.6.1.4.1.42.2.27.9.4.93.1
Japanese	ja	1.3.6.1.4.1.42.2.27.9.4.94.1
Korean	ko	1.3.6.1.4.1.42.2.27.9.4.97.1
Lithuanian	lt	1.3.6.1.4.1.42.2.27.9.4.100.1
Latvian	lv	1.3.6.1.4.1.42.2.27.9.4.101.1
Macedonian	mk	1.3.6.1.4.1.42.2.27.9.4.102.1
Dutch	nl	1.3.6.1.4.1.42.2.27.9.4.105.1
Dutch Netherlands	nl-NL	1.3.6.1.4.1.42.2.27.9.4.105.1
Dutch Belgium	nl-BE	1.3.6.1.4.1.42.2.27.9.4.106.1
Norwegian	no	1.3.6.1.4.1.42.2.27.9.4.107.1
Norwegian Norway	no-NO	1.3.6.1.4.1.42.2.27.9.4.107.1
Norwegian Nynorsk	no-NO-NY	1.3.6.1.4.1.42.2.27.9.4.108.1

TABLE 2 Supported Collation Rules *(Continued)*

Locale	Character Suffix	OID
Polish	pl	1.3.6.1.4.1.42.2.27.9.4.114.1
Portuguese	pt	1.3.6.1.4.1.42.2.27.9.4.115.1
Portuguese Portugal	pt-PT	1.3.6.1.4.1.42.2.27.9.4.115.1
Portugues Brazil	pt-BR	1.3.6.1.4.1.42.2.27.9.4.116.1
Romanian	ro	1.3.6.1.4.1.42.2.27.9.4.117.1
Russian	ru	1.3.6.1.4.1.42.2.27.9.4.118.1
Russian Russia	ru-RU	1.3.6.1.4.1.42.2.27.9.4.118.1
Slovak	sk	1.3.6.1.4.1.42.2.27.9.4.121.1
Slovenia	sl	1.3.6.1.4.1.42.2.27.9.4.122.1
Albanian	sq	1.3.6.1.4.1.42.2.27.9.4.127.1
Serbian	sr	1.3.6.1.4.1.42.2.27.9.4.128.1
Swedish	sv	1.3.6.1.4.1.42.2.27.9.4.129.1
Swedish Sweden	sv-SE	1.3.6.1.4.1.42.2.27.9.4.129.1
Thai	th	1.3.6.1.4.1.42.2.27.9.4.136.1
Turkish	tr	1.3.6.1.4.1.42.2.27.9.4.140.1
Ukrainian	uk	1.3.6.1.4.1.42.2.27.9.4.141.1
Vietnamese	vi	1.3.6.1.4.1.42.2.27.9.4.142.1
Chinese	zh	1.3.6.1.4.1.42.2.27.9.4.143.1
Chinese China	zh-CN	1.3.6.1.4.1.42.2.27.9.4.144.1
Chinese Hong Kong	zh-HK	1.3.6.1.4.1.42.2.27.9.4.145.1
Chinese Taiwan	zh-TW	1.3.6.1.4.1.42.2.27.9.4.148.1

Adding, Modifying, and Deleting Directory Data

The directory server provides a full set of LDAPv2- and LDAPv3-compliant client tools to manage directory entries. You can add, update, or remove entries by using the `ldapmodify` and `ldapdelete` utilities. The LDAP command-line utilities require LDAP Data Interchange Format (LDIF)-formatted input, entered through the command line or read from a file.

You can also modify directory data by using the control panel. For more information, see "Managing Directory Data With the Control Panel" on page 200.

Before you make modifications to directory data, make sure that you understand the following concepts:

- The privilege and access control mechanisms.

  For information about setting privileges, "Controlling Access To Data" on page 225.
- The structure of your directory server.
- The schema of your directory server.

Adding Directory Entries

You can add one or more entries to a directory server by using the `ldapmodify` command. `ldapmodify` opens a connection to the directory server, binds to it, and performs the modification to the database (in this case, an "add") as specified by the command-line options.

`ldapmodify` enables you to add entries in one of two ways:

- **Using the --defaultAdd option**. Use the `--defaultAdd` option to add new entries to the directory when data is entered on the command line. Press Ctrl-D (UNIX, Linux) or Ctrl-Z (Windows) when finished, or use an input file with your changes.
- **Using LDIF update statements**. LDIF update statements define how `ldapmodify` changes the directory entry. LDIF update statements contain the DN of the entry to be modified, *changetype* that defines how a specific entry is to be modified (add, delete, modify, modrdn), and a series of attributes and their changed values.

Note – Any newly added entry must conform to the directory's schema. If you add any entry that does not conform to the schema, the server responds with an Object Class Violation error. You can view the details of the error in the `errors` log.

▼ To Create a Root Entry

The root entry is the topmost entry in the directory and must contain the naming context, or root suffix. You can set up the root entry when you first install the directory server using the graphical user interface (GUI) or the command-line. If you install the directory without any data, create a root entry using the `ldapmodify` command with the `--defaultAdd` option.

1 Create the root entry using `ldapmodify`.

```
$ ldapmodify --hostname localhost --port 1389 --defaultAdd \
  --bindDN "cn=Directory Manager" --bindPassword password
dn: dc=example,dc=com
objectclass: domain
objectclass: top
dc: example
(Press Ctrl-D on Unix, Linux)
```

```
(Press Ctrl-Z on Windows), then press ENTER.

Processing ADD request for dc=example,dc=com
ADD operation successful for DN dc=example,dc=com
```

Note – The `--bindDN` and `--bindPassword` options specify the bind DN and password, respectively, of the user with permissions to add new entiries. You can provide the clear-text version of the password. The server encrypts this value and store only the encrypted one. Be sure to limit read permissions to protect clear passwords that appear in LDIF files. To avoid this security issue, use SSL or startTLS.

---

2 Verify the change by using the `ldapsearch` command.

```
$ ldapsearch --hostname localhost --port 1389 --baseDN "dc=example,dc=com" \
  --searchScope base --bindDN "cn=Directory Manager" --bindPassword password \
  "(objectclass=*)"
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
```

▼ To Add an Entry Using the `--defaultAdd` Option With `ldapmodify`

1 Create your directory entry in LDIF format.

Before you add an entry, ensure that the suffix to which you want to add the entry exists in your database (for example, `ou=People,dc=example,dc=com`).

For this example, create an input file called `new.ldif` with the following contents:

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
cn: Marcia Garza
sn: Garza
givenName: Marcia
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
ou: Accounting
ou: People
l: Santa Clara
uid: mgarza
mail: mgarza@example.com
roomnumber: 5484
userpassword: donuts
```

2 **Add the entry using `ldapmodify` with the `--defaultAdd` option.**

```
$ ldapmodify --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password --defaultAdd --filename /tmp/new.ldif
```

▼ To Add Entries Using an LDIF Update Statement With `ldapmodify`

1 **Create the entry in LDIF format with the `changetype:add` element.**

Make sure that there are no trailing spaces after `add`. If a space exists after `add`, the server base-64 encodes the value to represent the space, which can cause problems.

For this example, create an input LDIF file named `new.ldif`.

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: add
cn: Marcia Garza
sn: Garza
givenName: Marcia
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
ou: Accounting
ou: People
l: Santa Clara
uid: mgarza
mail: mgarza@example.com
roomnumber: 5484
userpassword: donuts
```

2 **Add the entry using `ldapmodify`.**

Do not include the `-a` option as the `changetype` attribute specifies the action.

```
$ ldapmodify --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password --filename /tmp/new.ldif

Processing ADD request for uid=Marcia Garza,ou=People,dc=example,dc=com
ADD operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

Adding Attributes

The LDIF `changetype:add` statement adds an entry to the directory. To add attributes to an entry, use the `changetype:modify` statement, as shown in the following examples. You can combine multiple commands within a file by separating each command with a dash ("-").

To Add an Attribute to an Entry

1 **Create the entry in LDIF format with the `changetype:modify` element.**

Use the `modify` change type, because you are modifying an existing entry with the addition of a new attribute. Make sure that there are no trailing spaces after `modify`. After the `changetype`, specify `add:` *newAttributeName* and, on the following line, the value of the new attribute.

For this example, create an input LDIF file called `add_attribute.ldif`, as follows:

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
add: telephonenumber
telephonenumber: +1 408 555 8283
```

Note – To add multiple attributes, separate the attributes with a dash (-), for example:

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
add: telephonenumber
telephonenumber: +1 408 555 8283
-
add: building
building: sc09
```

---

2 **Add the attribute by using `ldapmodify`.**

```
$ ldapmodify --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password --filename /tmp/add_attribute.ldif

Processing MODIFY request for uid=Marcia Garza,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

To Add an ACI Attribute

You can use `ldapmodify` to add access control instructions (ACIs) to manage access rights for a user's account. For more information, see "Controlling Access To Data" on page 225 and "ACI Syntax" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

The following example allows a user to modify her own directory attributes.

1 **Create the LDIF file containing the ACI.**

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (target="ldap:///uid=Marcia Garza,ou=People,dc=example,dc=com")
 (targetattr="*")(version 3.0; acl "mgarza rights"; allow (write)
 userdn="ldap:///self";)
```

2 **Add the attribute by using `ldapmodify`.**

```
$ ldapmodify --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password --filename /tmp/add_aci.ldif

Processing MODIFY request for uid=Marcia Garza,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

▼ To Add an International Attribute

The directory server represents international locales using a language tag in the form *attribute;language-subtype*. For example, `homePostalAddress;lang-jp:address` specifies the postal address with the locale in Japan (`subtype=jp`).

- **Use `ldapmodify` to add the attribute.**

  Affix the language subtype, `lang-`*cc*, where *cc* is the country code.

```
$ ldapmodify --hostname localhost --port 1389 --bindDN "cn=Directory Manager" \
  --bindPassword password
dn: uid=jarrow,ou=People,dc=example,dc=com
changetype: modify
add: homePostalAddress;lang-jp
homePostalAddress;lang-jp: 1-8-15 Azuchimachi, Chuo-ku
(Press Ctrl-D on Unix, Linux)
(Press Ctrl-Z on Windows), then press ENTER.
```

Note – If the attribute value contains non-ASCII characters, they must be UTF-8 encoded.

Modifying Directory Entries

Use the LDIF update statement `changetype:modify` to make changes to existing directory data. The following procedures provide examples of modifying directory entries.

For more information, see "ldapmodify" in *Sun OpenDS Standard Edition 2.0 Command-Line Usage Guide*.

▼ To Modify an Attribute Value

- **Use `ldapmodify` to change the entry, using the `changetype:modify` and `replace` elements.**

  Ensure that there are no trailing spaces after `modify`.

  This example modifies a user's existing telephone number.

```
$ ldapmodify -h localhost -p 1389 D "cn=Directory Manager" -w password \
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
```

```
replace: telephonenumber
telephonenumber: +1 408 555 8288

Processing MODIFY request for uid=Marcia Garza,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

Note – To modify multiple attributes, separate the attributes with a dash (-), for example:

```
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
replace: telephonenumber
telephonenumber: +1 408 555 6465
-
add: facsimiletelephonenumber
facsimiletelephonenumber: +1 408 222 4444
-
replace: l
l: Sunnyvale
```

▼ To Modify an Attribute With Before and After Snapshots

The `ldapmodify` command provides the options, `--preReadAttribute` and `--postReadAttribute`, that return the modified attribute value with a *before* and *after* snapshot, respectively.

- **Use `ldapmodify` with the `--preReadAttribute` and `--postReadAttribute` options.**

  This example modifies a user's existing telephone number.

```
$ ldapmodify -h localhost -p 1389 D "cn=Directory Manager" -w password \
  --preReadAttributes telephoneNumber --postReadAttributes telephoneNumber
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
replace: telephonenumber
telephonenumber: +1 408 555 8288

Processing MODIFY request for uid=Marcia Garza,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com

Target entry before the operation:
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
telephonenumber: +1 408 555 4283

Target entry after the operation:
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
telephonenumber: +1 408 555 8288
```

To Delete an Attribute

This example deletes the location (l) attribute from an entry.

- **Use the `ldapmodify` to delete the attribute.**

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: modify
delete: l
(Press CTRL-D for Unix, Linux) (Press CTRL-Z for Windows), then press ENTER.

Processing MODIFY request for uid=Marcia Garza,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

Note – Type control-D (UNIX, Linux) or control-Z (Windows) to complete the input.

To Change an RDN

The distinguished name (DN) of an entry uniquely identifies and describes that entry. A distinguished name consists 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.

The relative distinguished name (RDN) is the leftmost element in an entry DN. For example, the RDN for `uid=Marcia Garza,ou=People,dc=example,dc=com` is `uid=Marcia Garza`. To change an RDN, use the `changetype:moddn` LDIF update statement.

You can specify if the old RDN should be retained in the directory by using the `deleteoldrdn` attribute. A `deleteoldrdn` value of `0` indicates that the existing RDN should be retained in the directory. A value of `1` indicates that the existing RDN should be replaced by the new RDN value.

Note – You cannot rename an RDN if it has any children, due to the possible orphaning of the subtree elements. This is a violation of the LDAP protocol.

1. **Use the `ldapmodify` command to rename the entry.**

   In this example, an employee `Marcia Garza` wants to change to her married name, `Marcia Peters`.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password
dn: uid=Marcia Garza,ou=Marketing,dc=example,dc=com
changetype: moddn
newrdn: uid=Marcia Peters
deleteoldrdn: 1
Processing MODIFY DN request for uid=Marcia Garza,ou=People,dc=example,dc=com
MODIFY DN operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

2 Change any other attributes as necessary.

In this example, certain attributes might still list the user's previous name.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password
dn: uid=Marcia Peters,ou=People,dc=example,dc=com
changetype: modify
replace: sn
sn: Peters
-
replace: cn
cn: Marcia Peters
-
replace: uid
uid: mpeters
uid: Marcia Peters
-
replace: mail
mail: mpeters@example.com
(Press Ctrl-D on Unix, Linux)
(Press Ctrl-Z on Windows), then press ENTER.

Processing MODIFY request for uid=Marcia Peters,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=Marcia Peters,ou=People,dc=example,dc=com
```

▼ To Move an Entry

If you are moving an entry from one parent to another, extend the access control instruction (ACI) rights on the parent entries. On the current parent entry of the entry to be moved, ensure that the ACI allows the export operations by using the syntax `allow(export ...)`. On the future parent entry of the entry to be moved, ensure that the ACI allows the import operations by using the syntax `allow(import...)`.

In this example, move `uid=sgarza` from the `ou=Contractors,dc=example,dc=com` suffix to the `ou=People,dc=example,dc=com` subtree.

1 Use `ldapmodify` with the `moddn` changetype to move the entry.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password
dn: uid=sgarza,ou=Contractors,dc=example,dc=com
changetype: moddn
newrdn: uid=sgarza
deleteoldrdn: 0
newsuperior: ou=People,dc=example,dc=com
--filename move_entry.ldif
Processing MODIFY DN request for uid=sgarza,ou=Contractors,dc=example,dc=com
MODIFY DN operation successful for DN uid=sgarza,ou=Contractors,dc=example,dc=com
```

2 **Change any other attribute values, as required.**

The following example provides before and after snapshot changes for the ou attribute.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --preReadAttributes ou --postReadAttributes ou
dn: uid=sgarza,ou=People,dc=example,dc=com
changetype: modify
replace: ou
ou: People
ou: Product Testing
(Press Ctrl-D on Unix, Linux)
(Press Ctrl-Z on Windows), then press ENTER.

Processing MODIFY request for uid=sgarza,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=sgarza,ou=People,dc=example,dc=com

Target entry before the operation:
dn: uid=sgarza,ou=People,dc=example,dc=com
ou: Contractors
ou: Product Testing

Target entry after the operation:
dn: uid=sgarza,ou=People,dc=example,dc=com
ou: People
ou: Product Testing
```

Deleting Directory Entries

You can use `ldapmodify` and `ldapdelete` to remove entries from the directory. The `ldapmodify` command removes entries and attributes by using the LDIF update statements `changetype:delete` and `changetype:modify` with the `delete` attribute, respectively. The `ldapdelete` tool removes only entries.

Note – You cannot delete an entry that has children entries. If you want to delete an entry that has children, first delete all the children entries below the targeted entry, then delete the entry.

For more information, see "ldapdelete" in *Sun OpenDS Standard Edition 2.0 Command-Line Usage Guide*.

▼ To Delete an Entry With `ldapmodify`

- **Use the `ldapmodify` command with the `changetype:delete` statement.**

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password
dn: uid=Marcia Garza,ou=People,dc=example,dc=com
changetype: delete
```

```
(Press CTRL-D for Unix)
(Press CTRL-Z for Windows), then press ENTER.

Processing DELETE request for uid=Marcia Garza,ou=People,dc=example,dc=com
DELETE operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
The number of entries deleted was 1
```

▼ To Delete an Entry With `ldapdelete`

- **Use the `ldapdelete` command and specify the entry that you want to delete.**

```
$ ldapdelete -h localhost -p 1389 -D "cn=Directory Manager" -w password
"uid=mgarza,ou=People,dc=example,dc=com"

Processing DELETE request for uid=Marcia Garza,ou=People,dc=example,dc=com
DELETE operation successful for DN uid=Marcia Garza,ou=People,dc=example,dc=com
```

▼ To Delete Multiple Entries by Using a DN File

1. **Create a file that contains a list of DNs to be deleted.**

   In this example, the file is named `delete.ldif`. The file must list each DN on a separate line, for example:

```
uid=mgarza,ou=People,dc=example,dc=com
uid=wsmith,ou=People,dc=example,dc=com
uid=jarrow,ou=People,dc=example,dc=com
uid=mbean,ou=People,dc=example,dc=com
```

2. **Delete the entries by passing the file as an argument to the `ldapdelete` command.**

```
$ ldapdelete -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --continueOnError --filename delete.ldif

Processing DELETE request for uid=mgarza,ou=People,dc=example,dc=com
DELETE operation successful for DN uid=mgarza,ou=People,dc=example,dc=com
Processing DELETE request for uid=wsmith,ou=People,dc=example,dc=com
DELETE operation successful for DN uid=wsmith,ou=People,dc=example,dc=com
Processing DELETE request for uid=jarrow,ou=People,dc=example,dc=com
DELETE operation successful for DN uid=jarrow,ou=People,dc=example,dc=com
Processing DELETE request for uid=mbean,ou=People,dc=example,dc=com
DELETE operation successful for DN uid=mbean,ou=People,dc=example,dc=com
```

Note – The `--continueOnError` option specifies that if an error occurs, the command continues to the next search item.

Indexing Directory Data

This section describes how to index attributes using the `dsconfig` command-line tool. Indexes are configured per server and index configuration is not replicated.

You can use `dsconfig` to create local database indexes and Virtual List View (VLV) indexes. A local database index is used to find entries that match search criteria. A VLV index is used to process searches efficiently with VLV controls.

Unindexed searches are denied by default, unless the user has the `unindexed-search` privilege. For more information, see "To Change a Root User's Privileges" on page 270.

You can determine whether a search is indexed in two ways:

- Try to perform the search anonymously. (The server rejects unindexed anonymous searches by default.)
- Use the `debugsearchindex` operational attribute. This attribute provides the indexes used in the search, the number of candidate entries from each index, and the final indexed status. Include the `debugsearchindex` attribute in your `ldapsearch` command, as follows:

```
$ ldapsearch -h localhost -p 1389 -b "dc=example,dc=com" "(objectClass=*)" debugsearchindex
```

Configuring Indexes on the Local DB Back End

The Local DB back end supports the following index types:

- `approximate` — Improves the efficiency of searches using approximate search filters.
- `equality` - Improves the efficiency of searches using equality search filters.
- `ordering` - Improves the efficiency of searches using "greater than or equal to" or "less than or equal to" search filters. In the future, this index type might also be used for server-side sorting.
- `presence` - Improves the efficiency of searches using presence search filters.
- `substring` - Improves the efficiency of searches using substring search filters.

The directory server does not currently support indexing for extensible matching operations.

When you create a new local DB back end with `dsconfig`, the following default indexes are created automatically:

- `aci` (presence index)
- `ds-sync-hist` (ordering index)
- `entryuuid` (equality index)
- `objectclass` (equality index)

To Create a New Local DB Index

This procedure demonstrates the steps for creating a new local DB index.

Note – After you have created a new index, you must rebuild the indexes using the `rebuild-index` utility. The directory server cannot use the new index until the indexes have been rebuilt. For more information, see "rebuild-index" in *Sun OpenDS Standard Edition 2.0 Command-Line Usage Guide*.

---

1 Create the new index.

```
$ dsconfig -D "cn=directory manager" -w password -n create-local-db-index \
  --backend-name backend --index-name attribute \
  --set index-type:index-type
```

2 Check that the index was created by listing the local DB indexes for that back end.

```
$ dsconfig -D "cn=directory manager" -w password -n list-local-db-indexes \
  --backend-name backend
```

3 Configure any specific index properties.

```
$ dsconfig -D "cn=directory manager" -w password -n set-local-db-index-prop \
  --backend-name backend --index-name attribute \
  --set property:value
```

4 List the index properties to verify your change.

```
$ dsconfig -D "cn=directory manager" -w password -n get-local-db-index-prop \
  --backend-name backend --index-name attribute
```

5 Stop the server and rebuild the index.

```
$ stop-ds
$ rebuild-index --baseDN baseDN --index attribute
```

6 Restart the server.

```
$ start-ds
```

Example 12 Creating a New Equality Index

This example creates a new equality index for the `employeeNumber` attribute, verifies the index properties, and sets the index entry limit to 5000.

```
$ dsconfig -D "cn=directory manager" -w password -n create-local-db-index \
  --backend-name userRoot --index-name employeeNumber \
  --set index-type:equality
$ dsconfig -D "cn=directory manager" -w password -n list-local-db-indexes \
  --backend-name userRoot
```

```
Local DB Index : Type : index-type
---------------:---------:----------------------------
...
employeeNumber : generic : equality
...
$ dsconfig -D "cn=directory manager" -w password -n get-local-db-index-prop \
  --backend-name userRoot --index-name employeeNumber
Property : Value(s)
--------------------:---------------
attribute : employeenumber
index-entry-limit : 4000
index-type : equality
$ dsconfig -D "cn=directory manager" -w password -n set-local-db-index-prop \
  --backend-name userRoot --index-name employeeNumber --set index-entry-limit:5000
$ dsconfig -D "cn=directory manager" -w password -n get-local-db-index-prop \
  --backend-name userRoot --index-name employeeNumber
Property : Value(s)
--------------------:---------------
attribute : employeenumber
index-entry-limit : 5000
index-type : equality
$ stop-ds
$ rebuild-index -b "dc=example,dc=com" --index employeeNumber
$ start-ds
```

Example 13 Adding a Substring Index

This example adds a substring index to the index created in the previous example. Note that you need to specify the `index-type` property twice to avoid overwriting the equality index created previously.

```
$ dsconfig -D "cn=directory manager" -w password -n set-local-db-index-prop \
  --backend-name userRoot --index-name employeeNumber --set index-type:equality \
  --set index-type:substring
$ dsconfig -D "cn=directory manager" -w password -n get-local-db-index-prop \
  --backend-name userRoot --index-name employeeNumber
Property : Value(s)
--------------------:---------------
attribute : employeenumber
index-entry-limit : 5000
index-type : equality, substring
$ stop-ds
$ rebuild-index -b "dc=example,dc=com" --index employeeNumber
$ start-ds
```

Configuring VLV Indexes

A VLV index applies to a particular search on a given base entry and its subtree. The sort order, scope of the index, base DN, and filter must be defined when you create the index.

Note – After you have created a new VLV index, you must rebuild the indexes using the `rebuild-index` command, appending `vlv.` in front of the index name. The directory server cannot use the new index until the indexes have been rebuilt. For more information, see "rebuild-index" in *Sun OpenDS Standard Edition 2.0 Command-Line Usage Guide*.

▼ To Create a New VLV Index

1 Use `dsconfig` to create a new VLV index as follows:

```
$ dsconfig -D "cn=directory manager" -w password -n create-local-db-vlv-index \
  --backend-name backend --index-name name --set sort-order:attributes \
  --set scope:scope --set base-dn:baseDN --set filter:filter
```

where:

- `index-name` specifies a unique index name, which cannot be altered after the VLV index is created.
- `sort-order` specifies the names of the attributes by which the entries are sorted and their order of precedence, from highest to lowest.
- `scope` specifies the LDAP scope of the query being indexed and can be one of `base-object`, `single-level`, `subordinate-subtree`, or `whole-subtree`.
- `base-dn` specifies the base DN used in the search query being indexed.
- `filter` specifies the LDAP filter used in the query being indexed and can be any valid LDAP filter.

2 Check that the index was created by listing the existing VLV indexes.

```
$ dsconfig -D "cn=directory manager" -w password -n list-local-db-vlv-indexes \
  --backend-name backend
```

3 Display the index properties to verify your change.

```
$ dsconfig -D "cn=directory manager" -w password -n get-local-db-vlv-index-prop \
  --backend-name backend --index-name name
```

4 Stop the server and rebuild the index.

```
$ stop-ds
$ rebuild-index -b baseDN --index vlv.name
```

5 **Restart the server.**

```
$ start-ds
```

Example 14 Creating a New VLV Index

The following example creates a new VLV index to sort entries first by surname and then by common name for queries `sn=*`. The example then stops the server, rebuilds the index, and restarts the server.

```
$ dsconfig -D "cn=directory manager" -w password -n create-local-db-vlv-index \
  --backend-name userRoot --index-name myVLVIndex --set sort-order:"sn cn" \
  --set scope:base-object --set base-dn:dc=example,dc=com --set filter:sn=*
$ stop-ds
$ rebuild-index -b "dc=example,dc=com" --index vlv.myVLVIndex
$ start-ds
```

Managing Indexes With the Control Panel

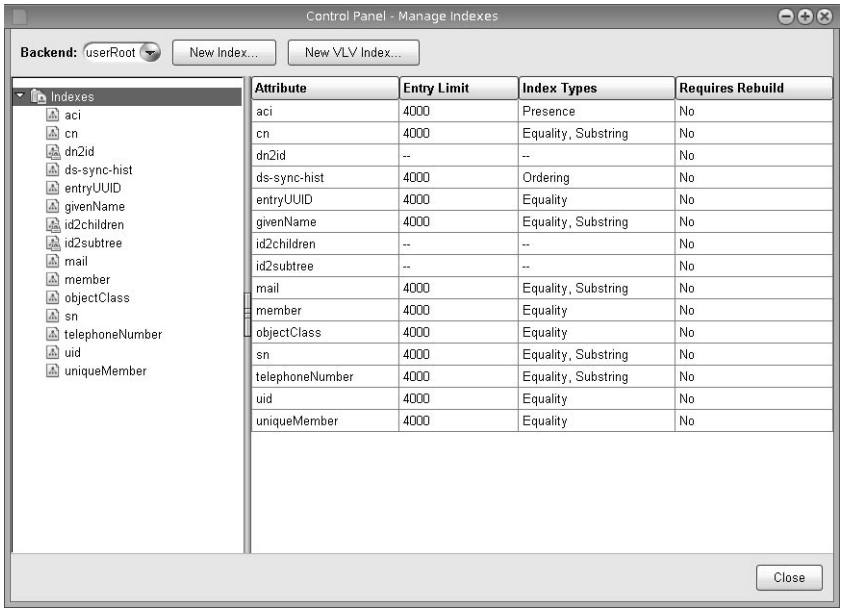
You can use the Control Panel to create, configure, and delete local database indexes and Virtual List View (VLV) indexes. A local database index is used to find entries matching search criteria. A VLV index is used to process searches efficiently with VLV controls. Unindexed searches are denied by default unless the user has the `unindexed-search` privilege. For more information, see "Root Users and the Privilege Subsystem" on page 266.

▼ To Display a List of Indexes

This procedure shows how to display a list of all the indexes configured for the directory server.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Click the Manage Indexes link under the Indexes menu on the left side of the Control Panel window. The Manage Indexes window appears, displaying all configured indexes in a list on its left side.**



To Add an Index

If your clients search on particular attribute frequently, you can add an index for that attribute to return results more efficiently.

1. **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2. **Click the Manage Indexes link under the Indexes menu on the left side of the Control Panel window. The Manage Indexes window appears, displaying all configured indexes in a list on its left side.**

3. **Click the New Index button,**

   The New Index window appears.

4. **Select the attribute that you want to index from the Attributes list.**

5. **If necessary, specify a number in the Entry Limit fields other than the default, `4000`.**

   The entry limit is the maximum number of entries allowed in the index before the directory server stops maintaining the index.

6. **Select one or more Index Type options.**

   - approximate-Used to help improve the efficiency of searches using approximate equality search filers.

- equality-Used to help improve the efficiency of searches using equality search filters.
- ordering-Used to help improve the efficiency of searches using "greater than or equal to" or "less than or equal to" search filters. In the future, this index type might also be used for server-side sorting.
- presence-Used to help improve the efficiency of searches using presence search filters.
- substring-Used to help improve the efficiency of searches using substring search filters.

7 Click the OK button.

The Index Rebuild Required window appears.

8 To rebuild the index now, click the Yes button

Otherwise, to rebuild the index later, click the No button. However, the index does not improve search efficiency until it is built.

If you select the Yes button, the New Index window displays the progress of the rebuild operation.

9 When the operation is complete, click the Close button to close the New Index window.

▼ To Add a VLV Index

1 Start the Control Panel, as described in "To Start the Control Panel" on page 28.

2 Click the Manage Indexes link under the Indexes menu on the left side of the Control Panel window. The Manage Indexes window appears, displaying all configured indexes in a list on its left side.

3 Click the New VLV Index button.

The New VLV Index window appears.

4 Specify the following information in the fields of the New VLV Index window:

Name	A unique name to identify the VLV index
Base DN	The base DN where you want to apply the new VLV index. If you select `Other`, enter the DN in the empty field to the right.
Search Scope	The search target where you want to apply the index
Filter	The LDAP filter used in the query being indexed, which can be any valid LDAP filter
Max Block Size	The number of entry IDs to store in a single sorted set before it must be split.

Sort Order	The names of the attributes by which the entries are sorted and their order of precedence, from highest to lowest. Select the name of each attribute from the list of available attributes, select a sort order for the attribute (either Ascending or Descending) and then click the Add button to add the attribute to the list below. Adjust the position of a selected attribute in the sort order list as needed by clicking the Move Up and Move Down buttons. Remove a selected attribute from the sort order list by clicking the Remove button.

5 **Click the OK button.**

The VLV index is configured, but it is not used for search operations until it has been rebuilt for the first time.

To Delete an Index

If an index is not used, you can delete it and release the resources that it uses.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Click the Manage Indexes link under theIndexes menu on the left side of the Control Panel window. The Manage Indexes window appears, displaying all configured indexes in a list on its left side.**

3 **In the list of indexes, select the index that you want to delete.**

4 **Click the Delete Index button.**

5 **A window appears, asking you to confirm that you want to delete the selected index.**

6 **Click the Yes button.**

7 **The Delete Index window appears, displaying the progress of the operation.**

To Verify Indexes

The contents of an index can become incorrectly organized. You can verify that any or all indexes are correctly prepared for effective searching. If the verification operation shows that an index is not correctly organized it, rebuild indexes as described in "To Rebuild Indexes" on page 198.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2. **Click the Verify Indexes link under the Indexes menu on the left side of the Control Panel window.**

   The Verify Indexes window appears, displaying all configured indexes in a list on its left side.

3. **If needed, select the base DN whose indexes you want to verify in the Base DN field.**

4. **Select one of the verification options:**

   - Verify Entry Contents are Properly Indexed
   - Verify All Index Key ID's are Clean and Refer to Existing Entries

5. **If you selected Verify Entry Contents are Properly Indexed, add the indexes that you want to verify to the Selected Indexes list by using one of these steps**

   - **Double-click the name of the index in the list of Available Indexes.**
   - **Select the name of the index in the list of Available Indexes and then click the Add button.**

6. **Click the OK button.**

   The Verify Indexes window displays the status of the operation

7. **When the operation completes, click the Close button.**

To Rebuild Indexes

Over time, as new entries are added, an index can become less efficiently organized. You can rebuild any or all indexes to organize them most efficiently.

1. **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2. **Click the Rebuild Indexes link under theIndexes menu on the left side of the Control Panel window.**

   The Rebuild Indexes window appears, displaying all configured indexes in a list on its left side.

3. **Add the indexes that you want to rebuild to the Selected Indexes list by using one of these steps:**

   - **Double-click the name of the index in the list of Available Indexes.**
   - **Select the name of the index in the list of Available Indexes and then click the Add button.**

4. **Click the OK button.**

   The Confirmation Required window appears, asking you to confirm that you want to rebuild the selected indexes.

5 **Click the Yes button.**

The Rebuild Indexes window displays the status of the operation.

While the indexes are being rebuilt, the back end is disabled and none of its suffixes can be accessed.

6 **When the operation completes, click the Close button.**

Reducing Stored Data Size

The directory server provides two mechanisms for reducing the size of stored data:

- **Compact encoding**. When compact encoding is enabled, the back end uses a compact form when encoding entries by compressing the attribute descriptions and object class sets. This property applies only to the entries themselves and does not impact the index data. Compact encoding is enabled by default but can be disabled if required. You might want to disable compact encoding where user-supplied capitalization is required because user-supplied capitalization is not preserved in compacted entries. The compaction does, however, provide a performance gain and is therefore beneficial in deployments where user-supplied capitalization can be sacrificed for performance, or is not required.
- **Entry compression**. Entry compression uses a deflater to compress the data before it is stored. When entry compression is enabled, the back end attempts to compress entries before storing them in the database. This property also applies only to the entries themselves and does not impact the index data. The effectiveness of entry compression is based on the type of data contained in the entry.

You can enable one or both of these mechanisms to reduce the size of the stored data. Because enabling these mechanisms affects future writes only, the database might contain a mixture of compressed and uncompressed records. Either type of record can be read regardless of the compression settings.

▼ To Enable or Disable Compact Encoding

Compact encoding is configured by setting the `compact-encoding` property of a back end. Changes to this setting will only take effect for writes that occur after the change is made. Existing data is not changed retroactively.

● **Disable compact encoding on the "userRoot" back end.**

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X -n \
  set-backend-prop --backend-name="userRoot" --set compact-encoding:false
```

▼ To Enable or Disable Entry Compression

Entry compression is configured by setting the `entries-compressed` property of a back end. Changes to this setting will only take effect for writes that occur after the change is made. Existing data is not changed retroactively.

- **Enable entry compression on the "userRoot" back end.**

```
$ dsconfig -h localhost -p 4444 -D "cn=directory manager" -w password -X -n \
  set-backend-prop --backend-name="userRoot" --set entries-compressed:true
```

Managing Directory Data With the Control Panel

Managing Entries With the Control Panel

The following procedures describe how to use the Control Panel to manage entries in the directory.

▼ To Display A List of All Directory Entries

This procedure shows how to use the Control Panel to display a list of all entries in the directory.

1. **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**
2. **Expand the Directory Data menu on the left side of the Control Panel window to display its options, if needed.**
3. **Click the Browse Entries link under Directory Data. A list of all entries in the directory is displayed.**

   If needed, select from the Base DN field to extend or limit the viewable DNs.

   Similarly, select from the Filter fields to limit the viewable entries. In the first Filter field, select the kind of entry to view, and in the second Filter field, enter a string to require in entries to be viewed.

To Add a New Entry With the Control Panel

This procedure shows how to use the Control Panel to add a new entry.

1. **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**
2. **Click the Manage Entries link under the Directory Data menu. The Manage Entries window appears, displaying the base DN in a list on its left side.**
3. **Choose the command from the Entries menu for the kind of entry you want to create.**
   - New User
   - New Group
   - New Organizational Unit
   - New Organization
   - New Domain

   A window opens, displaying fields for the attributes that apply to the kind of entry you selected to add.
4. **Enter values for the entry into the fields of the window, and then click the OK button to create the new entry.**

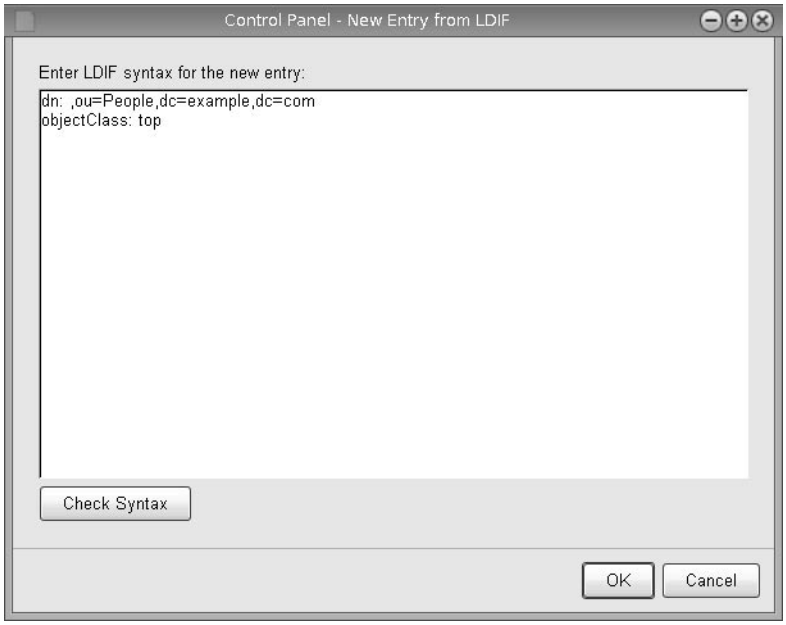
   The window displays the process of the operation.
5. **When the operation completes, click the Close button to close the window.**

To Add a New Entry From an LDIF Specification With the Control Panel

This procedure shows how to use the Control Panel to add a new entry from an LDIF specification.

1. **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**
2. **Click the Manage Entries link under the Directory Data menu on the left side of the Control Panel window. The Manage Entries window appears, displaying the base DN in a list on its left side.**
3. **Select the location where you want to add the new entry in the list of entries on the left side of the window.**
4. **Select the New from LDIF command from the Entries menu.**

   The New Entry from LDIF window appears.



5 **Enter the LDIF data that defines the entry that you want to create into the window.**

6 **Click the Check Syntax button to verify that the LDIF data that you entered can be processed.**

If the data cannot be processed, the Error window appears and describes the reason the data cannot be processed.

7 **Click the OK button.**

8 **The New Entry from LDIF window displays the progress of the operation.**

9 **When the operation is complete, click the Close button to close the New Entry from LDIF window.**

▼ To Change the Values of an Entry's Attributes With the Control Panel

This procedure shows how to use the Control Panel to change the values of an entry's attributes.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Click the Manage Entries link under the Directory Data menu on the left side of the Control Panel window. The Manage Entries window appears, displaying the base DN in a list on its left side.**

3. **Expand the list as needed, and select the entry whose attributes you want to change.**

   The attributes and values for the selected entry appear on the right side of the window.

4. **Edit the values for the entry as needed.**

   The attributes are displayed in fields where you can change the values. Change the values of attributes as needed in these fields.

   To be sure that all attributes are displayed, deselect the Only Show Attributes with Values check box on this window.

5. **To apply auxiliary object classes to the entry, click the Edit button next to the object class's name list.**

   To apply an auxiliary object class to the entry, select the object class's name in the Available list and click the Add button.

6. **Click the Save Changes button.**

   The Save Changes window appears, displaying the success or failure of the operation. Click the Close button to close the Save Changes window.

7. **Click the Close button to close the Manage Entries window.**

▼ To Delete an Entry With the Control Panel

This procedure shows how to use the Control Panel to delete an entry.

1. **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2. **Click the Manage Entries link under theDirectory Data menu on the left side of the Control Panel window. The Manage Entries window appears, displaying the base DN in a list on its left side.**

3. **Expand the list as needed, and select the object whose attributes you want to delete.**

   The attributes and values for the selected object appear on the right side of the window.

4. **Delete the selected entry.**

   - **Click the Delete Entry button.**
   - **Choose Delete Entry from the Entries menu.**

   A window opens, asking you to confirm that you want to delete the entry.

5 **Click the Yes button.**

The specified object is deleted, and the Delete Entry window appears, displaying the success or failure of the operation. Click the Details button to display additional information.

6 **Click the Close button to close the Manage Entries window.**

Managing Base DNs With the Control Panel

You can use the Control Panel to manage the hierarchy of distinguished names in a directory, as described in the following sections:

▼ Adding a New Base DN

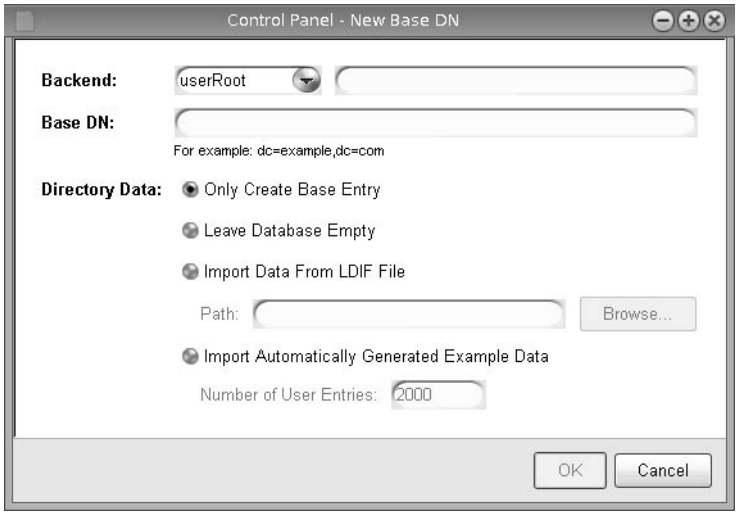
This procedure shows how to use the Control Panel to add a new base DN.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Expand the Directory Data menu on the left side of the Control Panel window to display its options, if needed.**

3 **Click the New Base DN link under Directory Data.**

The New Base DN window opens.



4 **Specify the settings for the new base DN as needed.**

5 **Click the OK button.**

The new base DN is created in the directory.

▼ Deleting a Base DN

This procedure shows how to use the Control Panel to delete a base DN.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

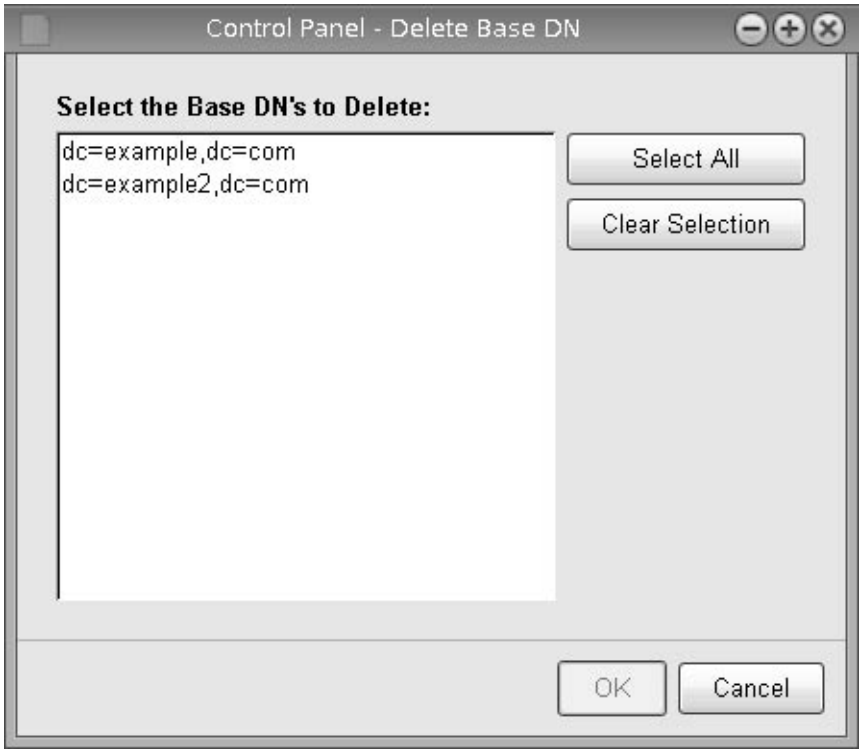
2 **Expand the Directory Data menu on the left side of the Control Panel window to display its options, if needed.**

3 **Click the Manage Entries link under Directory Data.**

The Manage Entries window opens.

4 **Select the Delete Base DN command under the Entries menu.**

The Delete Base DN window opens.



5 **Select the base DN that you want to delete.**

To deselect a base DN that you have selected, click the Clear Selection button.

6 **Click the OK button.**

The Confirmation Required window opens, asking you to confirm that you want to delete the selected base DN.

7 **Click the Yes button.**

The selected base DN is deleted from the directory.

▼ Copying an Entry's DN to the Clipboard

This procedure shows how to use the Control Panel to copy an entry's DN to the clipboard.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Click the Manage Entries link under Directory Data.**

The Manage Entries window opens.

3 **Expand the list as needed, and select the object whose DN you want to copy.**

The attributes and values for the selected object appear on the right side of the window.

4 **Choose Copy DN from the Entries menu.**

The clipboard now contains the DN of the selected entry, and you can paste it as needed.

