OpenDJ Administration Guide

Version 4.8.1-SNAPSHOT

Valery Kharseko
Mark Craig
Nemanja Lukić
Ludovic Poitou
Chris Ridd

ForgeRock AS
201 Mission St., Suite 2900
San Francisco, CA 94105, USA
+1 415-599-1100 (US)
`www.forgerock.com`

Abstract

Hands-on guide to configuring and using OpenDJ features. The OpenDJ project offers open source LDAP directory services in Java.

Table of Contents

Preface

This guide shows you how to configure, maintain, and troubleshoot OpenDJ directory services. This guide also describes file layouts, ports used, and standards, controls, extended operations, and languages supported for OpenDJ installations.

1 Who Should Read this Guide

This guide is written for directory designers and administrators who build, deploy, and maintain OpenDJ directory services for your organizations.

This guide starts by introducing the OpenDJ administrative interfaces and tools, and by showing how to manage OpenDJ server processes. It also demonstrates how to import and export directory data. This guide continues by showing how to configure and monitor the principle features of individual OpenDJ servers, and how to configure and monitor replicated server topologies for distributed high availability. It then demonstrates how to tune, troubleshoot, and move servers. This guide concludes with appendices of useful reference information for directory designers and administrators.

You do not need to be an LDAP wizard to learn something from this guide, though a background in directory services and maintaining server software can help. You do need some background in managing servers and services on your operating system of choice. You can nevertheless get started with this guide, and then learn more as you go along.

2 Formatting Conventions

Most examples in the documentation are created in GNU/Linux or Mac OS X operating environments. If distinctions are necessary between operating environments, examples are labeled with the operating environment name in parentheses. To avoid repetition file system directory names are often given only in UNIX format as in `/path/to/server`, even if the text applies to `C:\path\to\server` as well.

Absolute path names usually begin with the placeholder `/path/to/`. This path might translate to `/opt/`, `C:\Program Files\`, or somewhere else on your system.

Command line, terminal sessions are formatted as follows:

```
$ echo $JAVA_HOME
/path/to/jdk
```

Command output is sometimes formatted for narrower, more readable output even though formatting parameters are not shown in the command. In the following example, the query string parameter `_prettyPrint=true` is omitted and some of the output is replaced with an ellipsis (`...`):

```
$ curl https://bjensen:hifalutin@opendj.example.com:8443/users/newuser
{
  "_rev" : "000000005b337348",
  "_id" : "newuser",
  ...
}
```

Program listings are formatted as follows:

```
class Test {
    public static void main(String [] args)  {
        System.out.println("This is a program listing.");
    }
}
```

3 Accessing Documentation Online

Open Identity Platform core documentation, such as this document, aims to be technically accurate and complete with respect to the software documented.

Core documentation therefore follows a three-phase review process designed to eliminate errors:

- Product managers and software architects review project documentation design with respect to the readers' software lifecycle needs.
- Subject matter experts review proposed documentation changes for technical accuracy and completeness with respect to the corresponding software.
- Quality experts validate implemented documentation changes for technical accuracy, completeness in scope, and usability for the readership.

The review process helps to ensure that documentation published for a Open Identity Platform release is technically accurate and complete.

Fully reviewed, published core documentation is available at https://doc.openidentityplatform.org/. Use this documentation when working with a Open Identity Platform release.

You can find pre-release draft documentation at the online community resource center. Use this documentation when trying a nightly build.

4 Joining the Open Identity Platform Community

Visit the community resource center where you can find information about each project, download nightly builds, browse the resource catalog, ask and answer questions on the forums, find community events near you, and of course get the source code as well.

Chapter 1

Administration Interfaces & Tools

OpenDJ server software installs with a cross-platform, Java Swing-based Control Panel for many day-to-day tasks. OpenDJ server software also installs command-line tools for configuration and management tasks.

This chapter is one of the few to include screen shots of the control panel. Most examples make use of the command-line tools. Once you understand the concepts, and how to perform a task using the command-line tools, you no doubt need no more than to know where to start in the Control Panel to accomplish what you set out to do.

At a protocol level, administration tools and interfaces connect to servers through a different network port than that used to listen for traffic from other client applications.

This chapter takes a quick look at the tools for managing directory services.

1.1 Control Panel

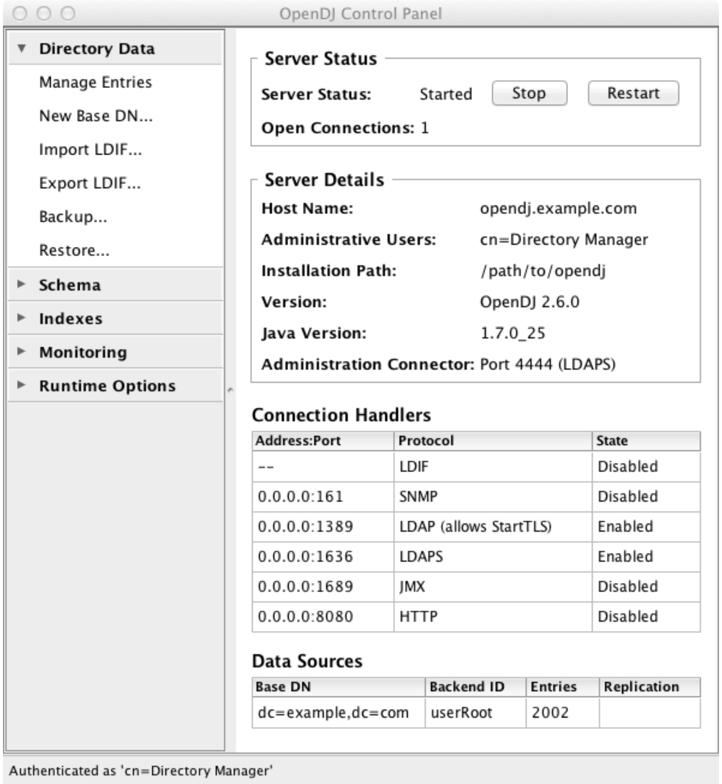
OpenDJ Control Panel offers a graphical user interface for managing both local and remote servers. You choose the server to manage when you start the Control Panel. The Control Panel connects to the administration server port, making a secure LDAPS connection.

Start OpenDJ Control Panel.

- (UNIX) Run **opendj/bin/control-panel**.

- (Windows) Double-click `opendj\bat\control-panel.bat`.
- (Mac OS X) Double-click `opendj/bin/ControlPanel.app`.

When you login to OpenDJ Control Panel, you authenticate over LDAP. This means that if users can run the Control Panel, they can use it to manage a running server. Yet, to start and stop the server process through OpenDJ Control Panel, you must start the Control Panel on the system where OpenDJ runs, as the user who owns the OpenDJ server files (such as the user who installed OpenDJ). In other words, the OpenDJ Control Panel does not do remote process management.



Server Status

Server Status: Started

Open Connections: 1

Server Details

Host Name:	opendj.example.com
Administrative Users:	cn=Directory Manager
Installation Path:	/path/to/opendj
Version:	OpenDJ 2.6.0
Java Version:	1.7.0_25
Administration Connector:	Port 4444 (LDAPS)

Connection Handlers

Address:Port	Protocol	State
--	LDIF	Disabled
0.0.0.0:161	SNMP	Disabled
0.0.0.0:1389	LDAP (allows StartTLS)	Enabled
0.0.0.0:1636	LDAPS	Enabled
0.0.0.0:1689	JMX	Disabled
0.0.0.0:8080	HTTP	Disabled

Data Sources

Base DN	Backend ID	Entries	Replication
dc=example,dc=com	userRoot	2002	

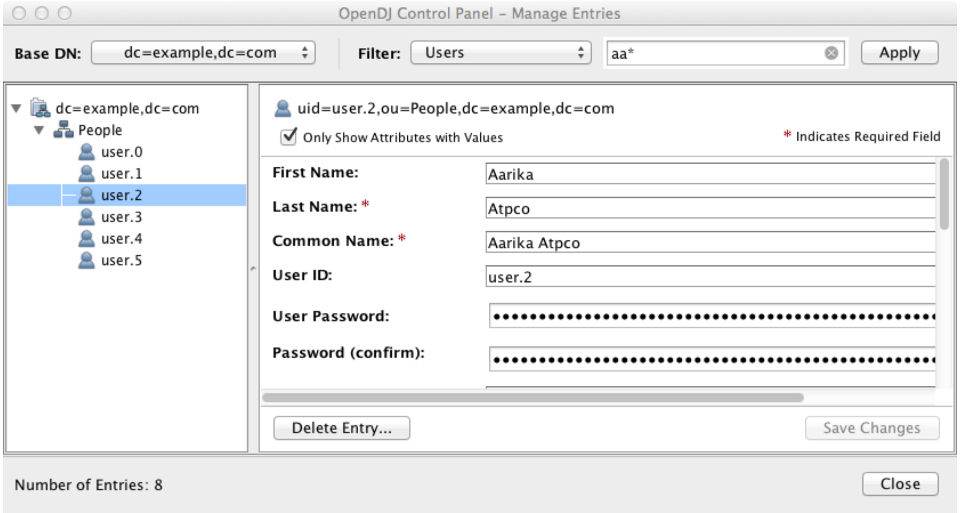
Authenticated as 'cn=Directory Manager'

OpenDJ Control Panel displays key information about the server.

Down the left side of OpenDJ Control Panel, notice what you can configure.

Directory Data

Directory data provisioning is typically not something you do by hand in most deployments. Usually entries are created, modified, and deleted through specific directory client applications. The Manage Entries window can be useful, however, both in the lab as you design and test directory data, and also if you modify individual ACIs or debug issues with particular entries.



OpenDJ Control Panel – Manage Entries

Base DN: dc=example,dc=com

Filter: Users aa*

uid=user.2,ou=People,dc=example,dc=com

☑ Only Show Attributes with Values

* Indicates Required Field

First Name: Aarika

Last Name: * Atpco

Common Name: * Aarika Atpco

User ID: user.2

User Password: ••

Password (confirm): ••

Number of Entries: 8

The Manage Entries window can check that your changes are valid before sending the request to the directory.

Additionally, the Directory Data list makes it easy to create a new base DN, and then import user data for the new base DN from LDIF. You can also use the tools in the list to export user data to LDIF, and to backup and restore user data.

Schema

The Manage Schema window lets you browse and modify the rules that define how data is stored in the directory. You can add new schema definitions such as new attribute types and new object classes while the server is running, and the changes you make take effect immediately.

Indexes

The Manage Indexes window gives you a quick overview of all the indexes currently maintained for directory attributes. To protect your directory resources from being absorbed by costly searches on unindexed attributes, you may choose to keep the default behavior, preventing unindexed searches, instead adding indexes required by specific applications. (Notice that if the number of user data entries is smaller than the default resource limits, you can still perform what appear to be unindexed searches. That is because the `dn2id` index returns all user data entries without hitting a resource limit that would make the search unindexed.)

OpenDJ Control Panel also allows you to verify and rebuild existing indexes, which you may have to do after an upgrade operation, or if you have reason to suspect index corruption.

Monitoring

The Monitoring list gives you windows to observe information about the system, the JVM used, and indications about how the cache is used, whether the work queue has been filling up, as well as details about the database. You can also view the numbers and types of requests arriving over the connection handlers, and the current tasks in progress as well.

Runtime Options

If you did not set appropriate JVM runtime options during the installation process, this is the list that allows you to do so through the Control Panel.

1.2 Command-Line Tools

All OpenDJ command-line tools take the `--help` option.

All commands call Java programs and therefore involve starting a JVM.

Setup, upgrade, and uninstall tools are located in the directory where you unpacked OpenDJ, such as `/path/to/opendj`. Find the additional command-line tools for your platform.

- (UNIX) In `opendj/bin`.
- (Windows) In `opendj\bat`.

The following list uses the UNIX names for the tools. On Windows all command-line tools have the extension .bat.

backup

Backup or schedule backup of directory data.

base64

Encode and decode data in base64 format.

Base64 encoding represents binary data in ASCII, and can be used to encode character strings in LDIF, for example.

create-rc-script (UNIX)

Generate a script you can use to start, stop, and restart the server either directly or at system boot and shutdown. Use **create-rc-script -f *`script-file`***.

dbtest

Debug JE databases.

dsconfig

The **dsconfig** command is the primary command-line tool for viewing and editing OpenDJ configuration. When started without arguments, **dsconfig** prompts you for administration connection information. Once connected it presents you with a menu-driven interface to the server configuration.

When you pass connection information, subcommands, and additional options to **dsconfig**, the command runs in script mode and so is not interactive.

You can prepare **dsconfig** batch scripts by running the tool with the `--commandFilePath` option in interactive mode, then reading from the batch file with the `--batchFile` option in script mode. Batch files can be useful when you have many **dsconfig** commands to run and want to avoid starting the JVM and setting up a new connection for each command.

In addition to the dsconfig reference that covers subcommands, the *Configuration Reference* covers the properties you can set using the **dsconfig** command.

dsjavaproperties

Apply changes you make to `opendj/config/java.properties`, which sets Java runtime options.

dsreplication

Configure data replication between directory servers to keep their contents in sync.

encode-password

Encode a clear text password according to one of the available storage schemes.

export-ldif

Export directory data to LDAP Data Interchange Format, a standard, portable, text-based representation of directory content.

import-ldif

Load LDIF content into the directory, overwriting existing data.

ldapcompare

Compare the attribute values you specify with those stored on entries in the directory.

ldapdelete

Delete one entry or an entire branch of subordinate entries in the directory.

ldapmodify

Modify the specified attribute values for the specified entries.

Use the **ldapmodify** command with the `-a` option to add new entries.

ldappasswordmodify

Modify user passwords.

ldapsearch

Search a branch of directory data for entries matching the LDAP filter that you specify.

ldif-diff

Display differences between two LDIF files, with the resulting output having LDIF format.

ldifmodify

Similar to the **ldapmodify** command, modify specified attribute values for specified entries in an LDIF file.

ldifsearch

Similar to the **ldapsearch** command, search a branch of data in LDIF for entries matching the LDAP filter you specify.

list-backends

List backends and base DNs served by OpenDJ.

make-ldif

Generate directory data in LDIF, based on templates that define how the data should appear.

The **make-ldif** command is designed to help you quickly generate test data that mimics data you expect to have in production, but without compromising private information.

manage-account

Lock and unlock user accounts, and view and manipulate password policy state information.

manage-tasks

View information about tasks scheduled to run in the server, and cancel specified tasks.

rebuild-index

Rebuild an index stored in a JE backend.

restore

Restore user data from backup.

start-ds

Start OpenDJ directory server.

status

Display information about the server.

stop-ds

Stop OpenDJ directory server.

verify-index

Verify that an index stored in a JE backend is not corrupt.

windows-service.bat (Windows)

Register OpenDJ as a Windows Service.

Chapter 2

Managing Server Processes

Using the OpenDJ Control Panel, you can start and stop local servers. You can also start and stop OpenDJ using command-line tools, and use the operating system's capabilities for starting OpenDJ at boot time.

This chapter demonstrates how to start and stop server processes with command line tools and using operating system capabilities. This chapter also describes what OpenDJ directory server does during startup and shutdown, and how it recovers following an abrupt shutdown such as happens during a system crash or when you kill the server process using system tools.

2.1 Starting a Server

Use one of the following techniques.

- Use the **start-ds** command.

  ```
  $ opendj/bin/start-ds
  ```

  Alternatively, you can specify the --no-detach option to start the server in the foreground.

- (UNIX) Create an RC script, and then use the script to start the server.

  Unless you run OpenDJ as root, use the --userName *userName* option to specify the user who installed OpenDJ.

```
$ sudo opendj/bin/create-rc-script
 --outputFile /etc/init.d/opendj
 --userName mark
$ sudo /etc/init.d/opendj start
```

For example, on Linux if you run OpenDJ as root, you can use the RC script to start the server at system boot, and stop the server at system shutdown.

```
$ sudo update-rc.d opendj defaults
update-rc.d: warning: /etc/init.d/opendj missing LSB information
update-rc.d: see <http://wiki.debian.org/LSBInitScripts>
 Adding system startup for /etc/init.d/opendj ...
   /etc/rc0.d/K20opendj -> ../init.d/opendj
   /etc/rc1.d/K20opendj -> ../init.d/opendj
   /etc/rc6.d/K20opendj -> ../init.d/opendj
   /etc/rc2.d/S20opendj -> ../init.d/opendj
   /etc/rc3.d/S20opendj -> ../init.d/opendj
   /etc/rc4.d/S20opendj -> ../init.d/opendj
   /etc/rc5.d/S20opendj -> ../init.d/opendj
```

- (Windows) Register OpenDJ as a Windows Service, and then manage the service through Windows administration tools.

```
C:\Users\Mark> opendj\bat\windows-service.bat --enableService
```

By default OpenDJ saves a compressed version of the server configuration used on successful startup. This ensures that the server provides a "last known good" configuration, which can be used as a reference or copied into the active configuration if the server fails to start with the current active configuration. It is possible, though not usually recommended, to turn this behavior off by changing the global server setting `save-config-on-successful-startup` to `false`.

2.2 Stopping a Server

Use one of the following techniques.

- Use the **stop-ds** command.

```
$ opendj/bin/stop-ds
```

- (UNIX) Create an RC script, and then use the script to stop the server.

```
$ sudo opendj/bin/create-rc-script
 --outputFile /etc/init.d/opendj
 --userName mark
$ sudo /etc/init.d/opendj stop
```

- (Windows) Register OpenDJ as a Windows Service, and then manage the service through Windows administration tools.

```
C:\Users\Mark> opendj\bat\windows-service.bat --enableService
```

2.3 Restarting a Server

Use one of the following techniques.

- Use the **stop-ds** command.

```
$ opendj/bin/stop-ds --restart
```

- (UNIX) Create an RC script, and then use the script to stop the server.

```
$ sudo opendj/bin/create-rc-script
 --outputFile /etc/init.d/opendj
 --userName mark
$ /etc/init.d/opendj restart
```

- (Windows) Register OpenDJ as a Windows Service, and then manage the service through Windows administration tools.

```
C:\Users\Mark> opendj\bat\windows-service.bat --enableService
```

2.4 Server Recovery

OpenDJ tends to show resilience when restarting after a crash or after the server process is killed abruptly. OpenDJ might have to replay the last few entries in a transaction log. Generally OpenDJ returns to service quickly.

You can find Berkeley Java Edition database recovery messages in the database log file, such as `/path/to/opendj/db/userRoot/je.info.0`. The following shows two example messages from that log, the first written at the beginning of the recovery process, the second written at the end of the process.

```
111104 10:23:48:967 CONFIG [/path/to/opendj/db/userRoot]Recovery
 underway, found end of log
...
111104 10:23:49:015 CONFIG [/path/to/opendj/db/userRoot]Recovery finished:
 Recovery Info ...
```

What can take some time during server startup is preloading database content into memory when the server starts. Objects cached in memory do not survive a crash. By default, OpenDJ does not cache objects in memory before starting to accept client requests. You can however set a `preload-time-limit` for the database cache of your backend if you do want to load objects into the database cache before OpenDJ begins accepting client connections.

Chapter 3

Importing & Exporting LDIF Data

LDAP Data Interchange Format provides a mechanism for representing directory data in text format. LDIF data is typically used to initialize directory databases, but also may be used to move data between different directories that cannot replicate directly, or even as an alternative backup format.

This chapter shows you how to import and export LDIF. This chapter also covers creating test data in LDIF format, and manipulating LDIF data with command-line tools.

3.1 Generating Test Data

When you install OpenDJ, you have the option of importing sample data generated during the installation. This procedure demonstrates how to generate LDIF using the **make-ldif** command.

Procedure 3.1. To Generate Test LDIF Data

The **make-ldif** command uses templates to provide sample data. Default templates are located in the `OpenDJ/config/MakeLDIF/` directory. The `example.template` file can be used to create a suffix with entries of the type `inetOrgPerson`. You can do the equivalent in OpenDJ Control Panel (Directory Data > New Base DN... > Import Automatically Generated Example Data).

1. Write a file to act as the template for your generated LDIF.

The resulting test data template depends on what data you expect to encounter in production. Base your work on your knowledge of the production data, and on the sample template, `OpenDJ/config/MakeLDIF/example.template`, and associated data.

See *make-ldif.template* for reference information about template files.

2. Create additional data files for the content in your template to be selected randomly from a file, rather than generated by an expression.

   Additional data files are located in the same directory as your template file.

3. Decide whether you want to generate the same test data each time you run the **make-ldif** command with your template.

   If so, provide the same `randomSeed` integer each time you run the command.

4. Before generating a very large LDIF file, make sure you have enough space on disk.

5. Run the **make-ldif** command to generate your LDIF file.

   ```
   $ make-ldif
    --randomSeed 0
    --templateFile /path/to/my.template
    --ldifFile /path/to/generated.ldif
   Processed 1000 entries
   Processed 2000 entries
   ...
   Processed 10000 entries
   LDIF processing complete.  10003 entries written
   ```

3.2 Importing & Exporting Data

You can use the OpenDJ Control Panel to import data (Directory Data > Import LDIF...) and to export data (Directory Data > Export LDIF...). The following procedures demonstrate how to use the **import-ldif** and **export-ldif** commands.

Procedure 3.2. To Import LDIF Data

The most efficient method of importing LDIF data is to take the OpenDJ server offline. Alternatively, you can schedule a task to import the data while the server is online.

1. (Optional) If you do not want to use the default `userRoot` backend, create a new JE backend for your data.

   See Section 3.4, “Creating a New Database Backend” for details.

2. The following example imports `dc=example,dc=org` data into the `userRoot` backend, overwriting existing data.

   - If you want to speed up the process—for example because you have millions of directory entries to import—first shut down the server, and then run the **import-ldif** command.

     ```
     $ stop-ds
     $ import-ldif
      --offline
      --includeBranch dc=example,dc=org
      --backendID userRoot
      --ldifFile /path/to/generated.ldif
     ```

   - If not, schedule a task to import the data while online.

     ```
     $ import-ldif
      --port 4444
      --hostname opendj.example.com
      --bindDN "cn=Directory Manager"
      --bindPassword password
      --includeBranch dc=example,dc=org
      --backendID userRoot
      --ldifFile /path/to/generated.ldif
      --trustAll
     ```

     Notice that the task is scheduled through communication over SSL on the administration port, by default `4444`. You can schedule the import task to start at a particular time using the `--start` option.

     The `--trustAll` option trusts all SSL certificates, such as a default self-signed certificate used for testing.

Procedure 3.3. To Export LDIF Data

- The following example exports `dc=example,dc=org` data from the `userRoot` backend.

  - If you want to speed up export, first shut down the server, and then export data using the **export-ldif** command.

    ```
    $ stop-ds
    $ export-ldif
     --offline
     --includeBranch dc=example,dc=org
     --backendID userRoot
     --ldifFile /path/to/backup.ldif
    ```

  - If not, schedule a task to export the data while online.

```
$ export-ldif
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --includeBranch dc=example,dc=org
 --backendID userRoot
 --ldifFile /path/to/backup.ldif
 --start 20111221230000
 --trustAll
```

The `--start 20111221230000` option tells OpenDJ to start the export at 11 PM on December 21, 2012.

If OpenDJ is stopped at this time, then when you start OpenDJ again, the server attempts to perform the task after starting up.

3.3 Other Tools For Working With LDIF Data

This section demonstrates the **ldifsearch**, **ldifmodify**, and **ldif-diff** tools.

3.3.1 Searching in LDIF With ldifsearch

The **ldifsearch** command lets you search LDIF files in a similar way to how you search LDAP directories with the **ldapsearch** command.

```
$ ldifsearch
 --baseDN dc=example,dc=org
 --ldifFile generated.ldif
 "(sn=Grenier)"
 mobile
dn: uid=user.4630,ou=People,dc=example,dc=org
mobile: +1 728 983 6669
```

The `--ldifFile` *`ldif-file`* option replaces the `--hostname` and `--port` options used to connect to an LDAP directory. Otherwise the command syntax and LDIF output is familiar to **ldapsearch** users.

3.3.2 Updating LDIF With ldifmodify

The **ldifmodify** command lets you apply changes to LDIF files, generating a new, changed version of the original file.

```
$ cat changes.ldif
dn: uid=user.0,ou=People,dc=example,dc=org
changetype: modify
replace: description
description: This is the new description for Aaccf Amar.
-
replace: initials
initials: AAA

$ ldifmodify
 --sourceLDIF generated.ldif
 --changesLDIF changes.ldif
 --targetLDIF new.ldif
```

Notice that the resulting new LDIF file is likely to be about the same size as the source LDIF file.

3.3.3 Comparing LDIF With ldif-diff

The **ldif-diff** command reports differences between two LDIF files in LDIF format.

```
$ ldif-diff --sourceLDIF old.ldif --targetLDIF new.ldif
dn: uid=user.0,ou=People,dc=example,dc=org
changetype: modify
add: initials
initials: AAA
-
delete: initials
initials: ASA
-
add: description
description: This is the new description for Aaccf Amar.
-
delete: description
description: This is the description for Aaccf Amar.
```

As the **ldif-diff** command reads both files into memory, constructing tree maps to perform the comparison, the command is designed to work with small files and fragments. The command can quickly run out of memory when calculating differences between large files.

3.4 Creating a New Database Backend

OpenDJ stores your data in a *backend*. OpenDJ stores directory data in backends. Backends are what you backup and restore. By default, OpenDJ stores your data in a backend named `userRoot`. You can create new backends using the **dsconfig** command. The following example creates a local backend named `testData`.

```
$ dsconfig create-backend --backend-name testData --type local-db
```

```
>>>> Configuring the "base-dn" property

    Specifies the base DN(s) for the data that the backend handles.

    A single backend may be responsible for one or more base DNs. Note that no
    two backends may have the same base DN although one backend may have a
    base DN that is below a base DN provided by another backend (similar to
    the use of sub-suffixes in the Sun Java System Directory Server). If any
    of the base DNs is subordinate to a base DN for another backend, then all
    base DNs for that backend must be subordinate to that same base DN.

    Syntax:  DN

Enter a value for the "base-dn" property: dc=example,dc=org

Enter another value for the "base-dn" property [continue]:


>>>> Configuring the "enabled" property

    Indicates whether the backend is enabled in the server.

    If a backend is not enabled, then its contents are not accessible when
    processing operations.

Select a value for the "enabled" property:

    1)  true
    2)  false

    ?)  help
    q)  quit

Enter choice: 1


>>>> Configure the properties of the Local DB Backend

        Property           Value(s)
        --------------------------------------
    1)  backend-id         testData
    2)  base-dn            "dc=example,dc=org"
    3)  compact-encoding   true
    4)  db-cache-percent   10
    5)  db-cache-size      0 b
    6)  db-directory       db
    7)  enabled            true
    8)  index-entry-limit  4000
    9)  writability-mode   enabled

    ?)  help
    f)  finish - create the new Local DB Backend
    q)  quit

Enter choice [f]:

The Local DB Backend was created successfully
```

Alternatively, you can create a new backend in OpenDJ Control Panel (Directory Data > New Base DN... > Backend > New Backend: *backend-name*).

3.5 Deleting a Database Backend

You delete a database backend by using the **dsconfig delete-backend** command.

When you delete a database backend by using the **dsconfig delete-backend** command, OpenDJ does not actually remove the database files for two reasons. First, a mistake could potentially cause lots of data to be lost. Second, deleting a large database backend could cause severe service degradation due to a sudden increase in I/O load.

Instead, after you run the **dsconfig delete-backend** command you must also manually remove the database backend files.

If you do run the **dsconfig delete-backend** command by mistake and have not yet deleted the actual files, then you can recover from the mistake by creating the backend again, reconfiguring the indexes that were removed, and rebuilding the indexes as described in the section on *Configuring & Rebuilding Indexes*.

Chapter 4

Configuring Connection Handlers

This chapter shows you how to configure OpenDJ directory server to listen for directory client requests, using connection handlers. You can view information about connection handlers in the OpenDJ Control Panel, and update the configuration using the **dsconfig** command.

4.1 LDAP Client Access

You configure LDAP client access by using the command-line tool **dsconfig**. By default you configure OpenDJ to listen for LDAP when you install.

The standard port number for LDAP client access is 389. If you install OpenDJ directory server as a user who can use port 389 and the port is not yet in use, then 389 is the default port number presented at installation time. If you install as a user who cannot use a port < 1024, then the default port number presented at installation time is 1389.

Procedure 4.1. To Change the LDAP Port Number

1. Change the port number using the **dsconfig** command.

```
$ dsconfig
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "LDAP Connection Handler"
 --set listen-port:11389
 --trustAll
 --no-prompt
```

This example changes the port number to 11389 in the configuration.

2. Restart the connection handler so the change takes effect.

   To restart the connection handler, you disable it, then enable it again.

```
$ dsconfig
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "LDAP Connection Handler"
 --set enabled:false
 --trustAll
 --no-prompt
$ dsconfig
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "LDAP Connection Handler"
 --set enabled:true
 --trustAll
 --no-prompt
```

4.2 Preparing For Secure Communications

One common way to protect connections between OpenDJ and client applications involves using StartTLS for LDAP or LDAPS to secure connections. OpenDJ and client applications use X.509 digital certificates to set up secure connections.

Both OpenDJ and client applications check that certificates are signed by a trusted party before accepting them. Merely setting up a secure connection therefore involves a sort of authentication using certificates. If either OpenDJ or the client application cannot trust the peer certificate, then the attempt to set up a secure connection must fail.

By default OpenDJ client tools prompt you if they do not recognize the server certificate. Other clients might not prompt you. OpenDJ server has no one to prompt when a client presents a certificate that cannot be trusted, so it must simply refuse to set up the connection.[1] In other words, it is important for both OpenDJ and client applications to be able to verify that peer certificates exchanged have been signed by a trusted party.

In practice this means that both OpenDJ and client applications must put the certificates that were used to sign each others' certificates in their respective trust stores. Conventionally, certificates are therefore signed by a Certificate Authority (CA). A CA is trusted to sign other certificates. The Java runtime environment for example comes with a trust store holding certificates from many well-known CAs.[2] If your client uses a valid certificate signed by one of these CAs, then OpenDJ can verify the certificate without additional configuration, because OpenDJ can find the CA certificate in the Java CA certificate trust store. Likewise if you set up StartTLS or LDAPS in OpenDJ using a valid certificate signed by one of these CAs, then many client applications can verify the OpenDJ server certificate without further configuration.

In summary, if you need a certificate to be recognized automatically, get the certificate signed by a well-known CA.

You can, however, choose to have your certificates signed some other way. You can set up your own CA. You can use a CA whose signing certificate is not widely distributed. You can also use self-signed certificates. In each case, you must add the signing certificates into the trust store of each peer making secure connections.

For OpenDJ directory server, you can choose to import your own CA-signed certificate as part of the installation process, or later using command-line tools. Alternatively, you can let the OpenDJ installation program create a self-signed certificate as part of the OpenDJ installation process. In addition, you can add a signing certificate to the OpenDJ trust store using the Java **keytool** command.

The following example shows the **keytool** command to add a client application's binary format, self-signed certificate to the OpenDJ trust store (assuming OpenDJ is already configured to use secure connections). This enables OpenDJ to recognize the self-signed client application certificate. (By definition a self-signed certificate is itself the signing certificate. Notice that the Owner and the Issuer are the same.)

[1] Unless you use the Blind Trust Manager Provider, which is recommended only for test purposes.

[2] `$JAVA_HOME/jre/lib/security/cacerts` holds the CA certificates. To read the full list, use the following command.

```
$ keytool
 -list
 -v
 -keystore $JAVA_HOME/jre/lib/security/cacerts
 -storepass changeit
```

```
$ keytool
 -import
 -alias myapp-cert
 -file myapp-cert.crt
 -keystore /path/to/opendj/config/truststore
 -storepass `cat /path/to/opendj/config/keystore.pin`
Owner: CN=My App, OU=Apps, DC=example, DC=com
Issuer: CN=My App, OU=Apps, DC=example, DC=com
Serial number: 5ae2277
Valid from: Fri Jan 18 18:27:09 CET 2013 until: Thu Jan 13 18:27:09 CET 2033
Certificate fingerprints:
  MD5:  48:AC:F9:13:11:E0:AB:C4:65:A2:83:9E:DB:FE:0C:37
  SHA1: F9:61:54:37:AA:C1:BC:92:45:07:64:4B:23:6C:BC:C9:CD:1D:44:0F
  SHA256: 2D:B1:58:CD:33:40:E9:ED:...:EA:C9:FF:6A:19:93:FE:E4:84:E3
  Signature algorithm name: SHA256withRSA
  Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 54 C0 C5 9C 73 37 85 4B   F2 3B D3 37 FD 45 0A AB  T...s7.K.;.7.E..
0010: C9 6B 32 95                                        .k2.
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

When working with a certificate in printable encoding format (.pem) rather than binary format, use the `-rfc` option, too.

Restart OpenDJ after adding certificates to the trust store to make sure that OpenDJ reads the updated trust store file.

On the client side, if your applications are also Java applications, then you can also import the OpenDJ signing certificate into the trust store for the applications using the **keytool** command.

The following example shows the **keytool** command to export the OpenDJ self-signed certificate in binary format.

```
$ keytool
 -export
 -alias server-cert
 -file server-cert.crt
 -keystore /path/to/opendj/config/keystore
 -storepass `cat /path/to/opendj/config/keystore.pin`
Certificate stored in file <server-cert.crt>
```

Importing the server certificate is similar to importing the client certificate, as shown above.

The following sections describe how to get and install certificates for OpenDJ directory server on the command line, for use when setting up StartTLS or LDAPS.

Procedure 4.2. To Request and Install a CA-Signed Certificate

First you create a server certificate in a Java Key Store. Next you issue a signing request to the CA, and get the CA-signed certificate as a reply. Then you set up the Key Manager Provider and Trust Manager Provider to rely on your new server certificate stored in the OpenDJ key store.

1. Generate the server certificate by using the Java **keytool** command.

   The CN attribute value is the FQDN for OpenDJ directory server, which you can see under Server Details in the OpenDJ Control Panel.

   ```
   $ keytool
    -genkey
    -alias server-cert
    -keyalg rsa
    -dname "CN=opendj.example.com,O=Example Corp,C=FR"
    -keystore /path/to/opendj/config/keystore
    -storepass changeit
    -keypass changeit
   ```

   Note

   Notice that the `-storepass` and `-keypass` options take identical password arguments. OpenDJ requires that you use the same password to protect both the keystore and also the private key.

2. Create a certificate signing request file for the certificate you generated.

   ```
   $ keytool
    -certreq
    -alias server-cert
    -keystore /path/to/opendj/config/keystore
    -storepass changeit
    -file server-cert.csr
   ```

3. Have the CA sign the request (`server-cert.csr`).

   See the instructions from your CA on how to provide the request.

   The CA returns the signed certificate.

4. If you have set up your own CA and signed the certificate, or are using a CA whose signing certificate is not included in the Java runtime environment, import the CA certificate into the key store so that it can be trusted.

   Otherwise, when you import the signed certificate in the reply from the (unknown) CA, **keytool** fails to import the signed certificate with the message `keytool error: java.lang.Exception: Failed to establish chain from reply`.

   The following example illustrates import of a CA certificate created with the **openssl** command. See the **openssl** documentation for instructions on creating CAs and on signing other certificates with the CA you created.

```
$ keytool
 -import
 -keystore /path/to/opendj/config/keystore
 -file ca.crt
 -alias ca-cert
 -storepass changeit
Owner: EMAILADDRESS=admin@example.com, CN=Example CA, O=Example Corp, C=FR
Issuer: EMAILADDRESS=admin@example.com, CN=Example CA, O=Example Corp, C=FR
Serial number: d4586ea05c878b0c
Valid from: Tue Jan 29 09:30:31 CET 2013 until: Mon Jan 24 09:30:31 CET 2033
Certificate fingerprints:
  MD5:  8A:83:61:9B:E7:18:A2:21:CE:92:94:96:59:68:60:FA
  SHA1: 01:99:18:38:3A:57:D7:92:7B:D6:03:8C:7B:E4:1D:37:45:0E:29:DA
  SHA256: 5D:20:F1:86:CC:CD:64:50:...:DF:15:43:07:69:44:00:FB:36:CF
  Signature algorithm name: SHA1withRSA
  Version: 3

Extensions:

#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: 30 07 67 7D 1F 09 B6 E6   90 85 95 58 94 37 FD 31  0.g........X.7.1
0010: 03 D4 56 7B                                        ..V.
]
[EMAILADDRESS=admin@example.com, CN=Example CA, O=Example Corp, C=FR]
SerialNumber: [    d4586ea0 5c878b0c]
]

#2: ObjectId: 2.5.29.19 Criticality=false
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 30 07 67 7D 1F 09 B6 E6   90 85 95 58 94 37 FD 31  0.g........X.7.1
0010: 03 D4 56 7B                                        ..V.
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

5. Import the signed certificate from the CA reply into the keystore where you generated the server certificate.

   In this example the certificate from the reply is `~/Downloads/server-cert.crt`.

   ```
   $ keytool
    -import
    -trustcacerts
    -alias server-cert
    -file ~/Downloads/server-cert.crt
    -keystore /path/to/opendj/config/keystore
    -storepass changeit
    -keypass changeit
   Certificate reply was installed in keystore
   ```

6. Configure the File Based Key Manager Provider for JKS to use the file name and key store PIN that you set up with the **keytool** command.

   ```
   $ dsconfig
    set-key-manager-provider-prop
    --hostname opendj.example.com
    --port 4444
    --bindDN "cn=Directory Manager"
    --bindPassword password
    --provider-name JKS
    --set enabled:true
    --set key-store-pin:changeit
    --remove key-store-pin-file:config/keystore.pin
    --trustAll
    --no-prompt
   ```

7. Configure the File Based Trust Manager Provider for JKS to use the key store and PIN as well.

   ```
   $ dsconfig
    set-trust-manager-provider-prop
    --hostname opendj.example.com
    --port 4444
    --bindDN "cn=Directory Manager"
    --bindPassword password
    --provider-name JKS
    --set enabled:true
    --set trust-store-file:config/keystore
    --set trust-store-pin:changeit
    --trustAll
    --no-prompt
   ```

   At this point, OpenDJ directory server can use your new CA-signed certificate, for example for StartTLS and LDAPS connection handlers.

8. If you use a CA certificate that is not known to clients, such as a CA that you set up yourself rather than a well-known CA whose certificate is included with the client system, import the CA certificate into the client application trust store. Otherwise the client application cannot trust the signature on the OpenDJ CA-signed server certificate.

Procedure 4.3. To Create & Install a Self-Signed Certificate

If you choose to configure LDAP Secure Access when setting up OpenDJ directory server, the setup program generates a key pair in the Java Key Store `/path/to/opendj/config/keystore`, and self-signs the public key certificate, which has the alias `server-cert`. The password for the key store and the private key is stored in clear text in the file `/path/to/opendj/config/keystore.pin`.

If you want to secure communications, but did not chose to configure LDAP Secure Access at setup time, this procedure can help. The following steps explain how to create and install a key pair with a self-signed certificate in preparation to configure LDAPS or HTTPS. First you create a key pair in a new Java Key Store, and then self-sign the certificate. Next, you set up the Key Manager Provider and Trust Manager Provider to access the new server certificate in the new key store.

If instead you want to *replace the existing server key pair with self-signed certificate*, then first use **keytool -delete -alias server-cert** to delete the existing keys before you generate a new key pair with the same alias. You can also either reuse the existing password in `keystore.pin`, or use a new password as shown in the steps below.

1. Generate the server certificate using the Java **keytool** command.

```
$ keytool
 -genkey
 -alias server-cert
 -keyalg rsa
 -dname "CN=opendj.example.com,O=Example Corp,C=FR"
 -keystore /path/to/opendj/config/keystore
 -storepass changeit
 -keypass changeit
```

   In this example, OpenDJ is running on a system with fully qualified host name `opendj.example.com`. The Java Key Store (JKS) is created in the `config` directory where OpenDJ is installed, which is the default value for JKS.

   Note

   Notice that the `-storepass` and `-keypass` options take identical password arguments. OpenDJ requires that you use the same password to protect both the key store and also the private key.

   Keep track of the password provided to the `-storepass` and `-keypass` options.

2. Self-sign the server certificate.

```
$ keytool
 -selfcert
 -alias server-cert
 -keystore /path/to/opendj/config/keystore
 -storepass changeit
```

3. Configure the File Based Key Manager Provider for JKS to access the Java Key Store with key store/private key password.

   In this example, the alias is `server-cert` and the password is `changeit`.

   If you are replacing a key pair with a self-signed certificate, reusing the `server-cert` alias and password stored in `keystore.pin`, then you can skip this step.

```
$ echo changeit > /path/to/opendj/config/keystore.pin
$ chmod 600 /path/to/opendj/config/keystore.pin
$ dsconfig
 set-key-manager-provider-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name JKS
 --set enabled:true
 --set key-store-file:config/keystore
 --set key-store-pin-file:config/keystore.pin
 --trustAll
 --no-prompt
```

4. Configure the File Based Trust Manager Provider for JKS to use the key store and PIN as well.

   If you skipped the previous step, you can also skip this step.

```
$ dsconfig
 set-trust-manager-provider-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name JKS
 --set enabled:true
 --set trust-store-file:config/keystore
 --set trust-store-pin-file:config/keystore.pin
 --trustAll
 --no-prompt
```

   At this point, OpenDJ directory server can use your new self-signed certificate, for example for StartTLS and LDAPS or HTTPS connection handlers.

4.3 LDAP Client Access With Transport Layer Security

StartTLS (Transport Layer Security) negotiations start on the unsecure LDAP port, and then protect communication with the client. You can opt to configure StartTLS during installation, or later using the **dsconfig** command.

Procedure 4.4. To Enable StartTLS on the LDAP Port

1. Make sure you have a server certificate installed.

```
$ keytool
 -list
 -alias server-cert
 -keystore /path/to/opendj/config/keystore
 -storepass `cat /path/to/opendj/config/keystore.pin`
server-cert, Jun 17, 2013, PrivateKeyEntry,
Certificate fingerprint (SHA1): 92:B7:4C:4F:2E:24:...:EB:7C:22:3F
```

2. Activate StartTLS on the current LDAP port.

```
$ dsconfig
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "LDAP Connection Handler"
 --set allow-start-tls:true
 --set key-manager-provider:JKS
 --set trust-manager-provider:JKS
 --trustAll
 --no-prompt
```

The change takes effect. No need to restart the server.

4.4 LDAP Client Access Over SSL

You configure LDAPS (LDAP/SSL) client access by using the command-line tool **dsconfig**. You can opt to configure LDAPS access when you install.

The standard port number for LDAPS client access is 636. If you install OpenDJ directory server as a user who can use port 636 and the port is not yet in use, then 636 is the default port number presented at installation time. If you install as a user who cannot use a port < 1024, then the default port number presented at installation time is 1636.

Procedure 4.5. To Set Up LDAPS Access

1. Make sure you have a server certificate installed.

   ```
   $ keytool
    -list
    -alias server-cert
    -keystore /path/to/opendj/config/keystore
    -storepass `cat /path/to/opendj/config/keystore.pin`
   server-cert, Jun 17, 2013, PrivateKeyEntry,
   Certificate fingerprint (SHA1): 92:B7:4C:4F:2E:24:...:EB:7C:22:3F
   ```

2. Configure the server to activate LDAPS access.

   ```
   $ dsconfig
    set-connection-handler-prop
    --hostname opendj.example.com
    --port 4444
    --bindDN "cn=Directory Manager"
    --bindPassword password
    --handler-name "LDAPS Connection Handler"
    --set listen-port:1636
    --set enabled:true
    --set use-ssl:true
    --trustAll
    --no-prompt
   ```

   This example changes the port number to 1636 in the configuration.

Procedure 4.6. To Change the LDAPS Port Number

1. Change the port number using the **dsconfig** command.

   ```
   $ dsconfig
    set-connection-handler-prop
    --hostname opendj.example.com
    --port 4444
    --bindDN "cn=Directory Manager"
    --bindPassword password
    --handler-name "LDAPS Connection Handler"
    --set listen-port:11636
    --trustAll
    --no-prompt
   ```

   This example changes the port number to 11636 in the configuration.

2. Restart the connection handler so the change takes effect.

   To restart the connection handler, you disable it, then enable it again.

```
$ dsconfig
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "LDAPS Connection Handler"
 --set enabled:false
 --trustAll
 --no-prompt
$ dsconfig
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "LDAPS Connection Handler"
 --set enabled:true
 --trustAll
 --no-prompt
```

4.5 Restricting Client Access

Using the OpenDJ directory server global configuration properties, you can add global restrictions on how clients access the server. These settings are per server, and so much be set independently on each server in replication topology.

These global settings are fairly coarse-grained. For a full discussion of the rich set of administrative privileges and fine-grained access control instructions that OpenDJ supports, see the chapter on *Configuring Privileges & Access Control*.

Consider the following global configuration settings.

`bind-with-dn-requires-password`

> Whether the directory server should reject any simple bind request that contains a DN but no password. Default: `true`
>
> To change this setting use the following command.

```
$ dsconfig
 set-global-configuration-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set bind-with-dn-requires-password:false
 --no-prompt
```

`max-allowed-client-connections`

> Restricts the number of concurrent client connections to the directory server. Default: 0, meaning no limit is set

To set a limit of 32768 use the following command.

```
$ dsconfig
 set-global-configuration-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set max-allowed-client-connections:32768
 --no-prompt
```

`reject-unauthenticated-requests`

Rejects any request (other than bind or StartTLS requests) received from a client that has not yet been authenticated, whose last authentication attempt was unsuccessful, or whose last authentication attempt used anonymous authentication. Default: `false`

To shut down anonymous binds use the following command.

```
$ dsconfig
 set-global-configuration-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set reject-unauthenticated-requests:true
 --no-prompt
```

`return-bind-error-messages`

Does not restrict access, but by default prevents OpenDJ directory server from returning extra information about why a bind failed, as that information could be used by an attacker. Instead, the information is written to the server errors log. Default: `false`

To have OpenDJ return additional information about why a bind failed use the following command.

```
$ dsconfig
 set-global-configuration-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set return-bind-error-messages:true
 --no-prompt
```

4.6 TLS Protocols & Cipher Suites

By default OpenDJ supports the SSL and TLS protocols and the cipher suites supported by the underlying Java virtual machine. For details see the documentation for the Java virtual machine in which you run OpenDJ. For Oracle Java, see the *Java Cryptography Architecture Oracle Providers Documentation* for the The `SunJSSE` Provider.

To list the available protocols and cipher suites, read the `supportedTLSProtocols` and `supportedTLSCiphers` attributes of the root DSE. Install unlimited strength Java cryptography extensions for stronger ciphers.

```
$ ldapsearch --port 1389 --baseDN "" --searchScope base "(objectclass=*)"
 supportedTLSCiphers supportedTLSProtocols
dn:
supportedTLSCiphers: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
supportedTLSCiphers: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
supportedTLSCiphers: TLS_RSA_WITH_AES_128_CBC_SHA256
supportedTLSCiphers: TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256
supportedTLSCiphers: TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256
supportedTLSCiphers: TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
supportedTLSCiphers: TLS_DHE_DSS_WITH_AES_128_CBC_SHA256
supportedTLSCiphers: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
supportedTLSCiphers: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
supportedTLSCiphers: TLS_RSA_WITH_AES_128_CBC_SHA
supportedTLSCiphers: TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA
supportedTLSCiphers: TLS_ECDH_RSA_WITH_AES_128_CBC_SHA
supportedTLSCiphers: TLS_DHE_RSA_WITH_AES_128_CBC_SHA
supportedTLSCiphers: TLS_DHE_DSS_WITH_AES_128_CBC_SHA
supportedTLSCiphers: TLS_ECDHE_ECDSA_WITH_RC4_128_SHA
supportedTLSCiphers: TLS_ECDHE_RSA_WITH_RC4_128_SHA
supportedTLSCiphers: SSL_RSA_WITH_RC4_128_SHA
supportedTLSCiphers: TLS_ECDH_ECDSA_WITH_RC4_128_SHA
supportedTLSCiphers: TLS_ECDH_RSA_WITH_RC4_128_SHA
supportedTLSCiphers: TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA
supportedTLSCiphers: TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
supportedTLSCiphers: SSL_RSA_WITH_3DES_EDE_CBC_SHA
supportedTLSCiphers: TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA
supportedTLSCiphers: TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA
supportedTLSCiphers: SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA
supportedTLSCiphers: SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA
supportedTLSCiphers: SSL_RSA_WITH_RC4_128_MD5
supportedTLSCiphers: TLS_EMPTY_RENEGOTIATION_INFO_SCSV
supportedTLSProtocols: SSLv2Hello
supportedTLSProtocols: SSLv3
supportedTLSProtocols: TLSv1
supportedTLSProtocols: TLSv1.1
supportedTLSProtocols: TLSv1.2
```

You can restrict the list of protocols and cipher suites used by setting the `ssl-protocol` and `ssl-cipher-suite` connection handler properties to include only the protocols or cipher suites you want.

For example, to restrict the cipher suites to `TLS_EMPTY_RENEGOTIATION_INFO_SCSV` and `TLS_RSA_WITH_AES_256_CBC_SHA` use the **dsconfig set-connection-handler-prop** command as shown in the following example.

```
$ dsconfig
   set-connection-handler-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "LDAPS Connection Handler"
 --add ssl-cipher-suite:TLS_EMPTY_RENEGOTIATION_INFO_SCSV
 --add ssl-cipher-suite:TLS_RSA_WITH_AES_256_CBC_SHA
 --no-prompt
 --trustAll
```

4.7 RESTful Client Access

OpenDJ offers two ways to give RESTful client applications HTTP access to directory data as JSON resources.

1. Enable the listener on OpenDJ directory server to respond to REST requests.

   With this approach, you do not need to install additional software.

2. Configure the external REST LDAP gateway Servlet to access your directory service.

   With this approach, you must install the gateway separately.

Procedure 4.7. To Set Up REST Access to OpenDJ Directory Server

OpenDJ directory server has a handler for HTTP connections, where it exposes the RESTful API demonstrated in the chapter on *Performing RESTful Operations*. The HTTP connection handler is not enabled by default.

You configure the mapping between JSON resources and LDAP entries by editing the configuration file for the HTTP connection handler, by default `/path/to/opendj/config/http-config.json`. The configuration is described in the appendix, *REST LDAP Configuration*. The default mapping works out of the box with Example.com data generated as part of the setup process and with Example.ldif.

1. Enable the connection handler.

```
$ dsconfig
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "HTTP Connection Handler"
 --set enabled:true
 --no-prompt
 --trustAll
```

2. Enable the HTTP access log.

   ```
   $ dsconfig
    set-log-publisher-prop
    --hostname opendj.example.com
    --port 4444
    --bindDN "cn=Directory Manager"
    --bindPassword password
    --publisher-name "File-Based HTTP Access Logger"
    --set enabled:true
    --no-prompt
    --trustAll
   ```

   This enables the HTTP access log, `opendj/logs/http-access`. For details on the format of the HTTP access log, see the section on *Server Logs*.

3. (Optional) Try reading a resource.

   The HTTP connection handler paths start by default at the root context, as shown in the following example.

   ```
   $ curl http://bjensen:hifalutin@opendj.example.com:8080/users/bjensen
    ?_prettyPrint=true
   {
     "_rev" : "00000000315fb731",
     "schemas" : [ "urn:scim:schemas:core:1.0" ],
     "manager" : [ {
       "_id" : "trigden",
       "displayName" : "Torrey Rigden"
     } ],
     "contactInformation" : {
       "telephoneNumber" : "+1 408 555 1862",
       "emailAddress" : "bjensen@example.com"
     },
     "_id" : "bjensen",
     "name" : {
       "familyName" : "Jensen",
       "givenName" : "Barbara"
     },
     "userName" : "bjensen@example.com",
     "displayName" : "Barbara Jensen"
   }
   ```

4. (Optional) If necessary, change the connection handler configuration using the **dsconfig** command.

   The following example shows how to set the port to 8443, and to configure the connection handler to do SSL (using the default server certificate). If you did not generate a default, self-signed certificate when installing OpenDJ directory server see the instructions, *To Create & Install a Self-Signed Certificate*, and more generally the section on *Preparing For Secure Communications* for additional instructions including how to import a CA-signed certificate.

   ```
   $ dsconfig
   ```

```
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "HTTP Connection Handler"
 --set listen-port:8443
 --set use-ssl:true
 --set key-manager-provider:JKS
 --set trust-manager-provider:"Blind Trust"
 --no-prompt
$ stop-ds --restart
Stopping Server...
.... The Directory Server has started successfully
$ keytool
 -export
 -rfc
 -alias server-cert
 -keystore /path/to/opendj/config/keystore
 -storepass `cat /path/to/opendj/config/keystore.pin`
 -file server-cert.pem
Certificate stored in file <server-cert.pem>
$ curl
 --cacert server-cert.pem
 --user bjensen:hifalutin
 https://opendj.example.com:8443/users/bjensen?_prettyPrint=true
{
  "_rev" : "0000000018c8b685",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "contactInformation" : {
    "telephoneNumber" : "+1 408 555 1862",
    "emailAddress" : "bjensen@example.com"
  },
  "_id" : "bjensen",
  "name" : {
    "familyName" : "Jensen",
    "givenName" : "Barbara"
  },
  "userName" : "bjensen@example.com",
  "displayName" : "Barbara Jensen",
  "manager" : [ {
    "_id" : "trigden",
    "displayName" : "Torrey Rigden"
  } ]
}
```

Procedure 4.8. To Set Up OpenDJ REST LDAP Gateway

Follow these steps to set up OpenDJ REST LDAP gateway Servlet to access your directory service.

1. Download and install the gateway as described in *To Install OpenDJ REST LDAP Gateway*.

2. Adjust the configuration for your directory service as described in *REST LDAP Configuration*.

4.8 DSML Client Access

Directory Services Markup Language (DSML) client access is implemented as a servlet that runs in a web application container.

You configure DSML client access by editing the `WEB-INF/web.xml` after you deploy the web application. In particular, you must at least set the `ldap.host` and `ldap.port` parameters if they differ from the default values, which are `localhost` and `389`.

The list of DSML configuration parameters, including those that are optional, consists of the following.

`ldap.host`

Required parameter indicating the host name of the underlying directory server. Default: `localhost`.

`ldap.port`

Required parameter indicating the LDAP port of the underlying directory server. Default: 389.

`ldap.userdn`

Optional parameter specifying the DN used by the DSML gateway to bind to the underlying directory server. Not used by default.

`ldap.userpassword`

Optional parameter specifying the password used by the DSML gateway to bind to the underlying directory server. Not used by default.

`ldap.authzidtypeisid`

This parameter can help you set up the DSML gateway to do HTTP Basic Access Authentication, given the appropriate mapping between the user ID, and the user's entry in the directory.

Required boolean parameter specifying whether the HTTP Authorization header field's Basic credentials in the request hold a plain ID, rather than a DN. If set to `true`, then the gateway performs an LDAP SASL bind using SASL plain, enabled by default in OpenDJ to look for an exact match between a `uid` value and the plain ID value from the header. In other words, if the plain ID is `bjensen`, and that corresponds in the directory server to Babs Jensen's entry with DN `uid=bjensen,ou=people,dc=example,dc=com`, then the bind happens as

Babs Jensen. Note also that you can configure OpenDJ identity mappers for scenarios that use a different attribute than `uid`, such as the `mail` attribute.

Default: `false`

`ldap.usessl`

Required parameter indicating whether `ldap.port` points to a port listening for LDAPS (LDAP/SSL) traffic. Default: `false`.