Managing Users

You can use the control panel to manage users defined in the directory, as described in the following sections:

▼ To Reset a User's Password

This procedure shows how to use the Control Panel to reset a user's password.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Expand the Directory Data menu on the left side of the Control Panel window to display its options, if needed.**

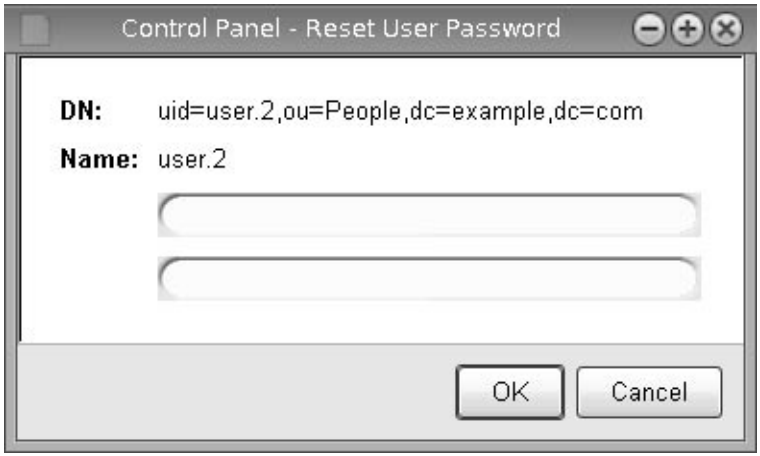
3 **Click the Manage Entries link under Directory Data.**

The Manage Entries window opens.

4 **In the list on the left side of the Control Panel window, select the entry for the user whose password you want to change.**

5 **Select the Reset User Password command from the Entries menu.**

The Reset User Password window opens.



6 **Type the new password into the first blank field.**

7 **Type the new password into the second blank field to confirm that it is recorded correctly.**

8 **Click the OK button.**

The Reset User Password window shows the progress of the operation.

9 **When the operation is complete, click the Close button to close the Reset User Password window.**

▼ To Create a Group

This procedure shows how to use the Control Panel to create a new group.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Expand the Directory Data menu on the left side of the Control Panel window to display its options, if needed.**

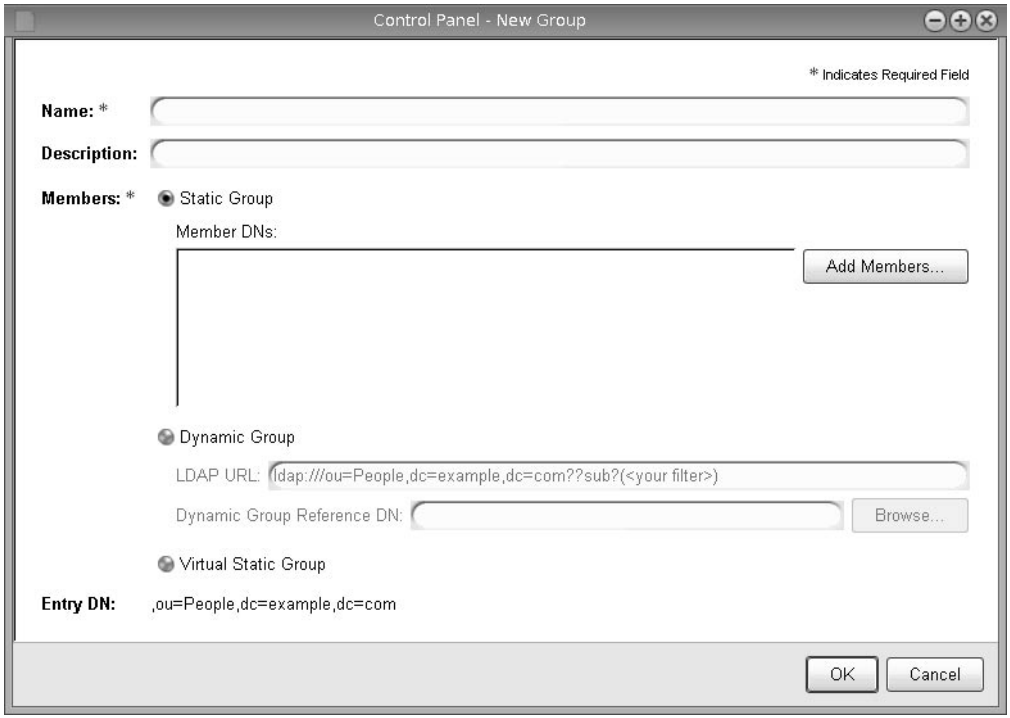
3 **Click the Manage Entries link under Directory Data.**

The Manage Entries window opens.

4 **In the list of entries on the left, select the entry where you want to add the group. For example, to create a group composed of members of the People DN, select `People` in the list.**

5 **Select the New Group command from the Entries menu.**

The New Group window opens



6 **Specify the settings for the new group as needed**

7 **If you are creating a static group, click the Add Members button to select at least one member of the group.**

The Add Members window opens.

Select at one or more entries to be the initial population of the group, and then click the OK button. The selected users are added in the Member DNs list.

8 **Click the OK button.**

The New Group window shows the progress of the operation.

9 **When the operation is complete, click the Close button to close the New Group window.**

▼ To Add a User to a Group

This procedure shows how to use the Control Panel to add a user to a group.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Expand the Directory Data menu on the left side of the Control Panel window to display its options, if needed.**

3 **Click the Manage Entries link under Directory Data.**

The Manage Entries window opens.

4 **In the list of entries on the left, select one of more entries to add to a group. For example, to create a group composed of members of the People DN, select `People` in the list.**

5 **Select the Add to Group command from the Entries menu.**

The Add to Group window opens. The entries that you selected appear in the Entries to Be Added list.

6 **Click the Add Groups button.**

The Choose Groups window opens.

7 **Select one or more groups where you want to add entries.**

8 **Click the OK button.**

The Choose Groups window closes, and the groups that you selected are added to the Groups list.

To delete a group from the Groups list, select the text of it

9 **Click the OK button.**

The Add to Group window shows the progress of the operation.

10 **When the operation is complete, click the Close button to close the Add to Group window.**

Deleting a Back End With the Control Panel

This procedure shows how to use the Control Panel to delete a back end.

▼ To Delete a Back End With the Control Panel

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

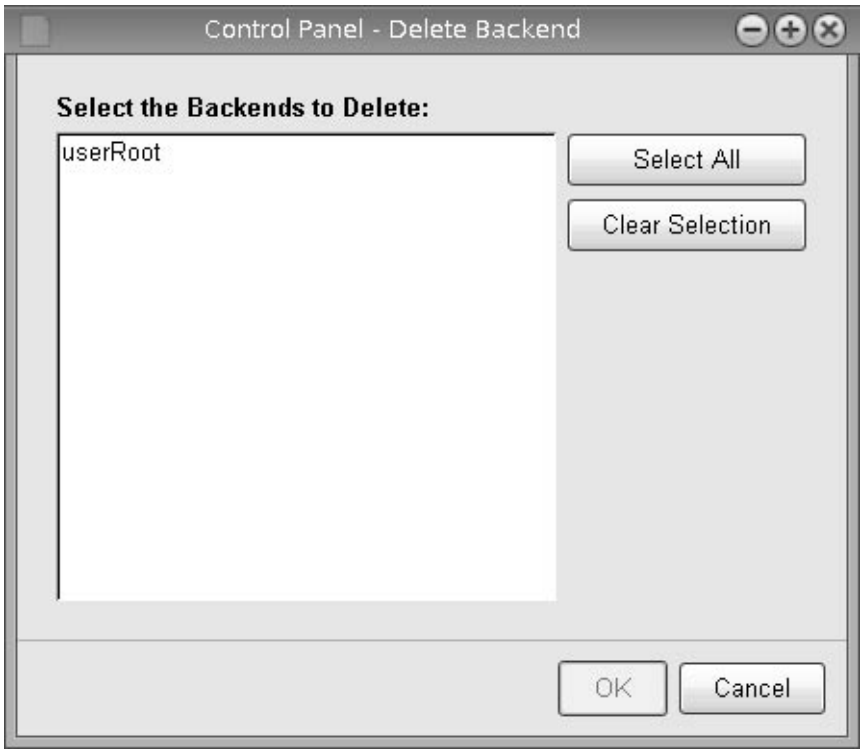
2 **Expand the Directory Data menu on the left side of the Control Panel window to display its options, if needed.**

3 **Click the Manage Entries link under Directory Data.**

The Manage Entries window opens.

4 **Select the Delete Backend command from the Entries menu.**

The Delete Backend window opens. The entries that you selected appear in the Entries to Be Added list.



5 **From the list of back ends, select the back end that you want to delete.**

6 **Click the OK button.**

The Confirmation Required window opens.

7 **If you are sure that you want to delete the selected back end, click the Yes button.**

The Delete Backend window shows the progress of the operation.

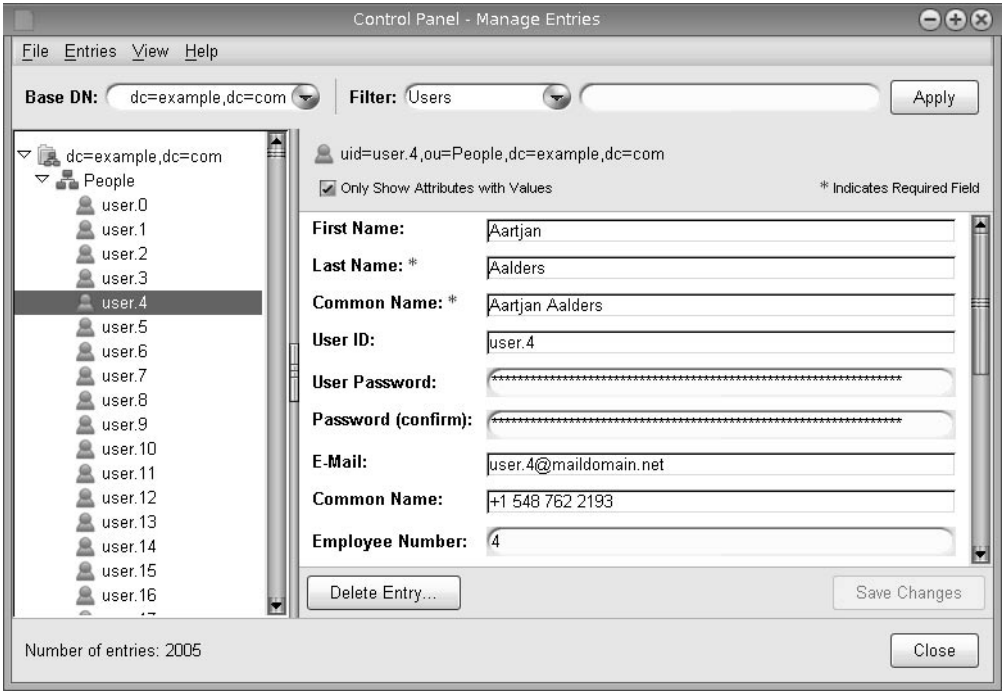
8 **When the operation is complete and the selected back end is deleted from the directory, click the Close button to close the Delete Backend window.**

Selecting a View of Entry Data

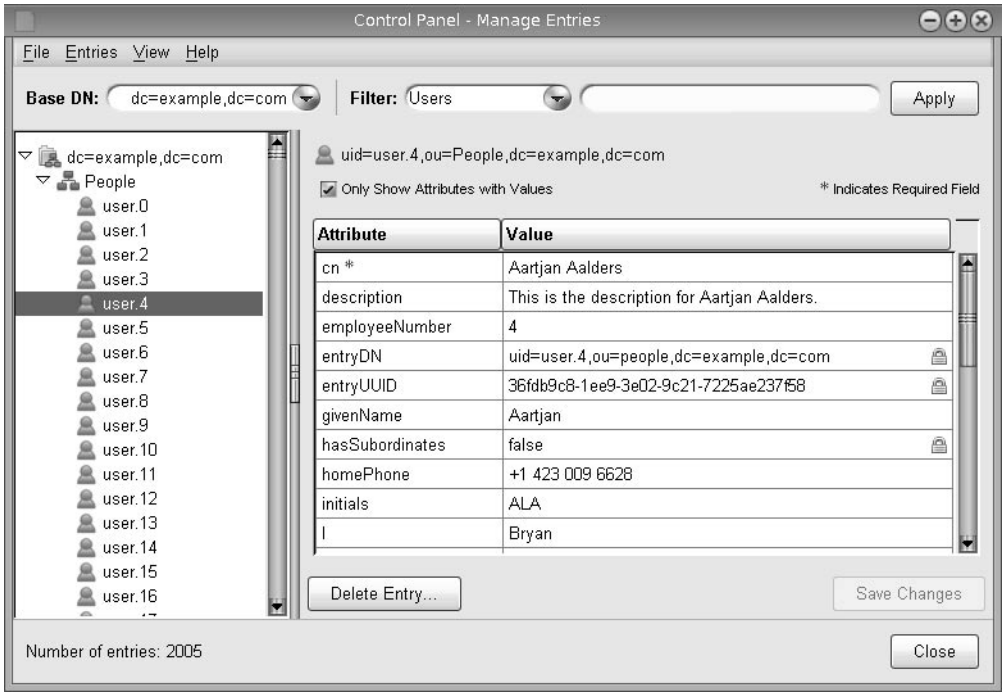
The Control Panel can display entry data in three different ways:

- Simplified view (the default view)
- Attribute view
- LDIF view

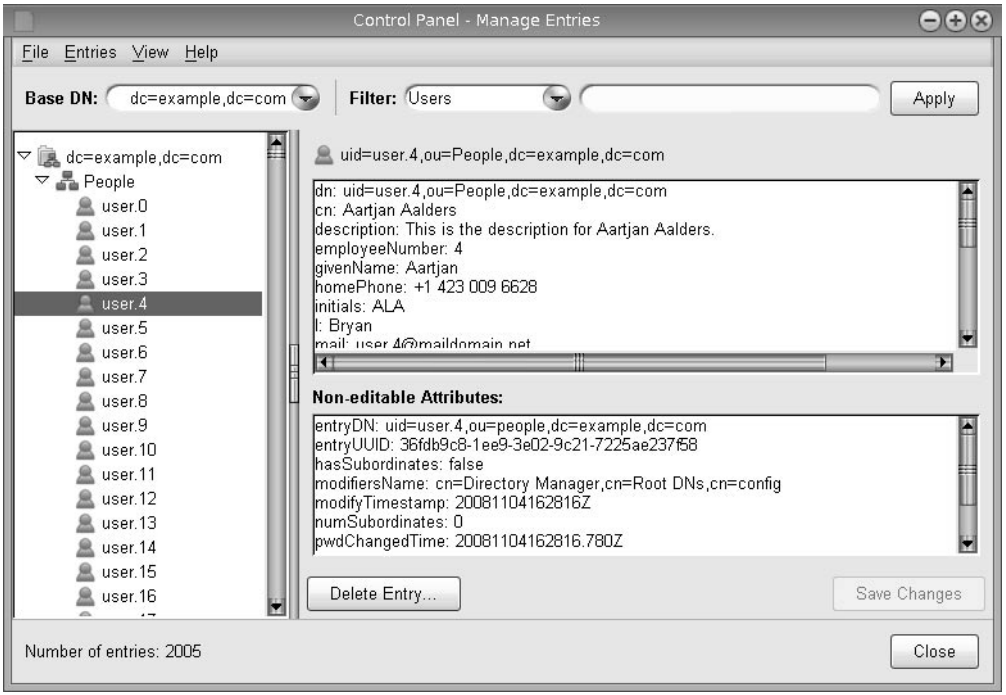
The following figures show the three different views of the same entry. The first figure shows the simplified view of the entry data.



The following figure shows the attribute view of the entry data.



The following figure shows the LDIF view of the entry data.



▼ To Select a View of Entry Data

1. **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**
2. **Expand the Directory Data menu on the left side of the Control Panel window to display its options, if needed.**
3. **Click the Manage Entries link under Directory Data.**

   The Manage Entries window opens.
4. **Select the view that you want to use from the View menu.**

Ensuring Attribute Value Uniqueness

A directory's structure requires that distinguished names be unique to identify the object and its place in the directory information tree. The directory server provides a *Unique Attribute* plug-in, which ensures that the value of an attribute is unique when the attribute is added, modified, or moved within the directory.

Overview of the Unique Attribute Plug-In

The unique attribute plug-in is disabled by default. You can enable the plug-in by using the `dsconfig` command and can define the suffix and attributes that it should check. When it is enabled, the plug-in identifies whether an LDAP add, modify, or modify DN operation causes two entries to have the same attribute value before the database is updated by the operation. If the server recognizes a conflict, the operation is terminated and an `LDAP_CONSTRAINT_VIOLATION` error is returned to the client.

When you enable attribute uniqueness on an existing directory, the server does not check for uniqueness among existing entries. After the plug-in is enabled, uniqueness is enforced when an entry is added, modified, or moved.

The unique attribute plug-in can be configured to enforce uniqueness in one or more subtrees in the directory or among entries of a specific object class. You can define several instances of the unique attribute plug-in if you want to enforce the uniqueness of other attributes. Typically, you define one plug-in instance for each attribute whose value must be unique. You can also have several plug-in instances for the same attribute to enforce "separate" uniqueness in several sets of entries.

The unique attribute plug-in is disabled by default, so that multi-master replication configuration is not affected. When the plug-in is enabled, it checks that the `uid` attribute is unique prior to any add, modify, or modify DN operations for stand-alone systems and checks for uniqueness after synchronization in replicated environments.

Like other plug-ins, the unique attribute plug-in is configured by using the `dsconfig` command. For more information, see "Configuring Plug-Ins With `dsconfig`" on page 20. The easiest way to configure plug-ins is to use `dsconfig` in interactive mode. Interactive mode functions like a wizard and walks you through the plug-in configuration. Because the interactive mode is self-explanatory, the examples in this section do not demonstrate interactive mode, but provide the equivalent complete `dsconfig` commands.

Configuring the Unique Attribute Plug-In Using `dsconfig`

The following procedures explain how to configure attribute value uniqueness.

▼ To Ensure Uniqueness of the Value of the `uid` Attribute

The unique attribute plug-in checks the `uid` attribute by default. The following task enables the unique attribute plug-in, and sets the base DN under which attribute value uniqueness for the `uid` attribute should be checked.

1 (Optional) Display the plug-ins that are currently defined in the server.

```
$ dsconfig -D "cn=directory manager" -w password -n list-plugins
Plugin                          : Type                           : enabled
--------------------------------:--------------------------------:--------
7-Bit Clean                     : seven-bit-clean                : false
Change Number Control           : change-number-control          : true
Entry UUID                      : entry-uuid                     : true
LastMod                         : last-mod                       : true
LDAP Attribute Description List : ldap-attribute-description-list : true
Password Policy Import          : password-policy-import          : true
Profiler                        : profiler                       : true
Referential Integrity           : referential-integrity          : false
UID Unique Attribute            : unique-attribute               : false
```

2 (Optional) Display the properties that are configured for the unique attribute plug-in

```
$ dsconfig -D "cn=directory manager" -w password -n get-plugin-prop \
  --plugin-name "UID Unique Attribute" \
Property : Value(s)
---------:---------
base-dn  : -
enabled  : false
type     : uid
```

3 Enable the unique attribute plug-in.

```
$ dsconfig -D "cn=directory manager" -w password -n set-plugin-prop \
  --plugin-name "UID Unique Attribute" --set enabled:true
```

4 Set the base DN under which uniqueness is checked.

```
$ dsconfig -D "cn=directory manager" -w password -n set-plugin-prop \
  --plugin-name "UID Unique Attribute" --set base-dn:ou=People,dc=example,dc=com
```

▼ To Ensure Uniqueness of the Value of Any Other Attribute

The unique attribute plug-in checks the `uid` attribute by default. If you want to ensure uniqueness for a different attribute, create a new instance of the unique attribute plug-in and set its `type` property.

This example creates a new instance of the unique attribute plug-in and ensures uniqueness of the `mail` attribute.

1 Create and enable a new instance of the unique attribute plug-in.

Set the `type` property to the name of the attribute that should be unique (in this case, `mail`.

```
$ dsconfig -D "cn=directory manager" -w password -n create-plugin \
  --type unique-attribute --plugin-name "MAIL unique attribute"
```

2. **Enable the new unique attribute plug-in.**

```
$ dsconfig -D "cn=directory manager" -w password -n set-plugin-prop \
  --plugin-name "MAIL Unique Attribute" --set enabled:true
```

3. **Set the base DN under which uniqueness is checked.**

```
$ dsconfig -D "cn=directory manager" -w password -n set-plugin-prop \
  --plugin-name "MAIL Unique Attribute" --set base-dn:ou=People,dc=example,dc=com
```

4. **Specify the attribute whose value must be unique.**

   This example specifies the `mail` attribute.

```
$ dsconfig -D "cn=directory manager" -w password -n set-plugin-prop \
  --plugin-name "MAIL Unique Attribute" --set type:mail
```

Next Steps To ensure that the values of more than one attribute are unique, create and enable multiple instances of the unique attribute plug-in.

Replication and the Unique Attribute Plug-In

The Unique Attribute plug-in does not check attribute uniqueness when an update is performed as part of a replication operation. To ensure attribute value uniqueness in a replication environment, enable the unique attribute plug-in for the same attribute in the same subtree on all servers in the topology.

Configuring Virtual Attributes

Virtual attributes are attributes whose values do not exist in persistent storage but are dynamically generated in some way.

Sun OpenDS Standard Edition supports the following virtual attribute types:

- `entryDN` virtual attribute
- `entryUUID` virtual attribute
- `hasSubordinates` virtual attribute
- `isMemberOf` virtual attribute
- `member` virtual attribute
- `numSubordinates` virtual attribute
- `subschemaSubentry` virtual attribute
- `User-defined` virtual attributes

Virtual attributes are configured by using the `dsconfig` command. `dsconfig` accesses the plug-in configuration over SSL via the "Managing Administration Traffic to the Server" on

page 5. The easiest way to configure virtual attributes is to use `dsconfig` in interactive mode. Interactive mode functions like a wizard and walks you through the virtual attribute configuration. Because the interactive mode is self-explanatory, the examples in this section do not demonstrate interactive mode, but provide the equivalent complete `dsconfig` commands.

For more information about using `dsconfig`, see "Configuring the Directory Server With `dsconfig`" on page 7.

To List the Existing Virtual Attributes

The directory server provides a number of virtual attribute rules by default. This example lists all configured virtual attribute rules.

- **Run the `dsconfig` command as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n list-virtual-attributes
Virtual Attribute           : Type               : enabled : attribute-type
----------------------------:--------------------:---------:------------------
entryDN                     : entry-dn           : true    : entrydn
entryUUID                   : entry-uuid         : true    : entryuuid
hasSubordinates             : has-subordinates   : true    : hassubordinates
isMemberOf                  : is-member-of       : true    : ismemberof
numSubordinates             : num-subordinates   : true    : numsubordinates
subschemaSubentry           : subschema-subentry : true    : subschemasubentry
Virtual Static member       : member             : true    : member
Virtual Static uniqueMember : member             : true    : uniquemember
```

The output of this command shows the following (from left to right):

- `Virtual Attribute`. The name of the virtual attribute, usually descriptive of what it does.
- `Type`. The type of virtual attribute. It is possible to define more then one virtual attribute of a specific type.
- `enabled`. Virtual attributes can either be enabled or disabled. Disabled virtual attributes remain in the server configuration, but their values are never generated.
- `attribute-type`. Specifies the type of attribute for which virtual values are generated.

To Create a New Virtual Attribute

This example creates and enables a virtual attribute rule that adds a virtual fax number of `+61 2 45607890` to any user entry with a location of Sydney (unless they already have a fax number in their entry):

- **Run the `dsconfig` command as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n create-virtual-attribute \
  --type user-defined --name "Sydney Fax Number" \
```

```
  --set attribute-type:facsimiletelephonenumber --set enabled:true \
  --set value:+61245607890 --set filter:"(&(objectClass=person)(l=Sydney))"
```

To Enable or Disable a Virtual Attribute

To enable a virtual attribute, set the `enabled` property to `true`. To disable a virtual attribute, set the `enabled` property to `false`. This example disables the virtual attribute created in the previous example:

- **Run the `dsconfig` command as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n set-virtual-attribute-prop \
  --name="Sydney Fax Number" --set enabled:false
```

To Display the Configuration of a Virtual Attribute

Use the `get-*-prop` subcommand of `dsconfig` to display the virtual attribute configuration. This example displays the properties of the virtual attribute created in the previous example:

- **Run the `dsconfig` command as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n get-virtual-attribute-prop \
  --name="Sydney Fax Number"
Property          : Value(s)
------------------:---------------------------------
attribute-type    : facsimiletelephonenumber
base-dn           : -
conflict-behavior : real-overrides-virtual
enabled           : false
filter            : (&(objectClass=person)(l=Sydney))
group-dn          : -
value             : +61245607890
```

To Change the Configuration of a Virtual Attribute

Use the `set-*-prop` subcommand of `dsconfig` to change the virtual attribute configuration. This example changes the behavior of the virtual attribute if a conflict occurs. By default, the value of a real attribute overwrites the value of the virtual attribute. With this change, the value of the real attribute and that of the virtual attribute are merged.

- **Run the `dsconfig` command as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n set-virtual-attribute-prop \
  --name="Sydney Fax Number" --set conflict-behavior:merge-real-and-virtual
```

Configuring Referrals

A *referral* is a pointer that is used to redirect a client's request to another server. Typically, referrals indicate to the client application that the requested entry or branch of the directory tree is not present on the server but is located on another remote server or at another branch of the directory tree. The client must then perform the operation again on the remote server named in the referral.

Referrals can be used in the following cases:

- When a client application requests an entry that does not exist on the local server, and the server has been configured to return the default referral.
- When an entire suffix has been disabled for maintenance, backup, or security reasons. The server will return the referrals defined by that suffix.
- When an object should be identified by different names. Referrals are useful to accommodate namespace changes.
- When "search paths" are needed for collecting results from multiple servers.

Configuring LDAP URLs

In all cases, a referral is an LDAP URL that contains the host name, port number, and optionally a DN on the local host or on another server.

Note – Unless an LDAP client provides authentication, any search request initiated by means of an LDAP URL is anonymous (unauthenticated).

The format of an LDAP URL is described in RFC 4516 and is summarized as follows:

```
ldap[s]://hostname:port/base_dn?attributes?scope?filter
```

An LDAP URL includes the following components:

`ldap[s]`	Indicates whether to connect to the server (`ldap:`), or connect to the server over SSL (`ldaps:`).
hostname	Specifies the host name or IP address of the LDAP server.
port	Specifies the port number of the LDAP server. If no port is specified, the default LDAP port (389) or LDAPS port (636) is used.
base_dn	Specifies the distinguished name (DN) of an entry in the directory. This DN identifies the entry that is the starting point of the search. If no base DN is specified, the search starts at the root of the directory tree.

attributes Returns the specified attributes. Use commas to separate more than one attribute. If no attributes are specified, the search returns all attributes.

scope Specifies the scope of the search:

- **base.** Search only the base entry specified by *base_dn*.
- **one.** Search one level below the base entry specified by base_dn
- **sub.** Search the base entry and all entries below the specified base_dn

If no scope is specified, the server performs a base search.

filter Specifies the search filter to apply to entries within the specified scope of the search. If no filter is specified, the server uses the default (`objectclass=*`).

Note – Any spaces must be escaped using a character appropriate to your shell.

Example LDAP URLs

- The following LDAP URL specifies a search for all entries that have the surname `Jensen` at any level under `dc=example,dc=com`. No port is specified, so the default (`389`) is used. No attributes are specified, so all attributes will be returned.

  ```
  ldap://example.com/dc=example,dc=com??sub?(sn=Jensen)
  ```

- The following LDAP URL specifies a search for the `cn` and `telephoneNumber` attributes at any level under `dc=example,dc=com`. The server contacts the remote server at port `2389`. Because no search filter is specified, the server uses the default filter `(objectclass=*)`.

  ```
  ldap://example.com:2389/dc=example,dc=com?cn,telephoneNumber?sub
  ```

▼ To Create a Referral

You can create a referral by adding a new entry that contains a `referral` object class and a `ref` attribute. The `ref` attribute must contain an LDAP URL.

This example creates a referral on server B for a user entry that exists on server A.

1 Locate the user entry on server A by running the following search command:

```
$ ldapsearch -h serverA -p 1389 -b dc=example,dc=com "uid=user.199" cn
dn: uid=user.199,ou=People,dc=example,dc=com
cn: Alfred Altay
```

2 Add a referral entry to the directory on server B.

```
$ ldapmodify -h serverB -p 2389 -D "cn=directory manager" -w password
dn: uid=aaltay,ou=People,dc=example,dc=com
changetype: add
```

```
objectclass: top
objectclass: extensibleObject
objectclass: referral
uid: aaltay
ref: ldap://serverA:1389/dc=example,dc=com??sub?(uid=user.199)

Processing ADD request for uid=aaltay,ou=People,dc=example,dc=com
ADD operation successful for DN uid=aaltay,ou=People,dc=example,dc=com
```

3 As a user with sufficient access rights, search for the user entry on server B.

```
$ ldapsearch -h serverB -p 2389 -D "cn=directory manager" -w password \
  -b dc=example,dc=com "uid=aaltay"
SearchReference(referralURLs={ldap://localhost:1389/dc=example,dc=com??sub?})
```

▼ To Modify a Referral

You can view or modify a referral by using `ldapsearch` or `ldapmodify` with the `manageDsaIT` control. This control informs the server that you intend to manage the referral object as a regular entry and prevents the server from sending a referral result for requests that read or update referral objects.

1 Use the `ldapsearch` command to view the referral.

```
$ ldapsearch -h serverB -p 2389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com --control managedsait "(uid=aaltay)" ref
dn: uid=aamar,ou=People,dc=example,dc=com
ref: ldap://serverA:1389/dc=example,dc=com??sub?(uid=user.199)
```

2 Use the `ldapmodify` command to modify the referral.

This example changes the server to which the referral points and the base DN under which the entry is located.

```
$ ldapmodify -h serverB -p 2389 -D "cn=Directory Manager" -w password \
  --control managedsait
dn: uid=aaltay,ou=People,dc=example,dc=com
changetype: modify
replace: ref
ref: ldap://serverC:1389/ou=People,dc=example,dc=com??sub?(uid=user.199)
Processing MODIFY request for uid=aaltay,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=aaltay,ou=People,dc=example,dc=com
```

▼ To Delete a Referral

You can delete a referral by using `ldapdelete` with the `manageDsaIT` control. This control informs the server that you intend to manage the referral object as a regular entry and prevents the server from sending a referral result for requests that read or update referral objects.

1 Use the `ldapsearch` command to view the referral.

```
$ ldapsearch -h serverB -p 2389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com --control managedsait "(uid=aaltay)" ref
dn: uid=aamar,ou=People,dc=example,dc=com
ref: ldap://serverA:1389/dc=example,dc=com??sub?(uid=user.199)
```

2 Use the `ldapdelete` command to delete the referral.

```
$ ldapdelete -h serverB -p 2389 -D "cn=Directory Manager" -w password \
  --control managedsait "uid=aaltay,ou=People,dc=example,dc=com"
Processing DELETE request for uid=aaltay,ou=People,dc=example,dc=com
DELETE operation successful for DN uid=aaltay,ou=People,dc=example,dc=com
```

Controlling Access To Data

Controlling access to directory contents is an integral part of creating a secure directory. Access to data is managed with access control instructions (ACIs) that specify the access right to individual entries, all sub-entries below an entry, or all entries on a global basis.

Numerous or complicated ACIs require greater processing resources than a few simple ACIs. You can significantly reduce the performance of your directory by specifying a large number of ACIs or extremely complicated ACIs.

The directory server includes the ability to view the effective rights of a given user for a given entry. This feature simplifies the administration of the complex and powerful access control mechanism.

For an overview of the ACI model, see "Access Control Principles" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

The following sections describe how to create ACIs to control access to data:

Managing Global ACIs With `dsconfig`

Global ACIs control access to the root of the DIT instead of to a particular sub-tree. Global ACIs apply to all entries in the directory. You can set, reset, and delete global ACIs with the `dsconfig` command and with the `ldapmodify` command. `dsconfig` accesses the server configuration over SSL, using the administration connector. For more information about `dsconfig`, see "Configuring the Directory Server With `dsconfig`" on page 7.

You cannot use `dsconfig` to manage ACIs that are applied to entries in sub-trees. To manage non-global ACIs, see "Managing ACIs With `ldapmodify`" on page 228.

Default Global ACIs

When you install the directory server, eight default global ACIs are defined. The effect of all the default global ACIs is to allow the following:

- Anyone has read access to certain controls and extended operations.
- Anyone has access to the directory for search, compare, and read operations on user attributes (except for the `userpassword` and `authPassword` attributes.)
- Authenticated users can modify their own entry in the directory, but not delete it.
- Anyone has access to key operational attributes including many in the root DSE and `cn=schema`, as well as other attributes that show up in entries throughout the server.

▼ To Display the Global ACIs

The global ACIs are all values of the `global-aci` property of the access control handler. You can use `dsconfig` to display the global ACIs currently configured on the server by viewing the `global-aci` property.

- **Run the `dsconfig` command as follows:**

```
$ dsconfig -D cn="Directory Manager" -w password -n get-access-control-handler-prop \
  --property global-aci
Property   : Value(s)
-----------:---------------------------------------------------------------------
global-aci : (extop="1.3.6.1.4.1.26027.1.6.1 || 1.3.6.1.4.1.26027.1.6.3 ||
: 1.3.6.1.4.1.4203.1.11.1 || 1.3.6.1.4.1.1466.20037 ||
: 1.3.6.1.4.1.4203.1.11.3") (version 3.0; acl "Anonymous extended
: operation access"; allow(read) userdn="ldap:///anyone";),
: "(target="ldap:///")(targetscope="base")(targetattr="objectClass||
: namingContexts||supportedAuthPasswordSchemes||supportedControl||su
: pportedExtension||supportedFeatures||supportedLDAPVersion||support
: edSASLMechanisms||vendorName||vendorVersion")(version 3.0; acl
: "User-Visible Root DSE Operational Attributes"; allow
: (read,search,compare) userdn="ldap:///anyone";)",
: "(target="ldap:///cn=schema")(targetscope="base")(targetattr="obje
: ctClass||attributeTypes||dITContentRules||dITStructureRules||ldapS
: yntaxes||matchingRules||matchingRuleUse||nameForms||objectClasses"
: )(version 3.0; acl "User-Visible Schema Operational Attributes";
: allow (read,search,compare) userdn="ldap:///anyone";)",
: (target="ldap:///dc=replicationchanges")(targetattr="*")(version
: 3.0; acl "Replication backend access"; deny (all)
: userdn="ldap:///anyone";),
: "(targetattr!="userPassword||authPassword")(version 3.0; acl
: "Anonymous read access"; allow (read,search,compare)
: userdn="ldap:///anyone";)", (targetattr="*")(version 3.0; acl
: "Self entry modification"; allow (write) userdn="ldap:///self";),
: "(targetattr="createTimestamp||creatorsName||modifiersName||modify
```

```
: Timestamp||entryDN||entryUUID||subschemaSubentry")(version 3.0;
: acl "User-Visible Operational Attributes"; allow
: (read,search,compare) userdn="ldap:///anyone";)",
: (targetcontrol="2.16.840.1.113730.3.4.2 ||
: 2.16.840.1.113730.3.4.17 || 2.16.840.1.113730.3.4.19 ||
: 1.3.6.1.4.1.4203.1.10.2 || 1.3.6.1.4.1.42.2.27.8.5.1 ||
: 2.16.840.1.113730.3.4.16") (version 3.0; acl "Anonymous control
: access"; allow(read) userdn="ldap:///anyone";)
```

To Delete a Global ACI

The easiest way to delete a global ACI is to use `dsconfig` in interactive mode. Interactive mode walks you through the ACI configuration, and is therefore not documented here. If you delete global ACIs in non-interactive mode, make sure that you escape all special characters in the ACI specification as required by your command line shell.

This example deletes the global ACI that allows anonymous access by using `dsconfig` in non-interactive mode.

- **Run the `dsconfig` command as follows.**

```
$ -D "cn=directory manager" -w password -n set-access-control-handler-prop \
  --remove global-aci:\(targetattr!=\"userPassword\|\|authPassword\"\) \
  \(version\ 3.0\;\ acl\ \"Anonymous\ read\ access\"\;\ allow\ \(read,search,compare\)
  \ userdn=\"ldap:///anyone\"\;\)
```

To Add a Global ACI

When you add a global ACI, make sure that you escape all special characters in the ACI specification as required by your command-line shell.

The following example adds the global ACI that was removed in the previous procedure, using `dsconfig` in non-interactive mode:

- **Run the `dsconfig` command as follows.**

```
$ dsconfig -D cn="Directory Manager" -w password -n set-access-control-handler-prop \
  --add global-aci:\(targetattr!=\"userPassword\|\|authPassword\"\) \
  \(version\ 3.0\;\ acl\ \"Anonymous\ read\ access\"\;\ allow\ \(read,search,compare\)
  \ userdn=\"ldap:///anyone\"\;\)
```

Managing ACIs With ldapmodify

You can create access control instructions (ACIs) manually using LDIF statements, and add them to your directory by using the ldapmodify command. Because ACI values can be very complex, it is useful to view existing values and copy them to help create new ones.

For additional sample ACIs to the ones illustrated here, see "Access Control Usage Examples" on page 229.

▼ To View ACI Attribute Values

ACIs are stored as one or more values of the aci attribute on an entry. The aci attribute is a multivalued operational attribute that can be read and modified by directory users, and should itself be protected by ACIs.

Administrative users are usually given full access to the aci attribute.

- **View the values of the aci attribute by running the following ldapsearch command:**

```
$ ldapsearch -h host -p port -D "cn=Directory Manager" -w password \
-b entryDN -s base "(objectclass=*)" aci
```

The result is LDIF text that you can copy into a new LDIF ACI definition for editing. Because the value of an ACI is a long string, the output from the ldapsearch operation is likely to be displayed over several lines, with the first space being a continuation marker. Take this into account when copying and pasting the LDIF output.

Next Steps To view the effect of an ACI value, in terms of the permissions that it grants or denies, see "Viewing Effective Rights" on page 235.

▼ To Add an ACI

You can add an ACI by specifying the ACI in an LDIF file and then applying the LDIF file with the ldapmodify command. The LDIF file must contain one or more aci attributes, each of which is composed of the aci: prefix followed by the ACI specification. For more information, see "ACI Syntax" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

1. **Create the ACI in an LDIF file.**

   The following sample LDIF file (aci.ldif) adds an ACI that grants a particular user (csmith) full access rights to the directory:

```
dn: ou=people,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="*")(version 3.0; acl "give csmith full rights"; allow(all)
userdn =  "ldap:///uid=csmith,ou=People,dc=example,dc=com";)
```

2 **Use the `ldapmodify` command to apply the ACI to the directory.**

The following command applies the ACI contained in the `aci.ldif` file to the directory:

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --filename aci.ldif
Processing MODIFY request for ou=people,dc=example,dc=com
MODIFY operation successful for DN ou=people,dc=example,dc=com
```

▼ To Remove an ACI

You can remove an ACI by specifying its value in an LDIF file, and then removing the value with the `ldapmodify` command.

1 **Remove the ACI in an LDIF file.**

The following sample LDIF file (`remove-aci.ldif`) removes the ACI that was added in the previous procedure:

```
dn: ou=people,dc=example,dc=com
changetype: modify
delete: aci
aci: (targetattr="*")(version 3.0; acl "give csmith full rights"; allow(all)
userdn =  "ldap:///uid=csmith,ou=People,dc=example,dc=com";)
```

2 **Use the `ldapmodify` command to apply the change to the directory.**

The following command applies the changes contained in the `remove-aci.ldif` file to the directory:

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
--filename remove-aci.ldif
Processing MODIFY request for ou=people,dc=example,dc=com
MODIFY operation successful for DN ou=people,dc=example,dc=com
```

Access Control Usage Examples

This section provides several sample ACIs that can be used to implement an access control policy.

Disabling Anonymous Access

The directory server allows anonymous access by default. There might be situations in which you want to disable anonymous access, particularly to sensitive data within your directory.

The following default ACI allows anonymous read access to all user attributes except for the `userpassword` and `authPassword` attributes:

```
aci: (targetattr!="userPassword||authPassword")(version 3.0; acl "
Anonymous read access"; allow (read,search,compare) userdn="ldap:///anyone";)
```

To disable anonymous access, remove this ACI from the default access control handler, as shown in the following example:

```
$ dsconfig -D cn="Directory Manager" -w password -n set-access-control-handler-prop \
  --remove global-aci:'(targetattr!="userPassword||authPassword") \
  (version 3.0; acl "Anonymous read access"; \
  allow (read,search,compare) userdn="ldap:///anyone";)'
```

Note – Depending on your shell, you might need to escape any quotations in the ACI itself.

Granting Write Access to Personal Entries

Many directory administrators want to allow internal users to change some but not all of the attributes in their own entry. The procedures in this section describe how to grant write access.

Granting Write Access Based on DNS

The following example ACI enables users internal to `example.com` to change their own password, home telephone number, and home address, but nothing else.

By allowing write access, you also grant users the right to delete attribute values.

```
aci: (targetattr="userPassword || homePhone || homePostalAddress")
(version 3.0; acl "Write example.com"; allow (write)
userdn="ldap:///self" and dns="*.example.com";)
```

This example assumes that the ACI is added to the `ou=People,dc=example,dc=com` entry.

Granting Write Access Based on Authentication Method

The following example enables any user to update his own personal information in the `example.com` tree provided that he establish an SSL connection to the directory.

By setting this permission, you are also granting users the right to delete attribute values.

```
aci: (targetattr="userPassword || homePhone || homePostalAddress")
(version 3.0; acl "Write SSL"; allow (write)
userdn= "ldap://self" and authmethod="ssl";)
```

This example assumes that the `aci` is added to the `ou=subscribers,dc=example,dc=com` entry.

Granting a Group Full Access to a Suffix

Most directories have a group that is used to identify certain corporate functions. These groups can be given full access to all or part of the directory. By applying the access rights to the group, you can avoid setting the access rights for each member individually. Instead, you grant users these access rights by adding them to the group.

The following sample ACI allows a group named the `HRgroup` full access to the `ou=People` branch of the directory so that they can update employee information:

```
aci: (targetattr="*") (version 3.0; acl "HR"; allow (all)
groupdn= "ldap:///cn=HRgroup,ou=People,dc=example,dc=com";)
```

This example assumes that the ACI is added to the `ou=People,dc=example,dc=com` entry.

Granting Rights to Add and Delete Group Entries

Some organizations want to allow employees to create entries in the tree if it can increase their efficiency, or if it can contribute to the corporate dynamics. The following examples assume that `example.com` has a social committee that is organized into various clubs (tennis, swimming, skiing, and so on).

Creating a "`Create Group`" ACI

This sample ACI allows any `example.com` employee to create a group entry representing a new club, under the `ou=social committee` branch.

```
aci: (target="ldap:///ou=social committee,dc=example,dc=com")
(targetattr="*")(targattrfilters="add=objectClass:
(|(objectClass=groupOfNames)(objectClass=top))")
(version 3.0; acl "Create Group"; allow (read,search,add)
userdn= "ldap:///uid=*,ou=People,dc=example,dc=com")
and dns="*.example.com";)
```

This example assumes that the ACI is added to the `ou=social committee, dc=example,dc=com` entry.

Note – This ACI does not grant write permission, which means that the entry creator cannot modify the entry. Because the server adds the value `top` behind the scenes, you must specify `objectClass=top` in the `targattrfilters`

Creating a "`Delete Group`" ACI

This sample ACI ensures that only the group owner can modify or delete a group entry under the ou=Social Committee branch.

```
aci: (target="ou=social committee,dc=example,dc=com")
(targetattr = "*")
(targattrfilters="del=objectClass:(objectClass=groupOfNames)")
(version 3.0; acl "Delete Group"; allow (write,delete)
userattr="owner#GROUPDN";)
```

This example assumes that the ACI is added to the `ou=social committee,dc=example,dc=com` entry.

Allowing Users to Add or Remove Themselves From a Group

Many directories set ACIs that allow users to add or remove themselves from groups. This is useful, for example, for allowing users to add and remove themselves from mailing lists. The following sample ACI enables all employees to add themselves to any group entry under the `ou=social committee` subtree:

```
aci: (targettattr="member")(version 3.0; acl "Group Members";
allow (selfwrite)
(userdn= "ldap:///uid=*,ou=People,dc=example,dc=com") ;)
```

This example assumes that the ACI is added to the `ou=social committee, dc=example,dc=com` entry.

Granting Conditional Access to a Group

In many cases, when you grant a group privileged access to the directory, you want to ensure that those privileges are protected from intruders trying to impersonate the privileged users. Therefore, in many cases, access control rules that grant critical access to a group or role are often associated with a number of conditions.

The following sample ACI grants the Directory Administrators group full access to the corporate clients branch of the directory tree, provided the following conditions are fulfilled:

- The connection is authenticated using a certificate over SSL
- Access is requested between 08:00 and 18:00, Monday through Thursday
- Access is requested from a specified IP address

```
aci: (target="ou=corporate-clients,dc=example,dc=com")
(targetattr = "*") (version 3.0; acl "corporate-clients"; allow (all)
(groupdn="ldap:///cn=DirectoryAdmin,ou=corporate-clients,dc=example,dc=com")
and (authmethod="ssl") and (dayofweek="Mon,Tues,Wed,Thu") and
(timeofday >= "0800" and timeofday <= "1800") and (ip="255.255.123.234"); )
```

This example assumes that the ACI is added to the `ou=corporate-clients,dc=example,dc=com` entry.

Denying Access

If your directory holds business-critical information, you might specifically want to deny access to it. The following sample ACIs allow users to read certain "billing information", such as connection time and account balance, under their own entries, but prohibits them from changing this information.

This ACI allows users to read the information. The example assumes that the relevant attributes have been created in the schema.