`ldap.usestarttls`

Required parameter indicating whether to use StartTLS to connect to the specified `ldap.port`. Default: `false`.

`ldap.trustall`

Required parameter indicating whether blindly to trust all certificates presented to the DSML gateway when using secure connections (LDAPS or StartTLS). Default: `false`.

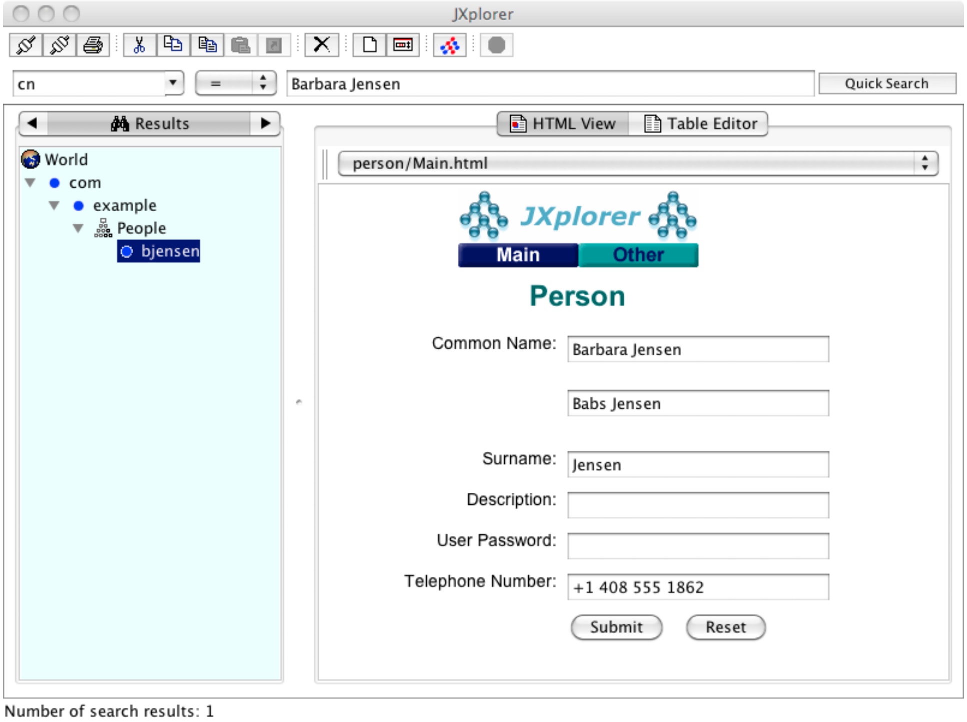
`ldap.truststore.path`

Optional parameter indicating the trust store used to verify certificates when using secure connections. If you want to connect using LDAPS or StartTLS, and do not want the gateway blindly to trust all certificates, then you must set up a trust store. Not used by default.

`ldap.truststore.password`

Optional parameter indicating the trust store password. If you set up and configure a trust store, then you need to set this as well. Not used by default.

The DSML servlet translates between DSML and LDAP, and passes requests to the directory server. For initial testing purposes, you might try JXplorer, where DSML Service: /*webapp-dir*/DSMLServlet. Here, *webapp-dir* refers to the name of the directory in which you unpacked the DSML .war file.



cn = Barbara Jensen

person/Main.html

Person

Common Name: Barbara Jensen

Babs Jensen

Surname: Jensen

Description:

User Password:

Telephone Number: +1 408 555 1862

Number of search results: 1

JXplorer accessing OpenDJ through DSML

4.9 JMX Client Access

You configure Java Management Extensions (JMX) client access by using the command-line tool, **dsconfig**.

Procedure 4.9. To Set Up JMX Access

1. Configure the server to activate JMX access.

```
$ dsconfig
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "JMX Connection Handler"
 --set enabled:true
 --trustAll
 --no-prompt
```

This example uses the default port number, 1689.

2. Restart the server so the change takes effect.

```
$ stop-ds --restart
```

Procedure 4.10. To Configure Access To JMX

After you set up OpenDJ directory server to listen for JMX connections, you must assign privileges in order to allow a user to connect over protocol.

1. Assign the privileges, `jmx-notify`, `jmx-read`, and `jmx-write` as necessary to the user who connects over JMX.

   See the section on *Configuring Privileges* for details.

2. Connect using the service URI, user name, and password.

   Service URI

   Full URI to the service including the hostname or IP address and port number for JMX where OpenDJ directory server listens for connections. For example, if the server IP is `192.168.0.10` and you configured OpenDJ to listen for JMX connections on port 1689, then the service URI is `service:jmx:rmi:///jndi/rmi://192.168.0.10:1689/org.opends.server.protocols.jmx.client-unknown`.

   User name

   The full DN of the user with privileges to connect over JMX such as `uid=kvaughan,ou=People,dc=example,dc=com`.

   Password

   The bind password for the user.

4.10 LDIF File Access

The LDIF connection handler lets you make changes to directory data by placing LDIF in a file system directory that OpenDJ server regularly polls for changes. The LDIF, once consumed, is deleted.

You configure LDIF file access by using the command-line tool **dsconfig**.

Procedure 4.11. To Set Up LDIF File Access

1. Activate LDIF file access.

```
$ dsconfig
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "LDIF Connection Handler"
 --set enabled:true
 --trustAll
 --no-prompt
```

The change takes effect immediately.

2. Add the directory where you put LDIF to be processed.

```
$ mkdir /path/to/opendj/config/auto-process-ldif
```

This example uses the default value of the `ldif-directory` property for the LDIF connection handler.

4.11 SNMP Access

For instructions on setting up the SNMP Connection Handler, see the section, *SNMP-Based Monitoring*.

Chapter 5

Configuring Privileges & Access Control

OpenDJ supports two mechanisms to protect access to the directory, *access control instructions* and *privileges*.

Access control instructions apply to directory data, providing fine-grained control over what a user or group member is authorized to do in terms of LDAP operations. Most access control instructions specify scopes (targets) to which they apply such that an administrative user who has all access to `dc=example,dc=com` need not have any access to `dc=example,dc=org`.

Privileges control the administrative tasks that users can perform, such as bypassing the access control mechanism, performing backup and restore operations, making changes to the configuration, and so forth. Privileges are implemented independently from access control. By default, privileges restrict administrative access to directory root users, though any user can be assigned a privilege. Privileges apply to a directory server, and do not have a scope.

Some operations require both privileges and also access control instructions. For example, in order to reset user's passwords, an administrator needs both the `password-reset` privilege and also access control to write `userPassword` values on the user entries. By combining an access control instruction with a privilege, you can effectively restrict the scope of that privilege to a particular branch of the Directory Information Tree.

This chapter covers both access control instructions and privileges, demonstrating how to configure both.

5.1 About Access Control Instructions

OpenDJ directory server access control instructions (ACIs) exist as operational `aci` attribute values on directory entries, and as global ACIs stored in the configuration. ACIs apply to a scope defined in the instruction, and set permissions that depend on what operation is requested, who requested the operation, and how the client connected to the server.

For example, the ACIs on the following entry allow anonymous read access to all attributes except passwords, and allow read-write access for directory administrators under `dc=example,dc=com`.

```
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
aci: (target ="ldap:///dc=example,dc=com")(targetattr !=
 "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";)
```

OpenDJ directory server's default behavior is that no access is allowed unless it is specifically granted by an access control instruction. In addition privileges assigned to certain users such as `cn=Directory Manager` allow them to bypass access control checks.

OpenDJ directory server provides several global ACIs out of the box to facilitate evaluation while maintaining a reasonable security policy. By default users are allow to read the root DSE, to read the schema, to use certain controls and extended operations, to modify their own entries, to bind, and so forth. Global ACIs are defined on the access control handler, and apply to the entire directory server. You must adjust the default global ACIs to match the security policies for your organization, for example to restrict anonymous access.

ACI attribute values use a specific language described in this section. Although ACI attribute values can become difficult to read in LDIF, the basic syntax is simple.

targets(version 3.0;acl "*name*";*permissions subjects*;)

The following list briefly explains the variables in the syntax above.

targets

> The *targets* specifies entries, attributes, controls, and extended operations to which the ACI applies.

To include multiple *targets*, enclose each individual target in parentheses, (). When you specify multiple targets, all targets must match for the ACI to apply (`AND`).

name

Supplies a human-readable description of what the ACI does.

permissions

Defines which actions to allow, and which to deny. Paired with *subjects*.

subjects

Identify clients to which the ACI applies depending on who connected, and when, where, and how they connected. Paired with *permissions*.

Separate multiple pairs of *permissions* *subjects* definitions with semicolons, `;`. When you specify multiple permissions-subjects pairs, at least one must match (`OR`).

5.1.1 ACI Targets

The seven types of ACI targets identify the objects to which the ACI applies.

`(target = "ldap:///DN")`
`(target != "ldap:///DN")`

Sets the scope to the entry with distinguished name *DN*, and to child entries.

You can use asterisks, `*`, to replace attribute types, attribute values, and entire DN components. In other words, the following specification targets both `uid=bjensen,ou=People,dc=example,dc=com` and also `cn=Frank Zappa, ou=Musicians,dc=example,dc=com`.

```
(target = "ldap:///*=*,*,dc=example,dc=com")
```

The *DN* must be in the subtree of the entry on which the ACI is defined.

If you do not specify `target`, then the entry holding this ACI will be affected. If `targetscope` is also omitted, then this entry and all subordinates will be affected.

`(targetattr = "attr-list")`
`(targetattr != "attr-list")`

Replace *attr-list* with a list of attribute type names, such as `userPassword`, separating multiple attribute type names with `||`.

This specification affects the entry where the ACI is located, or the entries specified by other targets in the ACI.

You can use an asterisk, *, to specify all non-operational attributes, although you will see better performance when explicitly including or excluding attribute types needed. You can use a plus, +, to specify all operational attributes.

If you do not include this target specification, then by default no attributes are affected by the ACI.

```
(targetfilter = "ldap-filter")
(targetfilter != "ldap-filter")
```

Sets the scope to match the *ldap-filter* dynamically, as in an LDAP search. The *ldap-filter* can be any valid LDAP filter.

```
(targattrfilters = "expression")
(targattrfilters != "expression")
```

Use this target specification when managing changes made to particular attributes.

Here *expression* takes one of the following forms. Separate expressions with semicolons, `;`.

```
op=attr1:filter1[&& attr2:filter2 …][;op=attr3:filter3[&& attr4:filter4 …] …]
```

Here *op* can be either `add` for operations creating attributes, or `delete` for operations removing them. Replace *attr* with an attribute type. Replace *filter* with an LDAP filter that corresponds to the *attr* attribute type.

```
(targetscope = "base|onelevel|subtree|subordinate")
```

Here `base` refers to the entry where the ACI is defined, `onelevel` to immediate children, `subtree` to the base entry and all children, and `subordinate` to all children only.

If you do not specify `targetscope`, then the default is `subtree`.

```
(targetcontrol = "OID")
(targetcontrol != "OID")
```

Replace *OID* with the object identifier for the LDAP control to target. Separate multiple OIDs with ||.

This target cannot be restricted to a specific subtree by combining it with another target.

```
(extop = "OID")
(extop != "OID")
```

Replace *OID* with the object identifier for the extended operation to target. Separate multiple OIDs with ||.

This target cannot be restricted to a specific subtree by combining it with another target.

5.1.2 ACI Permissions

ACI permission definitions take one of the following forms.

`allow(`*`action`*`[, `*`action`*` …])`

`deny(`*`action`*`[, `*`action`*` …])`

Tip

Although `deny` is supported, avoid restricting permissions by using `deny`. Instead, explicitly `allow` access only where needed. What looks harmless and simple in your lab examples can grow difficult to maintain in a real-world deployment with nested ACIs.

Replace *`action`* with one of the following.

`add`

Entry creation, as for an LDAP add operation

`all`

All permissions, except `export`, `import`, `proxy`

`compare`

Attribute value comparison, as for an LDAP compare operation

`delete`

Entry deletion, as for an LDAP delete operation

`export`

Entry export during a modify DN operation.

Despite the name, this action is unrelated to LDIF export operations.

`import`

Entry import during a modify DN operation.

Despite the name, this action is unrelated to LDIF import operations.

`proxy`

Access the ACI target using the rights of another user

`read`

Read entries and attributes

`search`

Search the ACI targets. Needs to be combine with `read` in order to read the search results.

`selfwrite`

Add or delete own DN from a group

`write`

Modify attributes on ACI target entries

5.1.3 ACI Subjects

ACI subjects match characteristics of the client connection to the server. Use subjects to restrict whether the ACI applies depending on who connected, and when, where, and how they connected.

```
authmethod = "none|simple|ssl|sasl mech"
authmethod != "none|simple|ssl|sasl mech"
```

Here you use `none` to mean do not check, `simple` for simple authentication, `ssl` for certificate-based authentication over LDAPS, `sasl` *`mech`* for SASL where *`mech`* is DIGEST-MD5, EXTERNAL, or GSSAPI.

```
dayofweek = "day[, day …]"
dayofweek != "day[, day …]"
```

Replace *`day`* with one of `sun`, `mon`, `tue`, `wed`, `thu`, `fri`, `sat`.

```
dns = "hostname"
dns != "hostname"
```

You can use asterisks, *, to replace name components, such as `dns = "*. myCompany.com"`.

```
groupdn = "ldap:///DN[|| ldap:///DN …]"
groupdn != "ldap:///DN[|| ldap:///DN …]"
```

Replace *`DN`* with the distinguished name of a group to permit or restrict access for members.

```
ip = "addresses"
ip != "addresses"
```

Here *addresses* can be specified for IPv4 or IPv6. IPv6 addresses are specified in brackets as `ldap://[address]/subnet-prefix` where */subnet-prefix* is optional. You can specify individual IPv4 addresses, addresses with asterisks (*) to replace subnets and host numbers, CIDR notation, and forms such as `192.168.0.*+255.255.255.0` to specify subnet masks.

```
ssf = "strength"
ssf != "strength"
ssf > "strength"
ssf >= "strength"
ssf < "strength"
ssf <= "strength"
```

Here the security strength factor pertains to the cipher key strength for connections using DIGEST-MD5, GSSAPI, SSL, or TLS. For example, to require that the connection must have at least 128 bits of encryption, specify `ssf >= 128`.

```
timeofday = "hhmm"
timeofday != "hhmm"
timeofday > "hhmm"
timeofday >= "hhmm"
timeofday < "hhmm"
timeofday <= "hhmm"
```

Here *hhmm* is expressed as on a 24-hour clock. For example, 1:15 PM is written `1315`.

```
userattr = "attr#value"
userattr != "attr#value"
userattr = ldap-url#LDAPURL"
userattr != ldap-url#LDAPURL"
userattr = "[parent[child-level]. ]attr#GROUPDN|USERDN"
userattr != "[parent[child-level]. ]attr#GROUPDN|USERDN"
```

The `userattr` subject specifies an attribute that must match on both the bind entry and the target of the ACI.

To match when the attribute on the bind DN entry corresponds directly to the attribute on the target entry, replace *attr* with the attribute type, and *value* with the attribute value.

To match when the target entry is identified by an LDAP URL, and the bind DN is in the subtree of the DN of the LDAP URL, use *ldap-url*#LDAPURL.

To match when the bind DN corresponds to a member of the group identified by the *attr* value on the target entry, use *attr*#GROUPDN.

To match when the bind DN corresponds to the *attr* value on the target entry, use *attr*#USERDN.

The optional inheritance specification, `parent[`*child-level*`].`, lets you specify how many levels below the target entry inherit the ACI. Here *child-level* is a number from 0 to 9, with 0 indicating the target entry only. Separate multiple *child-level* digits with commas (,).

```
userdn = "ldap-url++[|| ldap-url++ …]"
userdn != "ldap-url++[|| ldap-url++ …]"
```

To match the bind DN, replace *ldap-url++* with either a valid LDAP URL such as `ldap:///uid=bjensen,ou=People,dc=example,dc=com`, `ldap:///dc=example,dc=com??sub?(uid=bjensen)`, or a special LDAP URL-like keyword from the following list.

`ldap:///all`

Match authenticated users.

`ldap:///anyone`

Match anonymous and authenticated users.

`ldap:///parent`

Match when the bind DN is a parent of the ACI target.

`ldap:///self`

Match when the bind DN entry corresponds to ACI target.

5.1.4 How ACI is Evaluated

Understanding how OpenDJ evaluates the aci values is critical when implementing an access control policy. The rules the server follows are simple.

1. To determine if an operation is allowed or denied, the OpenDJ server looks in the directory for the target of the operation. It collects any aci values from that entry, and then walks up the directory tree to the suffix, collecting all aci values en route. Global aci values are then collected.

2. It then separates the aci values into two lists; one list contains all the aci values that matches the target and denies the required access, and the other list contains all the aci values that matches the target and allows the required access.

3. If the deny list contains any aci values after this procedure, access will be immediately denied.
4. If the deny list is empty, then the allow list is processed. If the allow list contains any aci values, access will be allowed.
5. If both lists are empty, access will be denied.

Note

Some operations require multiple permissions and involve multiple targets. Evaluation will therefore take place multiple times. For example a search operation requires the `search` permission for each attribute in the search filter. If all those are allowed, the `read` permission is used to decide what attributes and values can be returned.

5.1.5 ACI Required For LDAP Operations

The minimal access control information required for specific LDAP operations is described here.

Add

The ACI must allow the `add` permission to entries in the target. This implicitly allows the attributes and values to be set. Use `targetattrfilters` to explicitly deny access to any values if required.

For example, the ACI required to allow `uid=bjensen,ou=People,dc=example, dc=com` to add an entry is:

```
aci: (version 3.0;acl "Add entry"; allow (add)(userdn =
 "ldap:///uid=bjensen,ou=People,dc=example,dc=com");)
```

Bind

Because this is used to establish the user's identity and derived authorizations, ACI is irrelevant for this operation and is not checked. To prevent authentication, disable the account instead. For more information see *Managing Accounts Manually*.

Compare

The ACI must allow the `compare` permission to the attribute in the target entry.

For example, the ACI required to allow uid=bjensen,ou=People,dc=example, dc=com to compare values against the sn attribute is:

```
aci: (targetattr = "sn")(version 3.0;acl "Compare surname";
 allow (compare)(userdn =
 "ldap:///uid=bjensen,ou=People,dc=example,dc=com");)
```

Delete

The ACI must allow the delete permission to the target entry. This implicitly allows the attributes and values in the target to be deleted. Use targetattrfilters to explicitly deny access to the values if required.

For example, the ACI required to allow uid=bjensen,ou=People,dc=example, dc=com to delete an entry is:

```
aci: (version 3.0;acl "Delete entry"; allow (delete)
 (userdn = "ldap:///uid=bjensen,ou=People,dc=example,dc=com");)
```

Modify

The ACI must allow the write permission to attributes in the target entries. This implicitly allows all values in the target attribute to be modified. Use targetattrfilters to explicitly deny access to specific values if required.

For example, the ACI required to allow uid=bjensen,ou=People,dc=example, dc=com to modify the description attribute in an entry is:

```
aci: (targetattr = "description")(version 3.0;
 acl "Modify description"; allow (write)(userdn =
 "ldap:///uid=bjensen,ou=People,dc=example,dc=com");)
```

ModifyDN

If the entry is being moved to a newSuperior, the export permission must be allowed on the target, and the import permission must be allowed on the newSuperior entry.

The ACI must allow write permission to the attributes in the old RDN and the new RDN. All values of the old RDN and new RDN can be written implicitly; use targetattrfilters to explicitly deny access to values used if required.

For example, the ACI required to allow uid=bjensen,ou=People,dc=example, dc=com to rename entries named with the uid attribute to new locations:

```
aci: (targetattr = "uid")(version 3.0;acl "Rename uid= entries";
 allow (write, import, export)(userdn =
 "ldap:///uid=bjensen,ou=People,dc=example,dc=com");)
```

Search

ACI is required to process the search filter, and to determine what attributes and values may be returned in the results. The `search` permission is used to allow particular attributes in the search filter. The `read` permission is used to allow particular attributes to be returned. If `read` permission is allowed to any attribute, the server will automatically allow the `objectClass` attribute to also be read. All values of readable attributes can be implicitly read; to restrict this use `targetattrfilters`.

For example, the ACI required to allow `uid=bjensen,ou=People,dc=example,dc=com` to search for `uid` attributes, and also to read that attribute in matching entries is:

```
aci: (targetattr = "uid")(version 3.0;acl "Search and read uid";
 allow (search, read)(userdn =
 "ldap:///uid=bjensen,ou=People,dc=example,dc=com");)
```

5.2 About Privileges

Privileges provide access control for server administration independently from access control instructions.

Directory root users, such as `cn=Directory Manager`, are granted privileges in the following list and marked with an asterisk (*) by default. Other administrator users can be assigned privileges, too.

`backend-backup`*

Request a task to backup data

`backend-restore`*

Request a task to restore data from backup

`bypass-acl`*

Perform operations without regard to ACIs

`bypass-lockdown`*

Perform operations without regard to lockdown mode

`cancel-request`*

Cancel any client request

`config-read*`

Read the server configuration

`config-write*`

Change the server configuration

`data-sync`

Perform data synchronization

`disconnect-client*`

Close any client connection

`jmx-notify`

Subscribe to JMX notifications

`jmx-read`

Read JMX attribute values

`jmx-write`

Write JMX attribute values

`ldif-export*`

Export data to LDIF

`ldif-import*`

Import data from LDIF

`modify-acl*`

Change ACIs

`password-reset*`

Reset other users' passwords

`privilege-change*`

Change the privileges assigned to users

`proxied-auth`

Use the Proxied Authorization control

`server-lockdown`*

Put OpenDJ into, and take OpenDJ out of, lockdown mode

`server-restart`*

Request a task to restart the server

`server-shutdown`*

Request a task to stop the server

`subentry-write`*

Perform LDAP subentry write operations

`unindexed-search`*

Search using a filter with no correponding index

`update-schema`*

Change OpenDJ schema definitions

* = default directory root user privileges

5.3 Configuring Privileges

For root directory administrators, by default `cn=Directory Manager`, you configure privileges using the **dsconfig** command.

For non-root directory administrators, you add privileges with the **ldapmodify** command.

Procedure 5.1. To Change Root DN Privileges

1. Start **dsconfig** in interactive mode.

```
$ dsconfig
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
```

2. Select the Root DN menu.

3. Select View and edit the Root DN.

4. Edit the `default-root-privilege-name`.

5. Make sure you apply the changes when finished.

Procedure 5.2. To Add Privileges on an Individual Entry

Privileges are specified using the `ds-privilege-name` operational attribute, which you can change on the command-line using **ldapmodify**.

1. Determine the privileges to add.

   ```
   $ cat privilege.ldif
   dn: uid=kvaughan,ou=People,dc=example,dc=com
   changetype: modify
   add: ds-privilege-name
   ds-privilege-name: config-read
   ds-privilege-name: password-reset
   ```

   This example lets the user read the server configuration, and reset user passwords. In order for the user to be able to change a user password, you must also allow the modification using ACIs. For this example, Kirsten Vaughan is a member of the Directory Administrators group for Example.com, and already has access to modify user entries.

   Prior to having the privileges, Kirsten gets messages about insufficent access when trying to read the server configuration, or reset a user password.

   ```
   $ ldapsearch
    --port 1389
    --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
    --bindPassword bribery
    --baseDN cn=config
    "(objectclass=*)"
   SEARCH operation failed
   Result Code:  50 (Insufficient Access Rights)
   Additional Information:  You do not have sufficient privileges to perform
    search operations in the Directory Server configuration
   $ ldappasswordmodify
    --port 1389
    --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
    --bindPassword bribery
    --authzID "dn:uid=scarter,ou=People,dc=example,dc=com"
    --newPassword changeit
   The LDAP password modify operation failed with result code 50
   Error Message:  You do not have sufficient privileges to perform password
   reset operations
   ```

2. Apply the change as a user with the `privilege-change` privilege.

   ```
   $ ldapmodify
    --port 1389
    --bindDN "cn=Directory Manager"
    --bindPassword password
    --filename privilege.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
   ```

At this point, Kirsten can perform the operations requiring privileges.

```
$ ldapsearch
 --port 1389
 --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
 --bindPassword bribery
 --baseDN cn=config
 "(objectclass=*)"
dn: cn=config
ds-cfg-return-bind-error-messages: false
ds-cfg-default-password-policy: cn=Default Password Policy,cn=Password Policies,
 cn=config
…
$ ldappasswordmodify
 --port 1389
 --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
 --bindPassword bribery
 --authzID "dn:uid=scarter,ou=People,dc=example,dc=com"
 --newPassword changeit
The LDAP password modify operation was successful
```

Procedure 5.3. To Add Privileges For a Group of Administrators

For deployments with more than one administrator, you no doubt use a group to define adminstrative rights. You can use a collective attribute subentry to specify privileges for the administrator group.

Collective attributes provide a standard mechanism for defining attributes that appear on all the entries in a particular subtree. OpenDJ extends collective attributes to give you fine-grained control over the which entries in the subtree are targetted. Also, OpenDJ lets you use virtual attributes, such as `isMemberOf` to construct the filter for targetting entries to which the collective attributes apply. This allows you, for example, to define administrative privileges that apply to all users who belong to an administrator group.

1. Create an LDAP subentry that specifies the collective attributes.

```
$ cat collective.ldif
dn: cn=Administrator Privileges,dc=example,dc=com
objectClass: collectiveAttributeSubentry
objectClass: extensibleObject
objectClass: subentry
objectClass: top
cn: Administrator Privileges
ds-privilege-name;collective: config-read
ds-privilege-name;collective: config-write
ds-privilege-name;collective: ldif-export
ds-privilege-name;collective: modify-acl
ds-privilege-name;collective: password-reset
ds-privilege-name;collective: proxied-auth
subtreeSpecification: {base "ou=people", specificationFilter
  "(isMemberOf=cn=Directory Administrators,ou=Groups,dc=example,dc=com)" }

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename collective.ldif
Processing ADD request for cn=Administrator Privileges,dc=example,dc=com
ADD operation successful for DN cn=Administrator Privileges,dc=example,dc=com
```

The Directory Administrators group for Example.com includes members like Kirsten Vaughan.

2. Observe that the change takes effect immediately.

```
$ ldappasswordmodify
 --port 1389
 --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
 --bindPassword bribery
 --authzID "dn:uid=scarter,ou=People,dc=example,dc=com"
 --newPassword changeit
The LDAP password modify operation was successful
```

5.4 Configuring Access Control

Access control instructions are defined in the data, as values for `aci` attributes. They can be imported in LDIF. They can be modified over LDAP. Yet in order to make changes to ACIs users first need the `modify-acl` privilege described previously. By default, only the root DN user has the `modify-acl` privilege.

Global ACIs on `cn=Access Control Handler,cn=config` can be set using the **dsconfig** command. Global ACIs have attribute type `ds-cfg-global-aci`. Modify global ACIs from the Access Control Handler menu in **dsconfig**.

Default global ACIs set up the following access rules.

- Users can employ LDAP controls and perform extended operations.

- Anonymous read access is allowed for most user data attributes.
- Users can read password values on their own entries after binding. (Also by default, password values are hashed.)
- Anonymous read access is allowed for schema-related operational attributes.
- Anonymous read access is allowed for root DSE attributes describing what the server supports.
- Anonymous read access is allowed for operational attributes related to entry updates and entry identification.
- Access to replication data is denied.

Users with write access to add ACIs and with the `modify-acl` privilege can use the **ldapmodify** command to change ACIs located in user data.

This section therefore focuses on ACI examples, rather than demonstrating how to update the directory for each example. To update ACIs, either change them using the **ldapmodify** command, or using OpenDJ Control Panel.

If you use OpenDJ Control Panel, find the entry to modify in the Manage Entries window. Then try View > LDIF View to edit the entry. Control Panel checks your syntax and lets you know if you made an error before it saves any changes.

For hints on updating directory entries with **ldapmodify**, see the section on *Modifying Entry Attributes*, keeping in mind that the name of the ACI attribute is `aci` as shown in the examples that follow.

Example 5.1. ACI: Anonymous Reads & Searches

This works when the only attributes you do not want world-readable are password attributes.

```
aci: (target ="ldap:///dc=example,dc=com")(targetattr !=
 "authPassword || userPassword")(version 3.0;acl "Anonymous read-search access";
 allow (read, search, compare)(userdn = "ldap:///anyone");)
```

Example 5.2. ACI: Disable Anonymous Access

By default OpenDJ denies access unless an access control explicitly allows access.[1] However, OpenDJ also allows anonymous access by default to use some controls, to perform certain extended operations, to view root DSE operational

[1] This does not apply to the directory root user, such as `cn=Directory Manager`, who bypasses ACIs.

attributes, to view directory schema definitions, to view some other operational attibutes, and to perform compare and search operations.

These default capabilities are defined on the `global-aci` property of the access control handler, which you can read by using the **dsconfig get-access-control-handler-prop** command.

```
$ dsconfig
 get-access-control-handler-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --property global-aci
```

To disable anonymous read access for example, use the **dsconfig set-access-control-handler-prop** command.

First, remove the `global-aci` attribute value that allows anonymous read access. Do not wrap the lines in the following example if you use it as the basis for your script.

```
$ dsconfig \
 set-access-control-handler-prop \
 --remove global-aci:\(targetattr!=\"userPassword\|\|authPassword\|\|changes\|\
\|changeNumber\|\|changeType\|\|changeTime\|\|targetDN\|\|newRDN\|\
\|newSuperior\|\|deleteOldRDN\|\|targetEntryUUID\|\|changeInitiatorsName\|\
\|changeLogCookie\|\|includedAttributes\"\)\(version\ 3.0\;\ acl\ \"Anonymous\
\ read\ access\"\;\ allow\ \(read,search,compare\)\
\ userdn=\"ldap:///anyone\"\;\)\
 --hostname opendj.example.com \
 --port 4444 \
 --bindDN cn=Directory\ Manager \
 --bindPassword password \
 --trustAll \
 --no-prompt
```

If the `global-aci` does not match the ACI exactly then the command fails to remove the value. An alternative approach is to use the **dsconfig** command interactively, adding the `--commandFilePath` option. You can then use the command you capture to remove the property value on other servers for example. To use the **dsconfig** command this way, start the command as follows.

```
$ dsconfig
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --commandFilePath /tmp/captured-command.sh
```

Next, add a global ACI value that allows authenticated users to perform read operations. You can use **dsconfig** interactively to add the ACI value as shown below `(targetattr!= … userdn="ldap:///all";)`.

```
global-aci: (targetattr!="userPassword||
 authPassword||changes||changeNumber||changeType||changeTime||targetDN||newRDN||
 newSuperior||deleteOldRDN||targetEntryUUID||changeInitiatorsName||
 changeLogCookie||includedAttributes")(version 3.0; acl "Authenticated users
 read access"; allow (read,search,compare) userdn="ldap:///all";)
```

Notice that these changes are made to the OpenDJ directory server configuration, and so are not replicated to other servers. You must instead apply the changes separately to each server.

Example 5.3. ACI: Full Access for Administrators

Directory Administrators need privileges as well for full access to administrative operations.

```
aci: (target="ldap:///dc=example,dc=com") (targetattr =
 "* || +")(version 3.0;acl "Admins can run amok"; allow(
 all, proxy, import, export) groupdn =
 "ldap:///cn=Directory Administrators,ou=Groups,dc=example,dc=com";)
```

Notice both `targetattr = "* || +"`, which permits access to both all user attributes and all operational attributes, and `allow(all, proxy, import, export)`, which permits not only all user operations, but also proxy authorization as well as data import and export operations.

Example 5.4. ACI: Change Own Password

By default this capability is set in a global ACI.

```
aci: (target ="ldap:///ou=People,dc=example,dc=com")(targetattr =
 "authPassword || userPassword")(version 3.0;acl "Allow users to change pass
 words"; allow (write)(userdn = "ldap:///self");)
```

Example 5.5. ACI: Manage Own Group Membership

For some static groups such as carpoolers and social club members, you might choose to let users manage their own memberships.

```
aci: (target ="ldap:///ou=Self Service,ou=Groups,dc=example,dc=com")(
 targetattr = "member")(version 3.0;acl "Self registration"; allow(selfwrite)(
 userdn = "ldap:///uid=*,ou=People,dc=example,dc=com");)
```

Example 5.6. ACI: Manage Self Service Groups

Let users create and delete self-managed groups.

```
aci: (target ="ldap:///ou=Self Service,ou=Groups,dc=example,dc=com")(
 targattrfilters="add=objectClass:(objectClass=groupOfNames)")(version 3.0;
 acl "All can create self service groups"; allow (add)(userdn= "
 ldap:///uid=*,ou=People,dc=example,dc=com");)
aci: (target ="ldap:///ou=Self Service,ou=Groups,dc=example,dc=com")(version 3
 .0; acl "Owner can delete self service groups"; allow (delete)(userattr= "
 owner#USERDN");)
```

Example 5.7. ACI: Permit Clear Text Access Over Loopback Only

This ACI uses IP address and Security Strength Factor subjects.

```
aci: (target = "ldap:///dc=example,dc=com")(targetattr =
 "*")(version 3.0;acl "Use loopback only for LDAP in the clear"; deny (all)(
 ip != "127.0.0.1" and ssf <= "1");)
```

The `ssf` is one for example when using SSL but you have not configured a cipher, so the packets are checksummed for integrity checking by all content is nevertheless sent in clear text.

5.5 Viewing Effective Rights

Once you set up a number of ACIs, you might find it difficult to understand by inspection what rights a user actually has to a given entry. The Get Effective Rights control can help.

Note

The control OID, `1.3.6.1.4.1.42.2.27.9.5.2`, is not allowed by the default global ACIs.

In this example, Babs Jensen is the owner of a small group of people who are willing to carpool.

```
$ ldapsearch
 --port 1389
 --bindDN "uid=bjensen,ou=people,dc=example,dc=com"
 --bindPassword hifalutin
 --baseDN "ou=Self Service,ou=Groups,dc=example,dc=com"
 "cn=*"
dn: cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com
objectClass: groupOfNames
objectClass: top
member: uid=bjensen,ou=People,dc=example,dc=com
description: People who are willing to carpool
owner: uid=bjensen,ou=People,dc=example,dc=com
cn: Carpoolers
```

Performing the same search with the get effective rights control, and asking for the `aclRights` attribute, shows what rights Babs has on the entry.

```
$ ldapsearch
 --control effectiverights
 --port 1389
 --bindDN "uid=bjensen,ou=people,dc=example,dc=com"
 --bindPassword hifalutin
 --baseDN "ou=Self Service,ou=Groups,dc=example,dc=com"
 "cn=*"
 aclRights
dn: cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com
aclRights;entryLevel: add:0,delete:1,read:1,write:0,proxy:0
```

Requesting the `aclRightsInfo` attribute results in information about the ACIs applied to arrive at the results.

```
$ ldapsearch
 --control effectiverights
 --port 1389
 --bindDN "uid=bjensen,ou=people,dc=example,dc=com"
 --bindPassword hifalutin
 --baseDN "ou=Self Service,ou=Groups,dc=example,dc=com"
 "cn=*"
 aclRights
 aclRightsInfo
dn: cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com
aclRightsInfo;logs;entryLevel;read: acl_summary(main): access allowed(read) on e
 ntry/attr(cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com, objectClas
 s) to (uid=bjensen,ou=People,dc=example,dc=com) (not proxied) ( reason: evaluat
 ed allow , deciding_aci: Anonymous read-search access)
aclRightsInfo;logs;entryLevel;write: acl_summary(main): access not allowed(write
 ) on entry/attr(cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com, NULL
 ) to (uid=bjensen,ou=People,dc=example,dc=com) (not proxied) ( reason: no acis
 matched the subject )
aclRightsInfo;logs;entryLevel;add: acl_summary(main): access not allowed(add) on
  entry/attr(cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com, NULL) to
  (uid=bjensen,ou=People,dc=example,dc=com) (not proxied) ( reason: no acis matc
 hed the subject )
aclRightsInfo;logs;entryLevel;delete: acl_summary(main): access allowed(delete)
 on entry/attr(cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com, NULL)
 to (uid=bjensen,ou=People,dc=example,dc=com) (not proxied) ( reason: evaluated
 allow , deciding_aci: Owner can delete self service groups)
aclRights;entryLevel: add:0,delete:1,read:1,write:0,proxy:0
aclRightsInfo;logs;entryLevel;proxy: acl_summary(main): access not allowed(proxy
 ) on entry/attr(cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com, NULL
 ) to (uid=bjensen,ou=People,dc=example,dc=com) (not proxied) ( reason: no acis
 matched the subject )
```

You can also request the effective rights for another user by using the `--getEffectiveRightsAuthzid` (short form: `-g`) option, which takes the authorization identity of the other user as an argument. The following example shows Directory Manager checking anonymous user rights to the same entry. Notice that the authorization identity for an anonymous user is expressed as `"dn:"`.

```
$ ldapsearch
 --getEffectiveRightsAuthzid "dn:"
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --baseDN "ou=Self Service,ou=groups,dc=example,dc=com"
 "cn=*" aclRightsInfo
dn: cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com
aclRightsInfo;logs;entryLevel;read: acl_summary(main): access allowed(read) on e
 ntry/attr(cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com, objectClas
 s) to (anonymous) (not proxied) ( reason: evaluated allow , deciding_aci: Anony
 mous read-search access)
aclRightsInfo;logs;entryLevel;write: acl_summary(main): access not allowed(write
 ) on entry/attr(cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com, NULL
 ) to (anonymous) (not proxied) ( reason: no acis matched the subject )
aclRightsInfo;logs;entryLevel;add: acl_summary(main): access not allowed(add) on
  entry/attr(cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com, NULL) to
  (anonymous) (not proxied) ( reason: no acis matched the subject )
aclRightsInfo;logs;entryLevel;delete: acl_summary(main): access not allowed(dele
 te) on entry/attr(cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com, NU
 LL) to (anonymous) (not proxied) ( reason: no acis matched the subject )
aclRightsInfo;logs;entryLevel;proxy: acl_summary(main): access not allowed(proxy
 ) on entry/attr(cn=Carpoolers,ou=Self Service,ou=Groups,dc=example,dc=com, NULL
 ) to (anonymous) (not proxied) ( reason: no acis matched the subject )
```

When you need to check access to an attribute that might not yet exist on the entry, you can further use the `--getEffectiveRightsAttribute` (short form: `-e`) option, which takes an attribute list as an argument. The following example shows Directory Manager checking anonymous user access to the description attribute for the Self Service groups organizational unit entry. The description attribute is not present on the entry, yet.

```
$ ldapsearch
 --port 1389
 --baseDN "ou=Self Service,ou=groups,dc=example,dc=com"
 "ou=Self Service" description
dn: ou=Self Service,ou=Groups,dc=example,dc=com

$ ldapsearch
 --getEffectiveRightsAuthzid "dn:"
 --getEffectiveRightsAttribute description
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --baseDN "ou=Self Service,ou=groups,dc=example,dc=com"
 "ou=Self Service" aclRights
dn: ou=Self Service,ou=Groups,dc=example,dc=com
aclRights;attributeLevel;description: search:1,read:1,compare:1,write:0,selfwrit
 e_add:0,selfwrite_delete:0,proxy:0
aclRights;entryLevel: add:0,delete:0,read:1,write:0,proxy:0
```

Chapter 6

Performing LDAP Operations

OpenDJ comes with a Control Panel browser for managing entries and also command-line tools for performing LDAP operations. This chapter demonstrates how to use the command line tools to script LDAP operations.

6.1 Searching the Directory

Searching the directory resembles searching for a phone number in a paper phone book. You can look up a phone number because you know the last name of a subscriber's entry. In other words, you use the value of one attribute of the entry to find entries that have another attribute you want.

Yet whereas a paper phone book has only one index (alphabetical order by name), the directory has many indexes. For a search you therefore always specify which index to use, by specifying which attribute(s) you are using to lookup entries.

Your paper phone book might be divided into white pages for residential subscribers, and yellow pages for businesses. If you are looking up an individual's phone number, you limit your search to the white pages. Directory services divide entries in various ways, often to separate organizations, and to separate groups from user entries from printers for example, but potentially in other ways. When searching you therefore also specify where in the directory to search.

The **ldapsearch** command thus takes at minimum a search base DN option and an LDAP filter. The search base DN identifies where in the directory to search for entries that match the filter. For example, if you are looking for printers, you might specify the base DN as `ou=Printers,dc=example,dc=com`. Perhaps you are visiting the `GNB00` office and are looking for a printer.

```
$ ldapsearch --baseDN ou=Printers,dc=example,dc=com "(printerLocation=GNB00)"
```

In the example, the LDAP filter indicates to the directory that you want to lookup printer entries where the `printerLocation` attribute is equal to `GNB00`.

You also specify the host and port to access directory services, what protocol to use (for example, LDAP/SSL, or StartTLS to protect communication). If the directory service does not allow anonymous access to the data you want to search, you also identify who is performing the search and provide their credentials, such as a password or certificate. Finally, you can specify a list of attributes to return. If you do not specify attributes, then the search returns all user attributes for the entry.

Review the following examples in this section to get a sense of how searches work.

- Example 6.1, "Search: Simple Filter"
- Example 6.2, "Search: Complex Filter"
- Example 6.3, "Search: Return Operational Attributes"
- Example 6.4, "Search: Return Attributes for an Object Class"
- Example 6.5, "Search: Escaping Search Filter Characters"
- Example 6.6, "Search: List Active Accounts"
- Example 6.7, "Search: Language Subtype"

Example 6.1. Search: Simple Filter

The following example searches for entries with user IDs (`uid`) containing `jensen`, returning only DNs and user ID values.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com "(uid=*jensen*)" uid
dn: uid=ajensen,ou=People,dc=example,dc=com
uid: ajensen

dn: uid=bjensen,ou=People,dc=example,dc=com
uid: bjensen

dn: uid=gjensen,ou=People,dc=example,dc=com
uid: gjensen

dn: uid=jjensen,ou=People,dc=example,dc=com
uid: jjensen

dn: uid=kjensen,ou=People,dc=example,dc=com
uid: kjensen

dn: uid=rjensen,ou=People,dc=example,dc=com
uid: rjensen

dn: uid=tjensen,ou=People,dc=example,dc=com
uid: tjensen


Result Code:  0 (Success)
```

Example 6.2. Search: Complex Filter

The following example returns entries with `uid` containing `jensen` for users located in Santa Clara. The command returns the attributes associated with the `person` object class.

```
$ ldapsearch
 --port 1389
 --baseDN ou=people,dc=example,dc=com
 "(&(uid=*jensen*)(l=Santa Clara))"
 @person
dn: uid=ajensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: top
cn: Allison Jensen
telephoneNumber: +1 408 555 7892
sn: Jensen

dn: uid=gjensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: top
cn: Gern Jensen
telephoneNumber: +1 408 555 3299
sn: Jensen

dn: uid=kjensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
```

```
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: top
cn: Kurt Jensen
telephoneNumber: +1 408 555 6127
sn: Jensen

dn: uid=tjensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: top
cn: Ted Jensen
telephoneNumber: +1 408 555 8622
sn: Jensen
```

Complex filters can use both "and" syntax, `(&(`*`filtercomp`*`)(`*`filtercomp`*`))`, and "or" syntax, `(|(`*`filtercomp`*`)(`*`filtercomp`*`))`.

Example 6.3. Search: Return Operational Attributes

Use + in the attribute list after the filter to return all operational attributes. Alternatively, specify operational attributes by name.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com uid=bjensen +
dn: uid=bjensen,ou=People,dc=example,dc=com
numSubordinates: 0
structuralObjectClass: inetOrgPerson
pwdPolicySubentry: cn=Default Password Policy,cn=Password Policies,cn=config
subschemaSubentry: cn=schema
hasSubordinates: false
entryDN: uid=bjensen,ou=people,dc=example,dc=com
entryUUID: fc252fd9-b982-3ed6-b42a-c76d2546312c
```

Example 6.4. Search: Return Attributes for an Object Class

Use `@`*`objectClass`* in the attribute list after the filter to return the attributes associated with a particular object class.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com uid=bjensen @person
dn: uid=bjensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: top
cn: Barbara Jensen
cn: Babs Jensen
telephoneNumber: +1 408 555 1862
sn: Jensen
```

Example 6.5. Search: Escaping Search Filter Characters

RFC 4515: Lightweight Directory Access Protocol (LDAP): String Representation of Search Filters mentions a number of characters that you must handle with care when using them in search filters.

For a filter like `(attr=value)`, the following list indicates characters that you must replace with a backslash (\ ) followed by two hexadecimal digits when using them as part of the *value* string.

- Replace `*` with `\2a`.
- Replace `(` with `\28`.
- Replace `)` with `\29`.
- Replace `\` with `\5c`.
- Replace NUL (0x00) with `\00`.

The following example shows a filter with escaped characters matching an actual value.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com
 "(description=\28*\5c*\2a\29)" description
dn: uid=bjensen,ou=People,dc=example,dc=com
description: (A \great\ description*)
```

Example 6.6. Search: List Active Accounts

OpenDJ supports extensible matching rules, meaning you can pass in filters specifying a matching rule OID that extends your search beyond what you can do with standard LDAP. One specific matching rule of this type that OpenDJ supports is the generalized time based "later than" and "earlier than" matching rules. See the example, *Configure an Extensible Match Index*, showing how to build an index for these matching rules.

You can use these matching rules to list, for example, all users who have authenticated recently.

First set up an attribute to store a last login timestamp. You can do this by adding a schema file for the attribute.

```
$ ldapmodify
 --port 1389
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
dn: cn=schema
changetype: modify
add: attributeTypes
attributeTypes: ( lastLoginTime-oid
  NAME 'lastLoginTime'
  DESC 'Last time the user logged in'
  EQUALITY generalizedTimeMatch
  ORDERING generalizedTimeOrderingMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
  SINGLE-VALUE
  NO-USER-MODIFICATION
  USAGE directoryOperation
  X-ORIGIN 'OpenDJ example documentation' )

Processing MODIFY request for cn=schema
MODIFY operation successful for DN cn=schema
```

Configure the applicable password policy to write the last login timestamp when a user authenticates. The following command configures the default password policy to write the timestamp in generalized time format to the `lastLoginTime` operational attribute on the user's entry.

```
$ dsconfig
 set-password-policy-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --policy-name "Default Password Policy"
 --set last-login-time-attribute:lastLoginTime
 --set last-login-time-format:"yyyyMMddHH'Z'"
 --trustAll
 --no-prompt
```

Wait a while for users to authenticate again (or test it yourself) so that OpenDJ writes the timestamps. The following search then returns users who have authenticated in the last three months (13 weeks) after you configured OpenDJ to keep the last login timestamps.

```
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 "(lastLoginTime:1.3.6.1.4.1.26027.1.4.6:=13w)" mail
dn: uid=bjensen,ou=People,dc=example,dc=com
mail: bjensen@example.com

dn: uid=kvaughan,ou=People,dc=example,dc=com
mail: kvaughan@example.com
```

Example 6.7. Search: Language Subtype

OpenDJ directory server supports many language subtypes. See the chapter on *Localization* for a list.

When you perform a search you can request the language subtype by OID or by language subtype string. For example, the following search gets the French version of a common name. The example uses the **base64** command provided with OpenDJ directory server to decode the attribute value.

```
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 "(givenName:fr:=Frédérique)" cn\;lang-fr
dn: uid=fdupont,ou=People,dc=example,dc=com
cn;lang-fr:: RnJlZMOpcmlxdWUgRHVwb250

$ base64 decode -d RnJlZMOpcmlxdWUgRHVwb250
Fredérique Dupont
```

At the end of the OID or language subtype, you further specify the matching rule as follows:

- Add `.1` for less than
- Add `.2` for less than or equal to
- Add `.3` for equal to (default)
- Add `.4` for greater than or equal to
- Add `.5` for greater than
- Add `.6` for substring

The following table describes the operators you can use in LDAP search filters.

Table 6.1. LDAP Filter Operators

Operator	Definition	Example
`=`	Equality comparison, as in `(sn=Jensen)`. This can also be used with substring matches. For example, to match last names starting with `Jen`, use the filter `(sn=Jen*)`. Substrings are more expensive for the directory server to index. Substring searches therefore	`"(cn=My App)"` matches entries with common name `My App`. `"(sn=Jen*)"` matches entries with surname starting with `Jen`.

Operator	Definition	Example
	might not be permitted for many attributes.	
`<=`	Less than or equal to comparison, which works alphanumerically.	`"(cn<=App)"` matches entries with `commonName` up to those starting with App (case-insensitive) in alphabetical order.
`>=`	Greater than or equal to comparison, which works alphanumerically.	`"(uidNumber>=1151)"` matches entries with `uidNumber` greater than 1151.
`=*`	Presence comparison. For example, to match all entries having a `userPassword`, use the filter `(userPassword=*)`.	`"(member=*)"` matches entries with a `member` attribute.
`~=`	Approximate comparison, matching attribute values similar to the value you specify.	`"(sn~=jansen)"` matches entries with a surname that sounds similar to `Jansen` (Johnson, Jensen, and so forth).

`[:dn]`
`[:`*`oid`*`]:=`
	Extensible match comparison.
	At the end of the OID or language subtype, you further specify the matching rule as follows:
	• Add `.1` for less than
	• Add `.2` for less than or equal to
	• Add `.3` for equal to (default)
	• Add `.4` for greater than or equal to
	• Add `.5` for greater than
	• Add `.6` for substring
		`(uid:dn:=bjensen)` matches entries where `uid` having the value `bjensen` is a component of the entry DN.
		`(lastLoginTime: 1.3.6.1.4.1.26027.1.4.5:=-13w)` matches entries with a last login time more recent than 13 weeks.
		You also use extensible match filters with localized values. Directory servers like OpenDJ support a variety of internationalized locales, each of which has an OID for collation order, such as `1.3.6.1.4.1.42.2.27.9.4.76.1` for French. OpenDJ also lets you use the language subtype, such as `fr`, instead of the OID.
		`"(cn:dn:=My App)"` matches entries who have `My App` as the common

Operator	Definition	Example
		name and also as the value of a DN component.
`!`	NOT operator, to find entries that do not match the specified filter component. Take care to limit your search when using `!` to avoid matching so many entries that the server treats your search as unindexed.	`'!(objectclass=person)'` matches non-person entries.
`&`	AND operator, to find entries that match all specified filter components.	`'(&(l=Cupertino)(!(uid=bjensen)))'` matches entries for users in Cupertino other than the user with ID `bjensen`.
`|`	OR operator, to find entries that match one of the specified filter components.	`"|(sn=Jensen)(sn=Johnson)"` matches entries with surname Jensen or surname Johnson.

6.2 Comparing Attribute Values

The compare operation checks whether an attribute value you specify matches the attribute value stored on one or more directory entries.

Example 6.8. Compare: Checking `authPassword`

In this example, Kirsten Vaughan checks whether the hashed password value matches the stored value on `authPassword`.

```
$ ldapcompare
 --port 1389
 --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
 --bindPassword bribery
 'authPassword:MD5$dFHgpDxXUT8=$qlC4xMXvmVlusJLz9/WJ5Q=='
 uid=kvaughan,ou=people,dc=example,dc=com
Comparing type authPassword with value
 MD5$dFHgpDxXUT8=$qlC4xMXvmVlusJLz9/WJ5Q== in entry
 uid=kvaughan,ou=people,dc=example,dc=com
Compare operation returned true for entry
 uid=kvaughan,ou=people,dc=example,dc=com
```

6.3 Updating the Directory

Authorized users can change directory data using the LDAP add, modify, modify DN, and delete operations.

6.3.1 Adding Entries

With the **ldapmodify -a** command, authorized users can add entire entries from the same sort of LDIF file used to import and export data.

Example 6.9. Add: Two New Users

```
$ cat new-users.ldif
dn: cn=Arsene Lupin,ou=Special Users,dc=example,dc=com
objectClass: person
objectClass: top
cn: Arsene Lupin
telephoneNumber: +33 1 23 45 67 89
sn: Lupin

dn: cn=Horace Velmont,ou=Special Users,dc=example,dc=com
objectClass: person
objectClass: top
cn: Horace Velmont
telephoneNumber: +33 1 12 23 34 45
sn: Velmont

$ ldapmodify
 --defaultAdd
 --port 1389
 --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
 --bindPassword bribery
 --filename new-users.ldif
Processing ADD request for cn=Arsene Lupin,ou=Special Users,dc=example,dc=com
ADD operation successful for DN
 cn=Arsene Lupin,ou=Special Users,dc=example,dc=com
Processing ADD request for cn=Horace Velmont,ou=Special Users,dc=example,dc=com
ADD operation successful for DN
 cn=Horace Velmont,ou=Special Users,dc=example,dc=com
```

6.3.2 Modifying Entry Attributes

With the **ldapmodify** command, authorized users can change the values of attributes in the directory using LDIF as specified in RFC 2849.

Example 6.10. Modify: Adding Attributes

The following example adds a description and JPEG photo to Sam Carter's entry.

```
$ cat scarter-mods.ldif
dn: uid=scarter,ou=people,dc=example,dc=com
changetype: modify
add: description
description: Accounting Manager
-
add: jpegphoto
jpegphoto:<file:///tmp/Samantha-Carter.jpg

$ ldapmodify
 --port 1389
 --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
 --bindPassword bribery
 --filename scarter-mods.ldif
Processing MODIFY request for uid=scarter,ou=people,dc=example,dc=com
MODIFY operation successful for DN uid=scarter,ou=people,dc=example,dc=com
```

Example 6.11. Modify: Changing an Attribute Value

The following example replaces the description on Sam Carter's entry.

```
$ cat scarter-newdesc.ldif
dn: uid=scarter,ou=people,dc=example,dc=com
changetype: modify
replace: description
description: Accounting Director

$ ldapmodify
 --port 1389
 --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
 --bindPassword bribery
 --filename scarter-newdesc.ldif
Processing MODIFY request for uid=scarter,ou=people,dc=example,dc=com
MODIFY operation successful for DN uid=scarter,ou=people,dc=example,dc=com
```

Example 6.12. Modify: Deleting an Attribute Value

The following example deletes the JPEG photo on Sam Carter's entry.

```
$ cat /path/to/scarter-deljpeg.ldif
dn: uid=scarter,ou=people,dc=example,dc=com
changetype: modify
delete: jpegphoto

$ ldapmodify
 --port 1389
 --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
 --bindPassword bribery
 --filename scarter-deljpeg.ldif
Processing MODIFY request for uid=scarter,ou=people,dc=example,dc=com
MODIFY operation successful for DN uid=scarter,ou=people,dc=example,dc=com
```

Example 6.13. Modify: Optimistic Concurrency

Imagine you are writing an application that lets end users update user profiles through a browser. You store user profiles as OpenDJ entries. Your end users can look up user profiles and modify them. Your application assumes that the end users can tell the right information when they see it, and so aims to update profiles exactly as users see them on their screens.

Consider two users, Alice and Bob, both busy and often interrupted. Alice has Babs Jensen's new phone and room numbers. Bob has Babs's new location and description. Both assume that they have all the information that has changed. What can you do to make sure that your application applies the right changes when Alice and Bob simulaneously update Babs Jensen's profile?

OpenDJ offers a couple of features to help you in this situation. One of the features is the LDAP Assertion Control, used to tell OpenDJ to perform the modify only if an assertion you make stays true. The other feature is OpenDJ's support for entity tag (ETag) attributes, making it easy to check whether the entry in the directory is the same as the entry you read.

Alice and Bob both get Babs's entry. In LDIF the relevant attributes from the entry look like this. Notice the ETag.

```
dn: uid=bjensen,ou=People,dc=example,dc=com
telephoneNumber: +1 408 555 1862
roomNumber: 0209
l: Cupertino
ETag: 000000007a1999df
```

Bob prepares his changes in your application. Bob is almost ready to submit the new location and description when Carol stops by to ask Bob a few questions.