```
aci: (targetattr="connectionTime || accountBalance")
(version 3.0; acl "Billing Info Read"; allow (search,read)
userdn="ldap:///self";)
```

This ACI prevents users from changing the information. The example assumes that the relevant attributes have been created in the schema.

```
aci: (targetattr="connectionTime || accountBalance")
(version 3.0; acl "Billing Info Deny";
deny (write) userdn="ldap:///self";)
```

Defining Permissions for DNs That Contain a Comma

DNs that contain commas require special treatment within LDIF ACI statements. In the target and bind rule portions of the ACI statement, commas must be escaped by a single backslash (\). The following example illustrates this syntax:

```
dn: o=example.com Bolivia\, S.A.
objectClass: top
objectClass: organization
aci: (target="ldap:///o=example.com Bolivia\,S.A.")
(targetattr="*") (version 3.0; acl "aci 2"; allow (all)
groupdn = "ldap:///cn=Directory Administrators,
o=example.com Bolivia\, S.A.";)
```

Proxy Authorization ACIs

The proxy authorization method is a special form of authentication: a user that binds to the directory using his own identity is granted the rights of another user, through proxy authorization.

This example makes the following assumptions:

- The client application's bind DN is `uid=MoneyWizAcctSoftware,ou=Applications,dc=example,dc=com`.
- The targeted subtree to which the client application is requesting access is `ou=Accounting,dc=example,dc=com`.
- An Accounting Administrator with access permissions to the `ou=Accounting,dc=example,dc=com` subtree exists in the directory.

For the client application to gain access to the Accounting subtree (using the same access permissions as the Accounting Administrator), the application requires the following rights and controls:

- The Accounting Administrator must have access permissions to the `ou=Accounting,dc=example,dc=com` subtree. The following ACI grants all rights to the Accounting Administrator entry:

```
aci: (target="ldap:///ou=Accounting,dc=example,dc=com")
(targetattr="*") (version 3.0; acl "allow All-AcctAdmin"; allow
(all) userdn="ldap:///uid=AcctAdministrator,ou=Administrators,
dc=example,dc=com";)
```

- The client application must have proxy rights. The following ACI grants proxy rights to the client application:

```
aci: (target="ldap:///ou=Accounting,dc=example,dc=com")
(targetattr="*") (version 3.0; acl "allow proxy-
accounting software"; allow (proxy) userdn=
"ldap:///uid=MoneyWizAcctSoftware,ou=Applications,
dc=example,dc=com";)
```

- The client application must be allowed to use the proxy authorization control. The following ACI allows the client application to use the proxy authorization control:

```
aci: (targetcontrol = "2.16.840.1.113730.3.4.18")
(version 3.0; acl "allow proxy auth - accounting software";
allow (all) userdn="ldap:///uid=MoneyWizAcctSoftware,ou=Applications,
dc=example,dc=com";)
```

With these ACIs in place, the `MoneyWizAcctSoftware` client application can bind to the directory and send an LDAP command such as `ldapsearch` or `ldapmodify` that requires the access rights of the proxy DN.

In the previous example, if the client wanted to perform an `ldapsearch` command, the command would include the following controls:

```
$ ldapsearch -D "uid=MoneyWizAcctSoftware,ou=Applications,dc=example,dc=com" \
  -w password -Y "dn:uid=AcctAdministrator,ou=Administrators,dc=example,dc=com" \
```

```
  -b "ou=Accounting,dc=example,dc=com"  "objectclass=*"\
...
```

The base of the search must match the target of the ACIs. The client binds as itself but is granted the privileges of the proxy entry. The client does not need the password of the proxy entry.

For more information, see "To Search Using the Proxied Authorization Control" on page 165.

Viewing Effective Rights

When you maintain the access control policy on the entries of a directory, it is useful to know the effects on security of the ACIs that you define. The directory server enables you to evaluate existing ACIs and report the effective rights that they grant for a given user on a given entry.

The Get Effective Rights Control

The directory server responds to the Get Effective Rights control, which can be included in a search operation. The response to this control is to return the effective rights information about the entries and attributes in the search results. This extra information includes read and write permissions for each entry and for each attribute in each entry. The permissions may be requested for the bind DN used for the search or for an arbitrary DN, allowing administrators to test the permissions of directory users.

Effective rights functionality relies on an LDAP control. To view the effective rights when going through a proxy server, you must enable this control in the proxy chaining policy. You must also ensure that the proxy identity used to bind to the remote server is also allowed to access the effective rights attributes.

Using the Get Effective Rights Control

The behavior of the Get Effective Rights Control differs from the Internet draft Get Effective Rights Control in the following ways:

- There is no response control returned with the search results. Instead, the rights information is added to the result entries. Also, the format of the rights information is completely different from the draft and is described below.
- The request control only takes an `authzid`.

There are two ways to specify the Get Effective Rights control with the `ldapsearch` command:

1. Use the `-J "1.3.6.1.4.1.42.2.27.9.5.2"` option or simply `-J effectiverights`. If you specify a NULL value for the Get Effective Rights Control's `authzid` value, the bind user is used as the `authzid` and the rights for the attributes and entries being returned with the current `ldapsearch` operation are retrieved.

2. The simpler and preferred method is to use the `-g` option with or without the `-e` option:
3.
   a. `-g "dn: `*DN*`"`--The search results will show the effective rights of the user binding with the given *DN*. This option allows an administrator to check the effective rights of another user. The option `-g "dn:"` will show the effective rights for anonymous authentication.
   b. `-e` *attributeName1* `-e` *attributeName2* --The search results will also include the effective rights on the named attributes. This option can be used to specify attributes that would not appear in the search results for the entry. For example, this option can be used to determine if a user has permission to add an attribute that does not currently exist in an entry.

Note – The `-e` option requires the `-g` option and should not be used with the `-J` option.

If you use the `-g` option, do not use the `-J` option with the OID of the Get Effective Rights control.

Besides using one of these two ways to specify the Get Effective Rights Control, you must specify the type of information you want to view, either the simple rights or the more detailed logging information that explains how those rights are granted or denied. The type of information is determined by adding either `aclRights` or `aclRightsInfo`, respectively, as an attribute to return in the search results. You can request both attributes to receive all effective rights information, although the simple rights are redundant with the information in the detailed logging information.

Note – The `aclRights` and `aclRightsInfo` attributes have the behavior of virtual operational attributes. They are not stored in the directory, and they will not be returned unless explicitly requested. These attributes are generated by the directory server in response to the Get Effective Rights Control. For this reason, neither of these attributes can be used in filters or search operations of any kind.

The effective rights feature inherits other parameters that affect access control (such as time of day, authentication method, machine address, and machine name) from the user initiating the search operation.

The following example shows how a user, Carla Fuente, can view her rights in the directory. In the results, a `1` means that permission is granted, and a `0` means that permission is denied.

```
$ ldapsearch -J effectiverights -h rousseau.example.com -p 1389 \
  -D "uid=cfuente,ou=People,dc=example,dc=com" -w password \
  -b "dc=example,dc=com" "(objectclass=*)" aclRights
```

```
dn: dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: ou=Groups, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=HR Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: uid=bjensen,ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: uid=cfuente, ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:1,proxy:0
```

This result shows Carla Fuente the entries in the directory where she has at least read permission and that she can modify her own entry. The effective rights control does not bypass normal access permissions, so a user will never see the entries for which they do not have read permission. In the following example, the Directory Manager can see the entries to which Carla Fuente does not have read permission:

```
$ ldapsearch -h rousseau.example.com -p 1389 -D "cn=Directory Manager" -w password \
  -g "dn: uid=cfuente,ou=People,dc=example,dc=com" -b "dc=example,dc=com" \
  "(objectclass=*)" aclRights
dn: dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: ou=Groups, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=Directory Administrators, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:0,write:0,proxy:0
dn: ou=Special Users,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:0,write:0,proxy:0
dn: ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: cn=HR Managers,ou=groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: uid=bjensen,ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
dn: uid=cfuente, ou=People, dc=example,dc=com
aclRights;entryLevel: add:0,delete:0,read:1,write:1,proxy:0
```

In the output above, the directory manager can see that Carla Fuente cannot even view the Special Users nor the Directory Administrators branches of the directory tree. In the following example, the Directory Administrator can see that Carla Fuente cannot modify the `mail` and `manager` attributes in her own entry:

```
$ ldapsearch -h rousseau.example.com -p 1389 -D "cn=Directory Manager" -w password \
  -g "dn: uid=cfuente,ou=People,dc=example,dc=com" -b "dc=example,dc=com" \
  "(uid=cfuente)" aclRights "*"
version: 1
dn: uid=cfuente, ou=People, dc=example,dc=com
aclRights;attributeLevel;mail: search:1,read:1,compare:1,
write:0,selfwrite_add:0,selfwrite_delete:0,proxy:0
mail: cfuente@example.com
aclRights;attributeLevel;uid: search:1,read:1,compare:1,
write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
uid: cfuente
aclRights;attributeLevel;givenName: search:1,read:1,compare:1,
write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
givenName: Carla
aclRights;attributeLevel;sn: search:1,read:1,compare:1,
write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
sn: Fuente
aclRights;attributeLevel;cn: search:1,read:1,compare:1,
write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
cn: Carla Fuente
aclRights;attributeLevel;userPassword: search:0,read:0,
compare:0,write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
userPassword: {SSHA}wnbWHIq2HPiY/5ECwe6MWBGx2KMiZ8JmjF80Ow==
aclRights;attributeLevel;manager: search:1,read:1,compare:1,
write:0,selfwrite_add:0,selfwrite_delete:0,proxy:0
manager: uid=bjensen,ou=People,dc=example,dc=com
aclRights;attributeLevel;telephoneNumber: search:1,read:1,compare:1,
write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
telephoneNumber: (234) 555-7898
aclRights;attributeLevel;objectClass: search:1,read:1,compare:1,
write:1,selfwrite_add:1,selfwrite_delete:1,proxy:0
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
```

Understanding Effective Rights Results

Depending on the options specified, an effective rights request returns the following information:

- "Rights Information" on page 239
- "`write`, `selfwrite_add`, and `selfwrite_delete` Permissions" on page 240
- "Logging Information" on page 243

Rights Information

The effective rights information is presented according to the following subtypes:

`aclRights;entrylevel` - Presents entry-level rights information

`aclRights;attributelevel` - Presents attribute-level rights information

`aclRightsInfo;entrylevel` - Presents entry-level logging information

`aclRightsInfo;attributelevel` - Presents attribute-level logging information

The format of the `aclRights` string is as follows:

`aclRights;entryLevel:` *permission*:*value*(*permission*:*value*)*

and

`aclRights;attributeLevel:` *permission*:*value*(*permission*:*value*)*

The possible entry-level permissions are `add`, `delete`, `read`, `write`, and `proxy`. The possible values for each permission are `0` (permission not granted) and `1` (permission granted).

Entry-level Permission	Explanation
`add` and `delete`	The ability of a user to add and delete the entire entry.
`read`	The ability of a user to read and search attributes in the entry.
`write`	The ability of a user to add, delete, and replace attribute values in the entry.
`proxy`	The ability of a user to access the directory with the rights of the entry.

Note – For information about assigning these permissions in an ACI, see "ACI Syntax" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

The possible attribute-level permissions are `read`, `search`, `compare`, `write`, `selfwrite_add`, `selfwrite_delete`, and `proxy`. The possible values for each permission are `0` (permission not granted) and `1` (permission granted). For the case of the write permission, the value of "`?`" is also permitted.

Attribute-level Permission	Explanation
`read`	The ability of a user to read the attribute value in the entry.
`search`	The ability of a user to search the attribute value in the entry.
`compare`	The ability of a user to compare the attribute value in the entry with a value that is provided by the user.
`write`	The ability of a user to add, delete, and replace the attribute value in the entry. This applies to all attributes except the `authorization dn`.
`selfwrite_add`	The ability of a user to add the attribute, `authorization dn`.
`selfwrite_delete`	The ability of a user to delete the attribute, `authorization dn`.
`proxy`	The ability of a user to access the directory with the rights of the attribute in the entry.

Note – The `write`, `selfwrite_add`, and `selfwrite_delete` permissions are particularly complex. If you see a "`?`", consult the logging information to establish why the permissions will or will not be granted. For more information, see Table 3.

The format of the `aclRightsInfo` string is as follows:

`aclRightsInfo;logs;entryLevel;`*permission*`: acl_summary(main):`*permission-string*

and

`aclRightsInfo;logs;attributeLevel;`*permission*`;attribute: acl_summary(main):`*permission-string*

The entry-level and attribute-level permissions are described in the preceding section.

The *permission-string* contains detailed information about the effective rights at the entry-level and attribute-levels.

`write`, `selfwrite_add`, and `selfwrite_delete` Permissions

The attribute-level permission for `write` can be either `0`, `1`, or "`?`". Only `write` attribute-level permissions can have a value of "`?`", which usually results from a `targattrfilters` ACI component. For add and delete permissions, the entries that can be modified depend on the values of the attributes in the entry. Only the permission, `0` or `1`, is returned on the entries as they are returned with the `ldapsearch` operation.

For all attribute values except the `authorization dn`, if the value for a `write` permission is `1`, the permission is granted for both add and delete. Similarly, for all attribute values except the `authorization dn`, a value of `0` for a write permission means that the permission is not granted for either add or delete `ldapmodify` operations. The permission in force for the value of the `authorization dn` is returned explicitly in one of the `selfwrite` permissions, that is, either `selfwrite_add` or `selfwrite_delete`.

Although `selfwrite_add` and `selfwrite_delete` attribute-level permissions do not exist in the context of ACIs, a set of ACIs can grant a user `selfwrite` permission for just the add or just the delete part of a modify operation. For `selfwrite` permissions, the value of the attribute being modified is the `authorization dn`. The same distinction is not made for `write` permissions because the value of the attribute being modified for a write permission is undefined.

When the effective permission depends on a `targattrfilters` ACI, the "?" value indicates that the logging information should be consulted for more permission detail. Given the relative complexity of the interdependencies between the `write`, `selfwrite_add`, and `selfwrite_delete` permissions, The following table explains what the possible combinations of these three permissions mean.

The following table outlines the interdependencies of the various effective rights values.

TABLE 3 Effective Rights Permission Interdependencies

write	selfwrite_add	selfwrite_delete	Effective Rights Explanation
`0`	`0`	`0`	Cannot add or delete any values of this attribute.
`0`	`0`	`1`	Can only delete the value of the `authorization dn`.
`0`	`1`	`0`	Can only add the value of the `authorization dn`.
`0`	`1`	`1`	Can only add or delete the value of the `authorization dn`.
`1`	`0`	`0`	Can add or delete all values except the `authorization dn`.
`1`	`0`	`1`	Can delete all values including the `authorization dn` and can add all values excluding the authorization dn.

TABLE 3 Effective Rights Permission Interdependencies *(Continued)*

write	selfwrite_add	selfwrite_delete	Effective Rights Explanation
1	1	0	Can add all values including the `authorization dn` and can delete all values excluding the `authorization dn`.
1	1	1	Can add or delete all values of this attribute.
?	0	0	Cannot add or delete the `authorization dn` value, but might be able to add or delete other values. Refer to logging information for further details regarding the `write` permission.
?	0	1	Can delete but cannot add the value of the `authorization dn`, and might be able to add or delete other values. Refer to logging information for further details regarding the `write` permission.
?	1	0	Can add but cannot delete the value of the `authorization dn` and might be able to add or delete other values. Refer to logging information for further details regarding the `write` permission.
1	?	1	Can add and delete the value of the `authorization dn` and might be able to modify add, modify, or delete other values. Refer to logging information for further details regarding the `write` permission.

Logging Information

The effective rights logging information enables you to understand and debug access control difficulties. The logging information contains an access control summary statement, called the `acl_summary`, that indicates why access control has been allowed or denied. The access control summary statement includes the following information:

- Whether access was allowed or denied
- The permissions granted
- The target entry of the permissions
- The name of the target attribute
- The subject of the rights being requested
- Whether or not the request was made by proxy, and if so, the proxy authentication DN
- The reason for allowing or denying access (important for debugging purposes as explained in the following table)

The following table lists the effective rights logging information reasons and their explanations.

TABLE 4 Effective Rights Logging Information Reasons and Their Explanations

Logging Information Reason	Explanation
`no reason available`	No reason available to explain why access was allowed or denied.
`no allow acis`	No `allow` ACIs exist, which results in denied access.
`result cached deny`	Cached information was used to determine the access denied decision.
`result cached allow`	Cached information was used to determine the access allowed decision.
`evaluated allow`	An ACI was evaluated to determine the access allowed decision. The name of the ACI is included in the log information.
`evaluated deny`	An ACI was evaluated to determine the access denied decision. The name of the ACI is included in the log information.
`no acis matched the resource`	No ACIs match the resource or target, which results in denied access.
`no acis matched the subject`	No ACIs match the subject requesting access control, which results in denied access.

TABLE 4 Effective Rights Logging Information Reasons and Their Explanations *(Continued)*

Logging Information Reason	Explanation
`allow anyone aci matched anon user`	An ACI with a `userdn = "ldap:///anyone"` subject allowed access to the anonymous user.
`no matching anyone aci for anon user`	No ACI with a `userdn= "ldap:///anyone"` subject was found, so access for the anonymous user was denied.
`user root`	The user is root DN and is allowed access.

Note – Write permissions for virtual attributes are not provided, nor is any associated logging evaluation information, because virtual attributes cannot be updated.

Restricting Access to the Get Effective Rights Control

Viewing effective rights is itself a directory operation that should be protected and appropriately restricted.

The default ACI does not allow read access to the `aclRights` and `aclRightsInfo` operational attributes used to return effective rights. Create a new ACI for these attributes to enable access by directory users to this information.

For example, the following ACI allows members of the Directory Administrators group to get effective rights:

```
aci: (targetattr = "aclRights||aclRightsInfo")(version 3.0; acl "getEffectiveRights";
allow(all) groupdn = "ldap:///cn=Directory
Administrators,ou=Groups,dc=example,dc=com";)
```

In addition, access is needed to use the Get Effective Rights Control.

To enable access by directory users to the Get Effective Rights Control, create a new ACI target by using the OID (`1.3.6.1.4.1.42.2.27.9.5.2`) for this control. For additional ACI syntax information, see "Defining Targets" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

For example, the following ACI allows members of the Directory Administrators group to use the Get Effective Rights control:

```
aci: (targetcontrol = "1.3.6.1.4.1.42.2.27.9.5.2")(version 3.0;
acl "getEffectiveRights control access";
allow(all) groupdn = "ldap:///cn=Directory
Administrators,ou=Groups,dc=example,dc=com";)
```

Replicating Data

Replication enables copies of identical data to be available across multiple servers. The directory server uses a multi-master replication model, which means that all the directory servers within a replication topology can accept read and write operations.

The multi-master replication model is *loosely consistent* by default. This means that changes made on one server are replayed asynchronously to the other servers in the topology. The same entries can be modified simultaneously on different servers. When updates are sent between the two servers, any conflicting changes must be resolved. Various attributes of a WAN, such as latency, can increase the chance of replication conflicts. Conflict resolution generally occurs automatically. A number of conflict rules determine which change takes precedence. In some cases conflicts must be resolved manually.

Note – In certain deployment scenarios, the default loose consistency model might not be adequate. In these situations, you can configure replication to function in *assured* mode. For more information, see "Configuring Assured Replication" on page 253.

Replication always occurs over a secure connection. Both parties of a replication session must authenticate to the other using SSL certificates. No access control or privileges are enforced. The following sections describe how to configure replication in the directory server.

This section includes the following topics:

Configuring Replication With dsreplication

You can set up replication automatically using the QuickSetup utility when you first install the directory server. This section explains how to set up replication manually between two directory servers, using the dsreplication utility. dsreplication accesses the server configuration over SSL via the administration connector. For more information, see "Managing Administration Traffic to the Server" on page 5

In any topology, you should have two replication servers for availability, in case one replication server fails. Replication servers are responsible for keeping track of all changes in the environment. Each replication server contains a list of all other replication servers in the topology.

The examples in this section assume that you have already installed two directory servers and populated one with data.

To Enable Replication Between Two Servers

To enable replication, use the dsreplication enable command. This command creates a replication server instance on the same host as each directory server.

Note – You cannot run more than one instance of the dsreplication enable command to set up replication between multiple servers in parallel. Rather, run the dsreplication enable command successively for each pair of replicated servers in the topology.

The following command enables replication of the data under "dc=example,dc=com" between two directory servers, host1 and host2. Both servers use the default administration port (4444). The command creates a replication server instance on host1, port 8989, and a second replication server instance on host2, port 8990.

```
$ dsreplication enable \
  --host1 host1 --port1 4444 --bindDN1 "cn=Directory Manager" \
  --bindPassword1 password --replicationPort1 8989 \
  --host2 host2 --port2 4444 --bindDN2 "cn=Directory Manager" \
  --bindPassword2 password  --replicationPort2 8990 \
  --adminUID admin --adminPassword password --baseDN "dc=example,dc=com" -X -n
```

To Initialize a Replicated Server

To initialize a replicated server with the data from another replicated server, use the dsreplication initialize command. The following command initializes the base DN "dc=example,dc=com" on host2 with the data contained on host1:

```
$ dsreplication initialize --baseDN "dc=example,dc=com" \
  --adminUID admin --adminPassword password \
  --hostSource host1 --portSource 4444 \
  --hostDestination host2 --portDestination 4444 -X -n
```

To Initialize an Entire Topology

If there are more than two directory servers in the topology, use the `dsreplication intialize-all` command to initialize all replicas simultaneously. This command takes the details of the source host as arguments, and initializes all other servers for which replication is enabled.

The following command initializes all servers on which replication is enabled, from the contents of the base DN "`dc=example,dc=com`" on host1:

```
$ dsreplication initialize-all --hostname host1 --port 4444 \
  --baseDN "dc=example,dc=com" --adminUID admin --adminPassword password
```

▼ To Test Replication

The easiest way to test that replication is working is to apply changes on one directory server and to check that those changes have been replicated on another directory server. To test the replication topology set up in the previous procedures, do the following:

1. **Use `ldapmodify` to change an entry on host1.**
2. **Use `ldapsearch` to verify that the change was propagated to host2.**

To Obtain the Status of a Replicated Topology

Use the `dsreplication` status command to display a list of the directory servers in the topology, along with any missing changes between those servers. You can use the connection details of any directory server in the topology to obtain the status of the entire topology. The following command displays the status of the topology set up in the previous procedures:

```
$ dsreplication status -h host1 -p 4444 --adminUID admin \
  --adminPassword password

dc=example,dc=com - Replication Enabled
=======================================
Server         : Entries : M.C. (1) : A.O.M.C. (2) : Port (3) : Security (4)
---------------:---------:----------:--------------:----------:-------------
host1:4444     : 102     : 0        : 0            : 8989     : Disabled
host2:4444     : 102     : 0        : 0            : 8990     : Disabled
```

```
[1] The number of changes that are still missing on this server (and that have
been at least applied to one of the other servers).
[2] Age of oldest missing change: the age (in seconds) of the oldest change
that has not arrived to this server.
[3] The port used to communicate between the servers whose contents are being
replicated.
[4] Whether the replication communication through the replication port is
encrypted or not.
```

Modifying the Replication Configuration With dsconfig

This section describes how to change certain advanced properties of a replication configuration by using the dsconfig command. Advanced properties are usually optional, or have a default value that is acceptable in most cases. For general information about using dsconfig, see "Configuring the Directory Server With dsconfig" on page 7.

You cannot use dsconfig to set up replication between directory servers. Replication can be set up automatically using the QuickSetup utility when you first install the directory server, or manually, using the dsreplication command. For more information, see "Configuring Replication With dsreplication" on page 246.

Retrieving the Replication Domain Name

The *replication domain name* is generated by the directory server and includes a numeric unique identifier.

To obtain a list of the configured replication domains, use the list-replication-domains subcommand. For example:

```
$ dsconfig -D "cn=directory manager" -w password -n list-replication-domains \
  --provider-name "Multimaster Synchronization"
```

Changing the Replication Purge Delay

The replication changes database maintains a record of updates, which might or might not have been replicated. The replication purge delay is a property of the replication server, and specifies the period of time after which internal purge operations are performed on the replication changes database.

How Replication Changes Are Purged

Any change that is older than the purge delay is removed from the replication changes database, irrespective of whether that change has been applied. The default purge delay is one day. If the replication changes database is backed up less frequently than the purge delay, changes will be cleared before the changes database has been backed up. Changes can therefore be lost if you use the backup to restore data.

▼ To Change the Replication Purge Delay

1 (Optional) Display the current value of the replication purge delay.

```
$ dsconfig -D "cn=directory manager" -w password -n get-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --property replication-purge-delay

Property                : Value(s)
------------------------:---------
replication-purge-delay : 1 d
```

2 Change the purge delay.

The following command changes the purge delay to one week:

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --set replication-purge-delay:1w
```

Changing the Window Size

The window size is a property of the replication server and specifies the number of change requests that are sent to directory servers, without the replication server having to wait for an acknowledgment from the directory server before continuing.

The window size represents the maximum number of update messages that can be sent without immediate acknowledgment from the directory server. It is more efficient to send many messages in quick succession instead of waiting for an acknowledgment after each one. Using the appropriate window size, you can eliminate the time replication servers spend waiting for acknowledgments to arrive. The default window size is 100. If you notice that some directory servers are lagging behind in terms of replicated changes, increase the window size to a higher value and check replication performance again before making further adjustments.

▼ To Change the Window Size

1 (Optional) Display the current value of the window size:

```
$ dsconfig -D "cn=directory manager" -w password -n get-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced --property window-size
```

```
Property    : Value(s)
------------:---------
window-size : 100
```

2 Change the window size.

The following command changes the window size to 200.

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --set window-size:200
```

Changing the Heartbeat Interval

The heartbeat interval is a property of the replication domain and specifies the frequency with which the replication domain communicates with the replication server. The replication domain expects a regular heartbeat at this interval from the replication server. If the heartbeat is not received, the domain closes its connection and connects to another replication server in the topology.

The default heartbeat interval is ten seconds. If replication is running over a WAN or a network with slow response times, you might want to increase the heartbeat interval. In addition, if you observe an error similar to the following in the logs, it is probably necessary to increase the heartbeat interval.

```
[26/May/2009:16:32:50 +0200] category=SYNC severity=NOTICE msgID=15138913
 msg=Replication Heartbeat Monitor on RS rserver/192.157.197.62:8989 30382 for
 dc=example,dc=com in DS 10879 is closing the session because it could not
 detect a heartbeat
```

The heartbeat interval is sensitive to the settings of your JVM. If you require a lower heartbeat interval than the default, you must configure your JVM to have a low pause time during garbage collection by setting the `-XX:+UseConcMarkSweepGC` option. For more information, see "Configuring the JVM, Java Options, and Database Cache" in *Sun OpenDS Standard Edition 2.0 Installation Guide*.

▼ To Change the Heartbeat Interval

1 (Optional) Display the current value of the heartbeat interval.

```
$ dsconfig -D "cn=directory manager" -w password -n get-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 15853)" --advanced \
  --property heartbeat-interval
Property           : Value(s)
-------------------:---------
heartbeat-interval : 10 s
```

2 **Change the heartbeat interval.**

The following command changes the heartbeat interval to 5 seconds.

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 15853)" --set heartbeat-interval:5s
```

Changing the Isolation Policy

The isolation policy is a property of the replication domain and specifies the behavior of the directory server if replication is configured but none of the replication servers are up and running when an update is received. The default behavior of the directory server in this situation is to reject all updates.

▼ To Change the Isolation Policy

1 **(Optional) Display the current isolation policy.**

```
$ dsconfig -D "cn=directory manager" -w password get-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 15853)" \
  --advanced --property isolation-policy -n

Property         : Value(s)
-----------------:------------------
isolation-policy : reject-all-updates
```

2 **Change the isolation policy.**

The following command specifies that the directory server should accept all updates in this situation.

```
$ dsconfig -D "cn=directory manager" -w password set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 15853)" \
  --set isolation-policy:accept-all-updates -n
```

Configuring Encrypted Replication

By default, replication traffic is not encrypted. You can enable encryption by configuring the crypto manager.

▼ To Configure Encrypted Replication

- **Set the properties of the crypto manager.**

  The following command specifies that replication traffic should be encrypted.

```
$ dsconfig -D "cn=directory manager" -w password -n set-crypto-manager-prop \
  --set ssl-encryption:true
```

Configuring Replication Groups

Replication groups are designed to support multi-data center deployments and disaster recovery scenarios. For information about the design and implementation of replication groups in the directory server, see "Replication Groups" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

Note – Changing the replication group configuration has an impact on assured replication. For more information, see "Assured Replication" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

▼ To Configure A Replication Group

A replication group is configured on each directory server and replication server that should be part of the same group. On directory servers, a replication group is configured *per replicated domain*. On replication servers, the group is configured for the entire replication server.

Replication groups are configured by giving each replicated domain and replication server the same group ID. This example configures a replication group (1) for the replicated domain `dc=example,dc=com`.

1. **On each directory server that will be part of this group, set the group ID for the domain `dc=example,dc=com`.**

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --set group-id:1
```

2. **On each replication server that will be part of this group, set the group ID.**

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --set group-id:1
```

Configuring Assured Replication

In most deployment scenarios, the loosely consistent multi-master replication model is sufficient. However, certain scenarios might require tighter consistency between replicas. In such cases, you can configure *assured replication*, which provides the following benefits:

- **High availability of data.** If a server crashes immediately after a modification is received on that server, there is a risk that the modification will be lost before it is replayed to other servers in the topology. With assured replication, any modification is replayed to another server in the topology *before* an acknowledgement is sent to the client application. The risk of losing data in the event of a server crash is therefore minimized.
- **Immediacy of data availability.** Some applications might require modifications to be available on additional servers in the topology immediately after a modification is made.

Assured replication is an extension of the replication protocol and is configured *per replicated domain*. For more information, see "Retrieving the Replication Domain Name" on page 248.

Assured replication is not the same as *synchronous replication*. That is, changes do not occur simultaneously on all servers in the topology. However, assured replication can mimic the functionality of synchronous replication to an extent, as far as LDAP clients are concerned. This is achieved by delaying acknowledgements to the client application until a modification has been propagated to additional servers in the topology.

Note – Assured replication relies on *replication groups*. All replication servers and directory servers that function together in an assured replication configuration must be part of the same replication group.

Assured replication can function in two modes:

- **Safe data mode.** Any update must be propagated to a defined number of replication servers before the client receives an acknowledgement that the update has been successful.

  The number of replication servers that must be reached defines the *safe data level*. The higher the safe data level, the higher the overall data availability.
- **Safe read mode.** Any update must be propagated to all the directory servers in the topology before the client receives an acknowledgement that the update has been successful.

In both safe data mode and safe read mode, you can configure a timeout interval to prevent LDAP client calls from hanging if certain servers in the topology are not available.

- On each *directory server*, you can configure a global timeout that comes into effect when the directory server sends an update to its replication server, either safe data mode or safe read mode. If this timeout is reached, the LDAP client call returns immediately and a message is written to the replication log to track the event.

- On each *replication server*, you can configure a global timeout that comes into effect when the replication server sends an update to a peer replication server or to another directory server, either in safe data mode or in safe read mode. If this timeout is reached, the acknowledgement message that is returned to the initiating server (either a directory server or a replication server) includes a message that indicates the timeout. The initial directory server then logs a message that the timeout occurred for that update.

Note – The default timeout of two seconds for a directory server and one second for a replication server should be satisfactory for most deployments. *Only* change the timeout if you are viewing timeouts in the logs and if you have a complete understanding of the impact of such a change. The value of the timeout should reflect the anticipated time that an update requires to go through its full path to reach its destination.

The timeout value on a directory server should always be higher than the value on the replication server. For example: `DS1(timeout 2s) -> RS1(timeout 1s) -> RS2(timeout 1s) -> DS2`.

For a detailed explanation of the assured replication mechanism and the various configurable options, see "Assured Replication" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

▼ To Configure Assured Replication in Safe Data Mode

This procedure configures assured replication in safe data mode for a topology. The procedure assumes that replication has already been configured.

1 On each directory server in the topology:

a. Set the assured replication mode.

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --set assured-type:safe-data
```

b. Set the safe data level.

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --set assured-sd-level:2
```

If you have configured replication by using `setup` or `dsreplication`, your replication servers and directory servers will be on the same virtual machine. In this case, you must set the safe data level to 2 or higher.

c. (Optional) Set the assured replication timeout.

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced\
  --set assured-timeout:5s
```

Only change the timeout if you are viewing timeouts in the logs and if you have a complete understanding of the impact of such a change.

d. Verify the directory server group ID.

This should be the same for all replication servers and directory servers that form part of this replication group. For instructions on configuring the group ID, see "Configuring Replication Groups" on page 252.

e. (Optional) Display the current assured replication configuration.

```
$ dsconfig -D "cn=directory manager" -w password -n get-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --property assured-type --property assured-sd-level --property assured-timeout
Property         : Value(s)
-----------------:------------
assured-sd-level : 2
assured-timeout  : 5 s
assured-type     : safe-data
```

2 (Optional) On each replication server in the topology:

a. Display the current assured replication configuration.

```
$ dsconfig -D "cn=directory manager" -w password -n get-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --property assured-timeout --property group-id
Property                  : Value(s)
--------------------------:---------
assured-timeout           : 1 s
group-id                  : 1
```

b. Set the assured replication timeout.

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --set assured-timeout:5s
```

Only change the timeout if you are viewing timeouts in the logs and if you have a complete understanding of the impact of such a change.

c. Verify the replication server group ID.

This should be the same for all replication servers and directory servers that form part of this replication group. For instructions on configuring the group ID, see "Configuring Replication Groups" on page 252.

▼ To Configure Assured Replication in Safe Read Mode

Assured replication is configured *per replicated domain*. This procedure configures assured replication in safe read mode for a topology. The procedure assumes that replication has already been configured.

1 On each directory server in the topology:

a. Set the assured replication mode.

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --set assured-type:safe-read
```

b. (Optional) Set the assured replication timeout.

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --set assured-timeout:5s
```

Only change the timeout if you are viewing timeouts in the logs and if you have a complete understanding of the impact of such a change.

c. Verify the directory server group ID.

This should be the same for all replication servers and directory servers that form part of this replication group. For instructions on configuring the group ID, see "Configuring Replication Groups" on page 252. For more information about groups and assured replication, see "Assured Replication" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

d. (Optional) Display the current assured replication configuration.

```
$ dsconfig -D "cn=directory manager" -w password -n get-replication-domain-prop \
  --provider-name "Multimaster Synchronization" \
  --domain-name "dc=example,dc=com (domain 10233)" --advanced \
  --property assured-type --property assured-timeout --property group-id
Property         : Value(s)
-----------------:------------
assured-timeout  : 5 s
assured-type     : safe-read
group-id         : 1
```

2 **(Optional) On each replication server in the topology:**

a. Display the current assured replication configuration.

```
$ dsconfig -D "cn=directory manager" -w password -n get-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --property assured-timeout --property degraded-status-threshold \
  --property group-id
Property                  : Value(s)
--------------------------:---------
assured-timeout           : 1 s
degraded-status-threshold : 5000
group-id                  : 1
```

b. Set the assured replication timeout.

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --set assured-timeout:5s
```

Only change the timeout if you are viewing timeouts in the logs and if you have a complete understanding of the impact of such a change.

c. Set the degraded status threshold.

The degraded status threshold defines the stage at which the server is regarded as "too slow", based on the number of updates queued in the replication server for that directory server. For more information, see "Degraded Status" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

Do *not* adjust this value unless you observe timeouts in the logs.

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --advanced \
  --set degraded-status-threshold:2000
```

d. Verify the replication server group ID.

This should be the same for all replication servers and directory servers that form part of this replication group. For instructions on configuring the group ID, see "Configuring Replication Groups" on page 252. For more information about groups and assured replication, see "Assured Replication" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

Configuring Replication Status

Each replicated domain in a replicated topology has a certain *replication status*, depending on its connections within the topology, and on how up to date it is with regard to the changes that

have occurred throughout the topology. For more information, see "Replication Status" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

Replication status is generated automatically, based on how up to date a server is within the replicated topology. The only parameter that can be configured is the degraded status threshold. This parameter defines the maximum number of changes that can be in the replication server's queue for all domains of the directory servers that are connected to this replication server. When this number is reached, for a specific directory server, that server is assigned a degraded status. The degraded status remains until the number of changes drops beyond this value.

Note – The default value of the degraded status threshold should be adequate for most deployments. Only modify this value if you observe several timeout messages in the logs when assured replication is configured.

▼ To Configure the Degraded Status Threshold

The default number of changes defined by this threshold is 5000. This example sets the threshold to 6000, to take into account a network with more latency.

- **On the replication server, use `dsconfig` to set the degraded status threshold.**

```
$ dsconfig -D "cn=directory manager" -w password -n set-replication-server-prop \
  --provider-name "Multimaster Synchronization" --set degraded-status-threshold:6000
```

Initializing a Replicated Server With Data

This section describes how to initialize a replicated server with data by using the "dsreplication" in *Sun OpenDS Standard Edition 2.0 Command-Line Usage Guide* command. `dsreplication` accesses the server configuration over SSL via the administration connector. For more information, see "Managing Administration Traffic to the Server" on page 5.

This section references some of the information covered in "Populating a Stand-Alone Directory Server With Data" on page 95. It is worthwhile reading that section before you read this one.

Initializing a Single Replicated Server

The easiest way to initialize a single directory server in a replicated topology is to use the `dsreplication` command to copy the data over from another directory server in the topology. This command requires replication to have been enabled between the source server and the destination server. The command replaces all data under the specified base DN on the destination server with the data from the source server.

For example, the following command initializes the base DN "`dc=example,dc=com`" on `host2` with the data on `host1`.

```
$ dsreplication initialize --baseDN "dc=example,dc=com" \
  --adminUID admin --adminPassword password \
  --hostSource host1 --portSource 4444 \
  --hostDestination host2 --portDestination 4444 --trustAll
```

Initializing a New Replicated Topology

To initialize all directory servers in a new replicated topology, use one of the following options:

- Initialize all directory servers individually with the same data, using one of the methods described in "Populating a Stand-Alone Directory Server With Data" on page 95. When you have initialized all directory servers with data, enable replication between the servers.
- Initialize a single directory server using one of the methods described in "Populating a Stand-Alone Directory Server With Data" on page 95. Enable replication for all directory servers, then use the `dsreplication intialize-all` command to initialize all the remaining servers simultaneously. This command takes the details of the source server as arguments, and initializes all other servers for which replication is enabled.

  For example, the following command initializes all directory servers from the contents on `host1`.

  ```
  $ dsreplication initialize-all --hostname localhost --port 4444 --trustAll \
    --baseDN "dc=example,dc=com" --adminUID admin --adminPassword password
  ```

Adding a Directory Server to an Existing Replicated Topology

When you add a directory server to an existing replicated topology, the new server must be populated with the same *generation* of data as the existing directory servers in the topology. The data generation is an ID stored within the root entry of the replication domain. When the data generation does not exist, it is computed by the replication mechanism and stored. To ensure that the new directory server has the same data generation as the other servers in the topology, use one of the following methods to populate the directory server with data:

- Use the same original LDIF file, backup file, or binary copy that was used to populate the other directory servers.
- Use the result of an export, backup, or binary copy from another directory server in the topology.

If you install the new directory server using QuickSetup and specify that it will be part of the replicated topology, the server is initialized with the correct data generation automatically.

If you do not install the directory server using QuickSetup, and you use the `dsreplication` command to enable replication, you must initialize the server manually using one of the methods described in the previous section.

Note – If a directory server in the topology does not contain the same data generation as the rest of the topology, data cannot be replicated to or from the server. However, the directory server remains connected to the topology, enabling it to be initialized using the replication protocol. Replication on this directory server is said to be *downgraded*.

When a directory server with the correct data generation is added to an existing topology, the replication mechanism automatically replays any changes that occurred since the first directory server in the topology was initialized with data. This action ensures that the new directory server is synchronized with the rest of the topology.

Changing the Data Set in an Existing Replicated Topology

Changing the data set implies importing an entirely new set of data to every directory server in the topology. When the data set is changed, two tasks are performed:

- The new data is applied to each directory server in the topology.
- The replication servers are cleared of any changes they might contain. This task includes resetting the data generation on the directory servers so that the new data generation is used.

If you change the data set using the `dsreplication initialize` command, both of these tasks are performed automatically. However, if you use the `import-ldif` command or the binary copy method to change the data set, you must perform these tasks manually, as described in the following section.

▼ To Change the Data Set With `import-ldif` or Binary Copy

1 Clear the generation ID from the directory servers by running the `dsreplication pre-external-initialization` command.

It is sufficient to run this command on only one directory server in the topology. All directory servers in the topology will be updated, unless you specify that only one server should be updated. For example, the following command prepares all servers in the topology for initialization by using `import-ldif` or binary copy:

```
$ dsreplication pre-external-initialization -h host1 -p 4444 -X \
  -b dc=example,dc=com -I admin -w password

Are you going to initialize only the contents of server host1:4444 (type
'no' if you will initialize contents of all replicated servers for the given
Base DNs)? (yes / no) [no]:
Preparing base DN dc=example,dc=com to be initialized externally ..... Done.
Now you can proceed to the initialization of the contents of the base DNs on
```

```
all the replicated servers. You can use the command import-ldif or the binary
copy to do so. When the initialization is completed you must use the subcommand
{post-external-initialization} for replication to work with the new base DNs contents.
```

2 **Use `import-ldif` or binary copy to initialize all directory servers in the topology with data.**

3 **Reset the generation ID by running the `dsreplication post-external-initialization` command.**

It is sufficient to run this command on only one directory server in the topology. All other directory servers are updated. For example, the following command resets the generation ID for all directory servers in the topology after initialization using `import-ldif` or binary copy:

```
$ dsreplication post-external-initialization -h localhost \
  -p 4444 -b dc=example,dc=com -I admin -w password -X
Updating replication information on base DN dc=example,dc=com ..... Done.
Post initialization procedure completed successfully.
```

Configuring Schema Replication

Schema replication is enabled by default. When you configure replication as part of the server setup, the schema of the new server is automatically initialized with the schema of the existing server in the topology.

Specifying the Schema Source

When you configure replication with the `dsreplication enable` command, you can specify that the schema of the second directory server be used to initialize the schema of the first server. If you do not specify an option, the schema of the first directory server is used by default.

In the following example, the data of `host1` is used to initialize `host2` but the schema of `host2` is used to initialize the schema on `host1`:

```
$ dsreplication enable --host1 host1 --port1 4444 --bindDN1 "cn=Directory Manager" \
  --bindPassword1 password --replicationPort1 8989 --host2 host2 --port2 4444 \
  --bindDN2 "cn=Directory Manager" --bindPassword2 password --replicationPort2 8990 \
  --adminUID admin --adminPassword password --baseDN "dc=example,dc=com" \
  --useSecondServerAsSchemaSource -X
```

Disabling Schema Replication

In certain circumstances, you might not want the schema to be replicated. The schema is replicated under a separate base DN, "`cn=schema`".

To Specify That Schema Should Not Be Replicated

When you configure replication with the `dsreplication enable` command, you can specify that the schema should not be replicated, using the `--noSchemaReplication` option.

Note – If you use QuickSetup to enable replication, you cannot specify that the schema should not be replicated.

To Disable Schema Replication

In an existing topology in which the schema are being replicated, you can disable this functionality by disabling replication of the schema base DN. The following example disables schema replication from the directory server running on the local host on port `1389`:

```
$ dsreplication disable -h localhost -p 4444 -D "cn=directory manager" \
  -w password -b "cn=schema" -X
```

Note – The previous example does not disable schema replication for the entire topology. To disable schema replication for the entire topology, you must run the equivalent command for each directory server in the topology.

Replicating to a Read-Only Server

The directory server replication model is a multi-master model, that is, all the replication servers in the topology can process both read and write operations. However, you can configure a replication server to be read-only, in which case add, modify, and delete operations from LDAP clients are rejected on this server.

Note – A read-only replication server functions like a *consumer replica* does in the Sun Java System Directory Server replication model.

▼ To Configure a Replica as Read-Only

This example assumes a replication configuration with replication servers on two hosts, host1 and host2. The example makes the replication server on host2 a read-only replica. The example uses the `dsconfig` command, which accesses the server configuration via the administration connector. For more information, see "Managing Administration Traffic to the Server" on page 5.

- **Use the `dsconfig` command to set the `writability-mode` of host2.**

```
$ dsconfig -h host2 -p 4444 -D "cn=Directory Manager" -w password -X -n \
  set-global-configuration-prop --set writability-mode:internal-only
```

A writability mode of *internal-only* means that replication operations are processed on the server, but the server is not writeable directly by LDAP client applications.

Detecting and Resolving Replication Inconsistencies

Directory server replication has been designed to ensure that replicated databases remain consistent, even in the case of hardware faults, directory server restarts, or network failures. Despite these efforts, however, it is possible that hardware failures (disk errors, memory errors) or software errors (causing memory corruption) might lead to inconsistent databases.

These topics explain how to detect replication inconsistencies, and how to resolve them when they are identified.

Types of Replication Inconsistencies

When inconsistencies occur, they might remain hidden for some time or they might trigger replication or application errors. Examples of inconsistencies include the following:

- An entry is present on all but one directory server in the replication topology.
- An entry has a DN on one directory server that is different to its DN on all other directory servers.
- An entry has different attributes on one directory server than on other directory servers in the replication topology.

Detecting Inconsistencies

Use the following methods to check for replication inconsistencies:

- **Check for information in the replication log file**. The replication log file is configured by default and lists inconsistencies that are detected by the replication mechanism. Imagine, for example, that a modify operation is performed on an entry that is missing from one directory server in the topology. When replication attempts to replay this operation to that server, it will detect the problem and produce an error in the `logs/replication` error log. This kind of error will not stop replication, but the operation will not be replayed and the administrator will need to repair the inconsistency.
- **Pay attention to errors reported by client applications or users**. Client applications or users might experience errors when accessing the directory server that might be due to replication inconsistencies.
- **Make regular checks for database consistency**. With the current directory server release, these checks must be performed manually, using searches or database exports. A future directory server release is expected to provide tools to check databases for consistency.

Resolving Inconsistencies

If a replication inconsistency is found on a single directory server in the topology, it is not possible to fix this inconsistency using regular LDAP operations. This is because the LDAP operation itself would be replicated to the other directory servers in the topology and might cause damage on those servers. In addition, the fix might involve modifying attributes that are generated by the directory server, such as the `entryuuid` or `modifyTimestamp` attributes. Such attributes cannot be modified by regular LDAP operations.

Replication repair operations must therefore be done using LDAP operations that specify the Replication Repair Control (OID: `1.3.6.1.4.1.26027.1.5.2`).

Caution – Because the replication repair control allows you to skip several controls usually done by the directory server, it should be used with great care and only when consistency problems have been detected and asserted.

The repair control alters the regular processing of an operation as follows:

- The operation can modify attributes that might not normally be modified or added (`NO-USER-MODIFICATION`), such as `entryuuid` and `ds-sync-hist`.
- No replication change number is associated with the operation.
- The operation is not published to the replication server and is therefore a local-only operation.
- Replication does not try to resolve conflicts or to generate historical information for this operation.
- Most of the schema checks are not performed for this operation.

For example, the following `ldapmodify` operation repairs an entry on `host1` only, with the changes contained in the file `changes.ldif`:

```
$ ldapmodify -J 1.3.6.1.4.1.26027.1.5.2 -h localhost -p 1389 \
  -D "cn=Directory Manager" -w password -f changes.ldif
```

When you repair an entry, you must repair all of its regular attributes as well as the attributes generated by the directory server, such as `modifyTimestamp`, `modifiersName`, `createTimestamp`, `creatorsName`, and `ds-sync-hist`. The values of these attributes should be read from a directory server that contains the correct values, and recreated on the server with faulty values.

The `ds-sync-hist` attribute contains historical information that replication uses to solve modify conflicts. This attribute can only be viewed by an administrator.

Managing Users and Groups

The directory server provides a comprehensive user management model that includes password policies, identity mapping, and account status notification. This section describes how to configure these elements by using the `dsconfig` command.

The section covers the following topics:

Managing Root User, Global Administrator, and Administrator Accounts

The directory server provides a flexible Privilege Subsystem that allows you to configure root users, Global Administrators, and administrators for your directory. You can configure multiple root users and assign different root privileges to each administrator. For administrative domains, you can also configure multiple Global Administrators to manage administrative domains in your network or in a replicated environment.

The topics in this section describe the management of multiple root users and the privilege subsystem. The topics also provide instructions on how to configure and maintain the various user accounts required to administer your directory securely.

Before you start with the procedures outlined here, determine the following guidelines for your directory:

- Number of root users, their privileges, and resource limits if any
- Number of administrators, their privileges, and resource limits if any
- Guidelines for user accounts on your system
- Password policies for your directory server and for specific groups of users

Working With Multiple Root Users

The directory server provides one default root DN or root user, "cn=Directory Manager". The default root DN is a user entry assigned with specialized privileges with full read and write access to all data in the directory server. Comparable to a Unix root user or superuser, the root DN can bypass access controls to carry out tasks on the directory server. The root user is defined below the "cn=Root DNs,cn=config" branch of the server at cn=Directory Manager,cn=Root DNs,cn=config.

The directory server supports multiple root users who have their own entries and their own set of credentials on the directory. This allows you to assign privileges to a user who might need root access for a particular task but might not need the full set of root user privileges. With each entry, you can assign strong authentication such as the GSSAPI SASL mechanism, password policies, or add resource limits (if your schema allows it) to one root user while having a completely different configuration for another root user.

Root users differ from regular user entries in the following ways:

- **Configuration**. Root users are the only user accounts that can exist in the server configuration (cn=config).
- **Privilege inheritance**. Root users automatically inherit the set of default root user privileges. Regular users do not automatically receive any privileges unless explicitly granted. You can grant privileges using real, virtual root-privilege-name attributes, or both in the entry.
- **Lockdown mode**. Root users are the *only* users who can cause the server to enter or leave lockdown mode and only over the loopback interface.

The Privilege Subsystem supports the configuration of multiple root users.

Root Users and the Privilege Subsystem

The Privilege Subsystem allows you to assign refined privileges to users who might require only a specific set of root user access privileges. Root users are automatically granted a set of privileges defined in the default-root-privilege-name attribute in the "cn=Root DNs,cn=config" subtree.

The Privilege Subsystem is independent from the Access Control Subsystem, but some operations might be subject to access controls.

The following set of privileges are automatically assigned to the root user.

Privilege	Description
`bypass-acl`	Allows the user to bypass access control evaluation.
`modify-acl`	Allows the user to make changes to access control instructions defined in the directory server.
`config-read`	Allows the user to have read access to the server configuration.
`config-write`	Allows the user to have write access to the server configuration.
`ldif-import`	Allows the user to request the LDIF import task.
`ldif-export`	Allows the user to request the LDIF export task.
`backend-backup`	Allows the user to request the back-end backup task.
`backend-restore`	Allows the user to request the back-end restore task.
`server-shutdown`	Allows the user to request the server shutdown task.
`server-restart`	Allows the user to request the server restart task.
`disconnect-client`	Allows the user to terminate arbitrary client connections.
`cancel-request`	Allows the user to cancel arbitrary client requests.
`unindexed-search`	Allows the user to request unindexed search operations.
`password-reset`	Allows the user to reset the user passwords.
`update-schema`	Allows the user to update the directory server schema.
`privilege-change`	Allows the user to change the set of privileges assigned to a user, or to change the set of default root privileges.

The following privileges can be assigned to the root user.

Privilege	Description
`data-sync`	Allows the user to participate in data synchronization environment.
`jmx-read`	Allows the user to read JMX attribute values.
`jmx-write`	Allows the user to update JMX attribute values.
`jmx-notify`	Allows the user to subscribe to JMX notifications.

Privilege	Description
proxied-auth	Allows the user to use the proxied authorization control or to request an alternate SASL authorization ID.

Managing Root Users With dsconfig

Use the dsconfig command to manage root users. For more information, see "Configuring the Directory Server With dsconfig" on page 7.

▼ To View the Default Root User Privileges

The default root user has a number of privileges, which are stored as values of the default-root-privilege-name property.

- **View the default root user privileges by running the following dsconfig command:**

```
$ dsconfig -D "cn=directory manager" -w password -n \
  get-root-dn-prop
Property                    : Value(s)
----------------------------:-------------------------------------------------
default-root-privilege-name : backend-backup, backend-restore, bypass-acl,
                                     : cancel-request, config-read, config-write,
                                     : disconnect-client, ldif-export, ldif-import,
                                     : modify-acl, password-reset, privilege-change,
                                     : server-restart, server-shutdown,
                                     : unindexed-search, update-schema
```

▼ To Edit the Default Root User Privileges

The easiest way to manage root user privileges is to use dsconfig in interactive mode. Interactive mode walks you through the root user configuration, and is therefore not documented here.

To add or remove privileges for the default root user, add or remove the values of the default-root-privilege-name property. This property can hold the following values:

- backend-backup
- backend-restore
- bypass-acl
- cancel-request
- config-read
- config-write
- data-sync
- disconnect-client
- jmx-notify

- jmx-read
- jmx-write
- ldif-export
- ldif-import
- modify-acl
- password-reset
- privilege-change
- proxied-auth
- server-restart
- server-shutdown
- unindexed-search
- update-schema

This example adds the data-sync privilege to the default root user, by using dsconfig in non-interactive mode.

- **Run the dsconfig command as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n \
  set-root-dn-prop --add default-root-privilege-name:data-sync
```

To Create a Root User

Root users are stored below the entry cn=Root DNs,cn=config. To create a new root user, create the entry in LDIF and add it by using the ldapmodify command.

Root users automatically inherit the set of default root user privileges on the server. For information about adding or removing privileges for a specific root user, see "To Change a Root User's Privileges" on page 270.

1. **Create a root user entry below the cn=Root DNs,cn=config entry.**

   The following LDIF file represents a new root user named "Administration Manager". The entry is saved in a file named add-root-user.ldif.

```
dn: cn=MyRootUser,cn=Root DNs,cn=config
objectClass: inetOrgPerson
objectClass: person
objectClass: top
objectClass: ds-cfg-root-dn-user
objectClass: organizationalPerson
userPassword: password
cn: MyRootUser
sn: MyRootUser
ds-cfg-alternate-bind-dn: cn=MyRootUser
givenName: Directory
```

2. **Use the `ldapmodify` command to add the entry.**

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --defaultAdd --filename "add-root-user.ldif"
Processing ADD request for cn=MyRootUser,cn=Root DNs,cn=config
ADD operation successful for DN cn=MyRootUser,cn=Root DNs,cn=config
```

3. **(Optional) Use the `ldapsearch` command to display all the root users defined in the server.**

```
$ ldapsearch -p 1389 -b "cn=root DNs,cn=config" -D "cn=directory manager" -w password \
  "objectclass=*" dn
dn: cn=Root DNs,cn=config

dn: cn=MyRootUser,cn=Root DNs,cn=config

dn: cn=Directory Manager,cn=Root DNs,cn=config
```

▼ To Change a Root User's Password

1. **Create a password in a secure file.**

2. **Use `ldappasswordmodify` to change the password.**

```
$ ldappasswordmodify -h localhost -p 1389 -D "cn=MyRootUser" -w password \
  --newPasswordFile rootuser_pwd.txt
The LDAP password modify operation was successful
```

▼ To Change a Root User's Privileges

If you want to have a different set of privileges for a specific root user, add the `ds-privilege-name` attribute to that root user's entry.

The following example gives the root user "`cn=MyRootUser,cn=Root DNs,cn=config`" the ability to use proxied authorization. The example removes the ability to change user privileges or access the configuration. The minus sign before the privilege indicates that the privilege is being removed rather than granted.

- **Apply the following LDIF statement to the root user's entry:**

```
dn: cn=MyRootUser,cn=Root DNs,cn=config
changetype: modify
add: ds-privilege-name
ds-privilege-name: proxied-auth
ds-privilege-name: -config-read
ds-privilege-name: -config-write
```

In this example, the root user "`cn=MyRootUser,cn=Root DNs,cn=config`" would inherit all privileges automatically granted to root users with the exception of the `config-read` and `config-write` privileges. The user would also be given the `proxied-auth` privilege.

Setting Root User Resource Limits

You can set resource limits on the server for search operations by using the operational attributes on the client application that is binding to the directory. The following resource limits are available:

- **Look-through limit**. Specify the maximum number of entries that can be examined during a single search operation. Use the `ds-rlim-lookthrough-limit` operational attribute.
- **Size limit**. Specify the maximum number of entries that can be returned in a single search operation. Use the `ds-rlim-size-limit` operational attribute.
- **Time limit**. Specify the maximum length of time in seconds that the server can spend processing a search operation. Use the `ds-rlim-time-limit` operational attribute.

The following LDIF update statement sets resource limits for the new root user created in the previous section. This statement should be applied to the root user's entry.

```
dn: cn=MyRootUser,cn=Root DNs,cn=config
changetype: modify
add: ds-rlim-lookthrough-limit
ds-rlim-lookthrough-limit: 1000
-
add: ds-rlim-size-limit
ds-rlim-size-limit: 500
-
add: ds-rlim-time-limit
ds-rlim-time-limit: 300
```

Managing Global Administrators

When setting up replication servers using the graphical installer or the `dsreplication` command, you are prompted to set a user name and password for the Global Administrator. The Global Administrator is responsible for managing and maintaining administrative server domains in replicated environments.

The Global Administrator exists in the `cn=Administrators,cn=admin data` subtree. To view the Global Administrator entry, run the following `ldapsearch` command:

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b "cn=Administrators,cn=admin data" -s sub "(objectclass=*)"
dn: cn=Administrators,cn=admin data
objectClass: top
objectClass: groupofurls
description: Group of identities which have full access.
cn: Administrators
memberURL: ldap:///cn=Administrators,cn=admin data??one?(objectclass=*)
```

```
dn: cn=admin,cn=Administrators,cn=admin data
objectClass: person
objectClass: top
userPassword: {SSHA}+ed1wbhcWjxtv2zJ6OHEA2TuE9n1qIJGnuR94w==
description: The Administrator that can manage all the OpenDS instances.
cn: admin
sn: admin
```

Managing Administrators

If you prefer to retain the single root user and one or more administrators, you can create a new administration user with the same rights as a root user. Because only the root user can exist in the configuration, you must set up the administrators entries in the `userRoot` back end. You can then add the root user privileges using the `root-privilege-name` attribute, and add the relevant access controls to the entry.

▼ To Create a New Administrator

1. **Set up the administrator in the `cn=Administrators` group under `ou=People`.**
2. **Add the administrator by using the `ldapmodify` command.**
3. **Add root user privileges using the `privilege-name` attribute.**
4. **Add access controls to the entry.**
5. **Add resource limits, if required.**

Managing Password Policies

A *password policy* is a set of rules governing the use of passwords in the system and is an integral component of any security strategy employed for your directory.

The directory server includes a default password policy for general users and a default root users password policy. These default password policies reside in the directory server's configuration and can be modified. Because the directory server supports multiple password policies, you can also create and configure specialized password policies for a specific set of users in addition to using the default password policies.

This section outlines the components of password policies and provides procedures to configure and manage password policies.

Password Policy Components

All password policies involve the following configurable components:

- **Password complexity requirements**. Specifies the composition of the password and its required number of characters. Typically, you would specify the minimum number of characters used in a password, the type of characters allowed, and the required number of numeric characters. For example, many institutions require a minimum of seven or eight characters, one numeral, one special character, as well as a mix of uppercase and lowercase letters.
- **Password history**. Determines the number of unique passwords a user must use before an old password can be reused.
- **Maximum password age**. Determines how long a password can be used before the user is allowed or required to change it.
- **Minimum password age**. Determines how long a new password must be kept before the user can change it.
- **First Login**. Determines if the user will be required to change his password upon first logging in to the system.
- **Authorized password change**. Refers to the conditions under which a user can change his password. For example, before a user can change his password, the server can be configured to require the user to enter his current password to authenticate his identity before entering a new password.
- **Account lockout**. Determines the conditions under which an account is disabled for access by the user. For example, if a user fails to properly authenticate after three attempts, then the server can be configured to lock the account on the fourth attempt. The administrator will be required to manually unlock the account for user.
- **Password storage scheme**. Determines how the password is to be encrypted and stored on the server. You can configure storage schemes for certain accounts on the server. For example, root user passwords require strong encryption due to the importance of the account and its privileges. Thus, you can configure the use the SSHA-512 storage scheme to store root user passwords.

Password Policies in a Replicated Environment

All password policies reside in the directory server configuration (under `cn=config`). Configuration information is not replicated and is specific to each directory server instance. If you modify the default password policy, you must make the same changes on each directory server instance in a replicated topology. Similarly, specialized password policies are not replicated to other directory servers.

Additional considerations for using password policies in replicated environments include the following:

- The directory server replicates all password information (current password, password history, password expiration) that is stored in the user entry.
- If a user changes his password, the new password might take a while to be updated on all replicas.
- A user might receive multiple password expiration warnings, one from each replicated server.

▼ To View the List of Password Policies

You can view the list of password policies by using the `dsconfig` command. `dsconfig` accesses the server over SSL via the administration connector. For more information, see "Configuring the Directory Server With `dsconfig`" on page 7.

The easiest way to manage password policies is by using `dsconfig` in interactive mode. Interactive mode functions much like a wizard and walks you through each step of the password policy configuration. To use `dsconfig` in interactive mode, run the command without any options. Because the interactive mode is self-explanatory, the examples provided here do not describe interactive mode, but provide the full equivalent command.

- **Run the following `dsconfig` command to view the list of password policies:**

```
$ dsconfig -D "cn=directory manager" -w password -n list-password-policies
Password Policy         : Type    : password-attribute : default-password-
                                                                 storage-scheme
------------------------:---------:--------------------:------------------
Default Password Policy : generic : userpassword       : Salted SHA-1
Root Password Policy    : generic : userpassword       : Salted SHA-512
```

Properties of the Default Password Policy

The Default Password Policy includes a number of configurable properties. These are listed in the following table. For more information, see Password Policy Configuration.

Property	Description
`account-status-notification-handler`	The account status notification handler is used to send messages when events occur during the course of password policy processing. This property specifies the DNs of the account status notification handlers that should be used for this password policy.

Property	Description
allow-expired-password-changes	Not Recommended. Indicates whether users are allowed to change their passwords after the passwords have expired. The user needs to issue the request anonymously and include the current password in the request. If this property is enabled, this feature uses the Password Modify Extended Operation, which is enabled by default at initial configuration.
allow-user-password-changes	Indicates whether users are allowed to change their own passwords if they have access control rights to do so.
default-password-storage-scheme	Specifies the DNs for the password storage schemes that are used to encode clear-text passwords for this password policy.
deprecated-password-storage-scheme	Specifies the DNs for password storage schemes that are considered deprecated for this password policy. If a user with this password policy authenticates to the server and his password is encoded with any deprecated schemes, those values are removed and replaced with values encoded using the default password storage scheme.
expire-password-without-warning	Indicates whether user passwords are allowed to expire even if the user has not yet seen a password expiration warning. If this is set to false, the user is always guaranteed to see at least one warning message even if the password expiration time has passed. The expiration time will be reset to the current time plus the warning interval (ds-cfg-password-expiration-warning-interval).
force-change-on-add	Indicates whether users are required to change their passwords the first time they use their accounts and before they are allowed to perform any other operation.
force-change-on-reset	Indicates whether users are required to change their passwords after an administrative password reset and before they are allowed to perform any other operation.
grace-login-count	Specifies the maximum number of grace logins that a user should be given. A grace login makes it possible for a user to authenticate to the server even after the password has expired, but the user is not allowed to do anything else until he has changed his password.

Property	Description
idle-lockout-interval	Specifies the maximum length of time that a user account can remain idle (that is, that the user may go without authenticating to the directory) before the server locks the account. This action is enforced if last login time tracking is enabled and if the idle lockout interval is set to a nonzero value.
last-login-time-attribute	Specifies the name of the attribute in the user's entry that is used to hold the last login time for the user. If this is provided, the specified attribute must either be defined as an operational attribute in the server schema, or it must be allowed by at least one of the object classes in the user's entry. The ds-pwp-last-login operational attribute has been defined for this purpose. Last login time tracking is only enabled if the ds-cfg-last-login-time-attribute and ds-cfg-last-login-time-format attributes have been configured for the password policy.
last-login-time-format	Specifies the format string that should be used to generate the last login time values. This can be any valid format string that can be used in conjunction with the java.text.SimpleDateFormat class. Note that for performance reasons, it might be desirable to configure this attribute so that it only stores the date (format: yyyyMMdd) and not the time of the last login. Then, it only needs to be updated once per day, rather than each time the user may authenticate. Last login time tracking is only enabled if the ds-cfg-last-login-time-attribute and ds-cfg-last-login-time-format attributes have been configured for the password policy.
lockout-duration	Specifies the length of time that a user account should remain locked due to failed authentication attempts before it is automatically unlocked. A value of "0 seconds" indicates that any locked accounts are not automatically unlocked and must be reset by an administrator.
lockout-failure-count	Specifies the number of authentication failures required to lock a user account, either temporarily or permanently. A value of zero indicates that automatic lockout is not enabled.

Property	Description
`lockout-failure-expiration-interval`	Specifies the maximum length of time that a previously failed authentication attempt should be counted toward a lockout failure. Note that the record of all previous failed attempts is always cleared upon a successful authentication. A value of "`0 seconds`" indicates that failed attempts are never automatically expired.
`max-password-age`	Specifies the maximum length of time that a user is allowed to keep the same password before choosing a new one. This is often known as the *password expiration interval*. A value of "`0 seconds`" indicates that passwords never expire. If the `ds-cfg-expire-passwords-without-warning` attribute is set to `false`, the effective password expiration time is recalculated to be the time at which the first warning is received, plus the warning interval (`ds-cfg-password-expiration-warning-interval`). This behavior ensures that a user always has the full configured warning interval to change his password.
`max-password-reset-age`	Specifies the maximum length of time that users are allowed to change their passwords after they have been administratively reset and before they are locked out. This is only applicable if the `ds-cfg-force-change-on-reset` attribute is set to `true`. A value of "`0 seconds`" indicates that there are no limits on the length of time that users have to change their passwords after administrative resets.
`min-password-age`	Specifies the minimum length of time that a user is required to have a password value before it can be changed again. Providing a nonzero value ensures that users are not allowed to repeatedly change their passwords in order to flush their previous password from the history so it can be reused.
`password-attribute`	Specifies the attribute in the user's entry that holds the encoded passwords for the user. The specified attribute must be defined in the server schema, and it must have either the user password syntax or the authentication password syntax. Typically, you enter "userPassword" for the User Password syntax (OID: `1.3.6.1.4.1.26027.1.3.1`), which has been in use for Netscape, iPlanet, Sun ONE, or Sun Java Directory server. You can also specify, if your server supports it, the value `authPassword` for the authenticated password syntax (OID: `1.3.6.1.4.1.4203.1.1.2`).

Property	Description
password-change-requires-current-password	Indicates whether users are required to provide their current password when setting a new password. If this is set to true, then users are required to provide their current password when changing their existing password. This may be done using the password modify extended operation, or using a standard LDAP modify operation by deleting the existing password value and adding the new password value in the same modify operation.
password-expiration-warning-interval	Specifies the length of time before the password expires that the users should start to receive notification that it is about to expire. This must be given a nonzero value if the ds-cfg-expire-passwords-without-warning attribute is set to false.
password-generator	Specifies the DN for the password generator that should be used in conjunction with this password policy. The password generator is used in conjunction with the password modify extended operation to provide a new password for cases in which the client did not include one in the request. If no password generator DN is specified, then the password modify extended operation does not automatically generate passwords for users.

Property	Description
password-history-count	Specifies the maximum number of password values that should be maintained in the password history. Whenever a user's password is changed, the server checks the proposed new password against the current password and all passwords stored in the history. If a match is found, then the user is not allowed to use that new password. A value of zero indicates either that the server should not maintain a password history (that is, the password history duration has a value of "0 seconds") or that the password history list should be based entirely on duration and no maximum count should be enforced (that is, the password history duration has a value other than "0 seconds"). Note that if an administrator reduces the configured password history count to a smaller (but still nonzero) value, each user entry containing password history state information is not impacted until a password change is processed for that user. At that time, any excess history state values is purged from the entry. If the history count is reduced to zero and the password history duration is also set to "0 seconds," any state information in the user's entry is retained in case the feature is re-enabled.
password-history-duration	Specifies the maximum length of time that a formerly used password should remain in effect in the user's password history. Whenever a user's password is changed, the server checks the proposed new password against the current password and all passwords stored in the history. If a match is found, the user is not allowed to use that new password. A value of "0 seconds" indicates either that the server should not maintain a password history (that is, the password history count has a value of "0") or that the password history list should be based entirely on count and no maximum duration should be enforced (that is, the password history count has a value other than "0").
password-validator	Specifies the DNs for password validators that should be used in conjunction with this password policy. The password validators are invoked whenever a user attempts to provide a new password in order to determine whether that new password is acceptable.

Property	Description
previous-last-login-time-format	Specifies the format string that was used in the past for older last login time values. This value is not necessary unless the last login time feature is enabled and the format in which the values are stored has been changed.
require-change-by-time	Specifies a time by which all users with this password policy are required to change their passwords. This option works independently of password expiration (that is, force all users to change their passwords at some point even if password expiration is disabled).
require-secure-authentication	Indicates whether users with this password policy are required to authenticate in a secure manner using a secure communication mechanism like SSL, or a secure SASL mechanism like DIGEST-MD5, EXTERNAL, or GSSAPI that does not expose the password in the clear.
require-secure-password-changes	Indicates whether users with this password policy are required to make password changes in a secure manner, such as over a secure communication channel like SSL.

▼ To View the Properties of the Default Password Policy

- **Run the following `dsconfig` command to view the properties of the default password policy.**

```
$ dsconfig -D "cn=directory manager" -w password -n get-password-policy-prop \
  --policy-name "Default Password Policy"
Property                                : Value(s)
----------------------------------------:-------------------------
account-status-notification-handler     : -
allow-expired-password-changes          : false
allow-user-password-changes             : true
default-password-storage-scheme         : Salted SHA-1
deprecated-password-storage-scheme      : -
expire-passwords-without-warning        : false
force-change-on-add                     : false
force-change-on-reset                   : false
grace-login-count                       : 0
idle-lockout-interval                   : 0 s
last-login-time-attribute               : -
last-login-time-format                  : -
lockout-duration                        : 0 s
lockout-failure-count                   : 0
lockout-failure-expiration-interval     : 0 s
max-password-age                        : 0 s
```

```
max-password-reset-age                    : 0 s
min-password-age                          : 0 s
password-attribute                        : userpassword
password-change-requires-current-password : false
password-expiration-warning-interval      : 5 d
password-generator                        : Random Password Generator
password-history-count                    : 0
password-history-duration                 : 0 s
password-validator                        : -
previous-last-login-time-format           : -
require-change-by-time                    : -
require-secure-authentication             : false
require-secure-password-changes           : false
```

Configuring Password Policies

The easiest way to configure a password policy is to by using the `dsconfig` command to set the password policy properties. The following examples configure various properties of the default password policy.

For a complete list of password policy configuration properties and their values, see the Password Policy Configuration.

EXAMPLE 15 Configuring Account Lockout

The following account lockout features can be configured:

- **Lockout failure count.** Specifies the number of authentication failures required to lock a user account.
- **Lockout duration.** Determines the length of time that the account is in a locked state after failed authentication attempts. After the duration time, the account is automatically unlocked. A value of zero indicates that the account is not be automatically unlocked.
- **Lockout failure expiration interval.** Determines the maximum length of time that a previously failed authentication attempt should be counted toward a lockout failure. A value of zero indicates that failed attempts never automatically expire.
- **Idle lockout interval.** Specifies the maximum length of time that a user account can go without authenticating to the directory before the server locks the account. This property is enforced if the `last-login-time` is enabled and `idle-lockout-interval` is set to a nonzero value.

The following command sets the account lockout properties for the default password policy.

```
$ dsconfig -D "cn=directory manager" -w password -n set-password-policy-prop \
  --policy-name "Default Password Policy" --set "lockout-failure-count:3" \
  --set "lockout-duration:15 minutes" --set "idle-lockout-interval:90 days" \
```

EXAMPLE 15 Configuring Account Lockout *(Continued)*

```
  --set "lockout-failure-expiration-interval:10 minutes"
```

EXAMPLE 16 Configuring Last Login

Last login is a basic security feature that helps the user to keep track of the login history. The directory server provides an operational attribute, `ds-pwp-last-login`, that holds the user's last login time. If you specify another attribute, the operational attribute must be defined in the server schema, or it must be allowed by at least one of the object classes in the user's entry.

The `last-login-time-format` property determines the time format. If the time format has changed and last login is enabled, the `previous-last-login-time-format` property is used.

The following command sets the last login properties for the default password policy.

```
$ dsconfig -D "cn=directory manager" -w password -n set-password-policy-prop \
  --policy-name "Default Password Policy" \
  --set "last-login-time-attribute:ds-pwp-last-login-time" \
  --set "last-login-time-format:yyyyMMdd" \
  --set "previous-last-login-time-format:yyyyMMdd"
```

EXAMPLE 17 Configuring Password History Count and Duration

The `password-history-count` property specifies the number of past passwords that should be maintained in the history. A value of zero indicates that the server does not maintain a password history.

The `password-history-duration` property specifies the maximum length of time that a previously used password should remain in the user's password history. A value of `0 seconds` indicates that the server should not maintain a password history.

The following command configures password history count and duration for the default password policy.

```
$ dsconfig -D "cn=directory manager" -w password -n set-password-policy-prop \
  --policy-name "Default Password Policy" --set "password-history-count:3" \
  --set "password-history-duration:5 seconds"
```

▼ To Create a New Password Policy

You can configure and store multiple password policies with different configuration options. When you set up a directory server instance, the instance uses the default password policy and applies it to all user entries, except root users (for example, the `cn=Directory Manager` account).

You can change the default password policy or you can create new password policies for specific groups in your directory. If a specific property is not present in a password policy, the server reads that property from the default password policy, in other words, all password policies inherit their default values from the default password policy.

The following command creates a new password policy and sets the `default-password-storage-scheme`, `lockout-duration`, `lockout-failure-count`, and `password-change-requires-current-password` properties. The remaining properties are inherited from the default Password Policy.

- **Use the `dsconfig` command to create a new password policy, as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n create-password-policy \
  --policy-name "Temp Password Policy" --set password-attribute:userPassword \
  --set default-password-storage-scheme:"Salted SHA-1" \
  --set lockout-duration:300s --set lockout-failure-count:3 \
  --set password-change-requires-current-password:true
```

To Create a First Login Password Policy

The First Login Password Policy is a specialized password policy that requires a user to change his password when first logging in to the system. Typically, an administrator sets up a new temporary password for newly created accounts, and the user is required to create his password after first logging in with the temporary password.

- **Use the `dsconfig` command to create a first login password policy.**

```
$ dsconfig -D "cn=directory manager" -w password -n create-password-policy \
  --policy-name "First Login Password Policy" --set password-attribute:userpassword \
  --set default-password-storage-scheme:"Salted SHA-1" \
  --set allow-user-password-changes:true --set force-change-on-add:true \
  --set force-change-on-reset:true --set expire-password-without-expiration:false \
  --set password-expiration-warning-interval:86400 \
  --set min-password-age:0 --set max-password-age:259200 --set lockout-duration:3600 \
  --set lockout-failure-count:3 --set password-change-requires-current-password:true
```

To Assign a Password Policy to an Individual Account

You can assign a password policy to an individual by adding the `ds-pwp-password-policy-dn` attribute to the user's entry. The server then uses the configured password policy for that user.

1. **Use `ldapmodify` to add the `ds-pwp-password-policy-dn` attribute.**

```
$ ldapmodify --h localhost -p 1389 -D "cn=Directory Manager" -w password \
dn: uid=mgarcia,ou=Contractors,dc=example,dc=com
changetype: modify
add: ds-pwp-password-policy-dn
ds-pwp-password-policy-dn: cn=Temp Password Policy,cn=Password Policies,cn=config
```

2 **Verify the entry by using `ldapsearch`.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b "dc=example,dc=com" -s sub "(uid=mgarcia)" ds-pwp-password-policy-dn
```

To Prevent Password Policy Modifications

To prevent users from modifying their password policy, you must add an ACI to the root entry.

- **Use the `ldapmodify` command with the specific ACI.**

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr != "passwordPolicySubentry")(version 3.0; acl "Allow self
modification except for passwordPolicySubentry";
allow (write) (userdn = "ldap:///self");)
```

To Assign a Password Policy to a Group of Users

You can assign a password policy to a group of users by adding a virtual attribute that automatically assigns the `ds-pwp-password-policy-dn` attribute to all user entries that match the criteria associated with that virtual attribute. The criteria can be based entirely or in part on the group membership for a user.

- **Use `dsconfig` to create a virtual attribute that adds a password policy to a group of users.**

```
$ dsconfig -D "cn=directory manager" -w password -n create-virtual-attribute \
  --name "Add PWPolicy to Admins" --type user-defined --set enabled:true \
  --set attribute-type:ds-pwp-password-policy-dn \
  --set group-dn:cn=Admins,ou=Groups,dc=example,dc=com \
  --set conflict-behavior:real-overrides-virtual \
  --set value:"cn=Admins PWPolicy,cn=Password Policies,cn=config"
```

To Delete a Password Policy

You can delete any password policy, except the Default Password Policy and the Default Root User Policy, from the directory when it is no longer needed.

In practice, first check the users who have the password policy you plan to delete, move them to a new password policy, and then remove the old password policy. If a password policy is deleted, any users who have a deleted password policy continue to have the `ds-pwd-password-policy-dn` pointing to the old password policy. The server returns an error when any requests to access the entry occur.

- **Use `dsconfig` to delete a password policy.**

```
$ dsconfig -D "cn=directory manager" -w password -n \
  delete-password-policy --policy-name "Temp Password Policy"
```

Managing User Accounts

User accounts are essentially user entries that you create, modify, or remove in your directory. The directory server provides easy-to-use utilities to manage user accounts and passwords.

Before you begin to manage user accounts, ensure that you have the appropriate password policies set up on the directory server.

Changing Passwords

Directory administrators are often asked to create, reset, or remove passwords for other users. The `ldappasswordmodify` utility enables you to change or reset a user's password with the LDAP password modify extended operation. You can specify authorization IDs with the `--authzid` option by prefixing `dn:`, `u:`, or by specifying the full DN.

▼ To Change the Directory Manager's Password

- **Use the `ldappasswordmodify` command, as shown in the following example:**

```
$ ldappasswordmodify -h localhost -p 1389 \
  --authzID "dn:cn=Directory Manager" \
  --currentPassword mypassword --newPassword mynewpassword
The LDAP password modify operation was successful
```

▼ To Reset and Generate a New Password for a User

This example assumes that the user does not remember his/her existing password.

- **Use the `ldappasswordmodify` command, as shown in the following example:**

```
$ ldappasswordmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --authzID u:jvedder
The LDAP password modify operation was successful
Generated Password:  evx07npv
```

▼ To Change a User's Password

This example assumes that the user remembers his/her existing password. The new password is passed to the server in a specified file.

- **Use the `ldappasswordmodify` command, as shown in the following example:**

```
$ ldappasswordmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --authzID uid=jvedder,ou=People,dc=example,dc=com \
  --currentPassword password --newPasswordFile pwdFile
The LDAP password modify operation was successful
```

Managing a User's Account Information

You can use the `manage-account` command to display information about the user's account and any password policy that is applied to the user. You can also use this command to enable and disable a user's account. The `manage-account` command accesses the server over SSL via the administration port. For more information, see "Managing Administration Traffic to the Server" on page 5.

▼ To View a User's Account Information

The `manage-account` command returns the DN of the password policy in effect on a user account, as well as the account status, and password and login related information

1 To display all available information on a user account, use the `manage-account` command with the `get-all` subcommand, as shown in the following example:

```
$ manage-account -D "cn=directory manager" -w password get-all \
  --targetDN uid=kvaughan,ou=People,dc=example,dc=com
Password Policy DN:  cn=Default Password Policy,cn=Password Policies,cn=config
Account Is Disabled:  false
Account Expiration Time:
Seconds Until Account Expiration:
Password Changed Time:  19700101000000.000Z
Password Expiration Warned Time:
Seconds Until Password Expiration:  432000
Seconds Until Password Expiration Warning:  0
Authentication Failure Times:
Seconds Until Authentication Failure Unlock:
Remaining Authentication Failure Count:
Last Login Time:
Seconds Until Idle Account Lockout:
Password Is Reset:  false
Seconds Until Password Reset Lockout:
Grace Login Use Times:
Remaining Grace Login Count:  4
Password Changed by Required Time:
Seconds Until Required Change Time:
Password History:
```

2 To display just a single property of the account, substitute the `get-all` subcommand with the subcommand corresponding to the property you want to view.

For example, to view just the password history, run the following command:

```
$ manage-account -D "cn=directory manager" -w password get-password-history \
  --targetDN "uid=kvaughan,ou=People,dc=example,dc=com"
```

For a complete list of subcommands, run the following command:

```
$ manage-account --help
```

▼ To View Account Status Information

You can use the `manage-account` command to assess whether an account is enabled or disabled.

- **Use the `manage-account` command with the `get-account-is-disabled` subcommand, as shown in the following example:**

```
$ manage-account -D "cn=directory manager" -w password get-account-is-disabled \
  --targetDN "uid=kvaughan,ou=People,dc=example,dc=com"
Account Is Disabled:  false
```

▼ To Disable an Account

- **Use the `manage-account` command with the `set-account-is-disabled` subcommand, as shown in the following example:**

```
$ manage-account -h localhost -p 4444 -D "cn=directory manager" -w password -X \
set-account-is-disabled --operationValue true \
--targetDN "uid=kvaughan,ou=People,dc=example,dc=com"
Account Is Disabled:  true
```

▼ To Enable an Account

- **Use the `manage-account` command with the `clear-account-is-disabled` subcommand, as shown in the following example:**

```
$ manage-account -D "cn=directory manager" -w password clear-account-is-disabled \
  --targetDN "uid=kvaughan,ou=People,dc=example,dc=com"
Account Is Disabled:  false
```

Setting Resource Limits on a User Account

You can control search operations on the server for each client account by assigning resource limits to the entry. Resource limits are assigned by adding specific operational attributes to the user entry. The directory server then enforces the limits based on the account that the client uses to bind to the directory.

The resource limits that you set on specific user accounts take precedence over the resource limits set in the server-wide configuration. The following limits can be set:

- **Look-through limit**. Specifies the maximum number of entries examined for a search operation. Use the `ds-rlim-lookthrough-limit` operational attribute.
- **Size limit**. Specifies the maximum number of entries returned in response to a search operation. use the `ds-rlim-size-limit` operational attribute.
- **Time limit**. Specifies the maximum time spent processing a search operation. Use the `ds-rlim-time-limit` operational attribute.

Note – The Directory Manager can use unlimited resources by default.

▼ To Set Resource Limits on an Account

1 Modify the entry in an LDIF file, adding the operational attributes, as shown here:

```
dn: uid=kvaughan,ou=people,dc=example,dc=com
changetype: modify
add: ds-rlim-lookthrough-limit
ds-rlim-lookthrough-limit: 1000
-
add: ds-rlim-size-limit
ds-rlim-size-limit: 500
-
add: ds-rlim-time-limit
ds-rlim-time-limit: 300
```

2 Use the `ldapmodify` command to apply the changes, as shown here:

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --filename add_resource.ldif
Processing MODIFY request for uid=kvaughan,ou=people,dc=example,dc=com
MODIFY operation successful for DN uid=kvaughan,ou=people,dc=example,dc=com
```

Defining Groups

The directory server supports *groups*, which are collections of entries that are manageable as a single object. Typically, directory administrators configure groups of printers, groups of software applications, groups of employees, and so forth. Groups are especially useful when assigning special access privileges to a set of users. For example, you can assign access managers the privileges to employee data while restricting those same privileges to others in the company.

The directory server supports the following group types:

- **Static groups**. A static group defines its membership by providing explicit sets of distinguished names (DNs) using the `groupOfNames`, `groupOfUniqueNames`, or `groupOfEntries` object class. Statics groups are well supported by external clients and provide good performance.

  A disadvantage of static groups is that as the group membership increases, the ability to easily manage the data becomes more difficult. For every entry that changes, all groups containing the changed entry must also be changed. This task becomes more difficult as the number of members of a group grows large. As a result, static groups are best used for relatively small groups that change infrequently.
- **Dynamic groups**. A dynamic group defines its membership using a set of search criteria in the form of an LDAP URL, using the `groupOfUrls` object class. Compared to static groups, dynamic groups handle large numbers of members well (millions of entries). As entries are updated, all parent groups are updated automatically.

  A disadvantage of dynamic groups is that not all clients support them. Performance also is adversely affected if you need to query the whole list of entries. Thus, dynamic groups are best suited for groups with a very large number of entries or for clients that need to determine specific group membership for an entry.
- **Virtual static groups**. A virtual static group appears and behaves like a static group to external clients, except that each member is represented by a virtual attribute that defines its membership on the fly from another dynamic group. Virtual static groups provide an efficient way to manage large numbers of entries and avoid the scalability issues for clients that only support static groups.

Defining Static Groups

A static group is one whose entry contains a membership list of explicit DNs. Many clients support static groups, but static groups are difficult to manage as the number of members in a group increases in size. For example, if you have a member entry that requires a DN change, then you must change the user's DN for each group she belongs to.

Because a static group contains a list of explicit member DNs, its database footprint increases as the membership list grows. For this reason, a static group is best suited for small groups (less than 10,000) whose entries do not change frequently. Using large static groups can have a detrimental impact on performance. If you know that group membership will exceed 10,000, consider using dynamic groups instead.

The directory server supports the following three types of static groups, divided according to the object class they use:

- `groupOfNames` You can define a static group by using the `groupOfNames` object class and by explicitly specifying the member DNs using the `member` attribute.

  Note – RFC 4519 requires that the `member` attribute be mandatory within the `groupOfNames` object class. This membership requirement has traditionally caused data management problems when an administrator attempted to delete the last member in the group. The directory server solves this problem by allowing the `member` attribute to be optional. The optional membership requirement allows you to have an empty object class when you delete the last member of the group.

  ```
  dn: cn=Example Static Group 1,ou=Groups,dc=example,dc=com
  objectClass: top
  objectClass: groupOfNames
  member: uid=user1,ou=People,dc=example,dc=com
  member: uid=user2,ou=People,dc=example,dc=com
  cn: Example Static Group 1
  ```

- `groupOfUniqueNames` You can define a static group by using the `groupOfUniqueNames` object class and by explicitly specifying the member DNs using the `uniqueMember` attribute. The `groupOfUniqueNames` object class differs from the `groupOfNames` object class in that you can enumerate the group's members by specifying a unique DN plus an optional identifier. The identifier ensures that the unique objects can be identified when adding, deleting, or renaming any object.

  For example, you could delete or move an employee (`cn=Tom Smith`) and add a new employee who has the same name (`cn=Tom Smith`) to the directory. To distinguish the two, you must add a separate identifier by using a bit string. The following example shows two users with the same name, but the second `uniqueMember` has an optional identifier.

  ```
  uniqueMember: uid=tsmith,ou=People,dc=example,dc=com
  uniqueMember: uid=tsmith,ou=People,dc=example,dc=com#'0111101'B
  ```

Note – Few LDAP applications actually use the optional UID identifier.

RFC 4519 requires that the `uniqueMember` attribute be mandatory within the `groupOfUniqueNames` object class. This membership requirement has historically caused data management problems when an administrator tried to delete the last member in the group. The directory server solves this problem by allowing the `uniqueMember` attribute to be optional. The optional membership requirement allows you to have an empty object class when you delete the last member of the group.

```
dn: cn=Example Static Group 2,ou=Groups,dc=example,dc=com
objectClass: top
objectClass: groupOfUniqueNames
uniqueMember: uid=user1,ou=People,dc=example,dc=com
uniqueMember: uid=user2,ou=People,dc=example,dc=com
cn: Example Static Group 2
```

- `groupOfEntries` You can define a static group using the `groupOfEntries` object class. Based on the original specifications (RFC 4519 and draft-findlay-ldap-grouofentries-00.txt, which expired in March, 2008), the `groupOfEntries` object class differs from the `groupOfNames` and `groupOfUniqueNames` object classes in that attributes are optional. This allows you to specify an empty object class without any members.

Note – The directory server supports the `groupOfEntries` draft but also allows empty `groupOfNames` and `groupOfUniqueNames` object classes. As a result, you can create empty groups of any type (`groupOfEntries`, `groupOfNames`, and `groupOfUniqueNames`).

```
dn: cn=Example Static Group 3,ou=Groups,dc=example,dc=com
objectClass: top
objectClass: groupOfEntries
cn: Example Static Group 3
```

▼ To Create a Static Group With `groupOfNames`

1. **Create the group entry in LDIF, including the group name (`cn`) and the `groupOfNames` object class.**

   This example shows an LDIF file, named `static-group1.ldif`, that defines the new group.