Alice starts just after Bob, but manages to submit her changes without getting interrupted. Now Babs's entry looks like this.

```
dn: uid=bjensen,ou=People,dc=example,dc=com
description: Updated by Alice
telephoneNumber: +47 2108 1746
roomNumber: 1389
l: Cupertino
ETag: 00000000aec2c1e9
```

In your application, you use the ETag attribute value with the assertion control to prevent Bob's update from going through when the ETag value has changed. Your application tries the equivalent of the following commands with Bob's updates.

```
$ cat /path/to/bobs.ldif
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: modify
replace: l
l: Grenoble
-
add: description
description: Employee of the Month

$ ldapmodify
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --port 1389
 --filename /path/to/bobs.ldif
 --assertionFilter "(ETag=000000007a1999df)"
Processing MODIFY request for uid=bjensen,ou=People,dc=example,dc=com
MODIFY operation failed
Result Code:  122 (Assertion Failed)
Additional Information:  Entry uid=bjensen,ou=People,dc=example,dc=com
 cannot be modified because the request contained an LDAP assertion control
 and the associated filter did not match the contents of the that entry
```

Your application therefore reloads Babs's entry, also getting the new ETag value, 00000000aec2c1e9, and lets Bob try again. This time Bob's changes do not collide with other changes. Babs's entry is successfully updated.

```
dn: uid=bjensen,ou=People,dc=example,dc=com
description: Employee of the Month
telephoneNumber: +47 2108 1746
roomNumber: 1389
l: Grenoble
ETag: 00000000e882c35e
```

6.3.3 Filtering Add & Modify Operations

Some client applications send updates including attributes with names that differ from the attribute names defined in OpenDJ. Other client applications might try to update attributes they should not update, such as the operational attributes creatorsName, createTimestamp, modifiersName, and modifyTimestamp. Ideally you would fix the client application behavior, but that is not always feasible.

You can configure the attribute cleanup plugin to filter add and modify requests, renaming attributes in requests using incorrect names, and removing attributes that applications should not change.

Example 6.14. Renaming Incoming Attributes

The following example renames incoming email attributes to mail attributes. First, configure the attribute cleanup plugin to rename the inbound attribute.

```
$ dsconfig
 create-plugin
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --type attribute-cleanup
 --plugin-name "Rename email to mail"
 --set enabled:true
 --set rename-inbound-attributes:email:mail
 --trustAll
 --no-prompt
```

Next, see that it works as expected.

```
$ cat email.ldif
dn: uid=newuser,ou=People,dc=example,dc=com
uid: newuser
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
cn: New User
sn: User
ou: People
email: newuser@example.com
userPassword: changeme

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename email.ldif
Processing ADD request for uid=newuser,ou=People,dc=example,dc=com
ADD operation successful for DN uid=newuser,ou=People,dc=example,dc=com
$ ldapsearch --port 1389 --baseDN dc=example,dc=com uid=newuser mail
dn: uid=newuser,ou=People,dc=example,dc=com
mail: newuser@example.com
```

Example 6.15. Removing Incoming Attributes

The following example prevents client applications from adding or modifying `creatorsName`, `createTimestamp`, `modifiersName`, and `modifyTimestamp` attributes. First, set up the attribute cleanup plugin.

```
$ dsconfig
 create-plugin
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --type attribute-cleanup
 --plugin-name "Remove attrs"
 --set enabled:true
 --set remove-inbound-attributes:creatorsName
 --set remove-inbound-attributes:createTimestamp
 --set remove-inbound-attributes:modifiersName
 --set remove-inbound-attributes:modifyTimestamp
 --trustAll
 --no-prompt
```

Next, see that it works as expected.

```
$ cat badattrs.ldif
dn: uid=badattr,ou=People,dc=example,dc=com
uid: newuser
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
cn: Bad Attr
sn: Attr
ou: People
mail: badattr@example.com
userPassword: changeme
creatorsName: cn=Bad Attr
createTimestamp: Never in a million years.
modifiersName: cn=Directory Manager,cn=Root DNs,cn=config
modifyTimestamp: 20110930164937Z

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename badattrs.ldif
Processing ADD request for uid=badattr,ou=People,dc=example,dc=com
ADD operation successful for DN uid=badattr,ou=People,dc=example,dc=com
$ ldapsearch --port 1389 --baseDN dc=example,dc=com uid=badattr +
dn: uid=badattr,ou=People,dc=example,dc=com
numSubordinates: 0
structuralObjectClass: inetOrgPerson
pwdPolicySubentry: cn=Default Password Policy,cn=Password Policies,cn=config
subschemaSubentry: cn=schema
hasSubordinates: false
entryDN: uid=badattr,ou=people,dc=example,dc=com
entryUUID: 35e5cb0e-e929-49d8-a50f-2df036d60db9
pwdChangedTime: 20110930165959.135Z
creatorsName: cn=Directory Manager,cn=Root DNs,cn=config
createTimestamp: 20110930165959Z
```

6.3.4 Renaming Entries

The Relative Distinguished Name (RDN) refers to the part of an entry's DN that distinguishes it from all other DNs at the same level in the directory tree. For example `uid=bjensen` is the RDN of the entry having DN `uid=bjensen,ou=People, dc=example,dc=com`.

With the **ldapmodify** command, authorized users can rename entries in the directory.

When you change the RDN of the entry, you are renaming the entry, modifying the value of the naming attribute, but also modifying the entry's DN.

Example 6.16. Rename: Modifying the DN

Sam Carter is changing her last name to Jensen, and changing her login from `scarter` to `sjensen`. The following example renames and changes Sam Carter's entry accordingly. Notice the boolean field, `deleteoldrdn: 1`, which indicates that the previous RDN, `uid: scarter`, should be removed. (Setting `deleteoldrdn: 0` instead would preserve `uid: scarter` on the entry.)

```
$ cat /path/to/scarter-sjensen.ldif
dn: uid=scarter,ou=people,dc=example,dc=com
changetype: modrdn
newrdn: uid=sjensen
deleteoldrdn: 1

dn: uid=sjensen,ou=people,dc=example,dc=com
changetype: modify
replace: cn
cn: Sam Jensen
-
replace: sn
sn: Jensen
-
replace: homeDirectory
homeDirectory: /home/sjensen
-
replace: mail
mail: sjensen@example.com

$ ldapmodify
 --port 1389
 --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
 --bindPassword bribery
 --filename /path/to/scarter-sjensen.ldif
Processing MODIFY DN request for uid=scarter,ou=people,dc=example,dc=com
MODIFY DN operation successful for DN uid=scarter,ou=people,dc=example,dc=com
Processing MODIFY request for uid=sjensen,ou=people,dc=example,dc=com
MODIFY operation successful for DN uid=sjensen,ou=people,dc=example,dc=com
```

6.3.5 Moving Entries

When you rename an entry with child entries, the directory has to move all the entries underneath.

Note

The modify DN operation only works when moving entries in the same backend, under the same suffix. Also, depending on the number of entries you move, this can be a resource-intensive operation.

With the **ldapmodify** command, authorized users can move entries in the directory.

Example 6.17. Move: Merging Customer and Employees Under `ou=People`

The following example moves `ou=Customers,dc=example,dc=com` to `ou=People, dc=example,dc=com`, and then moves each employee under `ou=Employees, dc=example,dc=com` under `ou=People,dc=example,dc=com` as well, finally removing the empty `ou=Employees,dc=example,dc=com` container. Here, `deleteoldrdn: 1` indicates that the old RDN, `ou: Customers`, should be removed from the entry. For employees, `deleteoldrdn: 0` indicates that old RDNs, in this case `uid` attribute values, should be preserved.

```
$ cat move-customers.ldif
dn: ou=Customers,dc=example,dc=com
changetype: modrdn
newrdn: ou=People
deleteoldrdn: 1
newsuperior: dc=example,dc=com

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename move-customers.ldif
Processing MODIFY DN request for ou=Customers,dc=example,dc=com
MODIFY DN operation successful for DN ou=Customers,dc=example,dc=com
$ cat move-employees.pl
#!/usr/bin/perl -w

# For each employee, construct a spec to move under ou=People.
while (<>)
{
    # Next line folded for readability only. Should not be split.
    $_ =~ s/dn: (.*?)(,.*)/dn: $1$2\nchangetype: moddn\nnewrdn: $1\n
     deleteoldrdn: 0\nnewsuperior: ou=People,dc=example,dc=com/;
    print;
}
$ ldapsearch --port 1389 --baseDN ou=Employees,dc=example,dc=com uid=* - |
 move-employees.pl > /tmp/move-employees.ldif
$ head -n 6 /tmp/move-employees.ldif
dn: uid=abarnes,ou=Employees,dc=example,dc=com
changetype: moddn
newrdn: uid=abarnes
deleteoldrdn: 0
newsuperior: ou=People,dc=example,dc=com

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename /tmp/move-employees.ldif
Processing MODIFY DN request for uid=abarnes,ou=Employees,dc=example,dc=com
MODIFY DN operation successful for DN uid=abarnes,ou=Employees,dc=example,dc=com
Processing MODIFY DN request for uid=abergin,ou=Employees,dc=example,dc=com
MODIFY DN operation successful for DN uid=abergin,ou=Employees,dc=example,dc=com
...
Processing MODIFY DN request for uid=wlutz,ou=Employees,dc=example,dc=com
MODIFY DN operation successful for DN uid=wlutz,ou=Employees,dc=example,dc=com
$ ldapdelete
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 ou=Employees,dc=example,dc=com
Processing DELETE request for ou=Employees,dc=example,dc=com
DELETE operation successful for DN ou=Employees,dc=example,dc=com
```

6.3.6 Deleting Entries

With the **ldapmodify** command, authorized users can delete entries from the directory.

Example 6.18. Delete: Removing a Subtree

The following example uses the subtree delete option to remove all Special Users from the directory.

```
$ ldapdelete
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --deleteSubtree "ou=Special Users,dc=example,dc=com"
Processing DELETE request for ou=Special Users,dc=example,dc=com
DELETE operation successful for DN ou=Special Users,dc=example,dc=com
```

6.4 Changing Passwords

With the **ldappasswordmodify** command, authorized users can change and reset user passwords.

Example 6.19. Password Reset

The following example shows Kirsten Vaughan resetting Sam Carter's password. Kirsten has the appropriate privilege to reset Sam's password.

```
$ ldappasswordmodify
 --useStartTLS
 --port 1389
 --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
 --bindPassword bribery
 --authzID "dn:uid=scarter,ou=people,dc=example,dc=com"
 --newPassword ChangeMe
The LDAP password modify operation was successful
```

Tip

Whenever one user changes another user's password, OpenDJ considers it a password reset. That often means the user has to change her password again after the reset.

If you want your application to change a user's password, rather than reset a user's password, have your application request the password change as the user whose password is changing. To change the password as the user, either bind as the user or use proxied authorization. For instructions on the latter, see the section on *Configuring Proxied Authorization*.

You could also accomplish password reset with the following command, but **set-password-is-reset** is a hidden option, supported only for testing.

```
$ manage-account
 set-password-is-reset
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --targetDN uid=scarter,ou=people,dc=example,dc=com
 --operationValue true
Password Is Reset:  true
```

Example 6.20. Change Own Password

You can use the **ldappasswordmodify** command to change your password, as long as you know your current password.

```
$ ldappasswordmodify
 --port 1389
 --authzID "dn:uid=bjensen,ou=people,dc=example,dc=com"
 --currentPassword hifalutin
 --newPassword secret12
The LDAP password modify operation was successful
```

The same operation works for `cn=Directory Manager`.

```
$ ldappasswordmodify
 --port 1389
 --authzID "dn:cn=Directory Manager"
 --currentPassword password
 --newPassword secret12
The LDAP password modify operation was successful
```

Example 6.21. Passwords With Special Characters

OpenDJ expects passwords to be UTF-8 encoded (base64 encoded when included in LDIF).

```
$ echo $LANG
en_US.utf8
$ ldappasswordmodify
 --port 1389
 --bindDN uid=bjensen,ou=People,dc=example,dc=com
 --bindPassword hifalutin
 --currentPassword hifalutin
 --newPassword pàsswôrd
The LDAP password modify operation was successful
$ ldapsearch
 --port 1389
 --bindDN uid=bjensen,ou=People,dc=example,dc=com
 --bindPassword pàsswôrd
 --baseDN dc=example,dc=com
 "(uid=bjensen)" cn
dn: uid=bjensen,ou=People,dc=example,dc=com
userPassword: {SSHA}k0eEeCxj9YRXUp8yJn0Z/mwqe+wrcFb1N1gg2g==
cn: Barbara Jensen
cn: Babs Jensen
```

6.5 Configuring Default Settings

You can use `~/.opendj/tools.properties` to set the defaults for bind DN, host name, and port number as in the following example.

```
hostname=directory.example.com
port=1389
bindDN=uid=kvaughan,ou=People,dc=example,dc=com

ldapcompare.port=1389
ldapdelete.port=1389
ldapmodify.port=1389
ldappasswordmodify.port=1389
ldapsearch.port=1389
```

The location on Windows is `%UserProfile%/.opendj/tools.properties`.

6.6 Authenticating To the Directory Server

Authentication is the act of confirming the identity of a principal. Authorization is the act of determining whether to grant or to deny access to a principal. Authentication is done to make authorization decisions.

As explained in *Configuring Privileges & Access Control*, OpenDJ directory server implements fine-grained access control for authorization. What is authorized depends on who is requesting the operation. Directory servers like OpenDJ must first therefore authenticate the principals using the clients before they can authorize or deny access. The LDAP bind operation, where a directory client authenticates with the directory server, is therefore the first LDAP operation in every LDAP session.

Clients bind by providing both a means to find their principal's entry in the directory and also providing some credentials that the directory server can check against their entry.

In the simplest bind operation, the client provides a zero-length name and a zero-length password. This results in an anonymous bind, meaning the client is authenticated as an anonymous user of the directory. In the simplest examples in Section 6.1, “Searching the Directory”, notice that no authentication information is provided. The examples work because the client commands default to requesting anonymous binds when you provide no credentials, and because access controls for the sample data allow anonymous clients to read, search, and compare some directory data.

In a simple bind operation, the client provides an LDAP name, such as the DN identifying its entry, and the corresponding password stored on the `userPassword` attribute of the entry. In Section 6.3, “Updating the Directory”, notice that to change directory data the client provides the bind DN and bind password of a user who has permission to change directory data. The commands do not work with a bind DN and bind password because access controls for the sample data only allow authorized users to change directory data.

Users rarely provide client applications with DNs, however. Instead users might provide a client application with an identity string like a user ID or an email address for example. Depending on how the DNs are constructed, the client application can either build the DN directly from the user's identity string, or use a session where the bind has been done with some other identity to search for the user entry based on the user's identity string. Given the DN constructed or found, the client application can then perform a simple bind.

For example, suppose Babs Jensen enters her email address, `bjensen@example.com`, and her password in order to log in. The client application might search for the entry matching `(mail=bjensen@example.com)` under base DN `dc=example,dc=com`. Alternatively, the client application might know to extract the user ID `bjensen` from the address, and then build the corresponding DN, `uid=bjensen,ou=people,dc=example,dc=com` in order to bind.

When an identifier string provided by the user can readily be mapped to the user's entry DN, OpenDJ directory server can do the translation between the identifier string and the entry DN. This translation is the job of a component called an identity mapper. Identity mappers are used to perform PLAIN SASL authentication (with a user name and password), SASL GSSAPI authentication (Kerberos V5), SASL CRAM MD5 and DIGEST MD5 authentication. They also handle authorization IDs during password modify extended operations and proxied authorization.

One use of PLAIN SASL is to translate user names from HTTP Basic authentication to LDAP authentication. The following example shows PLAIN SASL authentication using the default Exact Match identity mapper. In this (contrived) example, Babs Jensen reads the hashed value of her password. (According to the access controls in the example data, Babs must authenticate to read her password.) Notice the authentication ID is her user ID, `u:bjensen`, rather than the DN of her entry.

```
$ ldapsearch
 --port 1389
 --useStartTLS
 --baseDN dc=example,dc=com
 --saslOption mech=PLAIN
 --saslOption authid=u:bjensen
 --bindPassword hifalutin
 "(cn=Babs Jensen)" cn userPassword
dn: uid=bjensen,ou=People,dc=example,dc=com
cn: Barbara Jensen
cn: Babs Jensen
userPassword: {SSHA}7S4Si+vPE513cYQ7otiqb8hjiCzU7XNTv0RPBA==
```

The Exact Match identity mapper searches for a match between the string provided (here, `bjensen`) and the value of a specified attribute (by default the `uid` attribute). If you know users are entering their email addresses, you could create an exact match identity mapper for email addresses, and then use that for PLAIN SASL authentication as in the following example.

```
$ dsconfig
 create-identity-mapper
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --mapper-name "Email Mapper"
 --type exact-match
 --set match-attribute:mail
 --set enabled:true
 --no-prompt
$ dsconfig
 set-sasl-mechanism-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name PLAIN
 --set identity-mapper:"Email Mapper"
 --no-prompt
$ ldapsearch
 --port 1389
 --useStartTLS
 --baseDN dc=example,dc=com
 --saslOption mech=PLAIN
 --saslOption authid=u:bjensen@example.com
 --bindPassword hifalutin
 "(cn=Babs Jensen)" cn userPassword
dn: uid=bjensen,ou=People,dc=example,dc=com
cn: Barbara Jensen
cn: Babs Jensen
userPassword: {SSHA}7S4Si+vPE513cYQ7otiqb8hjiCzU7XNTv0RPBA==
```

The Regular Expression identity mapper uses a regular expression to extract a substring from the string provided, and then searches for a match between the substring and the value of a specified attribute. In the case of example data where an email address is *`user ID`* + @ + *`domain`*, you can use the default Regular Expression identity mapper in the same way as the email mapper from the previous example. The default regular expression pattern is `^([^@]+)@.+$`, and the part of the identity string matching `([^@]+)` is used to find the entry by user ID.

```
$ dsconfig
 set-sasl-mechanism-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name PLAIN
 --set identity-mapper:"Regular Expression"
 --no-prompt
$ ldapsearch
 --port 1389
 --useStartTLS
 --baseDN dc=example,dc=com
 --saslOption mech=PLAIN
 --saslOption authid=u:bjensen@example.com
 --bindPassword hifalutin
 "(cn=Babs Jensen)" cn userPassword
dn: uid=bjensen,ou=People,dc=example,dc=com
cn: Barbara Jensen
cn: Babs Jensen
userPassword: {SSHA}7S4Si+vPE513cYQ7otiqb8hjiCzU7XNTv0RPBA==
```

Try the **dsconfig** command interactively to experiment with `match-pattern` and `replace-pattern` settings for the Regular Expression identity mapper. The `match-pattern` can be any regular expression supported by `javax.util.regex.Pattern`.

6.7 Configuring Proxied Authorization

Proxied authorization provides a standard control as defined in RFC 4370 (and an earlier Internet-Draft) for binding with the user credentials of a proxy, who carries out LDAP operations on behalf of other users. You might use proxied authorization, for example, to have your application bind with its credentials, and then carry out operations as the users who login to the application.

Suppose you have an administrative directory client application that has an entry in the directory with DN `cn=My App,ou=Apps,dc=example,dc=com`. You can give that application the access rights and privileges to use proxied authorization. The default access control for OpenDJ permits authenticated users to use the proxied authorization control.

Suppose also that when directory administrator, Kirsten Vaughan, logs in to your application to change Babs Jensen's entry, your application looks up Kirsten's entry, and finds that she has DN `uid=kvaughan,ou=People,dc=example,dc=com`. For the example commands in the following procedure. My App uses proxied authorization to make a change to Babs's entry as Kirsten.

Procedure 6.1. To Set Up Proxied Authorization

1. Grant access to applications that can use proxied authorization.

```
$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (target="ldap:///dc=example,dc=com") (targetattr ="*
 ")(version 3.0; acl "Allow apps proxied auth"; allow(all, proxy
 )(userdn = "ldap:///cn=*,ou=Apps,dc=example,dc=com");)

Processing MODIFY request for dc=example,dc=com
MODIFY operation successful for DN dc=example,dc=com
```

2. Grant the privilege to use proxied authorization to My App.

```
$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
dn: cn=My App,ou=Apps,dc=example,dc=com
changetype: modify
add: ds-privilege-name
ds-privilege-name: proxied-auth

Processing MODIFY request for cn=My App,ou=Apps,dc=example,dc=com
MODIFY operation successful for DN cn=My App,ou=Apps,dc=example,dc=com
```

3. Test that My App can use proxied authorization.

```
$ ldapmodify
 --port 1389
 --bindDN "cn=My App,ou=Apps,dc=example,dc=com"
 --bindPassword password
 --proxyAs "dn:uid=kvaughan,ou=People,dc=example,dc=com"
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: modify
replace: description
description: Changed through proxied auth

Processing MODIFY request for uid=bjensen,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=bjensen,ou=People,dc=example,dc=com
```

If you need to map authorization identifiers using the `u:` form rather than using `dn:`, you can set the identity mapper with the global configuration setting, `proxied-authorization-identity-mapper`. For example, if you get user ID values from the client, such as `bjensen`, you can use the Exact Match Identity Mapper to match those to DNs based on an attribute of the entry. Use the **dsconfig** command interactively to investigate the settings you need.

6.8 Authenticating Using a Certificate

One alternative to simple binds with user name/password combinations consists in storing a digital certificate on the user entry, and then using the certificate as credentials during the bind. You can use this mechanism for example to let applications bind without using passwords.

Simply by setting up a secure connection with a certificate, the client is in effect authenticating to the server. The server must close the connection if it cannot trust the client certificate. However, the process of establishing a secure connection does not in itself identify the client to OpenDJ directory server.

Instead when binding with a certificate, the client must request the SASL External mechanism by which OpenDJ directory server maps the certificate to the client entry in the directory. When it finds a match, OpenDJ sets the authorization identity for the connection to that of the client, and the bind is successful.

For the whole process of authenticating with a certificate to work smoothly, OpenDJ and the client must trust each others' certificates, the client certificate must be stored on the client entry in the directory, and OpenDJ must be configured to map the certificate to the client entry.

Procedure 6.2. To Add Certificate Information to an Entry

Before trying to bind to OpenDJ directory server using a certificate, create a certificate, and then add the certificate attributes to the entry.

Example.ldif includes an entry for `cn=My App,ou=Apps,dc=example,dc=com`. Examples in this section use that entry, and use the Java **keytool** command to manage the certificate.

1. Create a certificate using the DN of the client entry as the distinguished name string.

   ```
   $ keytool
    -genkey
    -alias myapp-cert
    -keyalg rsa
    -dname "cn=My App,ou=Apps,dc=example,dc=com"
    -keystore keystore
    -storepass changeit
    -keypass changeit
   ```

2. Get the certificate signed.

   If you cannot get the certificate signed by a Certificate Authority, self-sign the certificate.

```
$ keytool
 -selfcert
 -alias myapp-cert
 -validity 7300
 -keystore keystore
 -storepass changeit
 -keypass changeit
```

3. Make note of the certificate fingerprints.

   Later in this procedure you update the client application entry with the MD5 fingerprint, which in this example is `48:AC:F9:13:11:E0:AB:C4:65:A2:83:9E:DB:FE:0C:37`.

```
$ keytool
 -list
 -v
 -alias myapp-cert
 -keystore keystore
 -storepass changeit
Alias name: myapp-cert
Creation date: Jan 18, 2013
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=My App, OU=Apps, DC=example, DC=com
Issuer: CN=My App, OU=Apps, DC=example, DC=com
Serial number: 5ae2277
Valid from: Fri Jan 18 18:27:09 CET 2013 until: Thu Jan 13 18:27:09 CET 2033
Certificate fingerprints:
  MD5:  48:AC:F9:13:11:E0:AB:C4:65:A2:83:9E:DB:FE:0C:37
  SHA1: F9:61:54:37:AA:C1:BC:92:45:07:64:4B:23:6C:BC:C9:CD:1D:44:0F
  SHA256: 2D:B1:58:CD:33:40:E9:...:FD:61:EA:C9:FF:6A:19:93:FE:E4:84:E3
  Signature algorithm name: SHA256withRSA
  Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 54 C0 C5 9C 73 37 85 4B   F2 3B D3 37 FD 45 0A AB  T...s7.K.;.7.E..
0010: C9 6B 32 95                                        .k2.
]
]
```

4. Export the certificate to a file in binary format.

```
$ keytool
 -export
 -alias myapp-cert
 -keystore keystore
 -storepass changeit
 -keypass changeit
 -file myapp-cert.crt
Certificate stored in file </path/to/myapp-cert.crt>
```

5. Modify the entry to add attributes related to the certificate.

   By default, you need the `userCertificate` value.

   If you want OpenDJ to map the certificate to its fingerprint, use `ds-certificate-fingerprint`. This example uses the MD5 fingerprint, which corresponds to the default setting for the Fingerprint Certificate Mapper.

   If you want to map the certificate subject DN to an attribute of the entry, use `ds-certificate-subject-dn`.

   ```
   $ cat addcert.ldif
   dn: cn=My App,ou=Apps,dc=example,dc=com
   changetype: modify
   add: objectclass
   objectclass: ds-certificate-user
   -
   add: ds-certificate-fingerprint
   ds-certificate-fingerprint: 48:AC:F9:13:11:E0:AB:C4:65:A2:83:9E:DB:FE:0C:37
   -
   add: ds-certificate-subject-dn
   ds-certificate-subject-dn: CN=My App, OU=Apps, DC=example, DC=com
   -
   add: userCertificate;binary
   userCertificate;binary:<file:///path/to/myapp-cert.crt

   $ ldapmodify
    --port 1389
    --hostname opendj.example.com
    --bindDN "cn=Directory Manager"
    --bindPassword password
    --filename addcert.ldif
   Processing MODIFY request for cn=My App,ou=Apps,dc=example,dc=com
   MODIFY operation successful for DN cn=My App,ou=Apps,dc=example,dc=com
   ```

6. Check your work.

```
$ ldapsearch
 --port 1389
 --hostname opendj.example.com
 --baseDN dc=example,dc=com
 "(cn=My App)"
dn: cn=My App,ou=Apps,dc=example,dc=com
ds-certificate-fingerprint: 4B:F5:CF:2C:2D:B3:86:14:FF:43:A8:37:17:DD:E7:55
userCertificate;binary:: MIIDOzCCAiOgAwIBAgIESfC6IjANBgkqhkiG9w0BAQsFADBOMRMwEQY
 KCZImiZPyLGQBGRYDY29tMRcwFQYKCZImiZPyLGQBGRYHZXhhbXBsZTENMAsGA1UECxMEQXBwczEPMA
 0GA1UEAxMGTXkgQXBwMB4XDTEzMDExNzE3MTEwM1oXDTEzMDQxNzE3MTEwM1owTjETMBEGCgmSJomT8
 ixkARkWA2NvbTEXMBUGCgmSJomT8ixkARkWB2V4YW1wbGUxDTALBgNVBAsTBEFwcHMxDzANBgNVBAMT
 Bk15IEFwcDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAJQYq+jG4ZQdNkyBT4OQBZ0sFkl
 X5o2yBViDMGl1sSWIRGLpFwu6iq1chndPBJYTC+FkT66yEEOwWOpSfcYdFHkMQP0qp5A8mgP6bYkeH1
 ROvQ1nhLs0ILuksR10CVIQ5b1zv6bGEFhA9gSKmpHfQOSt9PXq8+kuz+4RgZk9Il28tgDNMm91wSJr7
 kqi5g7a2a7Io5s9L2FeLhVSBYwinWQnASk8nENrhcE0hHkrpGsaxdhIQBQQvm+SRC0dI4E9iwBGI3Lw
 lV3a4KTa5DlYD6cDREI6B8XlSdc1DaIhwC8CbsE0WJQoCERSURdjkuHrPck6f69HKUFRiC7JMT3dFbs
 CAwEAAaMhMB8wHQYDVR0OBBYEFFTAxZxzN4VL8jvTN/1FCqvJazKVMA0GCSqGSIb3DQEBCwUAA4IBAQ
 BXsAIEw7I5XUzLFHvXb2N0hmW/Vmhb/Vlv9LTT8JcCRJy4zaiyS9Q+Sp9zQUkrXauFnNAhJLwpAymjZ
 MCOq1Th1bw9LnIzbccPQ/1+ZHLKDU5pgnc5BcvaV6Zl6COLLH2OOt0XMZ/OrODBV1M6STfhChqcowff
 xp72pWMQe+kpZfzjeDBk4kK2hUNTZsimB9qRyrDAMCIXdmdmFv1o07orxjy8c/6S1329swiiVqFckBR
 aXIa8wCcXjpQbZacDODeKk6wZIKxw4miLg1YByCMa7vkUfz+Jj+JHgbHjyoT/G82mtDbX02chLgXbDm
 xJPFN3mwAC7NEkSPbqd35nJlf3
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: ds-certificate-user
objectClass: top
ds-certificate-subject-dn: CN=My App, OU=Apps, DC=example, DC=com
cn: My App
sn: App
```

7. When using a self-signed certificate, import the client certificate into the trust store for OpenDJ.

   When the client presents its certificate to OpenDJ, by default OpenDJ has to be able to trust the client certificate before it can accept the connection. If OpenDJ cannot trust the client certificate, it cannot establish a secure connection.

```
$ keytool
 -import
 -alias myapp-cert
 -file /path/to/myapp-cert.crt
 -keystore /path/to/opendj/config/truststore
 -storepass `cat /path/to/opendj/config/keystore.pin`
Owner: CN=My App, OU=Apps, DC=example, DC=com
Issuer: CN=My App, OU=Apps, DC=example, DC=com
Serial number: 5ae2277
Valid from: Fri Jan 18 18:27:09 CET 2013 until: Thu Jan 13 18:27:09 CET 2033
Certificate fingerprints:
  MD5:  48:AC:F9:13:11:E0:AB:C4:65:A2:83:9E:DB:FE:0C:37
  SHA1: F9:61:54:37:AA:C1:BC:92:45:07:64:4B:23:6C:BC:C9:CD:1D:44:0F
  SHA256: 2D:B1:58:CD:33:40:E9:...:FD:61:EA:C9:FF:6A:19:93:FE:E4:84:E3
  Signature algorithm name: SHA256withRSA
  Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 54 C0 C5 9C 73 37 85 4B   F2 3B D3 37 FD 45 0A AB  T...s7.K.;.7.E..
0010: C9 6B 32 95                                        .k2.
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

8. When using a certificate signed by a CA whose certificate is not delivered with the Java runtime environment[1], import the CA certificate either into the Java runtime environment trust store, or into the OpenDJ trust store as shown in the following example.

```
$ keytool
 -import
 -alias ca-cert
 -file ca.crt
 -keystore /path/to/opendj/config/truststore
 -storepass `cat /path/to/opendj/config/keystore.pin`
Owner: EMAILADDRESS=admin@example.com, CN=Example CA, O=Example Corp, C=FR
Issuer: EMAILADDRESS=admin@example.com, CN=Example CA, O=Example Corp, C=FR
Serial number: d4586ea05c878b0c
Valid from: Tue Jan 29 09:30:31 CET 2013 until: Mon Jan 24 09:30:31 CET 2033
Certificate fingerprints:
  MD5:  8A:83:61:9B:E7:18:A2:21:CE:92:94:96:59:68:60:FA
  SHA1: 01:99:18:38:3A:57:D7:92:7B:D6:03:8C:7B:E4:1D:37:45:0E:29:DA
  SHA256: 5D:20:F1:86:CC:CD:64:50:1E:54:...:DF:15:43:07:69:44:00:FB:36:CF
  Signature algorithm name: SHA1withRSA
```

[1] `$JAVA_HOME/jre/lib/security/cacerts` holds the certificates for many CAs. To get the full list, use the following command.

```
$ keytool
 -list
 -v
 -keystore $JAVA_HOME/jre/lib/security/cacerts
 -storepass changeit
```

```
  Version: 3

Extensions:

#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: 30 07 67 7D 1F 09 B6 E6   90 85 95 58 94 37 FD 31  0.g........X.7.1
0010: 03 D4 56 7B                                        ..V.
]
[EMAILADDRESS=admin@example.com, CN=Example CA, O=Example Corp, C=FR]
SerialNumber: [    d4586ea0 5c878b0c]
]

#2: ObjectId: 2.5.29.19 Criticality=false
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 30 07 67 7D 1F 09 B6 E6   90 85 95 58 94 37 FD 31  0.g........X.7.1
0010: 03 D4 56 7B                                        ..V.
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

9. If you updated the OpenDJ trust store to add a certificate, restart OpenDJ to make sure it reads the updated trust store and can recognize the certificate.

```
$ stop-ds --restart
Stopping Server...
...
... The Directory Server has started successfully
```

Procedure 6.3. To Configure Certificate Mappers

OpenDJ uses certificate mappers during binds to establish a mapping between a client certificate and the entry that corresponds to that certificate. The certificate mappers provided out of the box include the following.

Fingerprint Certificate Mapper

> Looks for the MD5 (default) or SHA1 certificate fingerprint in an attribute of the entry (default: `ds-certificate-fingerprint`).

Subject Attribute To User Attribute Mapper

> Looks for a match between an attribute of the certificate subject and an attribute of the entry (default: match `cn` in the certificate to `cn` on the entry, or match `emailAddress` in the certificate to `mail` on the entry).

Subject DN to User Attribute Certificate Mapper

Looks for the certificate subject DN in an attribute of the entry (default: `ds-certificate-subject-dn`).

Subject Equals DN Certificate Mapper

Looks for an entry whose DN matches the certificate subject DN.

If the default configurations for the certificate mappers are acceptable, you do not need to change them. They are enabled by default.

The following steps demonstrate how to change the Fingerprint Mapper default algorithm of MD5 to SHA1.

1. List the certificate mappers to retrieve the correct name.

```
$ dsconfig
 list-certificate-mappers
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password

Certificate Mapper                   : Type                                 : enabled
-------------------------------------:--------------------------------------:--------
Fingerprint Mapper                   : fingerprint                          : true
Subject Attribute to User Attribute : subject-attribute-to-user-attribute : true
Subject DN to User Attribute         : subject-dn-to-user-attribute         : true
Subject Equals DN                    : subject-equals-dn                    : true
```

2. Examine the current configuration.

```
$ dsconfig
 get-certificate-mapper-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --mapper-name "Fingerprint Mapper"

Property              : Value(s)
----------------------:---------------------------
enabled               : true
fingerprint-algorithm : md5
fingerprint-attribute : ds-certificate-fingerprint
user-base-dn          : -
```

3. Change the configuration as necessary.

```
$ dsconfig
 set-certificate-mapper-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --mapper-name "Fingerprint Mapper"
 --set fingerprint-algorithm:sha1
 --no-prompt
```

4. Set the External SASL Mechanism Handler to use the appropriate certificate mapper (default: Subject Equals DN).

   Clients applications use the SASL External mechanism during the bind to have OpenDJ set the authorization identifier based on the entry that matches the client certificate.

   ```
   $ dsconfig
    set-sasl-mechanism-handler-prop
    --port 4444
    --hostname opendj.example.com
    --bindDN "cn=Directory Manager"
    --bindPassword password
    --handler-name External
    --set certificate-mapper:"Fingerprint Mapper"
    --no-prompt
   ```

Example 6.22. Authenticate With Client Certificate

Instead of providing a bind DN and password as for simple authentication, use the SASL EXTERNAL authentication mechanism, and provide the certificate. As a test with example data you can try an anonymous search, and then try with certificate-based authentication.

Before you try this example, make sure OpenDJ is set up to accept StartTLS from clients, and that you have set up the client certificate as described above. Next, create a password .pin file for your client key store.

```
$ echo changeit > keystore.pin
$ chmod 400 keystore.pin
```

Also, if OpenDJ directory server uses a certificate for StartTLS that was not signed by a well-known CA, import the appropriate certificate into the client key store, which can then double as a trust store. For example, if OpenDJ uses a self-signed certificate, import the server certificate into the key store.

```
$ keytool
 -export
 -alias server-cert
 -file server-cert.crt
 -keystore /path/to/opendj/config/keystore
 -storepass `cat /path/to/opendj/config/keystore.pin`
$ keytool
 -import
 -trustcacerts
 -alias server-cert
 -file server-cert.crt
 -keystore keystore
 -storepass `cat keystore.pin`
```

If OpenDJ directory server uses a CA-signed certificate, but the CA is not well known, import the CA certificate into your keystore.

```
$ keytool
 -import
 -trustcacerts
 -alias ca-cert
 -file ca-cert.crt
 -keystore keystore
 -storepass `cat keystore.pin`
```

Now that you can try the example, notice that OpenDJ does not return the `userPassword` value for an anonymous search.

```
$ ldapsearch
 --port 1389
 --hostname opendj.example.com
 --baseDN dc=example,dc=com
 --useStartTLS
 --trustStorePath keystore
 --trustStorePasswordFile keystore.pin
 "(cn=My App)" userPassword
dn: cn=My App,ou=Apps,dc=example,dc=com
```

OpenDJ does let users read the values of their own `userPassword` attributes after they bind successfully.

```
$ ldapsearch
 --port 1389
 --hostname opendj.example.com
 --baseDN dc=example,dc=com
 --useStartTLS
 --useSASLExternal
 --certNickName myapp-cert
 --keyStorePath keystore
 --keyStorePasswordFile keystore.pin
 --trustStorePath keystore
 --trustStorePasswordFile keystore.pin
 "(cn=My App)" userPassword
dn: cn=My App,ou=Apps,dc=example,dc=com
userPassword: {SSHA}vy/vTthOQoV/wH3MciTOBKKR4OX+0dSN/a09Ew==
```

You can also try the same test with other certificate mappers.

```
# Fingerprint mapper
$ dsconfig
 set-sasl-mechanism-handler-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name External
 --set certificate-mapper:"Fingerprint Mapper"
 --no-prompt
$ ldapsearch
 --port 1389
 --hostname opendj.example.com
 --baseDN dc=example,dc=com
 --useStartTLS
 --useSASLExternal
 --certNickName myapp-cert
 --keyStorePath keystore
 --keyStorePasswordFile keystore.pin
 --trustStorePath keystore
 --trustStorePasswordFile keystore.pin
 "(cn=My App)" userPassword
dn: cn=My App,ou=Apps,dc=example,dc=com
userPassword: {SSHA}vy/vTthOQoV/wH3MciTOBKKR4OX+0dSN/a09Ew==

# Subject Attribute to User Attribute mapper
$ dsconfig
 set-sasl-mechanism-handler-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name External
 --set certificate-mapper:"Subject Attribute to User Attribute"
 --no-prompt
$ ldapsearch
 --port 1389
 --hostname opendj.example.com
 --baseDN dc=example,dc=com
 --useStartTLS
 --useSASLExternal
 --certNickName myapp-cert
 --keyStorePath keystore
 --keyStorePasswordFile keystore.pin
 --trustStorePath keystore
 --trustStorePasswordFile keystore.pin
 "(cn=My App)" userPassword
dn: cn=My App,ou=Apps,dc=example,dc=com
userPassword: {SSHA}vy/vTthOQoV/wH3MciTOBKKR4OX+0dSN/a09Ew==

# Subject DN to User Attribute mapper
$ dsconfig
 set-sasl-mechanism-handler-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name External
 --set certificate-mapper:"Subject DN to User Attribute"
 --no-prompt
$ ldapsearch
```

```
 --port 1389
 --hostname opendj.example.com
 --baseDN dc=example,dc=com
 --useStartTLS
 --useSASLExternal
 --certNickName myapp-cert
 --keyStorePath keystore
 --keyStorePasswordFile keystore.pin
 --trustStorePath keystore
 --trustStorePasswordFile keystore.pin
 "(cn=My App)" userPassword
dn: cn=My App,ou=Apps,dc=example,dc=com
userPassword: {SSHA}vy/vTthOQoV/wH3MciTOBKKR4OX+0dSN/a09Ew==
```

Chapter 7

Performing RESTful Operations

OpenDJ lets you access directory data as JSON resources over HTTP. This chapter demonstrates basic RESTful client operations using the default configuration and sample directory data imported into OpenDJ from Example.ldif. Before trying the examples, enable HTTP access to OpenDJ directory server as described in procedure, *To Set Up REST Access to OpenDJ Directory Server*.

Interface stability: Evolving

7.1 Understanding the OpenDJ REST API

The OpenDJ REST API is built on a common ForgeRock HTTP-based REST API for interacting with JSON Resources. APIs built on this common layer all let you perform the following operations.

Create

Add a resource that does not yet exist

Read

Retrieve a single resource

Update

Replace an existing resource

Delete

Remove an existing resource

Patch

Modify part of an existing resource

Action

Perform a predefined action

Query

List a set of resources

modifyPassword

Change Your Password

resetPassword

Reset a Password

The present implementation in OpenDJ maps JSON resources onto LDAP entries, meaning REST clients can in principle do just about anything an LDAP client can do with directory data.

In addition to query string parameters that depend on the operation, the examples in this chapter make use of the following parameters that apply to the JSON resource returned for all operations.

`_fields=`*`field`*`[,…]`

Retain only the specified fields in the JSON resource returned.

`_prettyPrint=true|false`

Make the JSON resource returned easy for humans to read.

7.2 Authenticating Over REST

When you first try to get a resource that you can read as an LDAP entry with an anonymous search, you might be surprised that you must authenticate.

```
$ curl http://opendj.example.com:8080/users/bjensen?_prettyPrint=true
{
  "code" : 401,
  "reason" : "Unauthorized",
  "message" : "Unauthorized"
}
```

HTTP status code 401 tells your HTTP client that the request requires user authentication. You can change this behavior by setting the HTTP connection handler property, `authentication-required`, to `false`.

```
$ dsconfig
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "HTTP Connection Handler"
 --set authentication-required:false
 --no-prompt
 --trustAll
```

Out of the box both the HTTP Connection Handler and also the REST LDAP gateway are configured to allow HTTP Basic authentication and HTTP header based authentication in the style of OpenIDM. The authentication mechanisms translate HTTP authentication to LDAP authentication on the directory server side.

When you install OpenDJ either with generated sample user entries or with data from Example.ldif, the relative distinguished name attribute for the sample user entries is the user ID (`uid`) attribute. For example, the DN and user ID for Babs Jensen are as follows.

```
dn: uid=bjensen,ou=People,dc=example,dc=com
uid: bjensen
```

Given this pattern in the user entries, the default REST to LDAP configuration assumes that the user name on the HTTP side is the value of the user ID, and that user entries can be found under `ou=People,dc=example,dc=com`. In other words, Babs Jensen authenticates as `bjensen` (password: `hifalutin`) over HTTP. This is mapped for an LDAP bind to the bind DN `uid=bjensen,ou=People,dc=example,dc=com`.

With HTTP Basic authentication, it looks like this.

```
$ curl
 --user bjensen:hifalutin
 http://opendj.example.com:8080/users/bjensen?_prettyPrint=true
{
  "_rev" : "0000000016cbb68c",
  ...
}
```

Or, using the HTTP Basic *`username`*:*`password`*@ form in the URL, it looks like this.

```
$ curl
http://bjensen:hifalutin@opendj.example.com:8080/users/bjensen
 ?_prettyPrint=true
{
  "_rev" : "0000000016cbb68c",
  ...
}
```

With HTTP header based authentication, it looks like this.

```
$ curl
 --header "X-OpenIDM-Username: bjensen"
 --header "X-OpenIDM-Password: hifalutin"
 http://opendj.example.com:8080/users/bjensen?_prettyPrint=true
{
  "_rev" : "0000000016cbb68c",
  ...
}
```

If your directory data are laid out differently, or if your user names are email addresses rather than user IDs for example, then you must update the configuration in order for authentication to work.

The REST LDAP gateway can also translate HTTP user name and password authentication to PLAIN SASL authentication on the LDAP side. Moreover, the gateway can fall back to proxied authorization as necessary, using a root DN authenticated connection to LDAP servers. See *REST LDAP Configuration* for details on all configuration choices.

7.3 Creating Resources

There are two ways to create resources.

- To create a resource using an ID that you specify, perform an HTTP PUT request with headers `Content-Type: application/json` and `If-None-Match: *`, and the JSON content of your resource.

  The following example creates a new user entry with ID `newuser`.

```
$ curl
 --request PUT
 --user kvaughan:bribery
 --header "Content-Type: application/json"
 --header "If-None-Match: *"
 --data '{
  "_id": "newuser",
  "contactInformation": {
    "telephoneNumber": "+1 408 555 1212",
    "emailAddress": "newuser@example.com"
  },
  "name": {
    "familyName": "New",
    "givenName": "User"
  },
  "displayName": "New User",
  "manager": [
    {
      "_id": "kvaughan",
      "displayName": "Kirsten Vaughan"
    }
  ]
 }'
 http://opendj.example.com:8080/users/newuser?_prettyPrint=true
{
  "_rev" : "000000005b337348",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "contactInformation" : {
    "telephoneNumber" : "+1 408 555 1212",
    "emailAddress" : "newuser@example.com"
  },
  "_id" : "newuser",
  "name" : {
    "familyName" : "New",
    "givenName" : "User"
  },
  "userName" : "newuser@example.com",
  "displayName" : "New User",
  "meta" : {
    "created" : "2013-04-11T09:58:27Z"
  },
  "manager" : [ {
    "_id" : "kvaughan",
    "displayName" : "Kirsten Vaughan"
  } ]
}
```

- To create a resource letting the server choose the ID, perform an HTTP POST with `_action=create` as described in Section 7.8, “Using Actions”.

7.4 Reading a Resource

To read a resource, perform an HTTP GET.

```
$ curl
 --request GET
 --user kvaughan:bribery
 http://opendj.example.com:8080/users/newuser?_prettyPrint=true
{
  "_rev" : "000000005b337348",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "contactInformation" : {
    "telephoneNumber" : "+1 408 555 1212",
    "emailAddress" : "newuser@example.com"
  },
  "_id" : "newuser",
  "name" : {
    "familyName" : "New",
    "givenName" : "User"
  },
  "userName" : "newuser@example.com",
  "displayName" : "New User",
  "meta" : {
    "created" : "2013-04-11T09:58:27Z"
  },
  "manager" : [ {
    "_id" : "kvaughan",
    "displayName" : "Kirsten Vaughan"
  } ]
}
```

7.5 Updating Resources

To update a resource, perform an HTTP PUT with the changes to the resource. For read-only fields, either include unmodified versions, or omit them from your updated version.

The following example adds a manager for Sam Carter.

```
$ curl
 --request PUT
 --user kvaughan:bribery
 --header "Content-Type: application/json"
 --data '{
   "contactInformation": {
     "telephoneNumber": "+1 408 555 4798",
     "emailAddress": "scarter@example.com"
   },
   "name": {
     "familyName": "Carter",
     "givenName": "Sam"
   },
   "userName": "scarter@example.com",
   "displayName": "Sam Carter",
   "groups": [
     {
       "_id": "Accounting Managers"
     }
   ],
   "manager": [
     {
       "_id": "trigden",
       "displayName": "Torrey Rigden"
     }
   ]
 }'
 http://opendj.example.com:8080/users/scarter?_prettyPrint=true
{
  "_rev" : "00000000a1923db2",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "contactInformation" : {
    "telephoneNumber" : "+1 408 555 4798",
    "emailAddress" : "scarter@example.com"
  },
  "_id" : "scarter",
  "name" : {
    "familyName" : "Carter",
    "givenName" : "Sam"
  },
  "userName" : "scarter@example.com",
  "displayName" : "Sam Carter",
  "manager" : [ {
    "_id" : "trigden",
    "displayName" : "Torrey Rigden"
  } ],
  "meta" : {
    "lastModified" : "2013-04-12T07:42:34Z"
  },
  "groups" : [ {
    "_id" : "Accounting Managers"
  } ]
}
```

To update a resource only if the resource matches a particular version, use an `If-Match: ` *`revision`* header.

```
 $ curl
 --user kvaughan:bribery
 http://opendj.example.com:8080/users/scarter?_fields=_rev
{"_rev":"00000000b017c5b8"}
$ curl
 --request PUT
 --user kvaughan:bribery
 --header "If-Match: 00000000b017c5b8"
 --header "Content-Type: application/json"
 --data '{
   "contactInformation": {
     "telephoneNumber": "+1 408 555 1212",
     "emailAddress": "scarter@example.com"
   },
   "name": {
     "familyName": "Carter",
     "givenName": "Sam"
   },
   "userName": "scarter@example.com",
   "displayName": "Sam Carter",
   "groups": [
     {
       "_id": "Accounting Managers"
     }
   ],
   "manager": [
     {
       "_id": "trigden",
       "displayName": "Torrey Rigden"
     }
   ]
 }'
 http://opendj.example.com:8080/users/scarter?_prettyPrint=true
{
  "_rev" : "00000000a1ee3da3",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "contactInformation" : {
    "telephoneNumber" : "+1 408 555 1212",
    "emailAddress" : "scarter@example.com"
  },
  "_id" : "scarter",
  "name" : {
    "familyName" : "Carter",
    "givenName" : "Sam"
  },
  "userName" : "scarter@example.com",
  "displayName" : "Sam Carter",
  "meta" : {
    "lastModified" : "2013-04-12T07:47:45Z"
  },
  "groups" : [ {
    "_id" : "Accounting Managers"
  } ],
  "manager" : [ {
    "_id" : "trigden",
    "displayName" : "Torrey Rigden"
  } ]
}
```

7.6 Deleting Resources

To delete a resource, perform an HTTP DELETE on the resource URL. On success, the operation returns the resource you deleted.

```
$ curl
 --request DELETE
 --user kvaughan:bribery
 http://opendj.example.com:8080/users/newuser?_prettyPrint=true
{
  "_rev" : "000000003a5f3cb2",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "contactInformation" : {
    "telephoneNumber" : "+1 408 555 1212",
    "emailAddress" : "newuser@example.com"
  },
  "_id" : "newuser",
  "name" : {
    "familyName" : "New",
    "givenName" : "User"
  },
  "userName" : "newuser@example.com",
  "displayName" : "New User",
  "meta" : {
    "created" : "2013-04-11T09:58:27Z"
  },
  "manager" : [ {
    "_id" : "kvaughan",
    "displayName" : "Kirsten Vaughan"
  } ]
}
```

To delete a resource only if the resource matches a particular version, use an `If-Match:` *`revision`* header.

```
$ curl
 --user kvaughan:bribery
 http://opendj.example.com:8080/users/newuser?_fields=_rev
{"_rev":"000000006d8d7358"}
$ curl
 --request DELETE
 --user kvaughan:bribery
 --header "If-Match: 000000006d8d7358"
 http://opendj.example.com:8080/users/newuser?_prettyPrint=true
{
  "_rev" : "00000000383f3cae",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "contactInformation" : {
    "telephoneNumber" : "+1 408 555 1212",
    "emailAddress" : "newuser@example.com"
  },
  "_id" : "newuser",
  "name" : {
    "familyName" : "New",
    "givenName" : "User"
  },
  "userName" : "newuser@example.com",
  "displayName" : "New User",
  "meta" : {
    "created" : "2013-04-11T12:48:48Z"
  },
  "manager" : [ {
    "_id" : "kvaughan",
    "displayName" : "Kirsten Vaughan"
  } ]
}
```

To delete a resource and all its children, you must change the configuration, get the REST LDAP gateway or HTTP Connection Handler to reload its configuration, and perform the operation as a user who has the access rights required. The following steps show one way to do this with the HTTP Connection Handler.

In this case the LDAP view of the user to delete shows two child entries.

```
$ ldapsearch --port 1389 --baseDN uid=nbohr,ou=people,dc=example,dc=com "(&)" dn
dn: uid=nbohr,ou=People,dc=example,dc=com

dn: cn=quantum dot,uid=nbohr,ou=People,dc=example,dc=com

dn: cn=qubit generator,uid=nbohr,ou=People,dc=example,dc=com
```

1. In the configuration file for the HTTP Connection Handler, by default `/path/to/opendj/config/http-config.json`, set `"useSubtreeDelete" : true`.

Note

After this change, only users who have access to request a tree delete can delete resources.

2. Force the HTTP Connection Handler to reread its configuration.

```
$ dsconfig
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "HTTP Connection Handler"
 --set enabled:false
 --no-prompt
$ dsconfig
 set-connection-handler-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "HTTP Connection Handler"
 --set enabled:true
 --no-prompt
```

3. Delete as a user who has rights to perform a subtree delete on the resource.

```
$ curl
 --request DELETE
 --user kvaughan:bribery
 http://opendj.example.com:8080/users/nbohr?_prettyPrint=true
{
  "_rev" : "000000003d912113",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "contactInformation" : {
    "telephoneNumber" : "+1 408 555 1212",
    "emailAddress" : "nbohr@example.com"
  },
  "_id" : "nbohr",
  "name" : {
    "familyName" : "Bohr",
    "givenName" : "Niels"
  },
  "userName" : "nbohr@example.com",
  "displayName" : "Niels Bohr"
}
```

7.7 Patching Resources

OpenDJ lets you patch JSON resources, updating part of the resource rather than replacing it. For example, you could change Babs Jensen's email address by issuing an HTTP PATCH request, as in the example that follows.

Notice that the data sent specifies the type of patch operation, the field to change, and a value that depends on the field you change and on the operation. A single-valued field takes an object, boolean, string, or number depending on its type, whereas a multi-valued field takes an array of values. Getting the type wrong results in an error. Also notice that the patch data is itself an array, since you could patch more than one part of the resource by using a set of patch operations in the same request.

```
$ curl
 --user kvaughan:bribery
 --request PATCH
 --header "Content-Type: application/json"
 --data '[
  {
    "operation": "replace",
    "field": "/contactInformation/emailAddress",
    "value": "babs@example.com"
  }
 ]'
 http://opendj.example.com:8080/users/bjensen?_prettyPrint=true
{
  "_rev" : "00000000f3fdd370",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "contactInformation" : {
    "telephoneNumber" : "+1 408 555 1862",
    "emailAddress" : "babs@example.com"
  },
  "_id" : "bjensen",
  "name" : {
    "familyName" : "Jensen",
    "givenName" : "Barbara"
  },
  "userName" : "babs@example.com",
  "displayName" : "Barbara Jensen",
  "meta" : {
    "lastModified" : "2013-05-13T14:35:31Z"
  },
  "manager" : [ {
    "_id" : "trigden",
    "displayName" : "Torrey Rigden"
  } ]
}
```

OpenDJ supports four types of patch operation.

"add"

> The add operation ensures that the target field contains the value provided, creating parent fields as necessary.

If the target field is single-valued and a value already exists, then that value is replaced with the value you provide. **Note that you do not get an error when adding a value to a single-valued field that already has a value.** A single-valued field is one whose value is not an array (an object, string, boolean, or number).

If the target field is multi-valued, then the array of values you provide is merged with the set of values already in the resource. New values are added, and duplicate values are ignored. A multi-valued field takes an array value.

"remove"

The remove operation ensures that the target field does not contain the value provided. If you do not provide a value, the entire field is removed if it already exists.

If the target field is single-valued and a value is provided, then the provided value must match the existing value to remove, otherwise the field is left unchanged.

If the target field is multi-valued, then values in the array you provide are removed from the existing set of values.

"replace"

The replace operation removes existing values on the target field, and replaces them with the values you provide. It is equivalent to performing a remove on the field, then an add with the values you provide.

"increment"

The increment operation increments or decrements the value or values in the target field by the amount you specify, which is positive to increment, negative to decrement. The target field must be a number or a set of numbers. The value you provide must be a single number.