```
dn: cn=Directory Administrators,ou=Groups,dc=example,dc=com
cn: Directory Administrators
objectclass: top
objectclass: groupOfNames
ou: Groups
member: uid=ttully,ou=People,dc=example,dc=com
member: uid=charvey,ou=People,dc=example,dc=com
```

```
member: uid=rfisher,ou=People,dc=example,dc=com
```

2 **Add the group by using `ldapmodify` to apply the LDIF file.**

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --defaultAdd --filename static-group1.ldif
Processing ADD request for cn=Directory Administrators,ou=Groups,dc=example,dc=com
ADD operation successful for DN cn=Directory
Administrators,ou=Groups,dc=example,dc=com
```

▼ To Create a Static Group With `groupOfUniqueNames`

1 **Create the group entry in LDIF, including the group name (`cn`) and the `groupOfUniqueNames` object class.**

This example shows an LDIF file, named `static-group2.ldif`, that defines the new group.

```
dn: cn=Directory Administrators2,ou=Groups,dc=example,dc=com
cn: Directory Administrators2
objectclass: top
objectclass: groupOfUniqueNames
ou: Groups
uniquemember: uid=alangdon,ou=People,dc=example,dc=com
uniquemember: uid=drose,ou=People,dc=example,dc=com
uniquemember: uid=polfield,ou=People,dc=example,dc=com
```

2 **Add the group by using `ldapmodify` to apply the LDIF file.**

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --defaultAdd --filename static-group2.ldif
```

3 **Verify the change by using `ldapsearch` and the `isMemberOf` attribute.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --baseDN dc=example,dc=com "(uid=rdaugherty)" isMemberOf
dn: uid=alangdon,ou=People,dc=example,dc=com
isMemberOf: cn=Directory Administrators2,ou=Groups,dc=example,dc=com
```

▼ To Create a Static Group With `groupOfEntries`

1 **Create the group entry in LDIF, including the group name (`cn`) and the `groupOfEntries` object class.**

This example shows an LDIF file, named `static-group3.ldif`, that defines the new group.

```
dn: cn=Directory Administrators3,ou=Groups,dc=example,dc=com
cn: Directory Administrators3
objectclass: top
objectclass: groupOfEntries
ou: Groups
member: uid=bfrancis,ou=People,dc=example,dc=com
```

```
member: uid=tjames,ou=People,dc=example,dc=com
member: uid=bparker,ou=People,dc=example,dc=com
```

2 **Add the group by using `ldapmodify` to apply the LDIF file.**

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --defaultAdd --filename static-group3.ldif
```

3 **Verify the change by using `ldapsearch` and the `isMemberOf` attribute.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --baseDN dc=example,dc=com "(uid=bparker)" isMemberOf
dn: uid=bparker,ou=People,dc=example,dc=com
isMemberOf: cn=Directory Administrators3,ou=Groups,dc=example,dc=com
```

▼ To List All Members of a Static Group

You can use the `isMemberOf` virtual attribute to search for a group. The attribute is added to the user entry at the start of the search and then removed after the search has finished. This functionality provides easy management of groups with fast read access.

- **Use the `ldapsearch` command with the virtual attribute `isMemberOf`.**

  This example searches for all users who are members of the group "Accounting Managers".

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com \
  "(isMemberOf=cn=Accounting Managers,ou=Groups,dc=example,dc=com)"
dn: uid=scarter,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Accounting
ou: People
sn: Carter
facsimiletelephonenumber: +1 408 555 9751
roomnumber: 4612
userpassword: {SSHA}3KiJ51sx2Ug7DxZoq0vA9ZY6uaomevbJUBm7OA==
l: Sunnyvale
cn: Sam Carter
telephonenumber: +1 408 555 4798
givenname: Sam
uid: scarter
mail: scarter@example.com
dn: uid=tmorris,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
```

```
ou: Accounting
ou: People
sn: Morris
facsimiletelephonenumber: +1 408 555 8473
roomnumber: 4117
userpassword: {SSHA}bjFFHv6k1kbI6fZoCEfqmTj9XOZxWR06gxpKpQ==
l: Santa Clara
cn: Ted Morris
telephonenumber: +1 408 555 9187
givenname: Ted
uid: tmorris
mail: tmorris@example.com
```

▼ To List All Static Groups of Which a User Is a Member

- **Search using `ldapsearch` and the virtual attribute `isMemberOf`, as shown in the following example:**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com "(uid=scarter)" isMemberOf
dn: uid=scarter,ou=People,dc=example,dc=com
isMemberOf: cn=Accounting Managers,ou=groups,dc=example,dc=com
```

▼ To Determine Whether a User is a Member of a Group

- **Search using `ldapsearch`, as shown in the following example:**

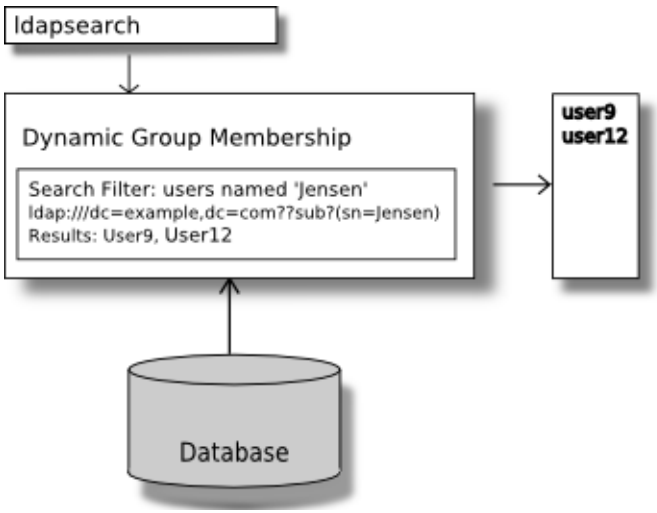
```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b "cn=Account Managers,ou=Groups,dc=example,dc=com" \
  "(&(objectclass=groupOfUniqueNames) \
  (uniquemember=uid=scarter,ou=People,dc=example,dc=com))"
dn: cn=Accounting Managers,ou=groups,dc=example,dc=com
objectClass: groupOfUniqueNames
objectClass: top
ou: groups
description: People who can manage accounting entries
cn: Accounting Managers
uniquemember: uid=scarter, ou=People, dc=example,dc=com
uniquemember: uid=tmorris, ou=People, dc=example,dc=com
```

Defining Dynamic Groups

A *dynamic group* is one whose membership, rather than being maintained explicitly in a list, is determined by search criteria using an LDAP URL. For example, suppose that you want to send an email to all managers in the `dc=example,dc=com` naming context. To do this, you create a dynamic group in which you specify `cn=Managers,ou=Groups,dc=example,dc=com`. You

further specify that you want only email addresses returned. When the email application queries the directory for that particular group, the directory server computes the membership dynamically and returns the corresponding list of email addresses.

Dynamic groups use the `groupOfURLs` object class and the `memberURL` attribute to define LDAP URLs with the criteria (search base, scope, and filter) to be used for determining members of the group. The mechanism for determining whether a user is a member of a dynamic group is a constant-time operation, so it is just as efficient for groups with millions of members as it is for a group with only a few members. However, care must be taken when specifying the search criteria as it can adversely affect performance if searching over a large set of data.



▼ To Create a Dynamic Group

1 **Create an LDIF file that specifies the group.**

This example specifies the dynamic group for employees located at Cupertino.

```
dn: cn=cupertinoEmployees,ou=Groups,dc=example,dc=com
cn: CupertinoEmployees
objectclass: top
objectclass: groupOfURLs
ou: Groups
memberURL: ldap:///ou=People,dc=example,dc=com??sub?(l=Cupertino)
```

2 **Add the group by using `ldapmodify` to process the LDIF file.**

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --defaultAdd --filename dynamic_group.ldif
Processing ADD request for cn=cupertionEmployees,ou=Groups,dc=example,dc=com
```

```
ADD operation successful for DN cn=cupertionEmployees,ou=Groups,dc=example,dc=com
```

To List All Members of a Dynamic Group

This procedure illustrates the use of the virtual attribute `isMemberOf`. Do not use this procedure for very large groups, because it adversely affects the directory server's performance.

- **Search using `ldapsearch` and the virtual attribute `isMemberOf`.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b "dc=example,dc=com" \
"(isMemberOf=cn=cupertinoEmployees,ou=Groups,dc=example,dc=com)"
dn: uid=abergin,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Product Testing
ou: People
sn: Bergin
facsimiletelephonenumber: +1 408 555 7472
roomnumber: 3472
userpassword: {SSHA}YcDl0pHLxkd/ouW2jslAk1XaT5SiY4ium5qh8w==
l: Cupertino
cn: Andy Bergin
telephonenumber: +1 408 555 8585
givenname: Andy
uid: abergin
mail: abergin@example.com
...(more entries)...
```

To List All Dynamic Groups of Which a User Is a Member

- **Search using `ldapsearch` and the virtual attribute `isMemberOf`.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com "(uid=abergin)" isMemberOf
dn: uid=abergin,ou=People,dc=example,dc=com
isMemberOf: cn=QA Managers,ou=groups,dc=example,dc=com
isMemberOf: cn=cupertinoEmployees,ou=Groups,dc=example,dc=com
```

To Determine Whether a User Is a Member of a Dynamic Group

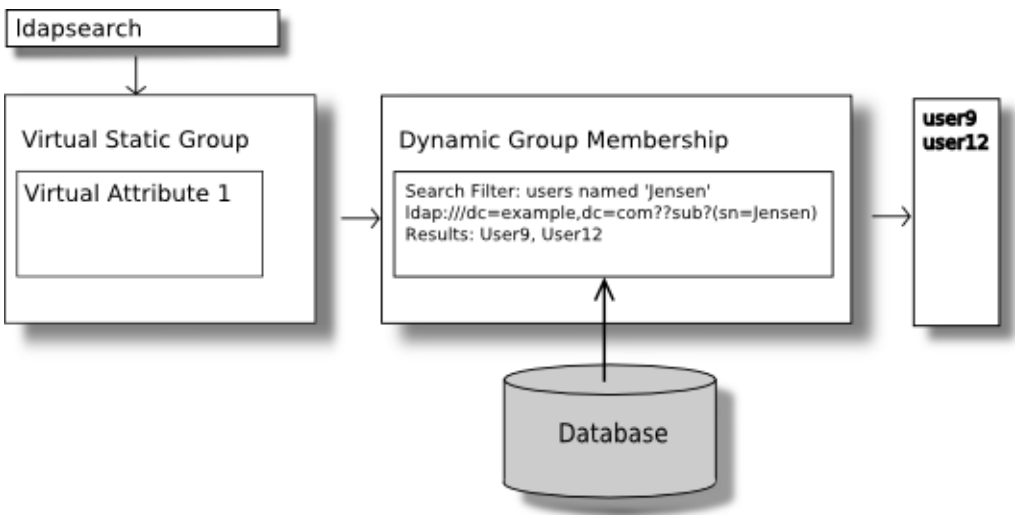
- **Search using `ldapsearch` and the virtual attribute `isMemberOf`.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com \
"(&(uid=abergin)(isMemberOf=cn=cupertinoEmployees,ou=Groups,dc=example,dc=com))"
```

```
dn: uid=abergin,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Product Testing
ou: People
sn: Bergin
facsimiletelephonenumber: +1 408 555 7472
roomnumber: 3472
userpassword: {SSHA}YcDl0pHLxkd/ouW2jslAk1XaT5SiY4ium5qh8w==
l: Cupertino
cn: Andy Bergin
telephonenumber: +1 408 555 8585
givenname: Andy
uid: abergin
mail: abergin@example.com
```

Defining Virtual Static Groups

A *virtual static group*, efficiently manages scalability for clients that can only support static groups. In a virtual static group, each entry behaves like a static group entry by using virtual attributes. The virtual attributes are dynamically determined when invoked, and the operations that determine group membership are passed to another group, such as a dynamic group, as shown in the following diagram.



Virtual static groups should include either the `groupOfNames` or `groupOfUniqueNames` object class but should not include the `member` or `uniqueMember` attribute. Virtual static groups should also contain the `ds-virtual-static-group` auxiliary object class and the `ds-target-group-dn` attribute. The `ds-target-group-dn` attribute is used to reference the actual group to mirror as a virtual static group and is used in place of the `member` or `uniquemember` attribute. For example:

```
dn: cn=Example Virtual Static Group,ou=Groups,dc=example,dc=com
objectClass: top
objectClass: groupOfUniqueNames
objectClass: ds-virtual-static-group
cn: Example Virtual Static Group
ds-target-group-dn: cn=Example Real Group,ou=Groups,dc=example,dc=com
```

Virtual static groups are most efficient when the application issues a search targeted at the membership attribute but does not actually retrieve the entire set of members. It is common for applications to use a filter such as the following to attempt to determine whether a user is a member of a given group:

```
(&(objectClass=groupOfUniqueNames)(uniqueMember=uid=john.doe,\
  ou=People,dc=example,dc=com))
```

For applications that retrieve the set of members, virtual static groups might not be ideal because the process of constructing the entire member list can be expensive.

▼ To Create a Virtual Static Group

1 Create an LDIF file that specifies the group.

This sample file, `virtual-static.ldif`, specifies a virtual static group named `cupertinoEmployees`.

```
dn: cn=virtualStatic,ou=Groups,dc=example,dc=com
cn: Virtual Static
objectclass: top
objectclass: groupOfUniqueNames
objectclass: ds-virtual-static-group
ou: Groups
ds-target-group-dn: cn=cupertinoEmployees,ou=Groups,dc=example,dc=com
```

2 Add the group by using `ldapmodify` to process the LDIF file.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --defaultAdd --filename virtual-static.ldif
Processing ADD request for cn=virtualStatic,ou=Groups,dc=example,dc=com
ADD operation successful for DN cn=virtualStatic,ou=Groups,dc=example,dc=com
```

To List All Members of a Virtual Static Group

Virtual static groups are best used in cases where the search is targeted at the membership attribute. This procedure is therefore not recommended but is included to show how to access the list.

This example procedure uses the dynamic group, `cupertinoEmployees`, created in the previous example.

- **Search using `ldapsearch` and the virtual attribute `isMemberOf`.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com "(isMemberOf=cn=virtualStatic,ou=Groups,dc=example,dc=com)"
dn: uid=abergin,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: top
objectClass: organizationalPerson
ou: Product Testing
ou: People
sn: Bergin
facsimiletelephonenumber: +1 408 555 7472
roomnumber: 3472
userpassword: {SSHA}YcDl0pHLxkd/ouW2jslAk1XaT5SiY4ium5qh8w==
l: Cupertino
cn: Andy Bergin
telephonenumber: +1 408 555 8585
givenname: Andy
uid: abergin
mail: abergin@example.com
...(more entries)...
```

To List All Virtual-Static Groups of Which a User Is a Member

- **Search using `ldapsearch` and the virtual attribute `isMemberOf`.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b dc=example,dc=com "(uid=abergin)" isMemberOf
dn: uid=abergin,ou=People,dc=example,dc=com
isMemberOf: cn=QA Managers,ou=groups,dc=example,dc=com
isMemberOf: cn=cupertinoEmployees,ou=Groups,dc=example,dc=com
isMemberOf: cn=virtualStatic,ou=Groups,dc=example,dc=com
```

To Determine Whether a User is a Member of a Virtual Static Group

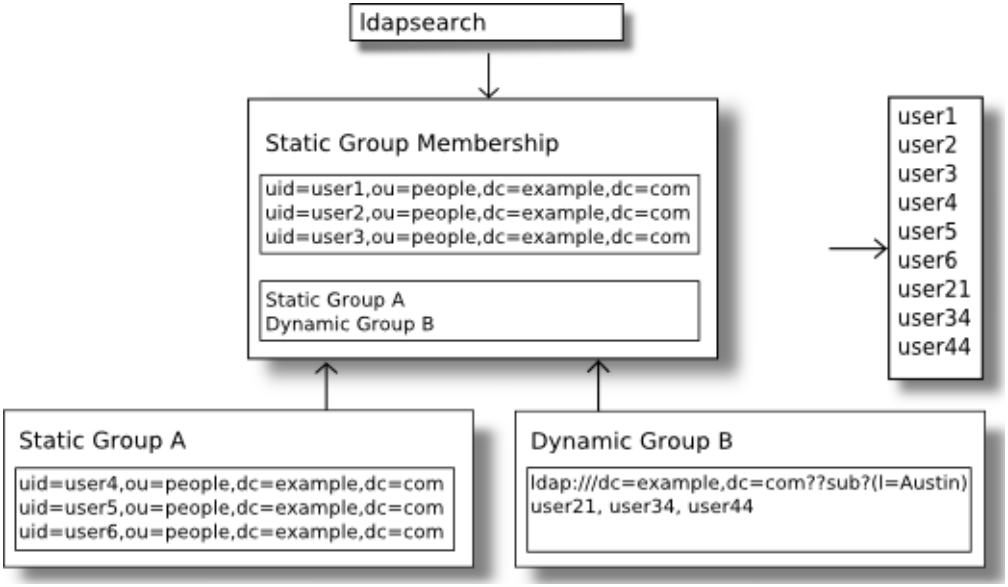
- **Search using `ldapsearch` and the `uniqueMember` attribute.**

```
$ ldapsearch -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  -b "cn=virtualStatic,ou=Groups,dc=example,dc=com" \
```

```
  "(&(objectclass=groupOfUniqueNames) \
  (uniquemember=uid=abergin,ou=People,dc=example,dc=com))"
dn: cn=virtualStatic,ou=Groups,dc=example,dc=com
objectClass: groupOfUniqueNames
objectClass: top
objectClass: ds-virtual-static-group
ou: Groups
ds-target-group-dn: cn=cupertinoEmployees,ou=Groups,dc=example,dc=com
cn: Virtual Static
cn: virtualStatic
```

Defining Nested Groups

Groups can be nested, where one group is defined as a child group entry whose DN is listed within another group, its parent. The nesting of groups allows you to set up inherited group memberships when performance is not a priority. You can add zero or more member attributes with their values set to the DNs of nested child groups, including both static and dynamic groups.



To Create a Nested Group

This example procedure creates a nested group using one static group and one dynamic group.

1 Create an LDIF file that specifies a static group.

This example file, `static-group.ldif`, specifies a virtual static group named `Dev Contractors`.

```
dn: cn=Contractors,ou=Groups,dc=example,dc=com
cn: Dev Contractors
objectclass: top
objectclass: groupOfUniqueNames
ou: Dev Contractors Static Group
uniquemember: uid=wsmith,ou=Contractors,dc=example,dc=com
uniquemember: uid=jstearn,ou=Contractors,dc=example,dc=com
uniquemember: uid=pbrook,ou=Contractors,dc=example,dc=com
uniquemember: uid=njohnson,ou=Contractors,dc=example,dc=com
uniquemember: uid=sjones,ou=Contractors,dc=example,dc=com
```

2 Add the group by using `ldapmodify` to process the LDIF file.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --defaultAdd --filename static-group.ldif
```

3 Create an LDIF file that specifies a dynamic group.

This example file, `dynamic-group.ldif`, specifies a dynamic group named `Developers`.

```
dn: cn=Developers,ou=Groups,dc=example,dc=com
cn: Developers
objectclass: top
objectclass: groupOfURLs
ou: Groups
memberURL: ldap:///ou=People,dc=example,dc=com??sub?(ou=Product Development)
```

4 Add the group by using `ldapmodify` to process the LDIF file.

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --defaultAdd --filename dynamic-group.ldif
```

5 Create an LDIF file that specifies a nested static group.

This example file, `nested-group.ldif`, specifies a nested group named `Developers Group`.

```
dn: cn=DevelopersGroup,ou=Groups,dc=example,dc=com
cn: Developers Group
objectclass: top
objectclass: groupOfUniqueNames
ou: Nested Static Group
uniquemember: cn=Contractors,ou=Groups,dc=example,dc=com
uniquemember: cn=Developers,ou=Groups,dc=example,dc=com
```

6 **Add the group by using `ldapmodify` to process the LDIF file,**

```
$ ldapmodify -h localhost -p 1389 -D "cn=Directory Manager" -w password \
  --defaultAdd --filename nested-group.ldif
```

Maintaining Referential Integrity

Referential integrity is a database mechanism for ensuring that all references are properly maintained after delete, rename, or move operations. For example, if an entry is removed from the directory, the directory server also removes the entry from any groups of which the entry is listed as a member.

The referential integrity mechanism is configured as a plug-in in the directory server and can be enabled using the `dsconfig` command. For more information, see "Configuring the Directory Server With `dsconfig`" on page 7.

Overview of the Referential Integrity Plug-In

By default, the referential integrity plug-in is disabled. When you enable the plug-in by using `dsconfig`, it performs integrity updates on the `member` and `uniquemember` attributes immediately after a delete, rename, or move operation. Whenever you delete, rename, or move a user or group entry in the directory, the operation is logged to the referential integrity log file, *install-dir*`/logs/referint`.

After a specified time, known as the *update interval*, the server performs a search on the specified attributes and matches the results with the DNs of the deleted or modified entries recorded in the log. If the log file shows that an entry was deleted, the corresponding attribute is deleted. If the log file shows that an entry was changed, the corresponding attribute value is modified accordingly.

You can configure the properties of the referential integrity plug-in to suit your requirements. The following properties can be configured:

- **Enabled**. Turn on the referential integrity plug-in.
- **plugin type**. By default, the delete, rename, and move operations are set. You can change a plug-in type to only delete, for example.
- **Attribute type**. By default, the attribute types are set to `member,uniquemember` but can be changed to some other attribute. If you use or define attributes containing DN values, you can use the referential integrity plug-in to monitor these attributes.
- **Base-DN**. By default, the scope is to use all public naming contexts but this can be changed to a specific context.

- **Log file**. By default, `logs/referint` is the log file. You can record the referential integrity updates in a different file. For example, if you want to record changes in a replicated environment, you can write to the *changelog* file on a replication server, so that it can be replicated to a consumer server.
- **Update interval**. By default, the update interval is set to 0 seconds, which will run referential integrity immediately after a delete, rename, or move operation. To minimize the impact of the updates on system performance, increase the amount of time between updates. Typical update intervals are as follows:
-
  - 0 seconds, update immediately
  - 90 seconds (updates every 90 seconds)
  - 3600 seconds (updates every hour)
  - 10,800 seconds (updates every 3 hours)
  - 28,800 seconds (updates every 8 hours)
  - 86,400 seconds (updates once a day)
  - 604,800 seconds (updates once a week)

▼ To Enable the Referential Integrity Plug-In

- **Set the `enabled` property of the plug-in to true.**

```
$ dsconfig -D "cn=directory manager" -w password -n set-plugin-prop \
  --plugin-name "Referential Integrity" --set enabled:true
```

Simulating DSEE Roles in an OpenDS Directory Server

Sun Java System Directory Server Enterprise Edition (DSEE) includes a roles subsystem that is used to provide a specialized type of grouping mechanism. This capability is not included directly in the OpenDS directory server, because it is based on non-standard functionality, uses Netscape™-proprietary schema elements, and is not widely used in LDAP-enabled applications.

However, the OpenDS directory server does provide all of the functionality offered by DSEE roles, and this functionality is available for use with standard grouping mechanisms. If you have an application that was specifically written to rely on the roles functionality available in DSEE and cannot work with standard grouping mechanisms, you can configure the OpenDS directory server to simulate DSEE roles to satisfy such applications.

Note – If your application needs to create and destroy role entries (for example, an entry containing one of the subordinates of the `nsRoleDefinition` object class), that functionality is currently not available with the OpenDS directory server. It could be added by creating a custom group implementation (that is, a subclass of the `org.opends.server.api.Group` class) to provide the necessary logic, but this task is currently not planned for inclusion in the OpenDS directory server.

▼ To Determine Whether a User is a Member of a Role

If the application needs only to determine whether a user is a member of a given role, it should only need to look at the `nsRole` attribute in the target user's entry to determine whether the DN of the appropriate role is present. In this case, you can simulate role functionality by following these steps.

After these steps are completed, the `nsRole` virtual attribute appears as an operational attribute in user entries, and should include the DNs of all groups in which that user is a member. Note that `nsRole` is an operational attribute, and must be explicitly requested for it to be returned in search results. You must also ensure that the authenticated user has permission to see that attribute.

1 Update the directory server to include the necessary schema for the DSEE roles implementation.

This schema is provided in the following LDIF file (named `03-dsee-roles.ldif`).

```
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License, Version 1.0 only
# (the "License").  You may not use this file except in compliance
# with the License.
#
# You can obtain a copy of the license at
# trunk/opends/resource/legal-notices/OpenDS.LICENSE
# or https://OpenDS.dev.java.net/OpenDS.LICENSE.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at
# trunk/opends/resource/legal-notices/OpenDS.LICENSE.  If applicable,
# add the following below this CDDL HEADER, with the fields enclosed
# by brackets "[]" replaced with your own identifying information:
#      Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
```

```
# This file contains schema definitions required to simulate DSEE role
# functionality in OpenDS.
dn: cn=schema
objectClass: top
objectClass: ldapSubentry
objectClass: subschema
attributeTypes: ( 2.16.840.1.113730.3.1.574 NAME 'nsRole'
  DESC 'Sun ONE defined attribute type' SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
  NO-USER-MODIFICATION USAGE directoryOperation
  X-ORIGIN 'Sun ONE Directory Server' )
attributeTypes: ( 2.16.840.1.113730.3.1.575 NAME 'nsRoleDN'
  DESC 'Sun ONE defined attribute type' SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
  USAGE directoryOperation X-ORIGIN 'Sun ONE Directory Server' )
```

- **Either copy the file into the `config/schema` directory of the directory server implementation and restart the server, or**
- **Use the `add schema file` task to cause the server to load the schema file into a running server instance.**

2. **Create a static or dynamic group to define role membership.**

   Make sure that the group has an appropriate set of members.

3. **Create a new instance of the `isMemberOf` virtual attribute to provide the `nsRole` virtual attribute.**

   The `nsRole` attribute will include a list of the DNs of all groups in which the target user is a member. Use the `dsconfig` command to create the virtual attribute, as follows:

```
$ dsconfig -D "cn=directory manager" -w password -n create-virtual-attribute \
  --type is-member-of --name nsRole --set attribute-type:nsRole --set enabled:true
```

▼ To Alter Membership by Using the `nsRoleDN` Attribute

Follow this procedure if the application you are using expects to be able to alter membership by placing the name of the corresponding role in the `nsRoleDN` virtual attribute in a user's entry.

After these steps are completed, any user entry that contains an `nsRoleDN` value of "`cn=Test Role,ou=Roles,dc=example,dc=com`" also has that DN present in the `nsRole` operational attribute.

1. **Create a dynamic group entry with the DN of the desired role.**

2 **Configure the group to include members that contain an `nsRoleDN` attribute with a value equal to the DN of the target role.**

For example, if the application is going to add an `nsRoleDN` value of "`cn=Test Role,ou=Roles,dc=example,dc=com`", add the following entry:

```
dn: cn=Test Role,ou=Roles,dc=example,dc=com
objectClass: top
objectClass: groupOfURLs
cn: Test Role
memberURL: ldap:///dc=example,dc=com??sub?(nsRoleDN=\
  cn=Test Role,ou=Roles,dc=example,dc=com)
```

Directory Server Monitoring

The directory server provides an extensible monitoring framework. The following sections describe how to configure monitoring:

Monitoring the Directory Server

Sun OpenDS Standard Edition provides a variety of methods to monitor the current state of the server for debugging or troubleshooting purposes.

The topics in this section assume that you have configured the following on the directory server:

- **JMX.** For more information, see "Monitoring the Directory Server With JConsole" on page 328.
- **Notifications.** For more information, see "Configuring Alerts and Account Status Notification Handlers" on page 342.
- **Logs.** For more information, see "Configuring Logs With `dsconfig`" on page 339.

Working With Monitor Providers

Monitor providers are enabled by default and are responsible for publishing information about the server that can be useful for monitoring or troubleshooting purposes. The `cn=monitor` entry contains the monitoring information that is published by the monitor providers.

Monitor providers can be configured by using the `dsconfig` command. For more information, see "Configuring the Directory Server With `dsconfig`" on page 7.

To View Monitor Providers

- **Run the `dsconfig` command as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n list-monitor-providers
Monitor Provider   : Type              : enabled
-------------------:-------------------:--------
Client Connections : client-connection : true
Entry Caches       : entry-cache       : true
JVM Memory Usage   : memory-usage      : true
JVM Stack Trace    : stack-trace       : true
System Info        : system-info       : true
Version            : version           : true
```

To Disable a Monitor Provider

- **Run the `dsconfig` command as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n set-monitor-provider-prop \
  --provider-name "Version Monitor Provider" --set enabled:false
```

To Create a Monitor Provider

This example creates and enables a new entry cache monitor provider, named "File System Entry Cache".

1. **Run the following `dsconfig` command:**

```
$ dsconfig -D "cn=directory manager" -w password -n create-monitor-provider \
  --provider-name "File System Entry Cache" --type entry-cache --set enabled:true
```

2. **(Optional) List the monitor providers to view the new monitor provider.**

```
$ dsconfig -D "cn=directory manager" -w password -n list-monitor-providers
Monitor Provider        : Type              : enabled
------------------------:-------------------:--------
Client Connections      : client-connection : true
Entry Caches            : entry-cache       : true
File System Entry Cache : entry-cache       : true
JVM Memory Usage        : memory-usage      : true
JVM Stack Trace         : stack-trace       : true
System Info             : system-info       : true
Version                 : version           : true
```

To Delete a Monitor Provider

This example deletes the entry cache monitor provider that was created in the previous example.

1. **Run the following `dsconfig` command:**

```
$ dsconfig -D "cn=directory manager" -w password -n delete-monitor-provider \
  --provider-name "File System Entry Cache"
```

2. **(Optional) List the monitor providers to ensure that the monitor provider was deleted.**

```
$ ./dsconfig -h VDPhost -p 4444 -D "cn=Directory Manager" -w password -X -n \
list-monitor-providers \

Monitor Provider   : Type              : enabled
-------------------:--------------------:--------
Client Connections : client-connection : true
Entry Caches       : entry-cache       : true
JVM Memory Usage   : memory-usage      : true
JVM Stack Trace    : stack-trace       : false
System Info        : system-info       : true
Version            : version           : true
```

3. **(Optional) List the monitor providers to ensure that the monitor provider was deleted.**

```
$ dsconfig -D "cn=directory manager" -w password -n list-monitor-providers
Monitor Provider        : Type              : enabled
------------------------:--------------------:--------
Client Connections      : client-connection : true
Entry Caches            : entry-cache       : true
JVM Memory Usage        : memory-usage      : true
JVM Stack Trace         : stack-trace       : true
System Info             : system-info       : true
Version                 : version           : true
```

Viewing Monitoring Information Using the `cn=monitor` Entry

The directory server records system, performance, and version information as an entry with the base DN of `cn=monitor`. This entry provides useful performance metrics and server state information that you can use to monitor and debug a directory server instance.

You can access the `cn=monitor` suffix over the regular LDAP port but there are advantages to using the administration port to access monitoring information. The main advantage of the administration connector is the separation of user traffic and administration traffic. For example, if you monitor the number of connections on the LDAP Connection Handler ("`cn=Client Connections,cn=LDAP Connection Handler 0.0.0.0 port` *port-number*`,cn=monitor`") over the regular LDAP port, your monitoring data are "polluted" by the monitoring request itself. All of the examples in this section use the administration port, over SSL. For more information, see "Managing Administration Traffic to the Server" on page 5.

To View the Available Monitoring Information

Use the ldapsearch command to inspect the attributes of cn=monitor. This example lists the base DNs of each monitor entry.

- **Run the ldapsearch command with a search scope of sub and the search attribute 1.1.**

  This search attribute indicates that no attributes should be included in the matching entries.

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll  -s sub -b "cn=monitor" "(objectclass=*)" "1.1"
dn: cn=monitor
dn: cn=Client Connections,cn=monitor
dn: cn=ads-truststore Backend,cn=monitor
dn: cn=Network Groups,cn=monitor
dn: cn=internal,cn=Network Groups,cn=monitor
dn: cn=default,cn=Network Groups,cn=monitor
dn: cn=LDAP Connection Handler 0.0.0.0 port 1389 Statistics,cn=monitor
dn: cn=Administration Connector 0.0.0.0 port 4444,cn=monitor
dn: cn=Client Connections,cn=Administration Connector 0.0.0.0 port 4444,cn=monitor
dn: cn=backup Backend,cn=monitor
dn: cn=Version,cn=monitor
dn: cn=Work Queue,cn=monitor
dn: cn=System Information,cn=monitor
dn: cn=userRoot Database Environment,cn=monitor
dn: cn=tasks Backend,cn=monitor
dn: cn=adminRoot Backend,cn=monitor
dn: cn=userRoot Backend,cn=monitor
dn: cn=schema Backend,cn=monitor
dn: cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor
dn: cn=admin,cn=Network Groups,cn=monitor
dn: cn=Client Connections,cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor
dn: cn=JVM Memory Usage,cn=monitor
dn: cn=Administration Connector 0.0.0.0 port 4444 Statistics,cn=monitor
dn: cn=JVM Stack Trace,cn=monitor
dn: cn=Entry Caches,cn=monitor
dn: cn=monitor Backend,cn=monitor
```

To Monitor General-Purpose Server Information

- **Use the ldapsearch command with a base DN of "cn=monitor".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=monitor" "(objectclass=*)"
dn: cn=monitor
startTime: 20090702103150Z
objectClass: extensibleObject
objectClass: top
objectClass: ds-monitor-entry
```

```
cn: monitor
vendorName: Sun Microsystems, Inc.
currentTime: 20090702103850Z
vendorVersion: Sun OpenDS Standard Edition 2.0.0
maxConnections: 1
productName: Sun OpenDS Standard Edition
currentConnections: 1
totalConnections: 3
upTime: 0 days 0 hours 7 minutes 0 seconds
```

▼ To Monitor System Information

- **Use the `ldapsearch` command with the base DN "`cn=System Information,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=System Information,cn=monitor" "(objectclass=*)"
dn: cn=System Information,cn=monitor
javaVersion: 1.6.0_10
jvmArchitecture: 32-bit
jvmArguments: "-Dorg.opends.server.scriptName=start-ds"
jvmVersion: 11.0-b15
classPath: /local/instances/SunOpenDS_SE2.0-standalone/classes:
/local/instances/SunOpenDS_SE2.0-standalone/resources/resources.jar:
/local/instances/SunOpenDS_SE2.0-standalone/lib/activation.jar:
/local/instances/SunOpenDS_SE2.0-standalone/lib/aspectjrt.jar:
/local/instances/SunOpenDS_SE2.0-standalone/lib/je.jar:
/local/instances/SunOpenDS_SE2.0-standalone/lib/mail.jar:
/local/instances/SunOpenDS_SE2.0-standalone/lib/OpenDS_de.jar:
/local/instances/SunOpenDS_SE2.0-standalone/lib/OpenDS_es.jar:
/local/instances/SunOpenDS_SE2.0-standalone/lib/OpenDS_fr.jar:
/local/instances/SunOpenDS_SE2.0-standalone/lib/OpenDS_ja.jar:
/local/instances/SunOpenDS_SE2.0-standalone/lib/OpenDS.jar:
/local/instances/SunOpenDS_SE2.0-standalone/lib/OpenDS_zh_CN.jar:
/local/instances/SunOpenDS_SE2.0-standalone/lib/quicksetup.jar
usedMemory: 83361792
freeUsedMemory: 21020432
objectClass: extensibleObject
objectClass: top
objectClass: ds-monitor-entry
javaVendor: Sun Microsystems Inc.
operatingSystem: SunOS 5.11 x86
cn: System Information
systemName: llandudno
workingDirectory: /local/instances/SunOpenDS_SE2.0-standalone/bin
maxMemory: 518717440
availableCPUs: 2
javaHome: /usr/jdk/instances/jdk1.6.0/jre
jvmVendor: Sun Microsystems Inc.
```

To Monitor Version Information

- **Use the `ldapsearch` command with base DN "`cn=Version,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -b "cn=Version,cn=Monitor" "(objectclass=*)"
dn: cn=Version,cn=monitor
revisionNumber: 5492
shortName: OpenDS
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject
compactVersion: OpenDS-2.0.0
pointVersion: 0
cn: Version
buildID: 20090630082738Z
majorVersion: 2
productName: Sun OpenDS Standard Edition
minorVersion: 0
fullVersion: Sun OpenDS Standard Edition 2.0.0
```

To Monitor the User Root Back End

The `userRoot` back end is the back-end database (the JE environment) for your data. The monitor displays the back end's general properties, such as writability mode, base DN, back-end IDs, entry count, and other properties.

- **Use the `ldapsearch` command with base DN "`cn=userRoot Backend,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=userRoot Backend,cn=monitor" "(objectclass=*)"
dn: cn=userRoot Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: FALSE
cn: userRoot Backend
ds-backend-writability-mode: enabled
ds-backend-entry-count: 2002
ds-backend-id: userRoot
ds-base-dn-entry-count: 2002 dc=example,dc=com
ds-backend-base-dn: dc=example,dc=com
```

To Monitor the Backup Back End

- **Use the `ldapsearch` command with base DN "`cn=backup Backend,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=backup Backend,cn=monitor" "(objectclass=*)"
```

```
dn: cn=backup Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: TRUE
cn: backup Backend
ds-backend-writability-mode: disabled
ds-backend-entry-count: 1
ds-backend-id: backup
ds-base-dn-entry-count: 1 cn=backups
ds-backend-base-dn: cn=backups
```

▼ To Monitor the Tasks Back End

Tasks are administrative functions (such as `import-ldif`, `export-ldif`, `backup`, and `restore`) that can be scheduled for processing at some future date or on a recurring basis. The monitor displays the tasks back end's general properties, such as writability mode, base DN, back-end IDs, entry count, and other properties.

- **Use the `ldapsearch` command with base DN "`cn=Tasks Backend,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=Tasks Backend,cn=monitor" "(objectclass=*)"
dn: cn=tasks Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: TRUE
cn: tasks Backend
ds-backend-writability-mode: enabled
ds-backend-entry-count: 3
ds-backend-id: tasks
ds-base-dn-entry-count: 3 cn=tasks
ds-backend-base-dn: cn=tasks
```

▼ To Monitor the `monitor` Back End

This monitor displays the back end's general properties, such as writability mode, base DN, back-end IDs, entry count, and other properties.

- **Use the `ldapsearch` command with base DN "`cn=monitor Backend,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=monitor Backend,cn=monitor" "(objectclass=*)"
dn: cn=monitor Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: TRUE
```

```
cn: monitor Backend
ds-backend-writability-mode: disabled
ds-backend-entry-count: 25
ds-backend-id: monitor
ds-base-dn-entry-count: 25 cn=monitor
ds-backend-base-dn: cn=monitor
```

▼ To Monitor the Schema Back End

This monitor displays the schema back end's general properties, such as writability mode, base DN, back-end IDs, entry count, and other properties.

- **Use the `ldapsearch` command with base DN "`cn=schema Backend,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=schema Backend,cn=monitor" "(objectclass=*)"
dn: cn=schema Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: TRUE
cn: schema Backend
ds-backend-writability-mode: enabled
ds-backend-entry-count: 1
ds-backend-id: schema
ds-base-dn-entry-count: 1 cn=schema
ds-backend-base-dn: cn=schema
```

▼ To Monitor the `adminRoot` Back End

This monitor displays the `adminRoot` back end's general properties, such as writability mode, base DN, back-end IDs, entry count, and other properties.

- **Use the `ldapsearch` command with base DN "`cn=adminRoot Backend,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=adminRoot Backend,cn=monitor" "(objectclass=*)"
dn: cn=adminRoot Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: TRUE
cn: adminRoot Backend
ds-backend-writability-mode: enabled
ds-backend-entry-count: 7
ds-backend-id: adminRoot
ds-base-dn-entry-count: 7 cn=admin data
ds-backend-base-dn: cn=admin data
```

To Monitor the ads-truststore Back End

The ads-truststore holds a mirror, or copy, of the remote Administrative Directory Service (ADS) host's ADS key entry, so that the new instance can establish trust with existing servers in the ADS domain. The monitor displays the back end's general properties, such as writability mode, base DN, back-end IDs, entry count, and other properties.

- **Use the ldapsearch command with base DN "cn=ads-truststore Backend,cn=monitor".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=ads-truststore Backend,cn=monitor" "(objectclass=*)"
dn: cn=ads-truststore Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: TRUE
cn: ads-truststore Backend
ds-backend-writability-mode: enabled
ds-backend-entry-count: 3
ds-backend-id: ads-truststore
ds-base-dn-entry-count: 3 cn=ads-truststore
ds-backend-base-dn: cn=ads-truststore
```

To Monitor Client Connections

This monitor represents *all* of the open client connections. Its contents are different to those of the DN "cn=Client Connections,cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor", which describes the open client connections on the LDAP connection handler only.

- **Use the ldapsearch command with base DN "cn=Client Connections,cn=monitor".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=Client Connections,cn=monitor" "(objectclass=*)"
dn: cn=Client Connections,cn=monitor
connection: connID="11" connectTime="20090702125632Z" source="127.0.0.1:54044"
destination="127.0.0.1:1389" ldapVersion="3" authDN="cn=Directory Manager,cn=Root DNs,
cn=config" security="none" opsInProgress="1"
cn: Client Connections
objectClass: extensibleObject
objectClass: top
objectClass: ds-monitor-entry
```

To Monitor the LDAP Connection Handler

This connection handler is used to interact with clients over LDAP.

- **Use the** `ldapsearch` **command with base DN** "`cn=LDAP Connection Handler 0.0.0.0 port` *port-number*`,cn=monitor`".

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor" \
  "(objectclass=*)"
dn: cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor
ds-connectionhandler-listener: 0.0.0.0:1389
ds-connectionhandler-num-connections: 1
ds-connectionhandler-protocol: LDAP
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-connectionhandler-monitor-entry
ds-mon-config-dn: cn=ldap connection handler,cn=connection handlers,cn=config
cn: LDAP Connection Handler 0.0.0.0 port 1389
ds-connectionhandler-connection: connID="22" connectTime="20090702133936Z"
source="127.0.0.1:39574" destination="127.0.0.1:1389" ldapVersion="3"
authDN="cn=Directory Manager,cn=Root DNs,cn=config" security="none" opsInProgress="1"
```

▼ To Monitor LDAP Connection Handler Statistics

- **Use the** `ldapsearch` **command with base DN** "`cn=LDAP Connection Handler 0.0.0.0 port` *port-number* `Statistics,cn=monitor`".

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=LDAP Connection Handler 0.0.0.0 port 1389 Statistics,cn=monitor" \
  "(objectclass=*)"
dn: cn=LDAP Connection Handler 0.0.0.0 port 1389 Statistics,cn=monitor
objectClass: ds-monitor-entry
objectClass: top
objectClass: extensibleObject
operationsCompleted: 37
compareRequests: 0
bytesWritten: 99488
extendedRequests: 0
addRequests: 0
bindRequests: 19
...(more output)
```

▼ To Monitor Connections on the LDAP Connection Handler

This monitor represents the open client connections on the LDAP connection handler.

- **Use the** `ldapsearch` **command with base DN** "`cn=Client Connections,cn=LDAP Connection Handler 0.0.0.0 port` *port-number*`,cn=monitor`".

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base \
  -b "cn=Client Connections,cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor" \
```

```
  "(objectclass=*)"
dn: cn=Client Connections,cn=LDAP Connection Handler 0.0.0.0 port 1389,cn=monitor
connection: connID="0" connectTime="20090706084747Z" source="127.0.0.1:57523" de
stination="127.0.0.1:1389" ldapVersion="3" authDN="" security="none" opsInProgr
ess="0"
connection: connID="1" connectTime="20090706084747Z" source="127.0.0.1:57524" de
stination="127.0.0.1:1389" ldapVersion="3" authDN="" security="none" opsInProgr
ess="0"
connection: connID="2" connectTime="20090706084747Z" source="127.0.0.1:57525" de
stination="127.0.0.1:1389" ldapVersion="3" authDN="" security="none" opsInProgr
ess="0"
connection: connID="3" connectTime="20090706084747Z" source="127.0.0.1:57526" de
stination="127.0.0.1:1389" ldapVersion="3" authDN="" security="none" opsInProgr
ess="0"
connection: connID="4" connectTime="20090706084747Z" source="127.0.0.1:57527" de
stination="127.0.0.1:1389" ldapVersion="3" authDN="" security="none" opsInProgr
ess="0"
```

▼ To Monitor the Administration Connector

This monitor provides basic information about the administration connector. For more information, see "Managing Administration Traffic to the Server" on page 5.

- **Use the `ldapsearch` command with base DN "`cn=Administration Connector 0.0.0.0 port` *admin-port-number*`,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=Administration Connector 0.0.0.0 port 4444,cn=monitor" \
  "(objectclass=*)"
dn: cn=Administration Connector 0.0.0.0 port 4444,cn=monitor
ds-connectionhandler-listener: 0.0.0.0:4444
ds-connectionhandler-num-connections: 0
ds-connectionhandler-protocol: LDAPS
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-connectionhandler-monitor-entry
cn: Administration Connector 0.0.0.0 port 4444
ds-mon-config-dn: cn=administration connector,cn=config
```

▼ To Monitor Administration Connector Statistics

This monitor provides extensive statistical information about operations that are performed through the administration connector. For more information, see "Managing Administration Traffic to the Server" on page 5.

- **Use the `ldapsearch` command with base DN "`cn=Administration Connector 0.0.0.0 port` *admin-port-number* `Statistics,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=Administration Connector 0.0.0.0 port 4444 Statistics,cn=monitor" \
  "(objectclass=*)"
dn: cn=Administration Connector 0.0.0.0 port 4444 Statistics,cn=monitor
compareResponses: 0
connectionsClosed: 1
searchResultsDone: 4
ds-mon-resident-time-mod-operations-total-time: 92257568
extendedResponses: 0
bindRequests: 2
operationsAbandoned: 0
bytesWritten: 45056
addResponses: 0
addRequests: 0
ds-mon-resident-time-moddn-operations-total-time: 0
ds-mon-extended-operations-total-count: 0
ds-mon-moddn-operations-total-count: 0
modifyResponses: 1
operationsCompleted: 7
...(more output)...
```

▼ To Monitor Connections on the Administration Connector

This monitor represents the open client connections on the Administration Connector.

- **Use the `ldapsearch` command with base DN "`cn=Client Connections,cn=Administration Connector 0.0.0.0 port` *port-number*`,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base \
  -b "cn=Client Connections,cn=Administration Connector 0.0.0.0 port 4444,cn=monitor" \
  "(objectclass=*)"
dn: cn=Client Connections,cn=Administration Connector 0.0.0.0 port 4444,cn=monitor
connection: connID="339" connectTime="20090707075218Z" source="127.0.0.1:48213"
destination="127.0.0.1:4444" ldapVersion="3" authDN="" security="TLS"
 opsInProgress="1"
cn: Client Connections
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject
```

▼ To Monitor the LDIF Connection Handler

The LDIF connection handler is used to process changes that are read from an LDIF file, using internal operations. Monitoring information for the LDIF connection handler is only available if the connection handler is enabled.

- **Use the `ldapsearch` command with base DN "`cn=LDIF Connection Handler,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=LDIF Connection Handler,cn=monitor" "(objectclass=*)"
dn: cn=LDIF Connection Handler,cn=monitor
ds-connectionhandler-num-connections: 0
ds-connectionhandler-protocol: LDIF
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-connectionhandler-monitor-entry
ds-mon-config-dn: cn=ldif connection handler,cn=connection handlers,cn=config
cn: LDIF Connection Handler
```

To Monitor the Work Queue

The work queue keeps track of outstanding client requests and ensures that they are processed.

- **Use the `ldapsearch` command with base DN "`cn=Work Queue,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=Work Queue,cn=monitor" "(objectclass=*)"
dn: cn=Work Queue,cn=monitor
currentRequestBacklog: 0
objectClass: extensibleObject
objectClass: top
objectClass: ds-monitor-entry
requestsSubmitted: 25
cn: Work Queue
maxRequestBacklog: 0
averageRequestBacklog: 0
requestsRejectedDueToQueueFull: 0
```

To Monitor the `userRoot` Database Environment

The `userRoot` database environment utilizes the Berkeley DB Java Edition back end. JE monitoring data (data under `cn=*Database Environment,cn=monitor`) is reliable only in the short term. During high server activity (for example, anywhere from an hour to several days depending on the counter), this data can overflow. In such cases, the JE monitoring data can reflect negative values or positive but incorrect values. This is a known issue and is expected to be fixed in the next major release of the Berkeley DB Java Edition. Oracle SR numbers 15979 and 15985 correspond to this issue.

- **Use the `ldapsearch` command with base DN "`cn=userRoot Database Environment,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=userRoot Database Environment,cn=monitor" "(objectclass=*)"
dn: cn=userRoot Database Environment,cn=monitor
EnvironmentNTempBufferWrites: 0
```

```
EnvironmentNNodesExplicitlyEvicted: 0
EnvironmentCleanerBacklog: 0
EnvironmentTotalLogSize: 5386067
EnvironmentLockBytes: 2000
EnvironmentNFullBINFlush: 2
EnvironmentNBINsStripped: 0
EnvironmentLastCheckpointEnd: 5385359
TransactionNCommits: 24
EnvironmentNCleanerEntriesRead: 0
EnvironmentNRepeatFaultReads: 2
TransactionNXACommits: 0
EnvironmentNClusterLNsProcessed: 0
TransactionNBegins: 24
LockNOwners: 25
...(more output)...
```

▼ To Monitor the Entry Cache

You can access the aggregated state of all active entry caches for your directory server instance by accessing the cn=Entry Caches,cn=Monitor entry. The server can also request the "per cache" monitor data for a given instance if the entry cache instances are enabled in the directory server configuration:

- cn=FIFO Entry Cache,cn=Monitor
- cn=Soft Reference Entry Cache,cn=Monitor
- cn=File System Entry Cache,cn=Monitor

Additionally, any arbitrarily named active entry cache instance should provide a monitor, which can be accessed by that instance name, for example cn=Any Arbitrary Name Entry Cache,cn=Monitor.

- **Use the** ldapsearch **command with base DN** "cn=Entry Caches,cn=monitor"**.**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=Entry Caches,cn=monitor" "(objectclass=*)"
dn: cn=Entry Caches,cn=monitor
entryCacheHits: 0
entryCacheTries: 0
currentEntryCacheCount: 0
objectClass: extensibleObject
objectClass: top
objectClass: ds-monitor-entry
entryCacheHitRatio: 0
cn: Entry Caches
```

To Monitor JVM Stack Trace Information

You can access JVM Stack Trace information for your directory server instance. This resource monitor is implemented in the `org.opends.server.monitors.StackTraceMonitorProvider` class and requires no custom configuration.

- **Use the `ldapsearch` command with the base DN "`cn=JVM Stack Trace,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=JVM Stack Trace,cn=monitor" "(objectclass=*)"
dn: cn=JVM Stack Trace,cn=monitor
cn: JVM Stack Trace
jvmThread: id=2 ---------- Reference Handler ----------
jvmThread: id=2 frame[0]=java.lang.Object.wait(Object.java:native)
jvmThread: id=2 frame[1]=java.lang.Object.wait(Object.java:485)
jvmThread: id=2 frame[2]=java.lang.ref.Reference$ReferenceHandler.run(Reference.
java:116)
jvmThread: id=3 ---------- Finalizer ----------
jvmThread: id=3 frame[0]=java.lang.Object.wait(Object.java:native)
jvmThread: id=3 frame[1]=java.lang.ref.ReferenceQueue.remove(ReferenceQueue.java
:116)
jvmThread: id=3 frame[2]=java.lang.ref.ReferenceQueue.remove(ReferenceQueue.java
:132)
jvmThread: id=3 frame[3]=java.lang.ref.Finalizer$FinalizerThread.run(Finalizer.j
ava:159)
jvmThread: id=4 ---------- Signal Dispatcher ----------
jvmThread: id=10 ---------- Time Thread ----------
jvmThread: id=10 frame[0]=sun.misc.Unsafe.park(Unsafe.java:native)
jvmThread: id=10 frame[1]=java.util.concurrent.locks.LockSupport.parkNanos(LockS
upport.java:198)
...(more output)...
```

To Monitor the JVM Memory Usage

- **Use the `ldapsearch` command with base DN "`cn=JVM Memory Usage,cn=monitor`".**

```
$ ldapsearch -h localhost -p 4444 -D "cn=directory manager" -w password --useSSL \
  --trustAll -s base -b "cn=JVM Memory Usage,cn=monitor" "(objectclass=*)"
dn: cn=JVM Memory Usage,cn=monitor
ps-eden-space-bytes-used-after-last-collection: 0
ps-mark-sweep-total-collection-count: 0
code-cache-bytes-used-after-last-collection: 0
ps-old-gen-current-bytes-used: 25260472
ps-perm-gen-bytes-used-after-last-collection: 0
ps-scavenge-recent-collection-duration: 3
ps-scavenge-total-collection-count: 17
ps-eden-space-current-bytes-used: 32001992
ps-perm-gen-current-bytes-used: 21179960
ps-old-gen-bytes-used-after-last-collection: 0
```

```
ps-mark-sweep-total-collection-duration: 0
ps-mark-sweep-average-collection-duration: 0
ps-scavenge-average-collection-duration: 26
ps-scavenge-total-collection-duration: 443
objectClass: extensibleObject
objectClass: top
objectClass: ds-monitor-entry
ps-mark-sweep-recent-collection-duration: 0
ps-survivor-space-bytes-used-after-last-collection: 622592
cn: JVM Memory Usage
code-cache-current-bytes-used: 2143680
ps-survivor-space-current-bytes-used: 622592
```

Monitoring Using JConsole

JConsole is a graphical user interface (GUI) tool that enables you to monitor the performance of programs running on the Java Virtual Machine (JVM). JConsole comes standard on the Java 2 Platform, Standard Edition (J2SE 5.0™). You can set up JMX on a directory server by using the `dsconfig` command. For more information, see "Monitoring the Directory Server With JConsole" on page 328.

Monitoring Using Managed Tasks

The directory server provides a tasks back end that provides a mechanism for scheduling and processing certain tasks, such as `import-ldif`, `export-ldif`, `backup`, and `restore`. You can schedule a task to run at specific times and at recurring periods. To monitor scheduled tasks, use the `manage-tasks` command. For more information, see "Configuring Commands As Tasks" on page 23.

Configuring Alert Notifications and Account Status Notification Handlers

The directory server provides mechanisms for transmitting alert and account status notifications by means of JMX extensions or SMTP extensions. You can receive alert notifications when a processing event occurs. For example, events could be startups, shutdowns, attempts to write to the configuration file, and so on.

You can receive account status notifications when an event occurs during password policy processing, such as when accounts are locked out, accounts expire, passwords expire, and so on. To configure alerts or account status notifications, use the `dsconfig` command. For more information, see "Configuring Alerts and Account Status Notification Handlers" on page 342.

Accessing Logs

The directory server provides logging mechanisms to record access, error, or debugging information for the directory server instance. Multiple loggers of a given type can be active at any time, which makes it possible to create logs for specific subtrees or different repositories. The directory server does not currently provide logging filters to restrict the type of information in the logs.

The following logs are provided:

- **Access logs**. Access logs record information about the types of operations processed by the directory server.
- **Audit logs**. Audit logs are a type of access log and record all activity on the directory server.
- **Debug logs**. Debug logs record information that can be used for troubleshooting directory server problems or for providing detailed information about the directory server's processing.
- **Error logs**. Error logs record all warnings, errors, or significant events that occur during directory server processing.
- **Replication repair logs**. Replication repair logs record inconsistencies on a single directory server in a topology, based on events relating to the replication of global index catalogs.

  The replication repair log is read-only and its use is restricted to enabling replication conflict resolution.

▼ To View the Access Logs

1 Change to the logs directory of the server instance.

```
$ cd install-dir/logs
```

2 Open the `access` file by using a text editor or the UNIX `cat` command.

```
$ cat access | more
[24/Oct/2008:16:02:52 -0500] CONNECT conn=0 from=127.0.0.1 to=127.0.0.1 protocol=LDAP
[24/Oct/2008:16:02:52 -0500] BIND conn=0 op=0 msgID=1 type=SIMPLE dn="cn=Directory
  Manager"
[24/Oct/2008:16:02:53 -0500] BIND conn=0 op=0 msgID=1 result="Success"
authDN="cn=Directory Manager,cn=Root DNs,cn=config" etime=57
...(more output)...
```

▼ To View the Audit Logs

1 Change to the logs directory of the server instance.

```
$ cd install-dir/logs
```

2 **Open the `audit` file by using a text editor or the UNIX `cat` command.**

```
$ cat audit | more
# 05/Nov/2008:13:32:58 -0600; conn=21; op=51
dn: cn=File-Based Audit Logger,cn=Loggers,cn=config
changetype: modify
replace: ds-cfg-enabled
ds-cfg-enabled: true
-
replace: modifiersName
modifiersName: cn=Directory Manager,cn=Root DNs,cn=config
-
replace: modifyTimestamp
modifyTimestamp: 20081105193257Z
# 05/Nov/2008:13:33:17 -0600; conn=21; op=57
dn: cn=File-Based Debug Logger,cn=Loggers,cn=config
changetype: modify
replace: ds-cfg-enabled
ds-cfg-enabled: true
-
replace: modifiersName
modifiersName: cn=Directory Manager,cn=Root DNs,cn=config
-
replace: modifyTimestamp
modifyTimestamp: 20081105193316Z
...(more output)...
```

▼ To View the Debug Logs

1 **Change to the logs directory of the server instance.**

```
$ cd install-dir/logs
```

2 **Open the `debug` file by using a text editor or the UNIX `cat` command.**

```
$ cat debug | more
[24/Oct/2008:16:02:52 -0500] CONNECT conn=0 from=127.0.0.1 to=127.0.0.1 protocol=LDAP
[24/Oct/2008:16:02:52 -0500] BIND conn=0 op=0 msgID=1 type=SIMPLE dn="cn=Directory
  Manager"
[24/Oct/2008:16:02:53 -0500] BIND conn=0 op=0 msgID=1 result="Success"
authDN="cn=Directory Manager,cn=Root DNs,cn=config" etime=57
...(more output)...
```

▼ To View the Error Logs

1 **Change to the logs directory of the server instance.**

```
$ cd install-dir/logs
```

2 **Open the `error` file by using a text editor or the UNIX `cat` command.**

```
$ cat error | more
[24/Oct/2007:16:02:10 -0500] category=CONFIG severity=NOTICE msgID=3605006
msg=Access control has been enabled and will use the
org.opends.server.authorization.dseecompat.AciHandler implementation
[24/Oct/2007:16:02:17 -0500] category=JEB severity=NOTICE msgID=8847402
msg=The database backend userRoot containing 160 entries has started
[24/Oct/2007:16:02:21 -0500] category=CORE severity=NOTICE msgID=458887
msg=The Directory Server has started successfully
...(more output)...
```

▼ To View the Replication Repair Logs

1 **Change to the logs directory of the server instance.**

```
$ cd install-dir/logs
```

2 **Open the `replication` file by using a text editor or the UNIX `cat` command.**

```
$ cat replication | more
[09/Oct/2007:11:02:22 -0500] category=SYNC severity=NOTICE msgID=15138878
msg=Replication Server sr1-uaus-08/129.123.131.98:8989 now used for Replication
Domain cn=admin data
[09/Oct/2007:11:02:23 -0500] category=SYNC severity=NOTICE msgID=15138878
msg=Replication Server sr1-uaus-08/129.123.131.98:8989 now used for Replication
Domain cn=schema
[09/Oct/2007:11:02:23 -0500] category=SYNC severity=NOTICE msgID=15138878
msg=Replication Server sr1-uaus-08/129.123.131.98:8989 now used for Replication
Domain dc=example,dc=com
...(more output)...
```

▼ To View the server.out Logs

1 **Change to the logs directory of the server instance.**

```
$ cd install-dir/logs
```

2 **Open the `server.out` file by using a text editor or the UNIX `cat` command.**

```
$ cat replication | more
[24/Jun/2009:18:32:18 +0200] category=CORE severity=INFORMATION msgID=132
msg=The Directory Server is beginning the configuration bootstrapping process
[24/Jun/2009:18:32:21 +0200] category=EXTENSIONS severity=INFORMATION msgID=1049147
msg=Loaded extension from file '$SERVER_ROOT/lib/extensions/distribution.jar'
 (build 1.0.0, revision 1057(20090618075417))
[24/Jun/2009:18:32:21 +0200] category=EXTENSIONS severity=INFORMATION msgID=1049147
msg=Loaded extension from file '$SERVER_ROOT/lib/extensions/globalindex.jar'
 (build 1.00, revision 1057(20090618075417))
[24/Jun/2009:18:32:21 +0200] category=EXTENSIONS severity=INFORMATION msgID=1049147
```

```
msg=Loaded extension from file '$SERVER_ROOT/lib/extensions/loadbalancing.jar'
(build 1.0.0, revision 1057(20090618075417))
[24/Jun/2009:18:32:22 +0200] category=EXTENSIONS severity=INFORMATION msgID=1049147
msg=Loaded extension from file '$SERVER_ROOT/lib/extensions/proxyldap.jar'
(build 1.0.0, revision 1057(20090618075417))
[24/Jun/2009:18:32:22 +0200] category=EXTENSIONS severity=INFORMATION msgID=1049147
msg=Loaded extension from file '$SERVER_ROOT/lib/extensions/snmp-mib2605.jar'
(build 2.0.0, revision 5452)
[24/Jun/2009:18:32:25 +0200] category=CORE severity=NOTICE msgID=458886
msg=Sun Virtual Directory Proxy 1.0.0 (OpenDS version = 2.0.0)
(build 20090615154012Z, R5452) starting up
...(more output)...
```

General Purpose Enterprise Monitoring Solutions

You can use a variety of general UNIX tools to monitor your server environment. For information about these tools, see the man pages on your UNIX system.

General UNIX Monitoring Tools

The following general purpose UNIX monitoring tools can be used with the directory server.

Tool	Description
`iostat`	Provides information about disk I/O and CPU usage.
`lsof`	Provides information about open file descriptors.
`lslk`	Provides information about file system locks.
`netstat`	Provides statistics about network functions.
`nslookup`	Allows you to query DNS servers for information about hosts and domains.
`ping`	Allows you to query the status of a remote host or network gateway.
`sar`	UNIX System V performance monitoring tool.
`tcpdump`	Allows you to debug and monitor network traffic.
`top`	Provides quick, easy monitoring of processes and CPU activities.
`trace`	Provides information about which system calls a process makes.

Tool	Description
`traceroute`	Provides the path a packet takes throughout the Internet to reach its final destination.
`vmstat`	Provides statistics about process, virtual memory, disk, trap, and CPU activity.

Solaris Monitoring Tools

The following Solaris monitoring tools can be used with the directory server.

Tool	Description
`lockstat`	Provides information about OS and application locking. Requires DTrace privileges.
`mpstat`	Provides statistics about each processor on the system.
`pmap`	Provides a breakdown of how much memory a process is using.
`proctool`	Monitors processes and threads.
`snoop`	Monitors network traffic. Indispensable when debugging low-level packets.
`SymbEL/Virtual\\Adrian`	Provides functionality of the above listed tools and more.
`truss`	Provides information about which system calls a process makes.

HP-UX Monitoring Tools

The following HP-UX monitoring tools can be used with the directory server.

Tool	Description
`glance`	Provides detailed system information about open file descriptors, locks, and threads.
`gpm`	GlancePlus is a graphical real-time performance diagnostic tool. Glance is the character-based component.
`tusc`	Provides a system call trapper.
`sysdef`	Provides information about kernel parameters.

Tool	Description
`landiag`	Monitors network statistics.
`sam`	Provides a general system administration tool.

Monitoring the Directory Server With JConsole

The JConsole (`jconsole`) Java utility is a JMX-compliant, graphical tool that connects to a running Java Virtual Machine that has been started with the management agent. This generic tool can be used to access directory server monitoring information.

▼ To Configure JMX on a Directory Server Instance

1. **Start the server.**

2. **Enable the JMX Connection Handler and set the port number to be used with JMX.**

   Choose a port that is not in use and to which the user that is running the server has access rights.

   ```
   $ dsconfig -D "cn=directory manager" -w password -n set-connection-handler-prop \
     --handler-name "JMX Connection Handler" --set enabled:true --set listen-port:1689
   ```

3. **Add the JMX read, write, and notify privileges to the root DN.**

   ```
   $ dsconfig -D "cn=directory manager" -w password -n set-root-dn-prop \
     --add default-root-privilege-name:jmx-read \
     --add default-root-privilege-name:jmx-write \
     --add default-root-privilege-name:jmx-notify
   ```

4. **Restart the directory server.**

Starting JConsole

Start the console by typing `jconsole` in a terminal window.

To run `jconsole` from the command line, you might have to add *JAVA_HOME*`/bin` to your path, where *JAVA_HOME* is the directory containing the JDK. Alternatively, you can enter the full path when you type the command.

For more information about using JConsole, see *Using JConsole*: Java 5 version, Java 6 version.

Accessing a Directory Server Instance From JConsole

How you access the directory server from JConsole depends on the version of Java that you use.

Using J2SE 5.0

To connect JConsole to a directory server instance, use the Advanced tab of the Connection window.

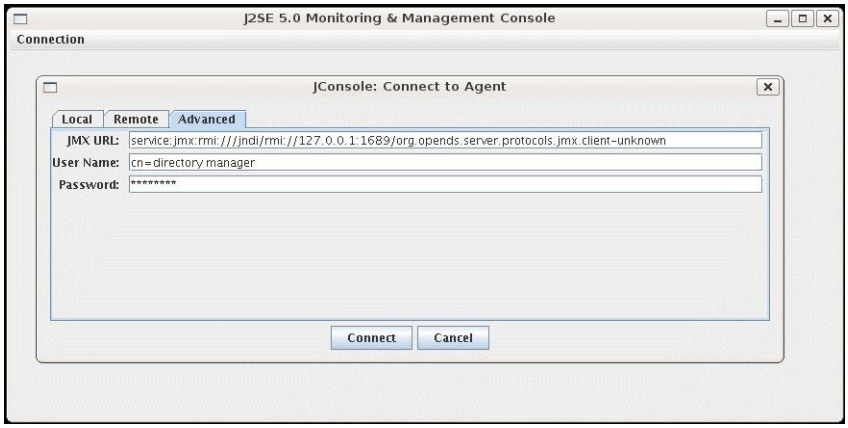


FIGURE 2 Connection screen for JConsole

The following fields are required:

- **JMX URL.**

  `service:jmx:rmi:///jndi/rmi://`*host*`:`*port*`/org.opends.server.protocols.jmx.client-unknown`

  - *host* is a host name, an IPv4 numeric host address, or an IPv6 numeric address enclosed in square brackets.
  - *port* is the decimal port number of the JMX connector. See "To Configure JMX on a Directory Server Instance" on page 328.

  The default JMX URL is `service:jmx:rmi:///jndi/rmi://127.0.0.1:1689/org.opends.server.protocols.jmx.client-unknown`.

- **User Name.** A valid LDAP user name.

  The default Directory Manager user name is `cn=Directory Manager`.

- **Password.** The user's LDAP password.

Using Java 6

To connect JConsole to a directory server instance, use the Remote Process fields.

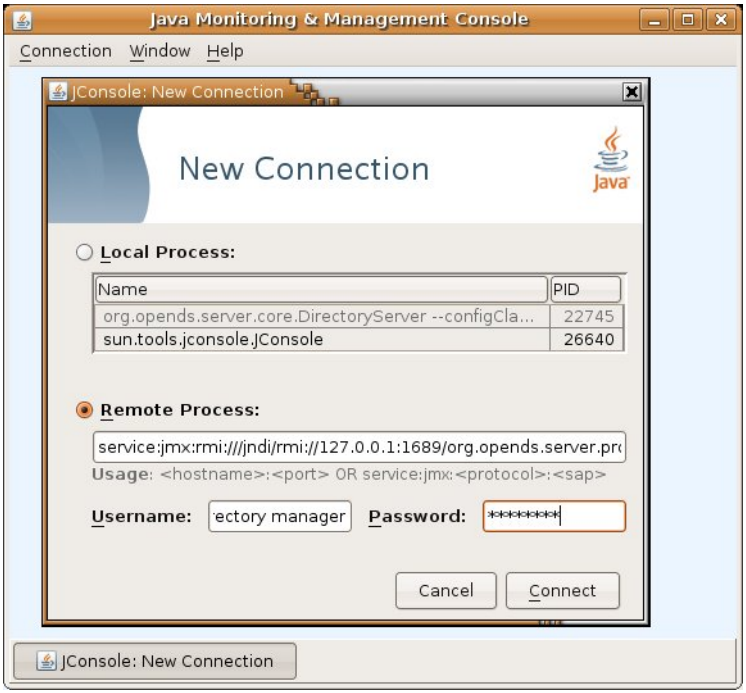


FIGURE 3 New connection to a directory server, using Java 6

The following fields are required:

- **JMX URL**:

  ```
  service:jmx:rmi:///jndi/rmi://''host'':''port''/
  org.opends.server.protocols.jmx.client-unknown
  ```

  - *host* is a host name, an IPv4 numeric host address, or an IPv6 numeric address enclosed in square brackets.
  - *port* is the decimal port number of the JMX connector. (See "Configuring Alerts and Account Status Notification Handlers" on page 342).

  The default JMX URL is:

  ```
  service:jmx:rmi:///jndi/rmi://127.0.0.1:1689/
  org.opends.server.protocols.jmx.client-unknown
  ```

- **User Name.** A valid LDAP user name.

  The default Directory Manager user name is `cn=Directory Manager`.

- **Password.** The user's LDAP password.

Viewing Directory Monitoring Information With JConsole

When JConsole is connected to a directory server instance, it displays management objects (MBeans). The tree on the left pane shows all MBeans currently available. You can access directory server monitoring information in the right pane by selecting the associated MBean.

The following examples show the attribute list for `cn=LDAP Connection Handler 0.0.0.0 port 1389 Statistics,cn=monitor` with the Java 5 and Java 6 `jconsole` implementations.

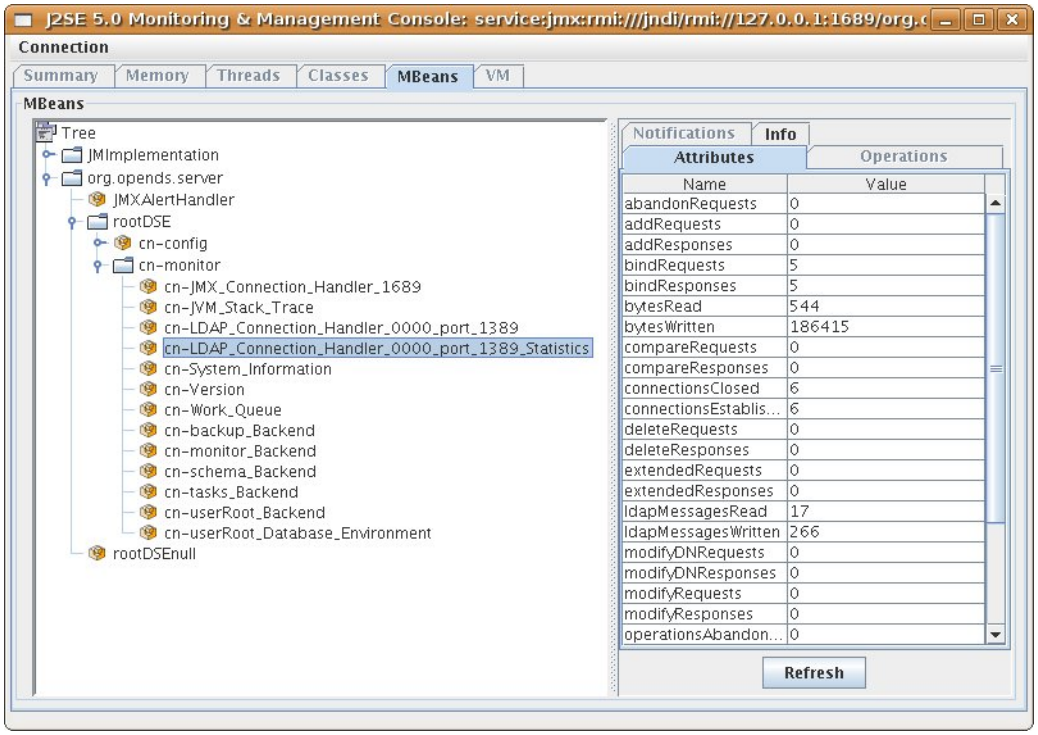


FIGURE 4 Using J2SE 5.0

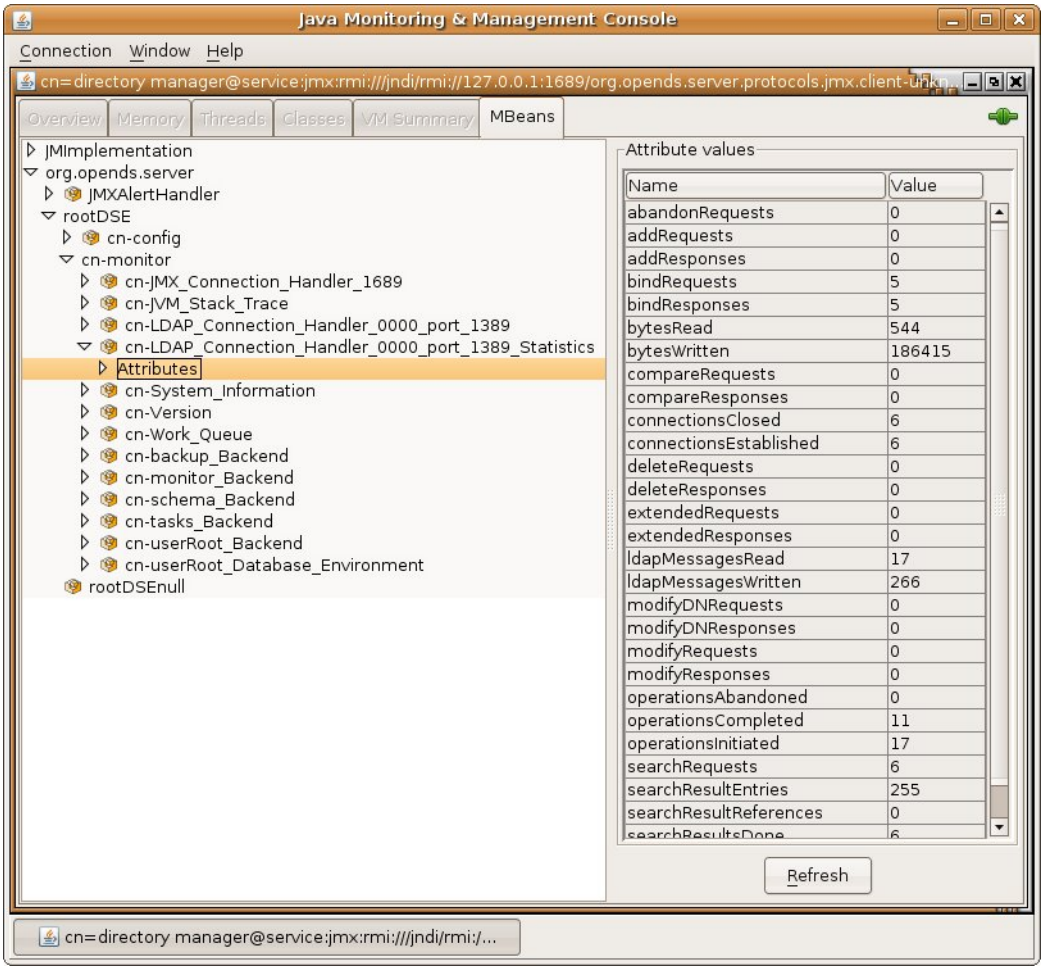


FIGURE 5 Using Java 6

Monitoring the Directory Server With SNMP

The directory server provides a jar file extension that contains a Simple Network Management Protocol (SNMP) connection handler for Management Information Base (MIB) 2605 support. The extension contains the SNMP connection handler, the required classes to support MIB 2605 objects and SNMP requests, and the SNMP adapter that allows an SNMP manager to access the directory server monitoring information.

Before you start on the procedures in this section, ensure that you have set up an SNMP-managed network for your particular system.

Configuring SNMP in the Directory Server

The directory server provides an SNMP connection handler that you can enable and configure. The SNMP connection handler is provided as a jar file extension and is located in *install-dir*`/lib/extensions/snmp-mib2605.jar`.

▼ To Configure SNMP in the Directory Server

The directory server can be configured for monitoring through the Simple Network Management Protocol (SNMP). The directory server uses the Java Dynamic Management Kit (JDMK) to create smart agents for the SNMP connection handler. To enable SNMP for your directory server, you must specify the file path for the JDMK jar file (`jdmkrt.jar`) that is bundled with the product distribution.

1 Verify that you have the SNMP connection handler.

Use `dsconfig` to view the list of current connection handlers.

```
$ dsconfig -D "cn=directory manager" -w password -n list-connection-handlers
Connection Handler       : Type : enabled : listen-port : use-ssl
-------------------------:------:---------:-------------:--------
JMX Connection Handler   : jmx  : false   : 1689        : false
LDAP Connection Handler  : ldap : true    : 1389        : false
LDAPS Connection Handler : ldap : false   : 636         : true
LDIF Connection Handler  : ldif : true    : -           : -
SNMP Connection Handler  : snmp : false   : 161         : -
```

2 Use the `dsconfig` command to enable SNMP for your directory server.

```
$ dsconfig -D "cn=Directory Manager" -w password -n set-connection-handler-prop \
  --handler-name "SNMP Connection Handler" --set enabled:true --set listen-port:8085 \
  --set opendmk-jarfile:install-dir/addons/jdmkrt.jar
```

▼ To View the SNMP Connection Handler Properties

- **Use the following `dsconfig` command.**

```
$ dsconfig -D "cn=directory manager" -w password -n get-connection-handler-prop \
  --handler-name "SNMP Connection Handler"
Property            : Value(s)
--------------------:----------------------------------------
allowed-client      : -
allowed-manager     : *
allowed-user        : *
community           : OpenDS
denied-client       : -
enabled             : false
listen-port         : 161
opendmk-jarfile     : -
```

```
registered-mbean    : false
security-agent-file : config/snmp/security/opends-snmp.security
security-level      : authnopriv
trap-port           : 162
traps-community     : OpenDS
traps-destination   : -
```

▼ To Access SNMP on a Directory Server Instance

1 Restart the server by using `stop-ds` and `start-ds`.

If the server was started and no modifications were made to the configuration, the restart operation is not required.

2 Check that your SNMP Connection Handler is up and running.

```
$ snmpwalk -v 2c -c OpenDS@OpenDS localhost:8085 mib-2.66
SNMPv2-SMI::mib-2.66.1.1.1.1 = STRING: "OpenDS Directory Server 1.3.0 -
  20090310152800Z"
SNMPv2-SMI::mib-2.66.1.1.2.1 = STRING: "Sun-OpenDS-SE-installation-directory/bin"
SNMPv2-SMI::mib-2.66.1.1.3.1 = Gauge32: 35
SNMPv2-SMI::mib-2.66.1.1.4.1 = Gauge32: 1
SNMPv2-SMI::mib-2.66.1.1.5.1 = Gauge32: 0
SNMPv2-SMI::mib-2.66.1.1.6.1 = Counter32: 0
SNMPv2-SMI::mib-2.66.1.1.7.1 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.1.1.1 = INTEGER: 1
SNMPv2-SMI::mib-2.66.2.1.1.1.2 = INTEGER: 2
SNMPv2-SMI::mib-2.66.2.1.1.1.3 = INTEGER: 3
SNMPv2-SMI::mib-2.66.2.1.2.1.1 = OID: SNMPv2-SMI::internet.27.3.8085
SNMPv2-SMI::mib-2.66.2.1.2.1.2 = OID: SNMPv2-SMI::internet.27.3.1389
SNMPv2-SMI::mib-2.66.2.1.2.1.3 = OID: SNMPv2-SMI::enterprises.42
SNMPv2-SMI::mib-2.66.2.1.3.1.1 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.3.1.2 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.3.1.3 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.4.1.1 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.4.1.2 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.4.1.3 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.5.1.1 = Counter32: 1
SNMPv2-SMI::mib-2.66.2.1.5.1.2 = Counter32: 1
...
```

The managed objects included in the MIB 2605 are divided into three tables: `dsTable`, `dsAppliIfOpsTable`, and `dsIntTable`. Currently, the `dsIntTable` table is not implemented.

SNMP Security Configuration

SNMP security configuration depends on the version of SNMP as you are using. This topic discusses security configuration for SNMP V1 and V2c, and vor V3.

SNMP Security Configuration : V1 and V2c

Under SNMP v1 and SNMP v2c, agents act as information servers, and the IP-based access control protects this information from unauthorized access. By default, the MIB 2605 is accessible in v1 and v2c by using the community string `OpenDS@OpenDS`. All managers are allowed to read the monitoring information exposed by the MIB 2605.

Note – Only read access is authorized on the MIB 2605.

You can configure SNMP v1 and SNMP v2c by setting the SNMP connection handler properties with the `dsconfig` command. Properties related to the SNMP v1 and SNMP v2c security configuration include:

- `allowed-manager`
- `community`

SNMP v1 traps are sent on server startup and server shutdown. By default, these traps are sent to `localhost` and use the trap community string "OpenDS".

Note – The default trap port might have to be changed to a value that is allowed by the system.

SNMP traps are also configured by setting the SNMP connection properties with the `dsconfig` command. Properties related to SNMP traps include:

- `trap-port`
- `traps-community`
- `traps-destination`

The ACL file that corresponds to the default values of the SNMP connection handler would be represented as follows:

```
acl = {
{
communities = OpenDS
access = read-only
managers = all
}
}
trap = {
{
traps-community = OpenDS
hosts = localhost
}
}
```

SNMP Security Configuration : V3

The SNMP v3 protocol provides more sophisticated security mechanisms than SNMP v1 and SNMP v2c. SNMP v3 implements a user-based security model (USM) that authenticates and encrypts the requests sent between agents and their managers, and provides user-based access control. A `defaultUser` template is provided for adding authorized users in the agent engine using the SNMP cloning mechanism.

Under SNMP v3, the community string described in the previous section is used as the "context" from which the MIB 2605 is registered. By default, the MIB2605 is accessible in v3 by using the context "OpenDS". All users have access to it.

The SNMP v3 UACL is configured by setting the SNMP connection handler properties with the `dsconfig` command-line utility. The properties related to SNMP v3 UACL configuration include:

- community
- allowed-user
- security-level

The UACL file corresponding to the default values of the SNMP connection handler would be represented as follows:

```
uacl = {
{
context-names = OpenDS
access = read-only
security-level = authNoPriv
users = *
}
}
```

SNMP USM Configuration: V3

The USM MIB (that is, the MIB that defines allowed users) is registered in the `null` context and only a `snmpAdmin` user with a security level `authNoPriv` has `read-write` access to it. This `snmpAdmin` user can add additional users who can access the MIB 2605 information.

The SNMP v3 USM configuration is read from a template file that is located at *install-dir*`/config/snmp/security/opends-snmp.security`. The template file is not encrypted.

To access the MIB 2605 in the directory server agent, use the SNMP clone mechanism to add a user in the security file. Use `snmpAdmin` to send the SNMP request for the clone mechanism as shown here. The user to clone is `defaultUser`. The `snmpAdmin` and `defaultUser` users cannot access the MIB 2605 information.

- Admin User to add and configure other users.

```
userEntry=localEngineID,snmpAdmin,null,usmHMACMD5AuthProtocol,passadmin
```

- Template user to be cloned with no read or write access.

```
userEntry=localEngineID,defaultUser,,usmHMACMD5AuthProtocol,password,,,3,true
```

Note – The security file is also used to make the users persistent.

Monitoring the Directory Server With the Control Panel

You can use the Control Panel to view monitoring information.

▼ To View Monitoring Information With the Control Panel

1. **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**
2. **Click one of the links under the Monitoring menu on the left side of the Control Panel window to display one of the monitoring windows, as shown in Figure 6 and Figure 7:**
   - General Information
   - Connection Handler

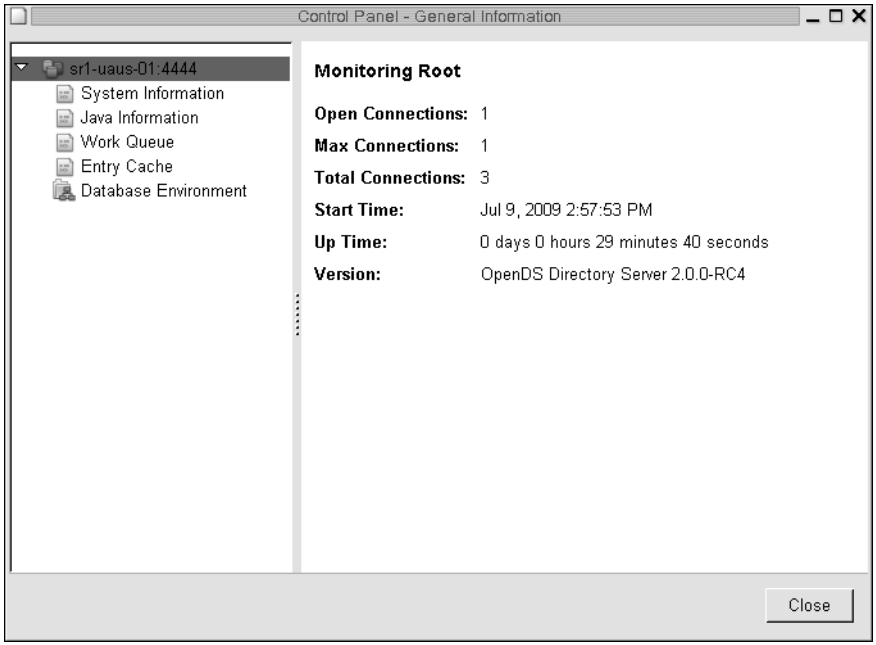


FIGURE 6 The Monitoring General Information Window of the Control Panel

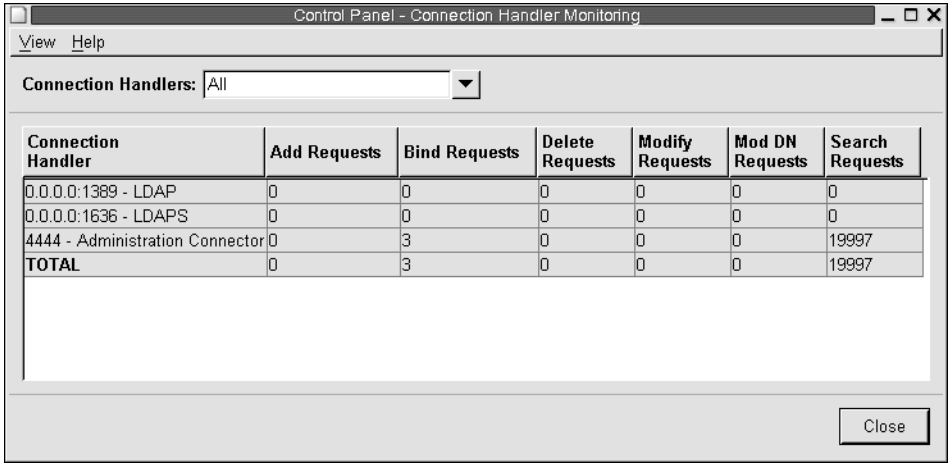


FIGURE 7 The Monitoring Connection Handler Window of the Control Panel

Configuring Logs With `dsconfig`

The directory server provides several kinds of logs: access logs, audit logs, error logs, debug logs, and a replication repair log. The replication repair log is read-only and its use is restricted to enabling replication conflict resolution. This section describes how to use `dsconfig` to configure access, audit, error, and debug logs.

Overview of Directory Server Logs

The easiest way to configure logging is to use the `dsconfig` command in interactive mode, which walks you through the configuration. Log configuration includes the definition of three configuration objects:

- **Log publisher.** A log publisher is defined for each logger. The log publisher type corresponds to the type of log. Currently, the directory server supports the following log publisher types:
  - File-based access
  - File-base audit
  - File-based debug
  - File-based error

  Any number of log publishers of any type can be defined and active at any time. This means that you can log to different locations or different types of repositories and that you can specify various sets of criteria for what to include in the logs.
- **Log retention policy.** The retention policy determines how long archived log files are stored.
- **Log rotation policy.** The rotation policy determines how often log files are rotated.

In addition, debug logs require the configuration of a debug target.

Configuring Log Publishers

The directory server provides several log publishers by default. To list the existing log publishers, type:

```
$ dsconfig list-log-publishers
```

To display the properties of a log publisher, type:

```
$ dsconfig get-log-publisher-prop
```

To create a new log publisher, type:

```
$ dsconfig create-log-publisher
```

Follow the prompts to configure the log publisher according to your requirements.

By default, no retention or rotation policies are configured for a log publisher. You can add one or more retention and rotation policies while creating the log publisher, or you can configure the retention and rotation policies later.

To configure an existing log publisher, type:

```
$ dsconfig set-log-publisher-prop
```

Check the configuration and make any additional configuration changes before you finish.

For more information about the configuration properties associated with log publishers, see the Log Publisher Configuration in *Sun OpenDS Standard Edition 2.0 Configuration Reference*.

Logging Internal Operations

By default, the `suppress-internal-logging` property for log publishers is set to `true`. If you need to log internal operations (such as operations performed by the LDIF connection handler and certain plug-ins), set `suppress-internal-logging` to `false`. The following example sets `suppress-internal-logging` to `false` for the file-based access logger:

```
$ dsconfig -D "cn=directory manager" -w password -n set-log-publisher-prop \
  --advanced --publisher-name "File-Based Access Logger" \
  --set suppress-internal-operations:false
```

▼ To Configure Log Retention Policies

1. **Create the log retention policy.**

   ```
   $ dsconfig create-log-retention-policy
   ```

2. **Follow the prompts to configure the retention policy according to your requirements.**

3. **Check the configuration and make any additional configuration changes before you finish.**

▼ To Configure Log Rotation Policies

1. **Create the log rotation policy.**

   ```
   $ dsconfig create-log-rotation-policy
   ```

2. **Follow the prompts to configure the rotation policy according to your requirements.**

3. **Check the configuration and make any additional configuration changes before you finish.**

▼ To Configure Debug Targets

Debug targets allow for fine-grained control of which messages are logged based on the package, class, or method that generated the message.