One key nuance in how patch works with OpenDJ has to do with multi-valued fields. Although JSON resources represent multi-valued fields as *arrays*, OpenDJ treats those values as *sets*. In other words, values in the field are unique, and the ordering of an array of values is not meaningful in the context of patch operations. If you reference array values by index, OpenDJ returns an error.[1]

Instead use the patch operations as if arrays values were sets. For example, you can include Barbara Jensen in a group by adding her to the set of members.

[1] OpenDJ does let you use a hyphen as the last element of the "field" JSON pointer value to add an element to the set, as in **curl --user kvaughan:bribery --request PATCH --header "Content-Type: application/json" --data '[{ "operation" : "add", "field" : "/members/-", "value" : { "_id" : "bjensen" } }]' http://opendj.example.com:8080/groups/Directory %20Administrators**.

```
$ curl
 --user kvaughan:bribery
 --request PATCH
 --header "Content-Type: application/json"
 --data '[
  {
    "operation": "add",
    "field": "/members",
    "value": [
      {
        "_id": "bjensen"
      }
    ]
  }
 ]'
 http://opendj.example.com:8080/groups/Directory%20Administrators
 ?_prettyPrint=true
{
  "_rev" : "00000000b70c881a",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "_id" : "Directory Administrators",
  "displayName" : "Directory Administrators",
  "meta" : {
    "lastModified" : "2013-05-13T16:40:23Z"
  },
  "members" : [ {
    "_id" : "kvaughan",
    "displayName" : "Kirsten Vaughan"
  }, {
    "_id" : "rdaugherty",
    "displayName" : "Robert Daugherty"
  }, {
    "_id" : "bjensen",
    "displayName" : "Barbara Jensen"
  }, {
    "_id" : "hmiller",
    "displayName" : "Harry Miller"
  } ]
}
```

Removing her from the group is similar.

```
$ curl
 --user kvaughan:bribery
 --request PATCH
 --header "Content-Type: application/json"
 --data '[
  {
    "operation": "remove",
    "field": "/members",
    "value": [
      {
        "_id": "bjensen"
      }
    ]
  }
 ]'
 http://opendj.example.com:8080/groups/Directory%20Administrators
 ?_prettyPrint=true
{
  "_rev" : "00000000e241797e",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "_id" : "Directory Administrators",
  "displayName" : "Directory Administrators",
  "meta" : {
    "lastModified" : "2013-05-13T16:40:55Z"
  },
  "members" : [ {
    "_id" : "kvaughan",
    "displayName" : "Kirsten Vaughan"
  }, {
    "_id" : "rdaugherty",
    "displayName" : "Robert Daugherty"
  }, {
    "_id" : "hmiller",
    "displayName" : "Harry Miller"
  } ]
}
```

You can use resource revision numbers in `If-Match:` *`revision`* headers to patch the resource only if the resource matches a particular version.

```
$ curl
 --user kvaughan:bribery
 "http://opendj.example.com:8080/users/bjensen?_prettyPrint=true&_fields=_rev"
{
  "_rev" : "00000000c1b6d4c7"
}
$ curl
 --user kvaughan:bribery
 --request PATCH
 --header "If-Match: 00000000c1b6d4c7"
 --header "Content-Type: application/json"
 --data '[
  {
    "operation": "add",
    "field": "/contactInformation/emailAddress",
    "value": "babs@example.com"
  }
 ]'
 http://opendj.example.com:8080/users/bjensen?_prettyPrint=true
{
  "_rev" : "00000000f946d377",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "contactInformation" : {
    "telephoneNumber" : "+1 408 555 1862",
    "emailAddress" : "babs@example.com"
  },
  "_id" : "bjensen",
  "name" : {
    "familyName" : "Jensen",
    "givenName" : "Barbara"
  },
  "userName" : "babs@example.com",
  "displayName" : "Barbara Jensen",
  "meta" : {
    "lastModified" : "2013-05-13T16:56:33Z"
  },
  "manager" : [ {
    "_id" : "trigden",
    "displayName" : "Torrey Rigden"
  } ]
}
```

The resource revision changes after you successfully perform the patch operation.

7.8 Using Actions

OpenDJ implements an action that lets the server set the resource ID on creation. To use this action, perform an HTTP POST with header `Content-Type: application/json`, `_action=create` in the query string, and the JSON content of your resource.

The following example creates a new user entry.

```
$ curl
 --request POST
 --user kvaughan:bribery
 --header "Content-Type: application/json"
 --data '{
  "_id": "newuser",
  "contactInformation": {
    "telephoneNumber": "+1 408 555 1212",
    "emailAddress": "newuser@example.com"
  },
  "name": {
    "familyName": "New",
    "givenName": "User"
  },
  "displayName": "New User",
  "manager": [
    {
      "_id": "kvaughan",
      "displayName": "Kirsten Vaughan"
    }
  ]
 }'
 "http://opendj.example.com:8080/users?_action=create&_prettyPrint=true"
{
  "_rev" : "0000000034a23ca7",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "contactInformation" : {
    "telephoneNumber" : "+1 408 555 1212",
    "emailAddress" : "newuser@example.com"
  },
  "_id" : "newuser",
  "name" : {
    "familyName" : "New",
    "givenName" : "User"
  },
  "userName" : "newuser@example.com",
  "displayName" : "New User",
  "meta" : {
    "created" : "2013-04-11T11:19:08Z"
  },
  "manager" : [ {
    "_id" : "kvaughan",
    "displayName" : "Kirsten Vaughan"
  } ]
}
```

7.9 Querying Resource Collections

To query resource collections, perform an HTTP GET with a `_queryFilter=`*`filter`* parameter in your query string.

For query operations, your *`filter`* expressions are constructed from the following building blocks. Make sure you URL encode the filter expressions, which are shown here without URL encoding to make them easier to read.

In these expressions the simplest *`json-pointer`* is a field of the JSON resource, such as `userName` or `id`. A *`json-pointer`* can however point to nested elements as described in the JSON Pointer Internet-Draft.

Comparison expressions

You can build filters using the following comparison expressions.

`json-pointer` `eq` *`json-value`*

Matches when the pointer equals the value, as in the following example.

```
$ curl --user kvaughan:bribery 'http://opendj.example.com:8080
 /users?_queryFilter=userName+eq+"bjensen@example.com"&_prettyPrint=true'
{
  "result" : [ {
    "_rev" : "00000000315fb731",
    "schemas" : [ "urn:scim:schemas:core:1.0" ],
    "manager" : [ {
      "_id" : "trigden",
      "displayName" : "Torrey Rigden"
    } ],
    "contactInformation" : {
      "telephoneNumber" : "+1 408 555 1862",
      "emailAddress" : "bjensen@example.com"
    },
    "_id" : "bjensen",
    "name" : {
      "familyName" : "Jensen",
      "givenName" : "Barbara"
    },
    "userName" : "bjensen@example.com",
    "displayName" : "Barbara Jensen"
  } ],
  "resultCount" : 1,
  "pagedResultsCookie" : null,
  "remainingPagedResults" : -1
}
```

json-pointer `co` *json-value*

Matches when the pointer contains the value, as in the following example.

```
$ curl --user kvaughan:bribery 'http://opendj.example.com:8080
 /users?_queryFilter=userName+co+"jensen"&_fields=userName&_prettyPrint=true'
{
  "result" : [ {
    "userName" : "ajensen@example.com"
  }, {
    "userName" : "bjensen@example.com"
  }, {
    "userName" : "gjensen@example.com"
  }, {
    "userName" : "jjensen@example.com"
  }, {
    "userName" : "kjensen@example.com"
  }, {
    "userName" : "rjensen@example.com"
  }, {
    "userName" : "tjensen@example.com"
  } ],
  "resultCount" : 7,
  "pagedResultsCookie" : null,
  "remainingPagedResults" : -1
}
```

json-pointer `sw` *json-value*

Matches when the pointer starts with the value, as in the following example.

```
$ curl --user kvaughan:bribery 'http://opendj.example.com:8080
 /users?_queryFilter=userName+sw+"ab"&_fields=userName&_prettyPrint=true'
{
  "result" : [ {
    "userName" : "abarnes@example.com"
  }, {
    "userName" : "abergin@example.com"
  } ],
  "resultCount" : 2,
  "pagedResultsCookie" : null,
  "remainingPagedResults" : -1
}
```

json-pointer `lt` *json-value*

Matches when the pointer is less than the value, as in the following example.

```
$ curl --user kvaughan:bribery 'http://opendj.example.com:8080
 /users?_queryFilter=userName+lt+"ac"&_fields=userName&_prettyPrint=true'
{
  "result" : [ {
    "userName" : "abarnes@example.com"
  }, {
    "userName" : "abergin@example.com"
  } ],
  "resultCount" : 2,
  "pagedResultsCookie" : null,
  "remainingPagedResults" : -1
}
```

json-pointer le *json-value*

Matches when the pointer is less than or equal to the value, as in the following example.

```
$ curl --user kvaughan:bribery 'http://opendj.example.com:8080
 /users?_queryFilter=userName+le+"ad"&_fields=userName&_prettyPrint=true'
{
  "result" : [ {
    "userName" : "abarnes@example.com"
  }, {
    "userName" : "abergin@example.com"
  }, {
    "userName" : "achassin@example.com"
  } ],
  "resultCount" : 3,
  "pagedResultsCookie" : null,
  "remainingPagedResults" : -1
}
```

json-pointer gt *json-value*

Matches when the pointer is greater than the value, as in the following example.

```
$ curl --user kvaughan:bribery 'http://opendj.example.com:8080
 /users?_queryFilter=userName+gt+"tt"&_fields=userName&_prettyPrint=true'
{
  "result" : [ {
    "userName" : "ttully@example.com"
  }, {
    "userName" : "tward@example.com"
  }, {
    "userName" : "wlutz@example.com"
  } ],
  "resultCount" : 3,
  "pagedResultsCookie" : null,
  "remainingPagedResults" : -1
}
```

json-pointer ge *json-value*

Matches when the pointer is greater than or equal to the value, as in the following example.

```
$ curl --user kvaughan:bribery 'http://opendj.example.com:8080
 /users?_queryFilter=userName+ge+"tw"&_fields=userName&_prettyPrint=true'
{
  "result" : [ {
    "userName" : "tward@example.com"
  }, {
    "userName" : "wlutz@example.com"
  } ],
  "resultCount" : 2,
  "pagedResultsCookie" : null,
  "remainingPagedResults" : -1
}
```

Presence expression

`json-pointer` `pr` matches any resource on which the *`json-pointer`* is present, as in the following example.

```
$ curl --user kvaughan:bribery 'http://opendj.example.com:8080
 /users?_queryFilter=userName%20pr&_prettyPrint=true'
{
  "result" : [ {
    "_rev" : "000000002210a544",
    "schemas" : [ "urn:scim:schemas:core:1.0" ],
    "manager" : [ {
      "_id" : "scarter",
      "displayName" : "Sam Carter"
    } ],
    "contactInformation" : {
      "telephoneNumber" : "+1 408 555 9445",
      "emailAddress" : "abarnes@example.com"
    },
    "_id" : "abarnes",
    "name" : {
      "familyName" : "Barnes",
      "givenName" : "Anne-Louise"
    },
    "userName" : "abarnes@example.com",
    "displayName" : "Anne-Louise Barnes"
  },… many entries omitted …
    "_id" : "newuser",
    "name" : {
      "familyName" : "New",
      "givenName" : "User"
    },
    "userName" : "newuser@example.com",
    "displayName" : "New User",
    "meta" : {
      "created" : "2013-03-26T10:52:42Z"
    }
  } ],
  "resultCount" : 152,
  "pagedResultsCookie" : null,
  "remainingPagedResults" : -1
}
```

Literal expressions

`true` matches any resource in the collection.

`false` matches no resource in the collection.

In other words you can list all resources in a collection as in the following example.

```
$ curl --user kvaughan:bribery 'http://opendj.example.com:8080
 /groups?_queryFilter=true&_fields=displayName&_prettyPrint=true'
{
  "result" : [ {
    "displayName" : "Accounting Managers"
  }, {
    "displayName" : "Directory Administrators"
  }, {
    "displayName" : "HR Managers"
  }, {
    "displayName" : "PD Managers"
  }, {
    "displayName" : "QA Managers"
  } ],
  "resultCount" : 5,
  "pagedResultsCookie" : null,
  "remainingPagedResults" : -1
}
```

Complex expressions

You can combine expressions using boolean operators `and`, `or`, and `!` (not), using parentheses, `(`*`expression`*`)`, to group expressions. The following example queries resources with last name Jensen and manager name starting with `Bar`. Notice that the filters use the JSON pointers `name/familyName` and `manager/displayName` to identify the fields that are nested inside the `name` and `manager` objects.

```
$ curl --user kvaughan:bribery 'http://opendj.example.com:8080
 /users?_queryFilter=(userName+co+"jensen"+and+manager/displayName+sw+"Sam")
 &_fields=displayName&_prettyPrint=true'
{
  "result" : [ {
    "displayName" : "Jody Jensen"
  }, {
    "displayName" : "Ted Jensen"
  } ],
  "resultCount" : 2,
  "pagedResultsCookie" : null,
  "remainingPagedResults" : -1
}
```

7.10 Change Your Password

Note

This action requires HTTPS to avoid sending the password over an insecure connection.

Perform an HTTPS POST with the header Content-Type: application/json, _action=modifyPassword in the query string, and the old and new passwords in JSON format as the POST data.

The JSON POST DATA must include the following fields:

oldPassword

The value of this field is the current password as a UTF-8 string.

newPassword

The value of this field is the new password as a UTF-8 string.

On success, the HTTP status code is 200 OK, and the response body is an empty JSON resource:

```
$ curl \
            --request POST \
            --cacert ca-cert.pem \
            --user bjensen:hifalutin \
            --header "Content-Type: application/json" \
            --data '{"oldPassword": "hifalutin", "newPassword": "chngthspwd"}' \
            --silent \
            https://localhost:8443/api/users/bjensen?_action=modifyPassword

            {}
```

7.11 Reset a Password

Whenever one user changes another user’s password, DS servers consider it a password reset. Often, password policies specify that users must change their passwords again after a password reset.

Note

This action requires HTTPS to avoid sending the password over an insecure connection.

Perform an HTTPS POST with the header Content-Type: application/json, _action=resetPassword in the query string, and an empty JSON document ({}) as the POST data.

The JSON POST DATA must include the following fields:

The following example demonstrates an administrator changing a user’s password. Before trying this example, make sure the password administrator has been given the password-reset privilege. Otherwise, the password administrator has insufficient access. On success, the HTTP status code is 200 OK, and the response body is a JSON resource with a generatedPassword containing the new password:

```
$ curl \
            --request POST \
            --cacert ca-cert.pem \
            --user kvaughan:bribery \
            --header "Content-Type: application/json" \
            --data '{}' \
            --silent \
            https://localhost:8443/api/users/bjensen?_action=resetPassword

            {"generatedPassword":"new-password"}
```

As password administrator, provide the new, generated password to the user.

Chapter 8

Indexing Attribute Values

OpenDJ provides several indexing schemes to speed up searches.

When a client requests a directory search operation, the client sends the server a filter expression such as `(&(uid=*jensen*)(l=Stavanger))`. The server then uses applicable indexes to find entries with attribute values likely to match the search. If no indexes are applicable, then the server potentially has to go through all entries to look for candidate matches.

Looking through all entries is resource-intensive for large directories. For this reason, the `unindexed-search` privilege, allowing users to request searches for which no applicable index exists, is reserved for the directory root user by default.

Rather than granting the `unindexed-search` privilege to more users and client applications, you configure indexes to correspond to the searches that clients need to perform.

This chapter first describes index types, then demonstrates how to index attribute values. This chapter also lists the default indexing configuration for OpenDJ directory server.

8.1 Index Types & What Each Does

OpenDJ provides several different index types, each corresponding to a different type of search.

8.1.1 Approximate Index

An approximate index is used to match values that "sound like" those provided in the filter. An approximate index on cn allows clients to find people even when they misspell names as in the following example.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com "(cn~=Babs Jansen)" cn
dn: uid=bjensen,ou=People,dc=example,dc=com
cn: Barbara Jensen
cn: Babs Jensen
```

8.1.2 Equality Index

An equality index is used to match values that correspond exactly (though generally without case sensitivity) to the value provided in the search filter. An equality index requires clients to match values without wildcards or misspellings.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com "(uid=bjensen)" mail
dn: uid=bjensen,ou=People,dc=example,dc=com
mail: bjensen@example.com
```

8.1.3 Ordering Index

An ordering index is used to match values for a filter that specifies a range. The ds-sync-hist has an ordering index by default because searches on that attribute often seek entries with changes more recent than the last time a search was performed.

The following example shows a search that specifies ranges.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com
 "(&(uidNumber>=1120)(roomNumber>=4500))" uid
dn: uid=charvey,ou=People,dc=example,dc=com
uid: charvey

dn: uid=eward,ou=People,dc=example,dc=com
uid: eward

dn: uid=mvaughan,ou=People,dc=example,dc=com
uid: mvaughan

dn: uid=pchassin,ou=People,dc=example,dc=com
uid: pchassin
```

8.1.4 Presence Index

A presence index is used to match the fact that an attribute is present on the entry, regardless of the value. The aci attribute is indexed for presence by default to allow quick retrieval of entries with ACIs.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com "(aci=*)" -
dn: dc=example,dc=com

dn: ou=People,dc=example,dc=com
```

8.1.5 Substring Index

A substring index is used to match values specified with wildcards in the filter. Substring indexes can be expensive to maintain, especially for large attribute values.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com "(cn=Barb*)" cn
dn: uid=bfrancis,ou=People,dc=example,dc=com
cn: Barbara Francis

dn: uid=bhal2,ou=People,dc=example,dc=com
cn: Barbara Hall

dn: uid=bjablons,ou=People,dc=example,dc=com
cn: Barbara Jablonski

dn: uid=bjensen,ou=People,dc=example,dc=com
cn: Barbara Jensen
cn: Babs Jensen

dn: uid=bmaddox,ou=People,dc=example,dc=com
cn: Barbara Maddox
```

8.1.6 Virtual List View (Browsing) Index

A VLV or browsing index are designed to help the server respond to client applications that need virtual list view results, for example to browse through a long list in a GUI. They also help the server respond to clients that request server-side sorting of the search results.

VLV indexes correspond to particular searches. Configure your VLV indexes using the Control Panel, and copy the command-line equivalent from the Details pane for the operation, if necessary.

8.1.7 Extensible Matching Rule Index

In some cases you need an index for a matching rule other than those described above. For example, OpenDJ supports generalized time based matching so applications can search for all times later than, or earlier than a specified time.

8.2 Configuring & Rebuilding Indexes

You modify index configurations using the **dsconfig** command. The configuration changes then take effect after you rebuild the index according to the new configuration, using the **rebuild-index**. The **dsconfig --help-database** command lists subcommands for creating, reading, updating, and deleting index configuration.

Tip

Indexes are per directory backend rather than per suffix. To maintain separate indexes for different suffixes on the same directory server, put the suffixes in different backends.

8.2.1 Configuring a Standard Index

You can configure standard indexes from the Control Panel, and also on the command line using the **dsconfig** command. After you finish configuring the index, you must rebuild the index for the changes to take effect.

Example 8.1. Create a New Index

The following example creates a new substring index for `description`.

```
$ dsconfig
 create-local-db-index
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --backend-name userRoot
 --index-name description
 --set index-type:substring
 --trustAll
 --no-prompt
```

Example 8.2. Configure an Approximate Index

The following example configures an approximate index for `cn` (common name).

```
$ dsconfig
 set-local-db-index-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --backend-name userRoot
 --index-name cn
 --set index-type:approximate
 --trustAll
 --no-prompt
```

Example 8.3. Configure an Extensible Match Index

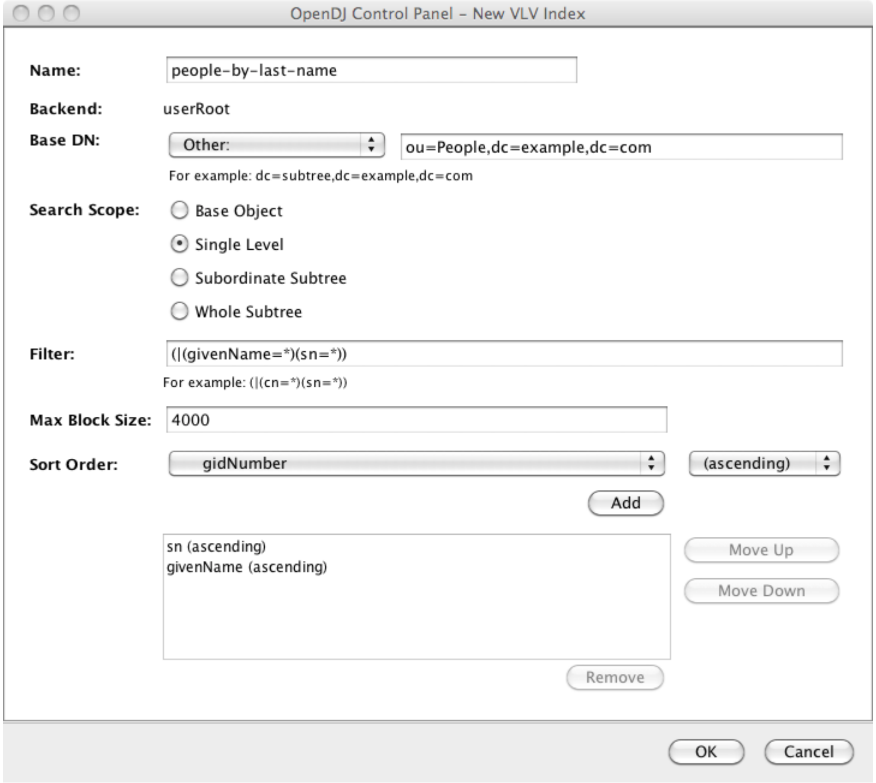
The OpenDJ Control Panel New Index window does not help you set up extensible matching rule indexes. Use the **dsconfig** command instead.

The following example configures an extensible matching rule index for "later than" and "earlier than" generalized time matching on a `lastLoginTime` attribute.

```
$ dsconfig
 create-local-db-index
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --backend-name userRoot
 --set index-type:extensible
 --set index-extensible-matching-rule:1.3.6.1.4.1.26027.1.4.5
 --set index-extensible-matching-rule:1.3.6.1.4.1.26027.1.4.6
 --index-name lastLoginTime
 --trustAll
 --no-prompt
```

8.2.2 Configuring a Virtual List View Index

In the OpenDJ Control Panel, select Manage Indexes > New VLV Index..., and then set up your VLV index using the New VLV Index window.



OpenDJ Control Panel – New VLV Index

Name: people-by-last-name

Backend: userRoot

Base DN: Other: ou=People,dc=example,dc=com

For example: dc=subtree,dc=example,dc=com

Search Scope:
○ Base Object
● Single Level
○ Subordinate Subtree
○ Whole Subtree

Filter: (|(givenName=*)(sn=*))

For example: (|(cn=*)(sn=*))

Max Block Size: 4000

Sort Order: gidNumber (ascending)

sn (ascending)
givenName (ascending)

After you finish configuring your index and click OK, the Control Panel prompts you to make the additional changes necessary to complete the VLV index configuration, and then to build the index.

You can also create the equivalent index configuration using the **dsconfig** command.

```
$ dsconfig
 create-local-db-vlv-index
 --port 4444
 --hostname opendj.example.com
 --bindDn "cn=Directory Manager"
 --bindPassword password
 --backend-name userRoot
 --index-name people-by-last-name
 --set base-dn:ou=People,dc=example,dc=com
 --set filter:"(|(givenName=*)(sn=*))"
 --set scope:single-level
 --set sort-order:"+sn +givenName"
 --trustAll
 --no-prompt
```

Note

When referring to a virtual list view (VLV) index after creation, you must add `vlv.` as a prefix. In other words, if you named the VLV index `people-by-last-name`, you refer to it as `vlv.people-by-last-name` when rebuilding indexes, changing index properties such as the index entry limit, or verifying indexes.

8.2.3 Rebuilding Indexes

After you change an index configuration, or when you find that an index is corrupt, you can rebuild the index. When you rebuild indexes, you specify the base DN of the data to index, and either the list of indexes to rebuild or `--rebuildAll`. You can rebuild indexes while the server is offline, or while the server is online. If you rebuild the index while the server is online, then you must schedule the rebuild process as a task.

Example 8.4. Rebuild Index

The following example rebuilds the `cn` index immediately with the server online.

```
$ rebuild-index
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --baseDN dc=example,dc=com
 --index cn
 --start 0
Rebuild Index task 20110607171639867 scheduled to start Jun 7, 2011 5:16:39 PM
```

Example 8.5. Rebuild Degraded Indexes

The following example rebuilds degraded indexes immediately with the server online.

```
$ rebuild-index
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --baseDN dc=example,dc=com
 --rebuildDegraded
...
[31/Jan/2012:16:43:25 +0100] severity="NOTICE" msgCount=7 msgID=8847510
 message="Due to changes in the configuration, index
 dc_example_dc_com_description is currently operating in a degraded state
 and must be rebuilt before it can be used"
[31/Jan/2012:16:43:25 +0100] severity="NOTICE" msgCount=8 msgID=8847591
 message="Rebuild of all degraded indexes started with 160 total entries
 to process"
...
[31/Jan/2012:16:43:25 +0100] severity="NOTICE" msgCount=10 msgID=8847493
 message="Rebuild complete. Processed 160 entries in 0 seconds (average
 rate 1860.5/sec)"
...
Rebuild Index task 20120131164324838 has been successfully completed
```

Example 8.6. Clear New, Unused, "Degraded" Indexes

When you add a new attribute as described in *Updating Directory Schema*, and then create indexes for the new attribute, the new indexes appear as degraded, even though the attribute has not yet been used, and so indexes are sure to be empty, rather than degraded.

In this special case, you can safely use the **rebuild-index** command `--clearDegradedState` option to avoid having to scan the entire directory backend to rebuild the new, unused index. This is shown in the following example, where an index has just been created for `newUnusedAttribute`.

```
$ dbtest
 list-index-status
 --backendID userRoot
 --baseDN dc=example,dc=com
 | grep newUnusedAttribute
newUnusedAttribute.equality   Index  ...newUnusedAttribute.equality   false...
newUnusedAttribute.presence   Index  ...newUnusedAttribute.presence   false...
newUnusedAttribute.substring  Index  ...newUnusedAttribute.substring  false...
$ rebuild-index
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --baseDN dc=example,dc=com
 --clearDegradedState
 --index newUnusedAttribute
 --start 0
Rebuild Index task 20130211175925012 scheduled to start Feb 11, 2013 5:59:25
 PM CET
$ dbtest
 list-index-status
 --backendID userRoot
 --baseDN dc=example,dc=com
 | grep newUnusedAttribute
newUnusedAttribute.equality   Index  ...newUnusedAttribute.equality   true...
newUnusedAttribute.presence   Index  ...newUnusedAttribute.presence   true...
newUnusedAttribute.substring  Index  ...newUnusedAttribute.substring  true...
```

If the newly indexed attribute has already been used, rebuild indexes instead.

8.2.4 Changing Index Entry Limits

As the number of entries in your directory grows, it can make sense not to maintain indexes for particular values. For example, every entry in the directory has the value `top` for the `objectClass` attribute, so maintaining a list of entries that match the filter `(objectClass=top)` is not a reasonable use of resources. In a very, very large directory, the same can be true for `(givenName=John)` and `(sn=Smith)`.

In an index, each index key points to a list of entries that are candidates to match. For the `objectClass` index key that corresponds to `=top`, the list of entries can include every entry in the directory.

OpenDJ directory server therefore defines an index entry limit. When the number of entries that an index key points to exceeds the index entry limit, OpenDJ stops maintaining the list of entries for that index key.

The default index entry limit value is 4000. 4000 is usually plenty large for all index keys, except for `objectClass` indexes. If you have clients performing searches with filters such as `(objectClass=person)`, you might suggest that they adjust the search to be more specific, such as `(&(mail=username@maildomain.net)(objectClass=person))`, so that the server can use an index, in this case equality for mail, to limit the number of candidate entries to check for matches.

You can change the index entry limit on a per index basis.

Example 8.7. Change Index Entry Limit

The following example changes the index entry limit for the `objectClass` index, and then rebuilds the index for the configuration change to take effect.