Note – Log file names include a time stamp with the suffix `Z`, indicating that the UTC (+0000) time zone is used.

---

1 Create a debug target.

```
$ dsconfig create-debug-target
```

2 Follow the prompts to configure the debug target according to your requirements.

The name of the debug target determines the directory server Java package, class, or method that is affected by the settings in this target definition.

For more information about the configuration properties associated with debug targets, see "Debug Log Publisher Configuration" in *Sun OpenDS Standard Edition 2.0 Configuration Reference*.

Logging Access Control Information

To obtain information on access control in the error logs, set the appropriate log level.

The following example shows how to configure the default file-based error logger to log access control messages using the default severity levels used for other log messages.

```
$ dsconfig -D "cn=directory manager" -w password -n set-log-publisher-prop \
  --publisher-name "File-Based Error Logger" \
  --set override-severity:access-control=severe-warning,severe-error,fatal-error,notice
```

For a complete list of configurable properties, see "Error Log Publisher Configuration" in *Sun OpenDS Standard Edition 2.0 Configuration Reference*.

Differences Between Logging in Sun OpenDS Standard Edition and Sun Java System Directory Server

The logging architecture of the Sun OpenDS Standard Edition directory server and the Sun Java System Directory Server differ significantly. The most notable differences include the following:

- The Sun OpenDS Standard Edition directory server allows for several loggers of any type to be defined and active at any time. This feature makes it possible to log messages to different locations and to different types of repositories. You can also define different sets of criteria for what to include in the logs. For example, one access log might hold everything, another might hold operations with a non-zero result code only, and yet another might hold write operations only.
- The Sun Java System Directory Server defines an *audit logger* that is used to hold information about the changes that are made to directory data. In the Sun OpenDS Standard Edition directory server, the audit logging capability is still present, but it is classified as a type of access logger.

Note – The current Sun OpenDS Standard Edition logging mechanism cannot easily be used to define filters that restrict the types of content to include in the log.

Configuring Alerts and Account Status Notification Handlers

You can configure the directory server to send alert notifications when an event occurs during processing. Typical server events include server starts and shut downs, or problems that are detected by the server, such as an attempt to write to the configuration file.

Alerts and account status notification handlers are configured by using the `dsconfig` command. For more information, see "Configuring the Directory Server With `dsconfig`" on page 7.

For additional information about the topics in this section, see "Managing Password Policies" on page 272 and "The Alert Handler Configuration" in *Sun OpenDS Standard Edition 2.0 Configuration Reference*.

Managing Alert Handlers

The directory server supports the following alert handlers:

- JMX alert handler for JMX notifications
- SMTP alert handler for email notifications.

- Custom alert handlers

To View All Configured Alert Handlers

The directory server stores alert handlers information in the configuration file under the `cn=Alert Handlers,cn=config` subtree. You can access the information using the `dsconfig` command.

- **To display a list of alert handlers, run the following `dsconfig` command:**

```
$ dsconfig -D "cn=directory manager" -w password -n list-alert-handlers
Alert Handler     : Type : enabled
------------------:------:--------
JMX Alert Handler : jmx  : false
```

To Enable an Alert Handler

The JMX alert handler is disabled by default. Before you begin, you must configure JMX on the directory server. For more information, see "Monitoring the Directory Server With JConsole" on page 328.

1. **To list the alert handler's properties, use the `dsconfig` command as follows.**

```
$ dsconfig -D "cn=directory manager" -w password -n get-alert-handler-prop \
  --handler-name "JMX Alert Handler"
Property            : Value(s)
--------------------:---------------------------------------------
disabled-alert-type : -
enabled             : false
enabled-alert-type  : -
```

2. **To enable the alert handler, use `dsconfig` as follows.**

```
$ dsconfig -D "cn=directory manager" -w password -n set-handler-prop \
  --handler-name "JMX Alert Handler" --set enabled:true
```

3. **(Optional) Verify the change by using `dsconfig`.**

```
$ dsconfig -D "cn=directory manager" -w password -n get-alert-handler-prop \
  --handler-name "JMX Alert Handler"
Property            : Value(s)
--------------------:---------------------------------------------
disabled-alert-type : -
enabled             : true
enabled-alert-type  : -
```

To Create a New Alert Handler

You can create a new alert handler by using `dsconfig`. This example configures a new SMTP handler. Before starting this procedure, ensure that you have configured an SMTP server for your directory.

1. **Use `dsconfig` with the `create-alert-handler` subcommand to create the handler.**

```
$ dsconfig -D "cn=directory manager" -w password -n create-alert-handler \
  --handler-name "my SMTP Handler" --type smtp --set enabled:true \
  --set message-body:"Alert Type: %%alert-type%%\n\nAlert ID: \
  %%alert-id%%\n\nAlert Message: %%alert-message%%" \
  --set message-subject:"Alert Message" \
  --set recipient-address:directorymanager@example.com \
  --set sender-address:OpenDS-Alerts@directory.example.com
```

2. **(Optional) View the list of alert handlers by using `dsconfig`.**

```
$ dsconfig -D "cn=directory manager" -w password -n list-alert-handlers
```

To Delete an Alert Handler

Note that you can *disable* an alert handler instead of deleting it. In this case, the alert handler is available if you need to enable it again in the future. This example removes an alert handler from the directory server.

- **Use the following `dsconfig` command.**

```
$ dsconfig -D "cn=directory manager" -w password -n delete-alert-handler \
  --handler-name "JMX Alert Handler"
```

Supported Alert Types

The directory server sends out message alerts when an alert type event occurs in the system. The supported alert types are defined in the following table.

Alert Type	Java Class	Description
Access Control Disabled	org.opends.server.AccessControlDisabled	Notify administrator that the access control handler has been disabled.
Access Control Enabled	org.opends.server.Enabled	Notify administrator that the access control handler has been enabled.

Alert Type	Java Class	Description
Access Control Parse Failed	org.opends.server.authentication.dseecompat.AciParseFailed	Notify administrator that the DSEE compatible access control subsystem failed to correctly parse one or more ACI rules when the server is first started.
Backend Environment Unusable	org.opends.server.BackendRunRecovery	Notify administrator that the JE back end throws a `RunRecoveryException` and the directory server needs to be restarted.
Cannot Copy Schema Files	org.opends.server.CannotCopySchemaFiles	Notify administrator if a problem occurs while attempting to create copies of the existing schema configuration before making a schema update, and the schema configuration is left in a potentially inconsistent state.
Cannot Find Recurring Task	org.opends.server.CannotFindRecurringTask	Notify administrator if the directory server is unable to locate a recurring task definition in order to schedule the next iteration once the previous iteration has completed.
Cannot Rename Current Task File	org.opends.server.CannotRenameCurrentTaskFile	Notify administrator if the directory server is unable to rename the current tasks backing file in the process of trying to write an updated version.
Cannot Rename New Task File	org.opends.server.CannotRenameNewTaskFile	Notify administrator if the directory server is unable to rename the new tasks backing file into place.
Cannot Schedule Recurring Iteration	org.opends.server.CannotScheduleRecurringIteration	Notify administrator if the directory server is unable to schedule an iteration of a recurring task.
Cannot Write Configuration	org.opends.server.CannotWriteConfig	Notify administrator if the directory server is unable to write its updated configuration for some reason and so the server cannot exhibit the new configuration if it is restarted.

Alert Type	Java Class	Description
Cannot Write New Schema Files	org.opends.server.CannotWriteNewSchemaFiles	Notify administrator if a problem occurs while attempting to write new versions of the server schema configuration files, and the schema configuration is left in a potentially inconsistent state.
Cannot Write Task File	org.opends.server.CannotWriteTaskFile	Notify administrator if the directory server is unable to write an updated tasks backing file for some reason.
Entering Lockdown Mode	org.opends.server.EnteringLockdownMode	Notify administrator that the directory server is entering lockdown mode, in which only root users will be allowed to perform operations and only over the loopback address.
LDAP Connection Handler Consecutive Failures	org.opends.server.LDAPHandlerDisabledByConsecutiveFailures	Notify administrator of consecutive failures that have occurred in the LDAP connection handler that have caused it to become disabled.
LDAP Connection Handler Uncaught Error	org.opends.server.LDAPHandlerUncaughtError	Notify administrator of uncaught errors in the LDAP connection handler that have caused it to become disabled.
LDIF Backend Cannot Write Update	org.opends.server.LDIFBackendCannotWriteUpdate	Notify administrator that an LDIF back end was unable to store an updated copy of the LDIF file after processing a write operation.
LDIF ConnHandler Parse Error	org.opends.server.LDIFConnectionHandlerParseError	Notify administrator that the LDIF connection handler encountered an unrecoverable error while attempting to parse an LDIF file.
LDIF ConnHandler IO Error	org.opends.server.LDIFConnectionHandlerIOError	Notify administrator that the LDIF connection handler encountered an I/O error that prevented it from completing its processing.
Leaving Lockdown Mode	org.opends.server.LeavingLockdownMode	Notify administrator that the directory server is leaving lockdown mode.

Alert Type	Java Class	Description
Manual Config Edit Handled	org.opends.server.ManualConfigEditHandled	Notify administrator if the directory server detects that its configuration has been manually edited with the server online and those changes were overwritten by another change made through the server. The manually-edited configuration will be copied off to another location.
Manual Config Edit Lost	org.opends.server.ManualConfigEditLost	Notify administrator if the directory server detects that its configuration has been manually edited with the server online and those changes were overwritten by another change made through the server. The manually-edited configuration could not be preserved due to an unexpected error.
Replication Unresolved Conflict	org.opends.server.replication.UnresolvedConflict	Notify administrator if the multimaster replication cannot automatically resolve a conflict.
Server Started	org.opends.server.DirectoryServerStarted	Notify administrator that the directory server has completed its startup process.
Server Shutdown	org.opends.server.DirectoryServerShutdown	Notify administrator that the directory server has begun the process of shutting down.
Uncaught Exception	org.opends.server.UncaughtException	Notify administrator if a directory server thread has encountered an uncaught exception that caused the thread to terminate abnormally. The impact that this problem has on the directory server depends on which thread was impacted and the nature of the exception.
Unique Attr Sync Conflict	org.opends.server.UniqueAttributeSynchronizationConflict	Notification that a unique attribute conflict has been detected during synchronization processing.

Alert Type	Java Class	Description
Unique Attr Sync Error	org.opends.server.UniqueAttributeSynchronizationError	Notification that an error occurred while attempting to perform unique attribute conflict detection during synchronization processing.

▼ To Disable an Alert Type

By default, all alert types are allowed. If you specify a value for the `enabled-alert-type` property, only alerts with one of those types are allowed. If you specify a value for the `disabled-alert-type` property, all alert types except for the values in that property are allowed. Alert types are specified by their Java class, as shown in this example.

- **To disable an alert type, specify its Java class as a value of the `disabled-alert-type` property.**

  This command disables the startup alert from the JMX Alert Handler.

```
$ dsconfig -D "cn=directory manager" -w password -n set-alert-handler-prop \
  --handler-name "JMX Alert Handler" \
  --set disabled-alert-type:org.opends.server.DirectoryServerStarted -n
```

Managing Account Status Notification Handlers

Account status notification handlers provide alerts on events during password policy processing. By default, the Error Log Account Status Notification handler is set to enabled upon initial configuration. The server writes a message to the server error log when one of the following events has been configured in the password policy and occurs during the course of password policy processing:

- `account-temporarily-locked`
- `account-permanently-locked`
- `account-unlocked`
- `account-idle-locked`
- `account-reset-locked`
- `account-disabled`
- `account-expired`
- `password-expired`
- `password expiring`
- `password-reset`
- `password-changed`

The error log is located at *install-dir*`/logs/errors`.

To View the Configured Account Status Notification Handlers

- **Use `dsconfig` with the `list-account-status-notification-handlers` subcommand.**

```
$ dsconfig -D "cn=directory manager" -w password -n \
  list-account-status-notification-handlers
Account Status Notification Handler : Type      : enabled
------------------------------------:-----------:--------
Error Log Handler                   : error-log : true
SMTP Handler                        : smtp      : false
```

To Enable Account Status Notification Handlers

You can enable an existing account status notification handler using the `dsconfig` command. By default, the directory server enables the Error Log Handler when the server is initially configured. This example enables the SMTP notification handler.

1. **Use `dsconfig` with the `get-account-status-notification-handler-prop` subcommand to view the `enabled` property.**

```
$ dsconfig -D "cn=directory manager" -w password -n \
  get-account-status-notification-handler-prop --handler-name "SMTP Handler" \
--property enabled
Property : Value(s)
---------:---------
enabled  : false
```

2. **2. Use `dsconfig` with the `set-account-status-notification-handler-prop` subcommand to enable the notification handler.**

```
$ dsconfig -D "cn=directory manager" -w password -n \
  set-account-status-notification-handler-prop --handler-name "SMTP Handler" \
  --set property:enabled
```

To Create a New Account Status Notification Handler

1. **Use `dsconfig` with the `create-account-status-notification-handler` subcommand to create the handler.**

   When you specify the type, you can use either `error-log` or `generic` (default).

```
$ dsconfig -D "cn=directory manager" -w password -n \
  create-account-status-notification-handler \
  --handler-name "My Password Reset Logger" --type error-log --set enabled:true \
  --set account-status-notification-type:password-reset
```

2. **(Optional) Use `dsconfig` to view the list of account status notification handlers.**

```
$ dsconfig -D "cn=directory manager" -w password -n \
  list-account-status-notification-handlers
```

```
Account Status Notification Handler : Type      : enabled
------------------------------------:-----------:--------
Error Log Handler                   : error-log : true
my Password Reset Logger            : error-log : true
SMTP Handler                        : smtp      : false
```

▼ To Delete an Account Status Notification Handler

You can disable an account status notification handler instead of deleting it. In this case, the alert handler is available if you need to enable it again in the future.

You can remove an account status notification handler entirely by using `dsconfig`.

- **Use `dsconfig` with the `delete-account-status-notification-handler` subcommand.**

```
$ dsconfig -D "cn=directory manager" -w password -n \
  delete-account-status-notification-handler --handler-name "My Password Reset Logger"
```

Monitoring a Replicated Topology

These topics describe how to monitor a replicated topology by using the `dsreplication status` command, and how to use the `ldapsearch` command to obtain more advanced monitoring information.

Monitoring Replication Status With `dsreplication`

The simplest way to monitor replication is to use the `dsreplication status` command. This command provides a tabular view of the replication status, including the following information:

- The topology and its connections
- The latency between replicated servers
- The data consistency across replicated servers
- The security configuration between replicated servers
- The replication protocol peer to peer

The examples in the remainder of this section assume the following simple replication topology.

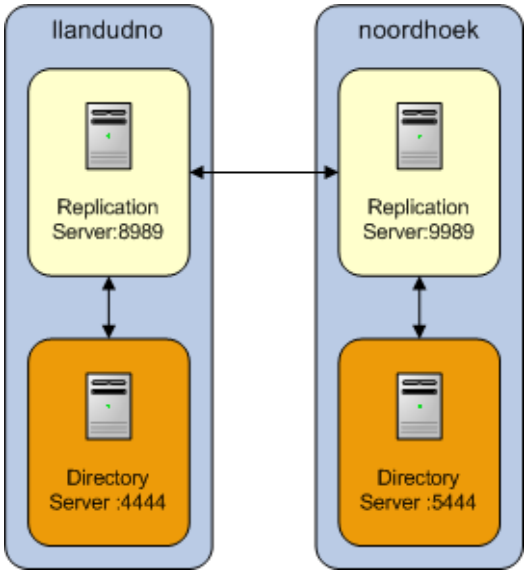


FIGURE 8 Simple replication topology

To obtain the replication status, run the following command:

```
$ dsreplication status -h llandudno -p 4444 -I admin -w password -X -n
dc=example,dc=com - Replication Enabled
======================================
Server         : Entries : M.C. (1) : A.O.M.C. (2) : Port (3) : Security (4)
---------------:---------:----------:--------------:----------:-------------
llandudno:4444 : 2002    : 0        : N/A          : 8989     : Disabled
noordhoek:5444 : 2002    : 0        : N/A          : 9989     : Disabled
[1] The number of changes that are still missing on this server (and that have
been applied to at least one of the other servers).
[2] Age of oldest missing change: the date on which the oldest change that has
not arrived on this server was generated.
[3] The port used to communicate between the servers whose contents are being
replicated.
[4] Whether the replication communication through the replication port is
encrypted or not.
```

The output of this command includes the following:

- **Server.** Lists the LDAP servers in the topology and the port on which they are listening for LDAP connections.
- **Entries.** Indicates the number of entries on each server for the specified base DN. If the information in this column is not the same across all the servers, the replication topology is not synchronized.

- **M.C.** Indicates the number of updates already pushed by the other LDAP servers in the topology, but not yet replayed on the specified LDAP server. If this number is high on a particular server, investigate the latency of that server.
- **A.O.M.C.** Specifies the approximate date of the oldest update pushed by the other directory servers in the topology, but not yet processed on the specified LDAP server.
- **Port.** Indicates the port of the replication server to which the specified LDAP server is directly connected.
- **Security.** Indicates whether SSL encryption is enabled between the LDAP server and its replication server.

Note – Additional replication monitoring information is available under the `cn=monitor` entry. You can use the `ldapsearch` command to track specific monitoring attributes, which will provide you with a comprehensive view of the replication status. For more information, see "Advanced Replication Monitoring" on page 352.

Advanced Replication Monitoring

The easiest way to monitor replication status is by using the `dsreplication status` command. However, in depth replication monitoring information is available under the `cn=monitor` entry. You can use the `ldapsearch` command to track specific monitoring attributes, which provide you with a comprehensive view of the replication status. Monitoring information is consolidated by replication servers. Therefore, monitoring information can only be retrieved by searching a directory server that hosts a running replication server.

The examples in the remainder of this section assume the following simple replication topology.

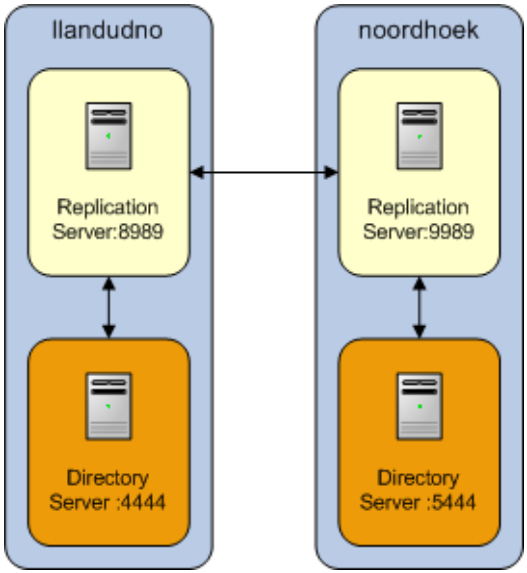


FIGURE 9 Simple replication topology

These examples access the `cn=monitor` entry on the administration port over SSL (`--useSSL`) and automatically trust the certificate that is presented by the server (`--trustAll`).

The information under `cn=monitor` can be filtered to include a single replicated base DN. You can do this in two ways:

- Specify the `domain-name` attribute as a filter, for example:

```
$ ldapsearch -p 4444 --useSSL --trustAll -b "cn=monitor" \
  "(domain-name=dc=example,dc=com)"
```

- Include the base DN in the search base, for example:

```
$ ldapsearch -p 4444 --useSSL --trustAll \
  -b "cn=dc_example_dc_com,cn=replication,cn=monitor" "(objectclass=*)"
```

To Monitor the Topology and Its Connections

Each directory server contains a list of candidate replication servers for each replicated base DN. However, a directory server is *connected* to only one replication server at a time.

To obtain an overview of the replication topology and its connections, run the following search on any directory server in the topology that hosts a replication server:

```
$ ldapsearch -p 4444 --useSSL --trustAll -b "cn=monitor" "(connected-to=*)" \
  "connected-to" "lost-connections"
```

```
dn: cn=Replication Domain 30839,cn=dc_example_dc_com,cn=replication,cn=monitor
lost-connections: 0
connected-to: llandudno/0:0:0:0:0:0:0:1:8989

dn: cn=Replication Domain 14142,cn=cn_schema,cn=replication,cn=monitor
lost-connections: 0
connected-to: llandudno/0:0:0:0:0:0:0:1:8989

dn: cn=Connected Replica llandudno 27742,cn=Replication Server 8989  1740,cn=cn_
admin data,cn=replication,cn=monitor
connected-to: Replication Server 8989 1740

dn: cn=Connected Replica llandudno 30839,cn=Replication Server 8989  1740,cn=dc_
example_dc_com,cn=replication,cn=monitor
connected-to: Replication Server 8989 1740

dn: cn=Connected Replica llandudno 14142,cn=Replication Server 8989  1740,cn=cn_
schema,cn=replication,cn=monitor
connected-to: Replication Server 8989 1740

dn: cn=Undirect Replica  22052,cn=Connected Replication Server noordhoek:9989 71
64,cn=Replication Server 8989  1740,cn=cn_schema,cn=replication,cn=monitor
connected-to: Connected Replication Server noordhoek:9989 7164,cn=Replication Se
rver 8989  1740,cn=cn_schema,cn=replication

dn: cn=Undirect Replica  19984,cn=Connected Replication Server noordhoek:9989 71
64,cn=Replication Server 8989  1740,cn=dc_example_dc_com,cn=replication,cn=moni
tor
connected-to: Connected Replication Server noordhoek:9989 7164,cn=Replication Se
rver 8989  1740,cn=dc_example_dc_com,cn=replication

dn: cn=Undirect Replica  30030,cn=Connected Replication Server noordhoek:9989 71
64,cn=Replication Server 8989  1740,cn=cn_admin data,cn=replication,cn=monitor
connected-to: Connected Replication Server noordhoek:9989 7164,cn=Replication Se
rver 8989  1740,cn=cn_admin data,cn=replication

dn: cn=Replication Domain 27742,cn=cn_admin data,cn=replication,cn=monitor
lost-connections: 0
connected-to: llandudno/0:0:0:0:0:0:0:1:8989
```

The `connected-to` attribute specifies the replication server to which each directory server is currently connected for a particular base DN. If a directory server is directly connected to the replication server, its DN includes `cn=Connected Replica`. A directory server that is in the topology but is connected to a different replication server has `cn=Undirect Replica` in its DN. Because all replication servers are permanently connected to all other replication servers, the `connected-to` attribute does not exist for replication servers.

The `lost-connections` attribute indicates the number of connection breaks between directory servers and replication servers. The value of this attribute on each directory server should be

close to the number of times that replication has been stopped on that server. If the value of this attribute is much higher, there are unexpected connection losses that must be investigated.

To Monitor Replication Latency

Monitoring replication latency enables you to establish whether a specific replication server is lagging behind other servers in the topology. This provides a complete view of any replication delays and the current quality of service.

To monitor replication latency, run the following search on any server in the topology that hosts a replication server:

```
$ ldapsearch -p 4444 --useSSL --trustAll -b "cn=monitor" \
  "domain-name=dc=example,dc=com" "missing-changes" \
  "approx-older-change-not-synchronized"
dn: cn=Replication Domain 30839,cn=dc_example_dc_com,cn=replication,cn=monitor

dn: cn=Replication Server 8989 1740,cn=dc_example_dc_com,cn=replication,cn=monitor
missing-changes: 0

dn: cn=Connected Replica llandudno 30839,cn=Replication Server 8989  1740,cn=dc_
example_dc_com,cn=replication,cn=monitor
missing-changes: 0

dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 8989
  1740,cn=dc_example_dc_com,cn=replication,cn=monitor
missing-changes: 0

dn: cn=Undirect Replica 19984,cn=Connected Replication Server noordhoek:9989
7164,cn=Replication Server 8989 1740,cn=dc_example_dc_com,cn=replication,cn=monitor
missing-changes: 0
```

The `missing-changes` attribute specifies the number of updates already pushed by the other directory servers in the topology, but not yet replayed on the specified directory server.

The `approx-older-change-not-synchronized` attribute specifies the approximate date of the oldest update pushed by the other directory servers in the topology, but not yet processed on the specified directory server.

Note – If the replication latency, as defined by these attributes, is high, look at the number of updates sent and received to identify the servers in the topology that are causing the latency. These attributes are described later in this document.

To Monitor Data Consistency

Monitoring data consistency enables you to establish whether each replication server in the topology is synchronized and up-to-date with the latest changes that have occurred in the topology.

To monitor the data consistency across the directory servers in the topology, run the following search on any server in the topology that hosts a replication server:

```
$ ldapsearch -p 4444 --useSSL --trustAll -b "cn=monitor" "(generation-id=*)" \
  "generation-id"
dn: cn=Replication Server 8989  1740,cn=cn_admin data,cn=replication,cn=monitor
generation-id: cn=admin data 94310

dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 8989
  1740,cn=cn_admin data,cn=replication,cn=monitor
generation-id: 94310

dn: cn=Replication Domain 30839,cn=dc_example_dc_com,cn=replication,cn=monitor
generation-id: 19399981

dn: cn=Replication Domain 14142,cn=cn_schema,cn=replication,cn=monitor
generation-id: 8468

dn: cn=Connected Replica llandudno 27742,cn=Replication Server 8989  1740,cn=cn_
admin data,cn=replication,cn=monitor
generation-id: 94310

dn: cn=Replication Server 8989  1740,cn=cn_schema,cn=replication,cn=monitor
generation-id: cn=schema 8468

dn: cn=Replication Server 8989  1740,cn=dc_example_dc_com,cn=replication,cn=monitor
generation-id: dc=example,dc=com 19399981

dn: cn=Connected Replica llandudno 30839,cn=Replication Server 8989  1740,cn=dc_
example_dc_com,cn=replication,cn=monitor
generation-id: 19399981

dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 8989
  1740,cn=cn_schema,cn=replication,cn=monitor
generation-id: 8468

dn: cn=Connected Replica llandudno 14142,cn=Replication Server 8989  1740,cn=cn_
schema,cn=replication,cn=monitor
generation-id: 8468

dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 8989
  1740,cn=dc_example_dc_com,cn=replication,cn=monitor
```

```
generation-id: 19399981

dn: cn=Replication Domain 27742,cn=cn_admin data,cn=replication,cn=monitor
generation-id: 94310
```

The `generation-id` attribute indicates the *version* of the data in each replicated base DN, for each directory server. Note that the generation ID on all servers for the base DN `dc=example,dc=com` is `19399981`. The consistency of the generation IDs means that the data on those servers is the same for that base DN.

Each directory server is also aware of the generation ID of the replication server to which it is connected. The generation ID of a replication server relates to the updates that are stored in its change log database for that base DN.

Replication is considered to be working correctly between two directory servers, for a specified base DN, when those servers and their replication server all have the same generation ID.

To Monitor Replication Security

A secure replication topology has SSL encryption enabled between servers, for a particular base DN.

To monitor replication security, run the following search on any server in the topology that hosts a replication server:

```
$ ldapsearch -p 4444 --useSSL --trustAll -b "cn=monitor" "(ssl-encryption=*)" \
  "ssl-encryption"
dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 89
89  1740,cn=cn_admin data,cn=replication,cn=monitor
ssl-encryption: true

dn: cn=Replication Domain 30839,cn=dc_example_dc_com,cn=replication,cn=monitor
ssl-encryption: true

dn: cn=Replication Domain 14142,cn=cn_schema,cn=replication,cn=monitor
ssl-encryption: true

dn: cn=Connected Replica llandudno 27742,cn=Replication Server 8989  1740,cn=cn_
admin data,cn=replication,cn=monitor
ssl-encryption: true

dn: cn=Connected Replica llandudno 30839,cn=Replication Server 8989  1740,cn=dc_
example_dc_com,cn=replication,cn=monitor
ssl-encryption: true

dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 89
89  1740,cn=cn_schema,cn=replication,cn=monitor
ssl-encryption: true
```

```
dn: cn=Connected Replica llandudno 14142,cn=Replication Server 8989  1740,cn=cn_
schema,cn=replication,cn=monitor
ssl-encryption: true

dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 8989
  1740,cn=dc_example_dc_com,cn=replication,cn=monitor
ssl-encryption: true

dn: cn=Replication Domain 27742,cn=cn_admin data,cn=replication,cn=monitor
ssl-encryption: true
```

The `ssl-encryption` attribute specifies whether the replication protocol is encrypted between two servers for a specified base DN. This information is available for each directory server or replication server. Authentication of replication sessions is not monitored.

To Monitor Replicated Updates

Monitoring the number of updates that have been sent and received by the servers in a topology provides an indication of how well replication is working.

To monitor sent and received updates, type the following command:

```
$ ldapsearch -p 4444 --useSSL --trustAll -b "cn=monitor" \
  "(&(sent-updates=*)(received-updates=*))" "sent-updates" "received-updates"
dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 8989
  1740,cn=cn_admin data,cn=replication,cn=monitor
sent-updates: 7
received-updates: 0

dn: cn=Replication Domain 30839,cn=dc_example_dc_com,cn=replication,cn=monitor
received-updates: 28
sent-updates: 0

dn: cn=Replication Domain 14142,cn=cn_schema,cn=replication,cn=monitor
received-updates: 0
sent-updates: 0

dn: cn=Connected Replica llandudno 27742,cn=Replication Server 8989  1740,cn=cn_
admin data,cn=replication,cn=monitor
sent-updates: 0
received-updates: 0

dn: cn=Connected Replica llandudno 30839,cn=Replication Server 8989  1740,cn=dc_
example_dc_com,cn=replication,cn=monitor
sent-updates: 28
received-updates: 0
```

```
dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 8989
  1740,cn=cn_schema,cn=replication,cn=monitor
sent-updates: 0
received-updates: 0

dn: cn=Connected Replica llandudno 14142,cn=Replication Server 8989  1740,cn=cn_
schema,cn=replication,cn=monitor
sent-updates: 0
received-updates: 0

dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 8989
  1740,cn=dc_example_dc_com,cn=replication,cn=monitor
sent-updates: 0
received-updates: 28

dn: cn=Replication Domain 27742,cn=cn_admin data,cn=replication,cn=monitor
received-updates: 0
sent-updates: 0
```

The `sent-updates` attribute indicates the number of updates that have been sent by this directory server or replication server.

The `received-updates` attribute indicates the number of updates that have been received by this directory server or replication server.

The values of these attributes assist in determining the flow of updates within a topology. When replication appears to be very slow, it is helpful to monitor these attributes. If the number of updates sent by one server is consistently much higher than the number of updates received by another server, it is likely that the second server is a bottleneck in the topology.

The replication protocol controls the flow of updates between two servers. This ensures that when a high number of updates is exchanged between two servers, the servers are not prevented from processing operations with a higher priority. This functionality relies on a window mechanism where the recipient server periodically provides the sending server with the number of updates that the sending server can send.

You can specify the size of the send and receive windows, by setting the `max-send-window` and `max-rcv-window` configuration attributes. For more information, see "Modifying the Replication Configuration With `dsconfig`" on page 248.

The `current-send-window` monitoring attribute indicates how many changes can be sent by the sending server to the recipient server at that specific time. If the value of the `current-send-window` attribute is often equal to 0, transmission is stopped and the recipient server is probably a bottleneck in the topology. If the value of the `current-send-window` attribute is often equal to the value of the `max-send-window` attribute, and you are experiencing high replication latency, it is likely that the sending server is a bottleneck in the topology.

To obtain the value of the `current-send-window` property, type the following command:

```
$ ldapsearch -p 4444 --useSSL --trustAll -b "cn=monitor" "(current-send-window=*)" \
  "current-send-window"
dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 8989
  1740,cn=cn_admin data,cn=replication,cn=monitor
current-send-window: 93

dn: cn=Replication Domain 30839,cn=dc_example_dc_com,cn=replication,cn=monitor
current-send-window: 100

dn: cn=Replication Domain 14142,cn=cn_schema,cn=replication,cn=monitor
current-send-window: 100

dn: cn=Connected Replica llandudno 27742,cn=Replication Server 8989  1740,cn=cn_
admin data,cn=replication,cn=monitor
current-send-window: 100

dn: cn=Connected Replica llandudno 30839,cn=Replication Server 8989  1740,cn=dc_
example_dc_com,cn=replication,cn=monitor
current-send-window: 72

dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 8989
  1740,cn=cn_schema,cn=replication,cn=monitor
current-send-window: 100

dn: cn=Connected Replica llandudno 14142,cn=Replication Server 8989  1740,cn=cn_
schema,cn=replication,cn=monitor
current-send-window: 100

dn: cn=Connected Replication Server noordhoek:9989 7164,cn=Replication Server 8989
  1740,cn=dc_example_dc_com,cn=replication,cn=monitor
current-send-window: 100

dn: cn=Replication Domain 27742,cn=cn_admin data,cn=replication,cn=monitor
current-send-window: 100
```

To Monitor Replication Conflicts

When multiple operations are performed on the same entry at the same time, replication conflicts can occur. In some cases, the replication mechanism is able to resolve these conflicts. In other cases, manual conflict resolution is required.

Three types of conflict attributes can be monitored:

- `unresolved-naming-conflicts`. Indicates the number of naming conflicts that could not be resolved by the replication mechanism.
- `resolved-naming-conflicts`. Indicates the number of naming conflicts that have been resolved.

- `resolved-modify-conflicts`. Indicates the number of modify conflicts that have been resolved.

To monitor resolved and unresolved replication conflicts, run the following command:

```
$ ldapsearch -p 4444 --useSSL --trustAll -b "cn=monitor" \
  "(&(unresolved-naming-conflicts=*)(resolved-naming-conflicts=*)\
  (resolved-modify-conflicts=*))" "unresolved-naming-conflicts" \
  "resolved-naming-conflicts" "resolved-modify-conflicts"
dn: cn=Replication Domain 30839,cn=dc_example_dc_com,cn=replication,cn=monitor
resolved-naming-conflicts: 0
unresolved-naming-conflicts: 0
resolved-modify-conflicts: 0

dn: cn=Replication Domain 14142,cn=cn_schema,cn=replication,cn=monitor
resolved-naming-conflicts: 0
unresolved-naming-conflicts: 0
resolved-modify-conflicts: 0

dn: cn=Replication Domain 27742,cn=cn_admin data,cn=replication,cn=monitor
resolved-naming-conflicts: 0
unresolved-naming-conflicts: 0
resolved-modify-conflicts: 0
```

Improving Performance

The directory server is designed for high performance "out-of-the-box". Nonetheless, it is possible to tune aspects of the server to improve the performance for specific deployments. The following topics describe strategies for improving performance.

Tuning Performance

The directory server aims to be high-performing and highly-scalable. While the server can achieve impressive results with the "out-of-the-box" server configuration and default JVM settings, it can often be improved significantly through some basic tuning.

General Performance Tuning

The following items can improve performance in specific deployment scenarios.

- **Java Version**. Use the most recent Java Runtime Environment™ (JRE) release available. Although the directory server is designed to work with Java SE 5 (minimum version 1.5.0_08), use the latest Java SE 6 for noticeably better performance.
- **Environment Variables**. The directory server uses the `OPENDS_JAVA_HOME` environment variable to point to your installed JRE. If you have multiple versions of Java installed on a system, set the `JAVA_HOME` environment variable to point to the root of the desired installation. In this way, the version of the JRE specified by the `JAVA_HOME` variable can be used by other applications but not by the directory server software. To specify a JRE installation for the directory server, do one of the following:
  - Use the `dsjavaproperties` command to set the appropriate environment variables.

    For more information, see "dsjavaproperties" in *Sun OpenDS Standard Edition 2.0 Command-Line Usage Guide*.
  - Set the `OPENDS_JAVA_BIN` environment variable (with the JAVA binary path).
  - Set the `OPENDS_JAVA_HOME` environment variable (with the JAVA installation path).

- **Use libumem (Solaris).** On Solaris systems, it can be useful to try the `libumem` memory manager by setting the `LD_PRELOAD` variable to `libumem.so`. For more information about `LD_PRELOAD`, see http://developers.sun.com/solaris/articles/linker.html.

Java Virtual Machine Settings

The default settings of the directory server are targeted initially for evaluators and developers who are running equipment with a limited amount of resources. For this reason, it is important to do initial tuning of the Java™ Virtual Machine (JVM) and the directory server software to improve scalability and performance (particularly for write operations).

You can use the `OPENDS_JAVA_ARGS` environment variable to provide global configuration arguments that can be passed to the JVM, or you can use the `java.properties` file. Any argument that can be used with the `java` command can be used with both methods.

For more information, see "dsjavaproperties" in *Sun OpenDS Standard Edition 2.0 Command-Line Usage Guide*.

For additional information about tuning the JVM, see the Java Performance Documentation site. Of particular use are the Java Tuning White Paper and Garbage Collection Tuning documents.

Some notable arguments related to performance and scalability follow.

Note – These recommendations apply to Sun's HotSpot VM and are not necessarily available with a JVM from another vendor.

`-d32` or `-d64`
: Select the 32-bit or 64-bit version of the JVM (if applicable for the underlying system).

`-server`
: Specify that the server VM should be used instead of the client VM. The client VM is better optimized for processes that run for a short period of time and need to start as quickly as possible, whereas the server VM can take longer to warm up but is faster in the long run.

`-Xms2g` and `-Xmx2g`
: Specify that the initial and maximum memory sizes available to the JVM should be 2 Gbytes. Increasing the amount of memory available can improve performance, but increasing it too high can sometimes have a detrimental effect in the form of longer pauses for full garbage collection runs. The initial and maximum sizes should generally be set to the same values.

  In general, you should allocate enough memory for the directory server runtime and the rest to the DB cache.

`-XX:+UseConcMarkSweepGC`
Use the Concurrent Mark Sweep (CMS) garbage collector. This option allows the JVM to minimize the response time of LDAP operations, but it could have a small impact on the overall performance (throughput) of the directory server.

`-XX:CMSInitiatingOccupancyFraction=<percentage>`
Specify the level at which the CMS garbage collection is started. The default value is approximately 68%. Use this value if you want to set the percentage to something other than the default value.

`-XX:+AggressiveOpts`
Specify the JVM to configure itself with a number of settings (for example, for garbage collection, compilation, and other types of optimizations) that can provide improved performance.

`-XX:+UseBiasedLocking`
Improve locking performance in the server in cases where there is not expected to be a high degree of contention.

`-XX:LargePageSizeInBytes=256m`
Use large pages for the information it stores in memory. This argument applies primarily to systems using the UltraSPARC T1® processor.

`-XX:+UseParallelGC`
Specify that the system should use parallel garbage collection, which is particularly useful on systems with a large number of CPUs.

`-XX:+UseParallelOldGC`
Specify that the JVM should use parallel garbage collection for the old (tenured) generation.

`-XX:ParallelGCThreads=8`
Specify that the JVM should use 8 threads when performing parallel garbage collection. The default is to use a number of threads equal to the number of CPUs, but this can be inappropriate on systems with a very large number of CPUs or on CMT-based systems like those using the UltraSPARC T1 processor.

Directory Server Configuration

- **Preload Database Content**. You can configure the directory server to preload some of the database contents into memory on startup for back ends that use the Berkeley DB JE (those entries with the `local-db-backend` object class). This can be done by setting the `preload-time-limit` configuration property to indicate the length of time that should be spent pre-loading data into the database cache. For more information, see "Local DB Backend Configuration" in the *Configuration Reference*.
- **Tuning Parameters**. Some JE database tuning parameters can be used to tune performance.

- Use the `db-cache-percent` and `db-cache-size` properties to configure the amount of memory that the Oracle™ Berkeley DB Java Edition (JE) cache uses. For best performance, consider setting your directory server so that the whole database can fit into the DB cache. For example, after importing 200,000 entries into the `userRoot` back end, you might run the following procedure to determine your DB cache:

  ```
  $ cd install-dir/db
  $ du -sk userRoot/
  910616 userRoot/
  ```

  Set the JVM heap to 2 Gbytes (`-Xms2g -Xmx2g`), and then set the `db-cache-percent` to 50, so that the DB cache will use 1 Gbyte of memory. To monitor the DB cache, observe the following properties under the "`dn:cn=userRoot Database Environment,cn=monitor`" entry through Jtrace and JMX:

  - Check that `EnvironmentCacheDataBytes` has a value consistent with the expected size of the DB cache.
  - Check that `EnvironmentNCacheMiss` does not have unexpected growth when loading the server.
- The `db-evictor-lru-only` configuration property can be used to control how the database cache retains information. Setting this value to `false` ensures that the internal nodes are maintained in cache, which provides better performance when the JE cache holds only a small percentage of the database contents.
- The `db-txn-no-sync` and `db-txn-write-no-sync` configuration properties can be used to configure durability for write operations. Reducing durability can increase write performance, but it can also increase the chance of data loss in the event of a JVM crash or system crash.
- The `db-log-file-max` configuration property can be used to control the size of JE log files. Increasing the file size can improve write performance, but it can also make it harder to maintain the desired utilization percentage.
- The `db-num-cleaner-threads` and `db-cleaner-min-utilization` properties control how the cleaner works, which keeps the database size down and keeps up with high write throughput.
- On systems with a large number of CPUs, the `db-num-lock-tables` configuration property improves concurrency within the database lock manager.

- **Configure the LDAP Connection Handler**. The LDAP connection handler (and also the LDAPS connection handler, if it is enabled) can be configured to use multiple threads for decoding client requests through the `num-request-handlers` configuration property. Increasing the number of threads on systems with a larger number of CPUs can improve performance. Also, in some cases disabling the `keep-stats` property can help reduce lock contention. For more information, see "LDAP Connection Handler Configuration" in the *Configuration Reference*.

- **Enable an Entry Cache**. In some cases, particularly those involving relatively small directories (for example, up to a few hundred thousand entries), it can be useful to enable an entry cache. In general the FIFO entry cache provides better results than the soft reference entry cache. For more information, see "Entry Cache Configuration" in the *Configuration Reference*.
- **Disable Unused Virtual Attributes**. If the functionality needed by one or more of the virtual attributes is not required, they can be disabled for a slight performance improvement when decoding entries. For more information, see "Virtual Attribute Configuration" in the *Configuration Reference*.
- **Disable Unused Access Logging**. If access logging is not necessary, disabling the server access logger can help improve performance. For more information, see "Logger Configuration" in the *Configuration Reference*.
- **Disable Unused Access Control Handlers**. If you do not need access control processing in the server, then you can disable it by setting the `enabled` configuration property to `false` for the Access Control Handler. You can set the property by using `dsconfig`.
- **Reduce Lock Contention**. On systems with large numbers of CPUs (for example, chip multi-threading (CMT) systems with several hardware threads per core), you can reduce lock contention by setting the `org.opends.server.LockManagerConcurrencyLevel` system property to be equal to the number of worker threads you intend to use.

Note – This property must be set as a JVM system property, because it can be required very early in the server startup process, even before accessing the server configuration.

Improving Performance When Importing Large Data Sets

The topics in this section provide tips on improving performance when importing large data sets. The examples use the `dsconfig` command to modify the server configuration. For more information see "Configuring the Directory Server With `dsconfig`" on page 7.

Data Import Overview

By default, the server imports data with a fixed set of parameters. You can change the default behavior in two ways:

- Adjust the appropriate configuration properties for the database to which the data will be imported.
- Use the `dsjavaproperties` command to set the appropriate Java arguments before running the `import-ldif` command.

A single thread reads the LDIF file to be imported and places the entries in a queue. The import thread then takes these queued entries and processes them into intermediate binary files that are eventually merged into the database.

Adjusting Import Parameters

The import parameters are properties of the specific back end (database) to which the data is imported. The following examples assume that data is being imported to the default `userRoot` back end. Use `dsconfig` to change the following properties:

- Number of import threads
- Queue size

▼ To Adjust the Number of Import Threads

The higher the number of import threads, the faster the queued entries are processed. The default number of import threads is 8.

- **To change the number of import threads, set the `import-thread-count` back-end property, as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n set-backend-prop \
  --backend-name userRoot --set import-thread-count:12
```

▼ To Adjust the Queue Size

The longer the queue, the more threads can pick entries up and write them to intermediate files before committing to the database. The default queue size is 100.

- **To change the queue size, set the `import-queue-size` back-end property, as follows:**

```
$ dsconfig -D "cn=directory manager" -w password -n set-backend-prop \
  --backend-name userRoot --set import-queue-size:200
```

Adjusting Memory Requirements to Match Configuration Settings

To calculate how much memory you need, take the import buffer size, add 10 percent for database dedicated memory, and add 10 percent to avoid hitting out-of-memory errors. For example, if you set the buffer size to 2 Gbytes, add 256 Mbytes for the database and another 256 Mbytes to avoid out-of-memory errors.

When you have calculated the memory requirement, perform the following steps:

1. Edit the `java.properties` file and set the following values:

```
overwrite-env-java-args=true
import-ldif.offline.java-args=-Xms2560M -Xmx2560M
```

2. Run the `dsjavaproperties` command:

```
$ bin/dsjavaproperties
```

Note – Running the `dsjavaproperties` command, or setting the `OPENDS_JAVA_ARGS` environment variable, only has a performance impact if the import is offline. If the server is already running and you perform an online import, changing the Java arguments has no impact on the import performance because the import is performed by the server JVM.

Advanced Administration

This section contains topics that are not considered everyday administrative activities, or are of particular interest to advanced directory server users:

The section covers the following topics:

Running the Directory Server as a Non-Root User

Like many network daemons, the Sun Java System directory server has a `setuid` capability that allows it to be started as a root user but then drop privileges to run as a user with fewer capabilities. The OpenDS directory server does not currently include this capability (and it would require native code to implement, which is not desirable). However, you can install, start, and run the directory server as a non-root user. Note that the information in this section applies primarily to UNIX-based platforms, because Windows systems do not historically place as many restrictions on non-administrative users.

Reasons for Running the Server as a Non-Root User

In many cases, running the directory server as a non-root user from the start is a more attractive option and provides greater functionality than the `setuid` equivalent. Running the server as a non-root user means that administrators do not need root access to the system, which is often desirable from an operational perspective. In addition, more administrative actions can be performed with the directory server online, because the server can do things that might not have been available after it had dropped root privileges.

The primary reason that directory servers are typically started and/or run as root users is so that they can listen on a privileged port (namely, ports between 1 and 1024). The standard port for LDAP communication is port 389, and the standard port for LDAPS is 636. On most UNIX-based systems only root users are able to create processes that listen on these ports. There can be other reasons for starting as a root user (for example, the ability to use a larger number of file descriptors), but it is generally easier to configure around these other limitations.

Although the standard LDAP and LDAPS ports are 389 and 636, the directory server is not required to run on those ports. In some environments, it is common to run the directory server

on ports above 1024 (such as 1389 and 1636) so that it is not necessary to be root to start it. Virtually all LDAP-enabled clients provide the ability to specify the port on which the server is listening. As long as the clients know what port the server is using, any value is allowed. For information about configuring the listen port, see "Configuring the LDAP Connection Handler" on page 17.

How to Run as a Non-Root User on the Standard LDAP Ports

If clients expect the server to be listening on port 389 or 636, other options are still available. The best option, available on Solaris 10, is to use the process rights management subsystem (also called *least privilege*). The privileges subsystem in Solaris makes it possible to give non-root users and roles capabilities normally available only to the root user (much like the Privilege Subsystem allows within the directory server). In particular, the `net_privaddr` privilege controls which users can bind to privileged ports If this privilege is granted to a non-root user, that user can bind to privileged ports. To configure a user with this privilege, run the following command, as the root user:

```
# usermod -K defaultpriv=basic,net_privaddr,sys_resource,-proc_info,-file_link_any \
  opends
```

This command configures the `opends` user so that it starts with the `basic` privilege set (which is what non-root users have by default). The command then adds the `net_privaddr` and `sys_resource` privileges, which allow the user to increase the number of file descriptors available, among other things. The command removes the `proc_info` privilege (which allows the user to see processes owned by other users) and the `file_link_any` privilege (which allows the user to create hard links to files that they do not own). After running this command, the `opends` user is able to start the directory server listening on a privileged port.

Even on systems without a capability like least privilege, it is possible to expose the directory server on a privileged port such as 389 or 636 without requiring root privileges to be able to start it. One possibility would be to run the server on an unprivileged port and use a directory proxy server listening on the privileged port to forward communication to the directory server on an unprivileged port. It is also possible to use network hardware to achieve the same purpose or to use firewall rules on the same system. For example, on Linux systems the following commands can be used to redirect traffic targeting port 389 to port 1389:

```
# iptables --append PREROUTING --table nat --protocol tcp --dport 389 \
  --jump REDIRECT --to-port 1389
# iptables -t nat -A OUTPUT -p tcp --dport 389 -j DNAT --to :1389
```

Working With Directory Schema

The *schema* defines and governs the types of information objects that can be stored in a directory. A schema defines the types of entries in the directory information tree, maintains element uniqueness, and prevents unchecked schema growth that can arise when new elements are added to the directory. This section provides instructions on viewing and extending the schema provided with the directory server.

Directory Schema Overview

The directory server reads the schema once at startup and then uses the schema information to match a search filter request or assertion to an entry's attributes to determine if any add or modify operations are permitted by the client.

In most cases, the default schema should be sufficient for most applications. However, you can take advantage of the flexibility of the directory server to extend the schema to suit your applications. The general procedure is not to relinquish the standard schema to a new custom schema, but to use the standard attributes or object classes wherever possible. If you require custom attributes or object classes that are not handled with the standard schema, you can create or extend the standard schema with auxiliary attributes and object classes required for your application.

The schema is stored in the directory under the suffix (`cn=schema`). The directory server also has a subschema subentry that defines the schema elements plus the set of operational attributes in the directory.

You can extend the schema in one of two ways:

- Extend the schema over LDAP.
- Create a custom schema definition file.

Designing and Extending the Schema

Before you consider extending the default schema, or designing your own schema, ensure that you have a solid understanding of schema syntax and design. For background information on schema architecture, see "Understanding the Directory Server Schema" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

The basic steps to design or extend a schema are as follows:

1. Map the data to the default schema. Where possible, use the existing schema elements that are defined in the directory server. Standard schema elements help to ensure compatibility with directory-enabled applications. Because the schema is based on the LDAP standard, it has been reviewed and agreed upon by a large number of directory users.
2. Identify unmatched data. The default schema was designed to accommodate a large variety of information objects. However, if the schema does not handle your specific data type, then make note of it and any other data types needed for your directory.
3. Extend the default schema to define new elements. For optimal performance, reuse existing schema elements wherever possible. Also, minimize the number of mandatory attributes that you define for each object class. Keep the schema as simple as possible. Do not define more than one object class or attribute for the same purpose.
4. Use schema checking. Schema checking ensures that attributes and object classes conform to the schema rules.
5. Select and apply a consistent data format. The LDAP schema allows you to place any data on any attribute value. However, you should store data consistently by selecting a format appropriate for your LDAP client application and directory users.

Default Schema Files

The default schema provided with the directory server is a collection of LDIF files stored under *install-dir*`/config/schema`. The directory server loads the schema files in alphanumeric order (numerals first) at directory server startup.

Caution – Never modify the standard schema definitions and internal operational attributes in these files.

The following table describes the default schema files and their contents.

TABLE 5 Default Schema Files

Schema File	Description
`00-core.ldif`	Contains the schema definitions for the LDAPv3 standard user and organization.
`01-pwpolicy.ldif`	Contains the schema definitions for password policies based on the `draftldappolicy` draft.
`02-config.ldif`	Contains the schema definitions for the attribute and object class definitions in the directory configuration file.

TABLE 5 Default Schema Files *(Continued)*

Schema File	Description
`03-changelog.ldif`	Contains the schema definitions for storing changes to directory data based on the `draftldap-changelog`.
`03-rfc2713.ldif`	Contains the schema definitions for representing Java objects in an LDAP directory based on RFC 2713.
`03-rfc2714.ldif`	Contains the schema definitions for representing CORBA object references in an LDAP directory based on RFC 2714. The Common Object Request Broker Architecture (CORBA) integrates machines in a multivendor, multiplatform environments using CORBA objects. A directory server can be a repository for CORBA object references, which allow for a centrally administered service for CORBA-compliant applications.
`03-rfc2739.ldif`	Contains the schema definitions for representing calendar attributes for a vCard directory based on RFC 2739. Calendar applications require a calendar user agent to locate a URI, located in a directory, for an individual's calendar. Note that the definition in RFC 2739 contains a number of errors. This schema file has been altered from the standard definition in order to fix a number of those problems.
`03-rfc2926.ldif`	Contains the schema definitions for mapping Service Location Protocol (SLP) advertisements based on RFC 2926. This specification allows directory servers to serve SLP directory agent back ends that create mappings between SLP templates and the LDAP directory schema.
`03-rfc3112.ldif`	Contains the schema definitions for the authentication password syntax based on RFC 3112.
`03-rfc3712.ldif`	Contains the schema definitions for storing printer information in the directory based on RFC 3712.
`03-uddiv3.ldif`	Contains the schema definitions for storing UDDI v3 information in the directory based on RFC 4403. Universal Description, Discovery and Integration (UDDI) is a platform-independent, XML-based registry for companies on the Internet. UDDI enables companies to publish service listings and defines which software applications interact together over the Internet.

TABLE 5 Default Schema Files *(Continued)*

Schema File	Description
`04-rfc2307bis.ldif`	Contains the schema definitions for storing naming service information in the directory based on `draftrfc2307bis`.

Configuring Schema Checking

The directory server provides a schema-checking mechanism that verifies whether newly-written or added entries conform to the directory server's schema. This mechanism ensures that data imported using `import-ldif`, or added using `ldapmodify`, meets the syntax rules of the schema.

The schema checking configuration is part of the advanced global configuration, and can be displayed with the following command:

```
$ dsconfig -D "cn=directory manager" -w password -n --advanced \
  get-global-configuration-prop
Property                              : Value(s)
--------------------------------------:--------------------------------------
...
check-schema                          : true
...
invalid-attribute-syntax-behavior     : reject
...
single-structural-objectclass-behavior : reject
...
```

The following configuration properties control schema-checking:

- `check-schema`. Possible values: `true` (default), `false`. This property controls whether the directory server should do schema-checking on newly imported or added entries. By default, the property is set to true. If you need to tune the server for maximum performance and you are certain that your clients will never make a change that causes a schema violation, you can set the property to `false`. The small performance benefits are minimal compared to the potential risks to your directory.
- `invalid-attribute-syntax-behavior`. Possible values are: `reject` (default), `accept`, and `warn`. This property controls how the server should behave if an attempt is made to use an attribute value that violates the associated syntax. By default, the server rejects any requests to use attributes that violate the schema. If this property is set to `accept`, the server silently accepts attribute violations. If this attribute is set to `warn`, the server accepts violations, but writes a message to the error log. If the `check-schema` property is set to `false`, invalid attribute syntax checking is not enforced.

- `single-structural-objectclass-behavior`. Possible values are: `reject` (default), `accept`, and `warn`. This property controls how the server should behave if an attempt is made to create or alter an entry that does not have exactly one structural object class. This means that object classes with no structural object classes or more than one are rejected by default. If this property is set to `accept`, entries with no structural object classes are allowed. If this property is set to `warn`, entries with no structural object classes (or more than one) are allowed, but a message is written to the error log. If the `check-schema` property is set to `false`, single structural object class checking is not enforced.

Caution – Changing the value of these properties from the default puts the integrity of the schema at risk, so in general do *not* alter these values.

Working With Object Identifiers (OIDs)

An object identifier (OID) is a numeric string used to uniquely identify an object in a directory. OIDs are used in directory schema, controls, and extended operations that require unique identification of elements.

LDAP object classes and attributes require a base object identifier (OID) that must be unique within your organization to avoid naming conflicts in the directory. If you plan to use your directory internally within your organization, use the OIDs provided in the directory server. If you plan to export your schema or publicly expose your schema in any way, you should consider entering a request for a unique OID for your organization. For more information, see "Obtaining a Base OID" on page 379.

After you have obtained a base OID, you can add branches to it for your organization's object classes and attributes. For example, the directory server uses an assigned base OID of `1.3.6.1.4.1.26027`. For each component type, the directory server provides unique branch numbers to the base OID for each schema component.

Note – The directory server provides a comprehensive set of OIDs that should be sufficient for most applications. You can also request OIDs for addition to the directory server repository.

The following table shows the base OIDs used for each schema component:

TABLE 6 Base OIDs Used for Each Schema Component

OID Value	Type
`1.3.6.1.4.1.26027.1.1`	Attribute
`1.3.6.1.4.1.26027.1.2`	Object classes

TABLE 6 Base OIDs Used for Each Schema Component *(Continued)*

OID Value	Type
1.3.6.1.4.1.26027.1.3	Attribute syntaxes
1.3.6.1.4.1.26027.1.4	Matching rules
1.3.6.1.4.1.26027.1.5	Controls
1.3.6.1.4.1.26027.1.6	Extended operations
1.3.6.1.4.1.26027.1.9	General use
1.3.6.1.4.1.26027.1.999	Experimental use

For each schema type, a unique branch number is added to the base OID. For example, attribute types use a branch number of `1` to form the OID of `1.3.5.1.4.1.26027.1.*1*`. For each specific attribute type, the directory server assigns another set of branch numbers, one for each attribute type.

The following table displays a (partial) list of assigned OID values for attribute types.

TABLE 7 Assigned OID Values for Attribute Types

OID Value	Attribute Type
1.3.6.1.4.1.26027.1.1.1	ds-cfg-java-class
1.3.6.1.4.1.26027.1.1.2	ds-cfg-enabled
1.3.6.1.4.1.26027.1.1.3	ds-cfg-allow-attribute-name-exceptions
1.3.6.1.4.1.26027.1.1.4	ds-cfg-allowed-client
1.3.6.1.4.1.26027.1.1.5	ds-cfg-allow-ldap-v2

Note – The directory server allows the use of non-numeric OIDs as long as a corresponding numeric OID is defined within the schema. For example, you can use a non-numeric OID, `mytestattribute-oid` for the named attribute, `myTestAttribute`. The non-numeric OID must be all lowercase with the `-oid` appended to the named attribute. The use of non-numeric OIDs is an LDAP-specification violation but is permissible for ease of use.

Obtaining a Base OID

If you plan to make your directory server publicly available, or if you plan to redistribute your schema definitions for custom applications, you can obtain a base OID for your organization. You can use your own OIDs in a custom schema file if you plan to create custom extensions to the directory server. Alternatively, you can modify the schema configuration files by adding your base OID with its respective branch number.

Note – Do not modify the default OIDs unless you are sure of what you are doing. Modifying the OIDs can potentially damage your directory server.

To obtain and create base OIDs for your organization, perform the following steps:

1. Point your browser to the Internet Assigned Numbers Authority (IANA) web site at `(http://www.iana.org)` or a national organization in your country that handles such tasks. In some countries, corporations already have OIDs assigned to them. If your organization does not already have an OID, you can fill out a request at the IANA web site.
2. Determine the unique object classes, attributes, names, and other schema elements. Ensure that the names are descriptive to make it easier to manage the schema. One trick is to add a custom prefix to your custom object classes and attributes. For example, if your organization is Example.com, you can add the prefix `Example` before each custom schema element, such as adding `Example` to a `Person` object class as in `ExamplePerson`.
3. Create an OID registry to keep track of OID assignments. The registry is nothing more than a list that you maintain to ensure that OIDs and their descriptions are unique within your directory. The registry should be sufficiently protected so that only a privileged administrator can modify the registry.
4. Create branches in the OID tree to accommodate the schema elements.
5. Shut down the directory servers in your topology.
6. Manually edit the schema configuration files on each directory server in your topology. Replace each OID with your company's OID. This avoids problems with schema replication seeing differences in the schema and attempting to synchronize the information.
7. Manually edit any custom schema extensions. Ideally, you should define any custom extensions in a separate file.

Extending the Directory Schema

The directory server supports multiple methods to extend the schema. The schema files are a set of LDIF files located in the *install-dir*`/config/schema` directory. Do not modify these files directly, because doing so can result in unpredictable server behavior.

You can extend the schema as follows:

- **Extend the schema over LDAP.** Define your schema extensions, write the definitions to an LDIF file, and then add the custom schema extensions by using the `ldapmodify` command. When you use this method, the directory server automatically writes the new schema definitions to a file, `99user.ldif`. If you want to specify a different schema file, include the `X-SCHEMA-FILE` element with the name of your schema file. For example, as part of your attribute type definition, include the element `X-SCHEMA-FILE '98myschema.ldif'`.
- **Create a custom schema file.** Create a custom schema file with your definitions, save it as `98myschema.ldif`, and then move the file to the *install-dir*`/config/schema` directory.
- **Modify an existing schema file.** You can add a custom schema extension to an existing custom schema file, such as `99user.ldif`.

Points to Consider When Extending the Schema

- When adding new schema elements, all attributes must be defined before they can be used in an object class.
- If you are creating several object classes that inherit from other object classes, you must create the parent object class first.
- Each custom attribute or object class that you create should be defined in only one schema file.
- When defining new schema definitions manually, the best practice is to add these definitions to the `99user.ldif` file or to your designated schema file.
- The directory loads schema files in alphanumeric order with numbers loaded first, so you should name custom schema files as follows: `[00-99]filename.ldif`.

Managing Attribute Types

You can add new attribute types to the schema by using the `ldapmodify` command. The attribute types syntax requires that you provide at least a valid OID to define a new element. In typical applications, you can optionally include the following identifiers for the attribute type. To see the full set of attribute type elements, see "Understanding Attribute Types" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

TABLE 8 Attribute Elements

Attribute Elements	Description
`OID`	Required. Specifies the OID that uniquely identifies the attribute type in the directory server. The LDAP v3 specification requires the OID to be a numeric number, but the directory server supports the use of non-numeric OIDs for easy identification as long as the schema is used internally within the organization. The format is *attributename*`-oid`, for example, `telephoneNumber-oid`. Each non-numeric OID must have its corresponding numeric OID defined in the schema.
`NAME`	Optional. Specifies the set of human-readable names that are used to refer to the attribute type. If there is a single name, enclose it in single quotes, for example, `'blogURL'`. If there are multiple names, enclose each name in single quotes separated by spaces, and then enclose the entire set of names within parentheses, for example, `( 'blog' 'blogURL' )`. Ensure that there is a space between the left parenthesis and the name, and a space before the closing parenthesis.
`SUP`	Optional. Specifies the superior attribute type when you want one attribute type to inherit elements from another attribute type. The matching rule and attribute syntax specifications from the superior attribute type can be inherited by the subordinate type if it does not override the superior attribute type definition. The OID, any of the human-readable names associated with the superior attribute type or both can be used to collectively reference all of the subordinate attribute types.
`DESC`	Optional. Specifies a human-readable description of the attribute type.
`SYNTAX`	Optional. Specifies the attribute syntax for use with the attribute type. If provided, it should be given as a numeric OID. The core syntaxes are defined in section 3.3. of RFC 4517 (http://www.ietf.org/rfc/rfc4517.txt) (`http://www.ietf.org/rfc/rfc4517.txt`) and in Appendix A of the same document.
`SINGLE-VALUE`	Optional. Specifies whether the attributes of that type are allowed to have only a single value in any entry in which they appear. If `SINGLE-VALUE` is not present, the attributes are allowed to have multiple distinct values in the same entry.

TABLE 8 Attribute Elements *(Continued)*

Attribute Elements	Description
NO-USER-MODIFICATION	Optional. Indicates that the values of the attributes of the given type cannot be modified by external clients (that is, the values can be modified only by internal processing within the directory server).
USAGE	Optional. Indicates how the attribute is to be used. Possible values are as follows: `userApplications`. Used to store user data. `directoryOperation`. Used to store data required for internal processing within the directory server. `distributeOperation`. Used to store operational data that must be synchronized across directory servers in the topology. `dSAOperation`. Used to store operational data that is specific to a particular directory server and should not be synchronized across the topology.
extensions	Optional. Specifies the extensions available to the attribute type. The directory server provides the following extensions: `X-ORIGIN`. Provides information on where the attribute type is defined. The element is a non-standard tool that the user can use to locate the schema element. Examples could include the RFC number (`RFC4517`), `Sun OpenDS SE Directory Server` and others. `X-SCHEMA-FILE`. Indicates which schema file contains the attribute type definition. Used for internal purposes only and is not exposed to clients. You can use this extension to specify where the directory server should store your custom schema definitions. `X-APPROX`. Indicates which approximate matching rule should be used for the attribute type. If specified, the value should be the name of the OID of a registered approximate matching rule.

For example, you can specify the addition of a new attribute type, `blogURL`, in an LDIF file that will be added to the schema.

```
$ cat blogURL.ldif
dn: cn=schema
changetype: modify
add: attributeTypes
attributeTypes: ( 1.3.6.1.4.1.26037.1.999.1000
NAME ( 'blog' 'blogURL' )
DESC 'URL to a personal weblog'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
```

```
X-ORIGIN 'OpenDS Directory Server'
USAGE userApplications )
```

Note – Pay special attention to the spaces in an attribute type declaration. The LDAP specification requires that a space exist between the opening parenthesis and the OID, and the value of the `USAGE` element and the closing parenthesis. Further, the LDIF specification states that LDIF parsers should ignore exactly one space at the beginning of each line. Therefore, it is a good practice to add two (2) spaces at the beginning of the line that starts with an element keyword. For example, add two spaces before `NAME`, `DESC`, `SYNTAX`, `SINGLE-VALUE`, `X-ORIGIN`, and `USAGE` in the previous example.

To View Attribute Types

The `cn=schema` entry has a multivalued attribute, `attributeTypes`, that contains definitions of each attribute type in the directory schema. You can view the schema definitions by using the `ldapsearch` command.

Manipulation of the `cn=schema` suffix is regarded as an administrative action and, as such, it is recommended that you use the administration connector when accessing this suffix. See "Managing Administration Traffic to the Server" on page 5 for more information.

1. **Use the `ldapsearch` command as follows:**

```
$ ldapsearch -h localhost -p 4444 -D "cn=Directory Manager" -w password -X --useSSL \
  --baseDN cn=schema --searchScope base \
"(objectclass=*)" attributeTypes
dn: cn=schema
attributeTypes: ( 2.5.4.41 NAME 'name' EQUALITY caseIgnoreMatch SUBSTR caseIgnore
eSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} X-ORIGIN 'RFC 4519
' )
attributeTypes: ( 2.5.4.49 NAME 'distinguishedName' EQUALITY distinguishedNameMa
tch SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 X-ORIGIN 'RFC 4519' )
attributeTypes: ( 2.5.4.0 NAME 'objectClass' EQUALITY objectIdentifierMatch SYNT
AX 1.3.6.1.4.1.1466.115.121.1.38 X-ORIGIN 'RFC 4512' )
...(more output)...
```

2. **(Optional) To view a specific attribute type, use the `-dontWrap` option and then use the `grep` command to search for the required attribute.**

```
$ ldapsearch -h localhost -p 4444 -D "cn=Directory Manager" -w password -X --useSSL \
  -b cn=schema -s base --dontWrap "(objectclass=*)" attributeTypes | grep "telexNumber"
attributeTypes: ( 2.5.4.21 NAME 'telexNumber'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.52 X-ORIGIN 'RFC 4519' )
```

To Create an Attribute Type

The `cn=schema` entry has a multivalued attribute, `attributeTypes`, that contains definitions of each attribute type in the directory schema. You add custom schema definitions by using the `ldapmodify` command. This example adds an attribute named `blog`.

Manipulation of the `cn=schema` suffix is regarded as an administrative action and, as such, it is recommended that you use the administration connector when accessing this suffix. See "Managing Administration Traffic to the Server" on page 5 for more information.

1 Using a text editor, create an LDIF file with your schema extensions.

```
dn: cn=schema
changetype: modify
add: attributeTypes
attributeTypes: ( 1.3.6.1.4.1.26037.1.999.1000
NAME ( 'blog' 'blogURL' )
DESC 'URL to a personal weblog'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
X-ORIGIN 'OpenDS Directory Server'
USAGE userApplications )
```

2 Use `ldapmodify` to add the file.

```
$ ldapmodify -h localhost -p 4444 -D "cn=Directory Manager" -w password -X --useSSL \
  -a -f blogURL.ldif
Processing MODIFY request for cn=schema
MODIFY operation successful for DN cn=schema
```

3 Verify the addition by displaying it using `ldapsearch`.

```
$ ldapsearch -h localhost -p 4444 -D "cn=Directory Manager" -w password -X --useSSL \
  -b cn=schema -s base --dontWrap "(objectclass=*)" attributeTypes | grep 'blog'
attributeTypes: ( 1.3.6.1.4.1.26037.1.999.1000 NAME ( 'blog' 'blogURL' )
DESC 'URL to a personal weblog' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE X-ORIGIN 'OpenDS Directory Server' USAGE userApplications )
```

Note – The directory server automatically adds new attribute definitions to the `99user.ldif` file.

To Delete an Attribute Type

The `cn=schema` entry has a multivalued attribute, `attributeTypes`, that contains definitions of each attribute type in the directory schema. You can delete definitions with `X-ORIGIN 'user defined'` by using the `ldapmodify` command. The directory server does not allow deletions to other definitions.

Caution – Be careful when deleting attribute types, because doing so can harm your directory. Do not delete an attribute type unless absolutely necessary.

Manipulation of the `cn=schema` suffix is regarded as an administrative action and, as such, it is recommended that you use the administration connector when accessing this suffix. For more information, see "Managing Administration Traffic to the Server" on page 5.

1 **Create the delete request in an LDIF file.**

```
dn: cn=schema
changetype: modify
delete: attributeTypes
attributeTypes: ( 1.3.6.1.4.1.26037.1.999.1000
NAME ( 'blog' 'blogURL' )
DESC 'URL to a personal weblog'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE
X-ORIGIN 'OpenDS Directory Server'
USAGE userApplications )
```

2 **Use the `ldapmodify` command to process the delete request.**

```
$ ldapmodify -h localhost -p 4444 -D "cn=Directory Manager" -w password -X --useSSL \
  --defaultAdd --fileName "remove_blogURL.ldif"
Processing MODIFY request for cn=schema
MODIFY operation successful for DN cn=schema
```

Managing Object Classes

Object classes are named sets of attribute definitions that are used to control the types of data stored in entries. You can add new object classes to the schema by using the `ldapmodify` command. The object class syntax requires that you provide at least a valid OID to define your new element. In typical applications, you will also include the following optional identifiers for the object class type. For more information about the object class definition, see "Understanding the Directory Server Schema" in *Sun OpenDS Standard Edition 2.0 Architectural Reference*.

TABLE 9 Object Class Elements

Object Class Elements	Description
`OID`	Required. Specifies the OID that uniquely identifies the object class in the directory server. The LDAP v3 specification requires the OID to be a numeric number, but Sun OpenDS SE supports the use of non-numeric OIDs for easy identification because the schema is used internally within the organization. For example, the format is *objectClassName*`\-oid`, such as `person-oid`.
`NAME`	Optional. Specifies the set of human-readable names that are used to refer to the object class. If there is a single name, enclose it in single quotes, for example, `'blogURL'`. If there are multiple names, enclose each name in single quotes separated by spaces, and then enclose the entire set of names within parentheses, for example, `( 'blog' 'blogURL' )`. Ensure that there is a space between the left parenthesis and the name, and a space before the closing parenthesis.
`DESC`	Optional. Specifies a human-readable description of the object class. If specified, the description should be enclosed in single quotation marks.
`SUP`	Optional. Specifies the superior object class when you want it to inherit elements from another object class. The directory server allows only one superior object class, although the LDAP v3 specification allows for multiple superior object classes.
`OBSOLETE`	Optional. Indicates whether the object class is active or not. If an object class is marked as `OBSOLETE`, then it should not be referenced by any new elements created in the directory server.
`SUP` *oids*	Optional. The `SUP` keyword should be followed by the OID of the superior class.
`KIND`	Optional. Indicates the type of object class that is being defined. Allowed values are `ABSTRACT`, `AUXILIARY` and `STRUCTURAL`.

TABLE 9 Object Class Elements *(Continued)*

Object Class Elements	Description
`MUST` *oids*	Optional. Specifies the set of attribute types that are required to be present (that is, have at least one value) in entries with that object class. If there is only a single required attribute, then the `MUST` keyword should be followed by the name or the OID of that attribute type. If there are multiple required attribute types, then separate them with dollar signs ($) and enclose the entire set of attribute types in parentheses. For example, `MUST ( sn $ cn )`.
`MAY` *oids*	Optional. Specifies the set of attribute types that are allowed but not required to be present in entries with that object class. If there is only a single required attribute, then the `MAY` keyword should be followed by the name or the OID of that attribute type. If multiple required attribute types are specified, then separate them by dollar signs ($) and enclose the entire set of attribute types in parentheses. For example, `MAY ( userPassword $ telephoneNumber $ seeAlso $ description )`.
extensions	Optional. Specifies the extensions available to the object class. The directory server provides the following extensions: `X-ORIGIN`. Provides information on where the object class is defined. The element is a non-standard tool that the user can use to conveniently locate the schema element. Examples could include the RFC number `RFC4517`, `OpenDS Directory Server` and others. `X-SCHEMA-FILE`. Indicates which schema file contains the object class definition. Used for internal purposes only and is not exposed to clients. You can use this extension to specify where the directory server is to store your custom schema definitions.

For example, you can specify the addition of a new object class, `blogger`, in an LDIF file to be added to the schema.

```
$ cat blogger.ldif
dn: cn=schema
changetype: modify
add: objectClasses
objectClasses: ( 1.3.6.1.4.1.26037.1.999.2000
NAME ( 'blogger' )
DESC 'Someone who has a blog'
SUP inetOrgPerson
```

```
STRUCTURAL
MAY blog
X-ORIGIN 'OpenDS Directory Server' )
```

Note – Pay special attention to the spaces in your object class declaration. The LDAP specification requires that a space exist between the opening parenthesis and the OID, and the value of the `X-ORIGIN` element and the closing parenthesis. Further, the LDIF specification states that LDIF parsers should ignore exactly one space at the beginning of each line. Therefore, it is a good practice to add two spaces before the line that begins with an element keyword, such as, `NAME`, `DESC`, `SUP`, `STRUCTURAL`, `MAY`, and `X-ORIGIN` in the previous example.

To View Object Classes

The `cn=schema` entry has a multivalued attribute, `objectClass`, that contains definitions of each object class in the directory schema. You can view the schema definitions by using the `ldapsearch` command.

Manipulation of the `cn=schema` suffix is regarded as an administrative action and, as such, it is recommended that you use the administration connector when accessing this suffix. See "Managing Administration Traffic to the Server" on page 5 for more information.

1 **Use the `ldapsearch` command.**

```
$ ldapsearch -h localhost -p 4444 -D "cn=Directory Manager" -w password -X --useSSL \
  -b cn=schema -s base "(objectclass=*)" objectClasses
dn: cn=schema
objectClasses: ( 2.5.6.0 NAME 'top' ABSTRACT MUST objectClass X-ORIGIN 'RFC 4512
' )
objectClasses: ( 2.5.6.1 NAME 'alias' SUP top STRUCTURAL MUST aliasedObjectName
X-ORIGIN 'RFC 4512' )
objectClasses: ( 2.5.6.2 NAME 'country' SUP top STRUCTURAL MUST c MAY ( searchGu
ide $ description ) X-ORIGIN 'RFC 4519' )
objectClasses: ( 2.5.6.3 NAME 'locality' SUP top STRUCTURAL MAY ( street $ seeAl
so $ searchGuide $ st $ l $ description ) X-ORIGIN 'RFC 4519' )
objectClasses: ( 2.5.6.4 NAME 'organization' SUP top STRUCTURAL MUST o MAY ( use
rPassword $ searchGuide $ seeAlso $ businessCategory $ x121Address $ registered
Address $ destinationIndicator $ preferredDeliveryMethod $ telexNumber $ telete
xTerminalIdentifier $ telephoneNumber $ internationaliSDNNumber $ facsimileTele
phoneNumber $ street $ postOfficeBox $ postalCode $ postalAddress $ physicalDel
iveryOfficeName $ st $ l $ description ) X-ORIGIN 'RFC 4519' )
...(more output)...
```

2 **(Optional) Use the `--dontWrap` option and the `grep` command to search for a specific object class.**

```
$ ldapsearch -h localhost -p 4444 -D "cn=Directory Manager" -w password -X \
  --useSSL -b cn=schema -s base --dontWrap "(objectclass=*)" \
  objectClasses | grep "inetOrgPerson"
```

```
objectClasses: ( 2.16.840.1.113730.3.2.2 NAME 'inetOrgPerson' SUP organizationalPerson
STRUCTURAL MAY ( audio $ businessCategory $ carLicense $ departmentNumber $ displayName
$ employeeNumber $ employeeType $ givenName $ homePhone $ homePostalAddress $ initials
$ jpegPhoto $ labeledURI $ mail $ manager $ mobile $ o $ pager $ photo $ roomNumber
$ secretary $ uid $ userCertificate $ x500UniqueIdentifier $ preferredLanguage
$ userSMIMECertificate $ userPKCS12 ) X-ORIGIN 'RFC 2798' )
```

▼ To Create an Object Class

The `cn=schema` entry has a multivalued attribute, `objectClasses`, that contains definitions of each object class in the directory schema. You add custom schema by using the `ldapmodify` command. This example adds an object class `blogger` based on the attribute created in the previous example.

Manipulation of the `cn=schema` suffix is regarded as an administrative action and, as such, it is recommended that you use the administration connector when accessing this suffix. See "Managing Administration Traffic to the Server" on page 5 for more information.

1 Using a text editor, create an LDIF file with your schema extensions.

```
dn: cn=schema
changetype: modify
add: objectClasses
objectClasses: ( 1.3.6.1.4.1.26037.1.999.2000
NAME ( 'blogger' )
DESC 'Someone who has a blog'
SUP inetOrgPerson
STRUCTURAL
MAY blog
X-ORIGIN 'OpenDS Directory Server' )
```

2 Use the `ldapmodify` command to add the file.

```
$ ldapmodify -h localhost -p 4444 -D "cn=Directory Manager" -w password -X --useSSL \
  -a -f blogger.ldif
Processing MODIFY request for cn=schema
MODIFY operation successful for DN cn=schema
```

3 Verify the addition by displaying it with `ldapsearch`.

```
$ ldapsearch -h localhost -p 4444 -D "cn=Directory Manager" -w password -X --useSSL \
  -b cn=schema -s base --dontWrap "(objectclass=*)" objectClasses | grep 'blogger'
```

Note – The directory server automatically adds new object class definitions to the `99user.ldif` file.

▼ To Delete an Object Class

The `cn=schema` entry has a multivalued attribute, `objectClasses`, that contains definitions for each object class in the directory schema. You can delete these definitions by using the `ldapmodify` command.

Caution – Be careful when deleting object classes, because doing so can harm your directory. Do not delete an object class unless absolutely necessary.

Manipulation of the `cn=schema` suffix is regarded as an administrative action and, as such, it is recommended that you use the administration connector when accessing this suffix. See "Managing Administration Traffic to the Server" on page 5 for more information.

1 Create the delete request in LDIF format.

```
dn: cn=schema
changetype: modify
delete: objectClasses
objectClasses: ( 1.3.6.1.4.1.26037.1.999.2000
NAME ( 'blogger' )
DESC 'Someone who has a blog'
SUP inetOrgPerson
STRUCTURAL
MAY blog
X-ORIGIN 'OpenDS Directory Server' )
```

2 Remove the object class by using `ldapmodify` to apply the LDIF file.

```
$ ldapmodify -h localhost -p 4444 -D "cn=Directory Manager" -w password -X --useSSL \
  --fileName "remove_objectclass_schema.ldif"
```

Extending the Schema With a Custom Schema File

You can extend the schema by using a schema file that contains customized definitions. In general, the best practice is to modify the existing `99user.ldif` file in the *install-dir*`/config/schema` directory to add new definitions. When you update schema elements using LDAP, the new definitions are written to the `99user.ldif` file.

Alternatively, you can create a custom schema file and save it to the *install-dir*`/config/schema` directory. The directory server loads schema files in alphanumeric order with numbers loaded first. As such, you should name custom schema files as follows: `00-99`*filename*`.ldif`. The number should be higher than any standard schema file that has already been defined. If you name custom schema files with a number that is lower than the standard schema files, the server might encounter errors when loading the schema.

Replicating Directory Schema

In a replicated topology, schema definitions are automatically replicated to ensure that all servers use a single schema. Schema modifications on any server are replicated to all other servers in the topology.

When you configure replication, the schema of the first server is used to initialize the schema of the second server by default. You can, however, specify that the schema of the second server be used to initialize the schema of the first server. You can also specify that schema replication be disable altogether. For more information, see "Configuring Schema Replication" on page 261.

Managing the Schema With the Control Panel

You can use the Control Panel to manage the directory's schema, as described in these sections:

▼ To Display Schema Items

This procedure shows how to use the Control Panel to display items defined in the schema.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Click the Manage Schema link under theSchema menu on the left side of the Control Panel window.**

The Manage Schema window appears, displaying all configured schema object classes in a list on its left side.

3 **You can change the view of the schema items:**

- Double-click on a category in the list to expand it and display the attributes and sub-categories that it contains.
- Click on an item in the list to display its specifications.
- Filter the objects that are displayed. Select a category to filter on from the drop-down list, type a string in the blank field, and click the Apply button. Only the objects that match the filter are displayed.
- Double-click an object or attribute in the right panel of the window to display its configuration.

▼ To Add a New Object Class

This procedure shows how to use the Control Panel to add a new object class to the schema.

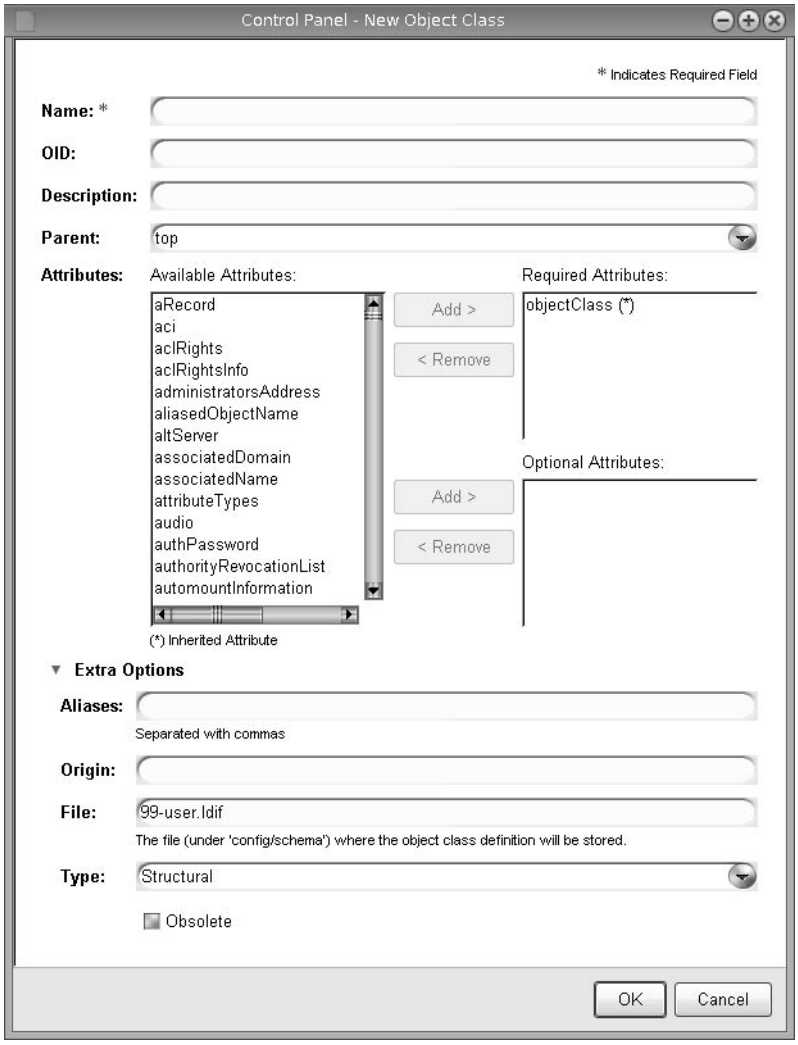
1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Click the Manage Schema link under the Schema menu on the left side of the Control Panel window.**

The Manage Schema window appears

3 **Click the New Object Class button.**

The New Object Class window appears.



4 **Specify the following information in the fields of the New Object Class window:**

Name A unique name to identify the new object class

OID	An OID that uniquely identifies the object class in the directory server. The LDAP v3 specification requires the OID to be a numeric number, but OpenDS supports the use of non-numeric OIDs for easy identification because the schema is used internally within the organization. The default format is *objectClassName*`-oid`, such as `person-oid`.
Description	A description of the object class
Parent	The superior object class from which the new object class inherits elements. OpenDS allows only one superior object class, although the LDAP v3 specification allows for multiple superior object classes.
Attributes	The set of attribute types that can be present (that is, have at least one value) in entries with the object class. Required attributes must be present (that is, have at least one value) in entries with that object class. Optional attributes can be present in such entries, but they are not required to be present. Select an attribute in the Available Attributes list and click one of the Add buttons to add it to either the Required Attributes list or the Optional Attributes list. Double-click an attribute to move it from the Available Attributes list to the Required Attributes list. double-click an attribute in either the Required Attributes list or the Optional Attributes list to return it to the Available Attributes list.

5 **Click the OK button.**

The New Object Class window displays the progress of the operation.

When the operation is complete, the new object class is created.

6 **When the operation is complete, click the Close button to close the New Object Class window.**

▼ To Add a New Attribute to the Schema

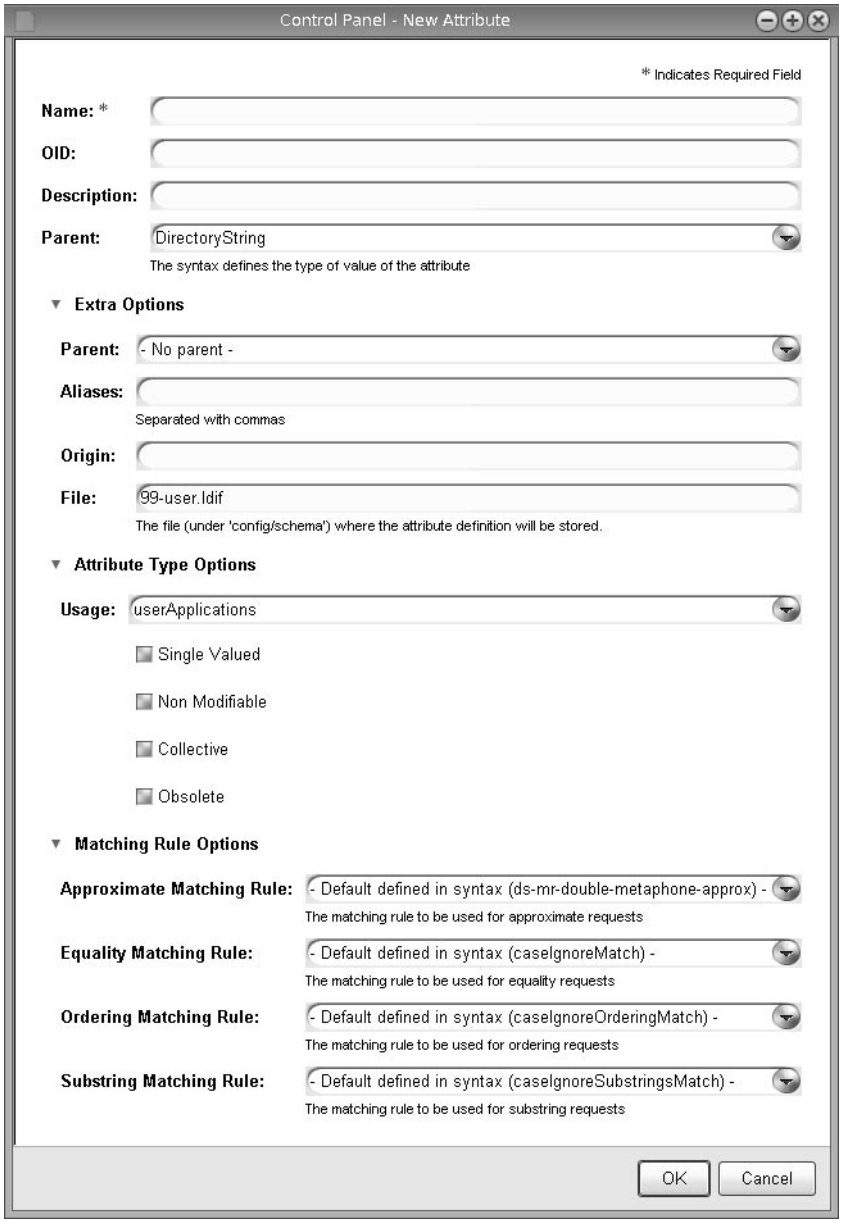
This procedure shows how to use the Control Panel to define a new attribute in the schema.

1 **Start the Control Panel, as described in "To Start the Control Panel" on page 28.**

2 **Click the Manage Schema link under the Schema menu on the left side of the Control Panel window. The Manage Schema window appears.**

3 **Click the New Attribute button.**

The New Attribute window appears.



4 Specify the following information in the fields of the New Attribute window:

Name	A unique name to identify the new attribute
OID	An OID that uniquely identifies the attribute in the directory server. The LDAP v3 specification requires the OID to be a numeric number, but OpenDS

supports the use of non-numeric OIDs for easy identification because the schema is used internally within the organization. The default format is *attributeName*`-oid`, such as `person-oid`.

Description — A description of the attribute

Parent — The superior object class from which the new attribute inherits elements.

OpenDS allows only one superior object class, although the LDAP v3 specification allows for multiple superior object classes.

Specify additional values in the fields for Extra Options, Attribute Type Options, and Matching Rule Options as needed.

5 Click the OK button.

The New Attribute window displays the progress of the operation.

When the operation is complete, the new attribute is created.

6 When the operation is complete, click the Close button to close the New Attribute window.