```
$ dsconfig
 set-local-db-index-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --backend-name userRoot
 --index-name objectClass
 --set index-entry-limit:5000
 --trustAll
 --no-prompt
$ rebuild-index
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --baseDN dc=example,dc=com
 --index objectclass
 --start 0
Rebuild Index task 20110607160349596 scheduled to start Jun 7, 2011 4:03:49 PM
```

Alternatively, you can configure the index entry limit for all indexes stored in a backend by using the **dsconfig set-backend-prop** command with the `--backend-name` *backendName* `--set index-entry-limit:`*limitValue* options.

8.3 Verifying Indexes

You can verify that indexes correspond to current directory data, and that indexes do not contain errors using the **verify-index** command.

Example 8.8. Verify Index

The following example verifies the `cn` (common name) index for completeness and for errors.

```
$ verify-index
 --baseDN dc=example,dc=com
 --index cn
 --clean
 --countErrors
[07/Jun/2011:16:06:50 +0200] category=BACKEND severity=INFORMATION
 msgID=9437595 msg=Local DB backend userRoot does not specify the number of
 lock tables: defaulting to 97
[07/Jun/2011:16:06:50 +0200] category=BACKEND severity=INFORMATION
 msgID=9437594 msg=Local DB backend userRoot does not specify the number of
 cleaner threads: defaulting to 24 threads
[07/Jun/2011:16:06:51 +0200] category=JEB severity=NOTICE msgID=8847461
 msg=Checked 1316 records and found 0 error(s) in 0 seconds
 (average rate 2506.7/sec)
[07/Jun/2011:16:06:51 +0200] category=JEB severity=INFORMATION
 msgID=8388710 msg=Number of records referencing more than one entry: 315
[07/Jun/2011:16:06:51 +0200] category=JEB severity=INFORMATION
 msgID=8388711 msg=Number of records that exceed the entry limit: 0
[07/Jun/2011:16:06:51 +0200] category=JEB severity=INFORMATION
 msgID=8388712 msg=Average number of entries referenced is 1.58/record
[07/Jun/2011:16:06:51 +0200] category=JEB severity=INFORMATION
 msgID=8388713 msg=Maximum number of entries referenced by any
 record is 32
```

Ignore the messages regarding lock tables and cleaner threads. The important information is whether any errors are found in the indexes.

8.4 Checking Indexes For a Search

When searching, you can improve performance by making sure your search is indexed as you expect. One way of checking is to request the `debugsearchindex` attribute in your results.

```
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 "(uid=bjensen)"
 debugsearchindex
dn: cn=debugsearch
debugsearchindex: filter=(uid=bjensen)[INDEX:uid.equality][COUNT:1]
 final=[COUNT:1]
```

When you request the `debugsearchindex` attribute, instead of performing the search, OpenDJ returns debug information indicating how it would process the search operation. In the example above you notice OpenDJ hits the equality index for `uid` right away.

A less exact search requires more work from OpenDJ. In the following example OpenDJ would have to return 160 entries.

```
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 "(uid=*)"
 debugsearchindex
dn: cn=debugsearch
debugsearchindex: filter=(uid=*)[NOT-INDEXED] scope=wholeSubtree[COUNT:160]
 final=[COUNT:160]
```

By default OpenDJ rejects unindexed searches when the number of candidate entries goes beyond the search or look-though limit.

8.5 Default Indexes

When you first install OpenDJ directory server and import your data from LDIF, the following indexes are configured.

Table 8.1. Default Indexes

Index	Approx.	Equality	Ordering	Presence	Substring	Entry Limit
`aci`	-	-	-	Yes	-	4000
`cn`	-	Yes	-	-	Yes	4000
`dn2id`	Non-configurable internal index					
`ds-sync-conflict`	-	Yes	-	-	-	4000
`ds-sync-hist`	-	-	Yes	-	-	4000
`entryUUID`	-	Yes	-	-	-	4000
`givenName`	-	Yes	-	-	Yes	4000
`id2children`	Non-configurable internal index					
`id2subtree`	Non-configurable internal index					
`mail`	-	Yes	-	-	Yes	4000
`member`	-	Yes	-	-	-	4000
`objectClass`	-	Yes	-	-	-	4000

Index	Approx.	Equality	Ordering	Presence	Substring	Entry Limit
`sn`	-	Yes	-	-	Yes	4000
`telephone-Number`	-	Yes	-	-	Yes	4000
`uid`	-	Yes	-	-	-	4000
`unique-Member`	-	Yes	-	-	-	4000

When you create a JE backend using the **dsconfig** command, OpenDJ creates the `aci` presence, `ds-sync-conflict` equality, `ds-sync-hist` ordering, `entryUUID` equality, and `objectClass` equality indexes automatically.

Chapter 9

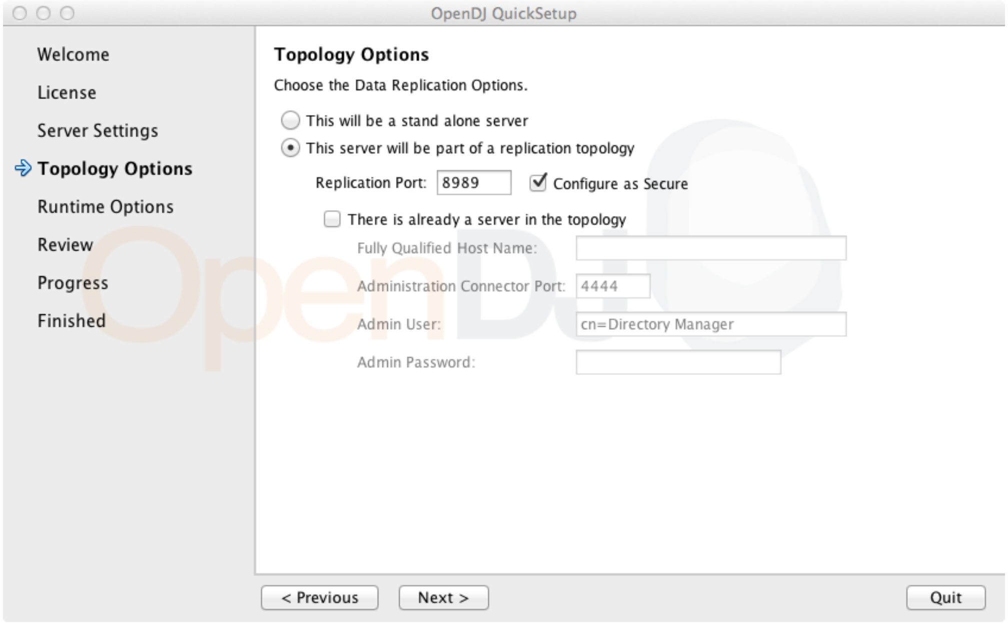
Managing Data Replication

OpenDJ uses advanced data replication with automated conflict resolution to help ensure your directory services remain available in the event a server crashes or a network goes down, and also as you backup or upgrade your directory service. You can configure data replication as part of OpenDJ installation, and in many cases let replication do its work in the background.

9.1 Replication Quick Setup

You can set up replication during installation by choosing to configure replication through the setup wizard.

In the Topology Options screen for the first server you set up, select This server will be part of a replication topology. If you also choose Configure as Secure, then replication traffic is protected by SSL.



Topology Options

Choose the Data Replication Options.

○ This will be a stand alone server

● This server will be part of a replication topology

Replication Port: 8989 ☑ Configure as Secure

□ There is already a server in the topology

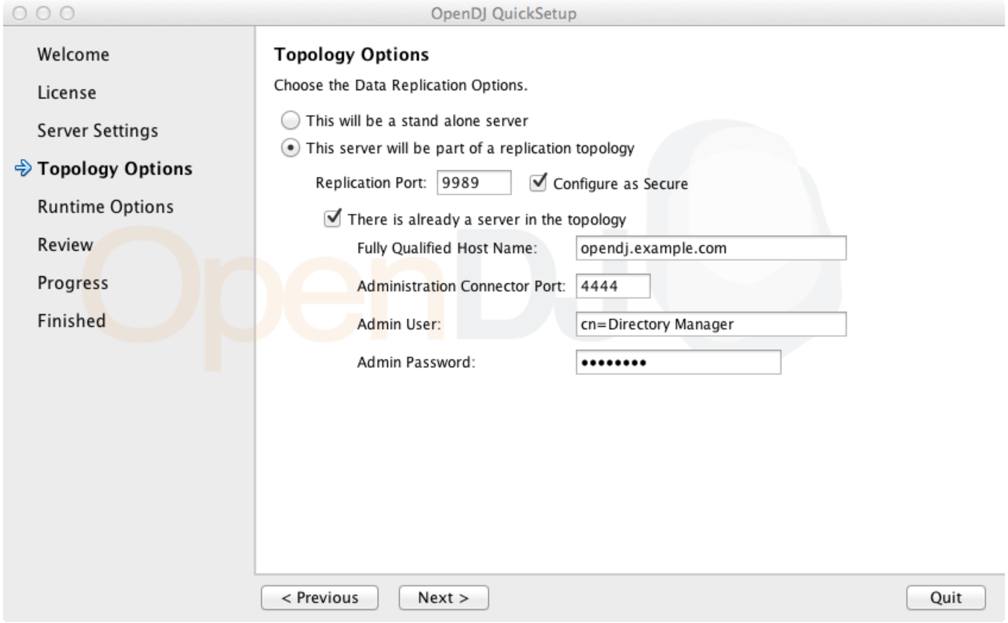
Fully Qualified Host Name:

Administration Connector Port: 4444

Admin User: cn=Directory Manager

Admin Password:

In the Topology Options screen for subsequent servers, also select There is already a server in the topology, providing the Host Name, Administration Connector Port number, Admin User, and Admin Password for the first replica you set up.



Topology Options

Choose the Data Replication Options.

○ This will be a stand alone server

● This server will be part of a replication topology

Replication Port: 9989 ☑ Configure as Secure

☑ There is already a server in the topology

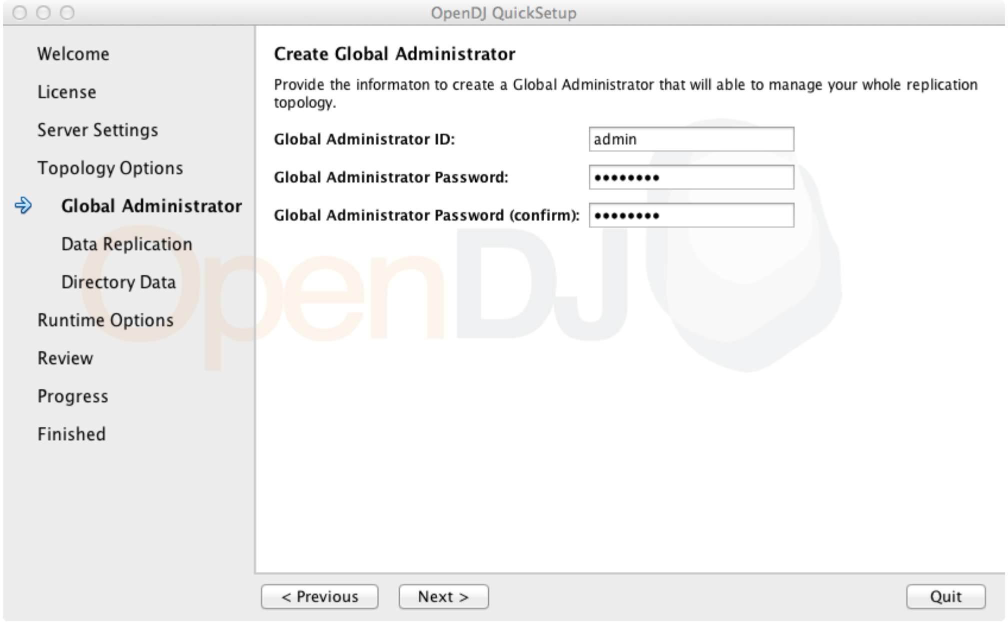
Fully Qualified Host Name: opendj.example.com

Administration Connector Port: 4444

Admin User: cn=Directory Manager

Admin Password: ••••••••

You also set up a global administrator account, stored under `cn=admin data` across replicas, used to manage replication in the topology.



Create Global Administrator

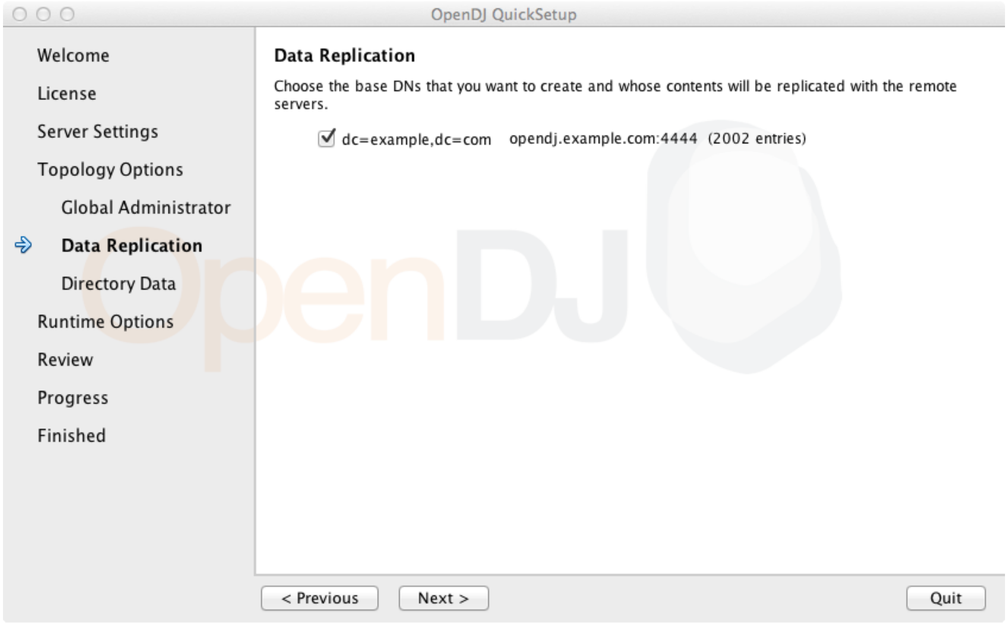
Provide the informaton to create a Global Administrator that will able to manage your whole replication topology.

Global Administrator ID: admin

Global Administrator Password: ••••••••

Global Administrator Password (confirm): ••••••••

You further set up what to replicate.

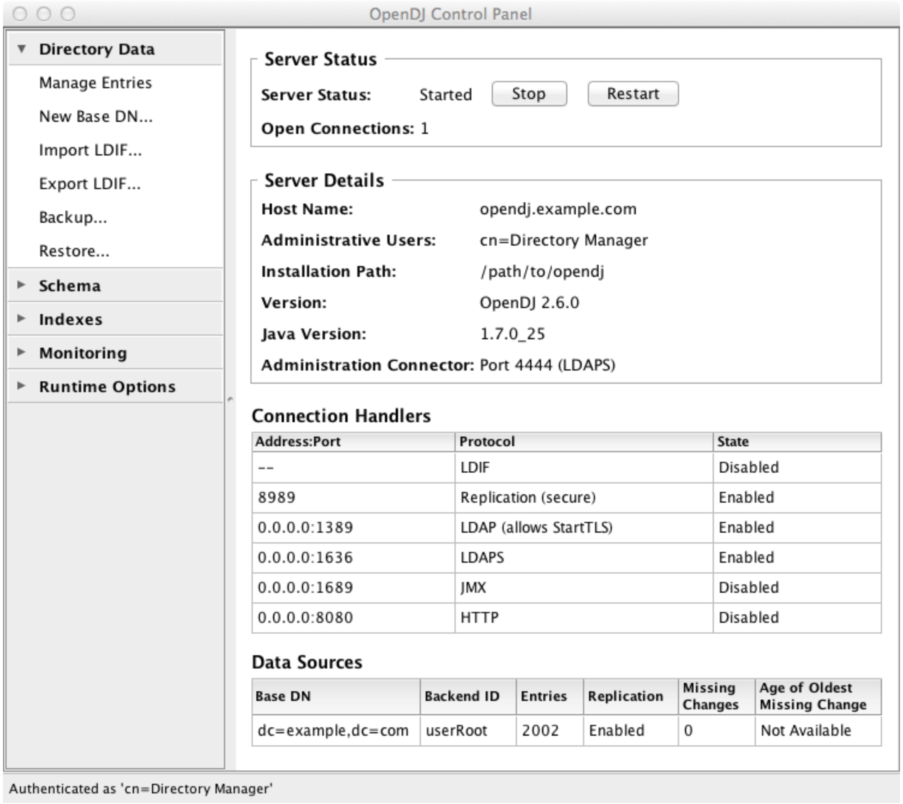


Data Replication

Choose the base DNs that you want to create and whose contents will be replicated with the remote servers.

☑ dc=example,dc=com opendj.example.com:4444 (2002 entries)

Once replication is set up, it works for all the replicas. You can monitor the replication connection and status through the OpenDJ Control Panel.



Server Status

Server Status: Started

Open Connections: 1

Server Details

Host Name:	opendj.example.com
Administrative Users:	cn=Directory Manager
Installation Path:	/path/to/opendj
Version:	OpenDJ 2.6.0
Java Version:	1.7.0_25
Administration Connector:	Port 4444 (LDAPS)

Connection Handlers

Address:Port	Protocol	State
--	LDIF	Disabled
8989	Replication (secure)	Enabled
0.0.0.0:1389	LDAP (allows StartTLS)	Enabled
0.0.0.0:1636	LDAPS	Enabled
0.0.0.0:1689	JMX	Disabled
0.0.0.0:8080	HTTP	Disabled

Data Sources

Base DN	Backend ID	Entries	Replication	Missing Changes	Age of Oldest Missing Change
dc=example,dc=com	userRoot	2002	Enabled	0	Not Available

Authenticated as 'cn=Directory Manager'

9.2 About Replication

Before you take replication further than setting up replication in the setup wizard, read this section to learn more about how OpenDJ replication works.

9.2.1 What Replication Is

Replication is the process of copying updates between OpenDJ directory servers such that all servers converge on identical copies of directory data. Replication is designed to let convergence happen over time by default. [1] Letting convergence happen over time means that different replicas can be momentarily out of sync, but it also means that if you lose an individual server or even an entire data

[1] Assured replication can require, however, that the convergence happen before the client application is notified that the operation was successful.

center, your directory service can keep on running, and then get back in sync when the servers are restarted or the network is repaired.

Replication is specific to the OpenDJ directory service. Replication uses a specific protocol that replays update operations quickly, storing enough historical information about the updates to resolve most conflicts automatically. For example, if two client applications separately update a user entry to change the phone number, replication can work out which was the latest change, and apply that change across servers. The historical information needed to resolve these issues is periodically purged to avoid growing larger and larger forever. As a directory administrator, you must ensure that you do not purge the historical information more often than you backup your directory data.

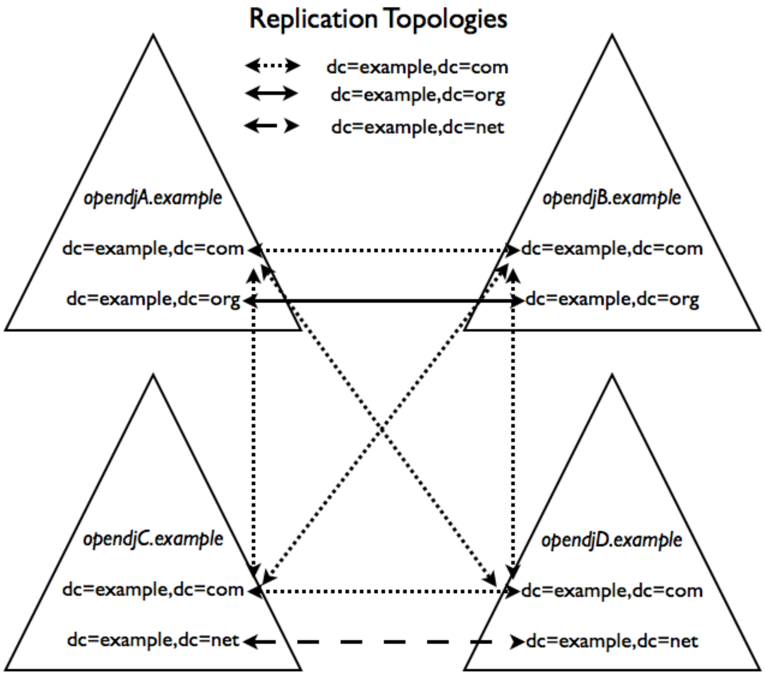
Keep server clocks synchronized for your topology. You can use NTP for example. Keeping server clocks synchronized helps prevent issues with SSL connections and with replication itself. Keeping server clocks synchronized also makes it easier to compare timestamps from multiple servers.

9.2.2 Replication Per Suffix

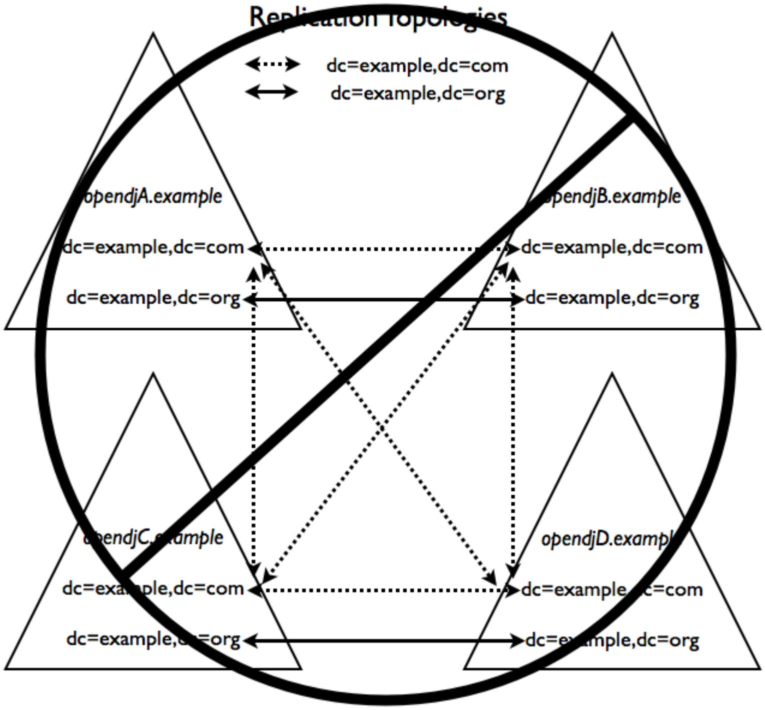
The primary unit of replication is the suffix, specified by a base DN such as `dc=example,dc=com`.[2] Replication also depends on the directory schema, defined on `cn=schema`, and the `cn=admin data` suffix with administrative identities and certificates for protecting communications. Thus that content gets replicated as well.

The set of OpenDJ servers replicating data for a given suffix is called a replication topology. You can have more than one replication topology. For example, one topology could be devoted to `dc=example,dc=com`, and another to `dc=example,dc=org`. OpenDJ servers are capable of serving more than one suffix. They are also capable of participating in more than one replication topology.

[2] When you configure partial and fractional replication, however, you can replicate only part of a suffix, or only certain attributes on entries. Also, if you split your suffix across multiple backends, then you need to set up replication separately for each part of suffix in a different backend.



Within a replication topology, the suffixes being replicated are identified to the replication servers by their DN. As all the replication servers are fully connected in a topology, a consequence is that it is impossible to have multiple "sub-topologies" within the overall set of servers as illustrated in the following diagram.



9.2.3 Replication Connection Selection

In order to understand what happens when individual servers stop responding due to a network partition or a crash, know that OpenDJ can offer both directory service and also replication service, and the two services are not the same, even if they can run alongside each other in the same OpenDJ server in the same Java Virtual Machine.

Replication relies on the replication service provided by OpenDJ replication servers, where OpenDJ directory servers publish changes made to their data, and subscribe to changes published by other OpenDJ directory servers. A replication server manages replication data only, handling replication traffic with directory servers and with other replication servers, receiving, sending, and storing only changes to directory data rather than directory data itself. Once a replication server is connected to a replication topology, it maintains connections to all other replication servers in that topology.

A directory server handles directory data. It responds to requests, stores directory data and historical information. For each replicated suffix, such as `dc=example,dc=com`, `cn=schema` and `cn=admin data`, the directory server publishes changes to a replication server, and subscribes to changes from that replication server. (Directory servers do not publish changes to other directory servers.) A directory server also resolves any conflicts that arise when reconciling changes from other directory servers, using the historical information about changes to resolve the conflicts. (Conflict resolution is the responsibility of the directory server rather than the replication server.)

Once a directory server is connected to a replication topology for a particular suffix, it connects to one replication server at a time for that suffix. The replication server provides the directory server with a list of all replication servers for that suffix. Given the list of possible replication servers to which it can connect, the directory server can determine which replication server to connect to when starting up, or when the current connection is lost or becomes unresponsive.

For each replicated suffix, a directory server prefers to connect to a replication server:

1. In the same group as the directory server

2. Having the same initial data for the suffix as the directory server

3. If initial data were the same, having all the latest changes from the directory server

4. Running in the same Java Virtual Machine as the directory server

5. Having the most available capacity relative to other eligible replication servers

   Available capacity depends on how many directory servers in the topology are already connected to a replication server, and what proportion of all directory servers in the topology ought to be connected to the replication server.

   To determine what proportion of the total number of directory servers should be connected to a replication server, OpenDJ uses replication server weight. When configuring a replication server, you can assign it a weight (default: 1). The weight property takes an integer that indicates capacity to provide replication service relative to other servers. For example, a weight of 2 would indicate a replication server that can handle twice as many connected servers as a replication server with weight 1.

   The proportion of directory servers in a topology that should be connected to a given replication server is equal to (replication server weight)/(sum of replication server weights). In other words, if there are 4 replication servers in a topology each with default weights, the proportion for each replication server is 1/4.

Consider a situation where 7 directory servers are connected to replication servers A, B, C, and D for `dc=example,dc=com` data. Suppose 2 directory servers each are connected to A, B, and C, and 1 directory server is connected to replication server D. Replication server D is therefore the server with the most available capacity relative to other replication servers in the topology. All other criteria being equal, replication server D is the server to connect to when an 8th directory server joins the topology.

The directory server regularly updates the list of replication servers in case it must reconnect. As available capacity of replication servers for each replication topology can change dynamically, a directory server can potentially reconnect to another replication server to balance the replication load in the topology. For this reason the server can also end up connected to different replication servers for different suffixes.

9.3 Configuring Replication

This section shows how to configure replication with command-line tools.

9.3.1 Enabling Replication

You can start the replication process by using the **dsreplication enable** command.

```
$ dsreplication
 enable
 --adminUID admin
 --adminPassword password
 --baseDN dc=example,dc=com
 --host1 opendj.example.com
 --port1 4444
 --bindDN1 "cn=Directory Manager"
 --bindPassword1 password
 --replicationPort1 8989
 --host2 opendj2.example.com
 --port2 4444
 --bindDN2 "cn=Directory Manager"
 --bindPassword2 password
 --replicationPort2 8989
 --trustAll
 --no-prompt

Establishing connections ..... Done.
Checking registration information ..... Done.
Updating remote references on server opendj.example.com:4444 ..... Done.
Configuring Replication port on server opendj2.example.com:4444 ..... Done.
Updating replication configuration for baseDN dc=example,dc=com on server
 opendj.example.com:4444 ..... Done.
Updating replication configuration for baseDN dc=example,dc=com on server
 opendj2.example.com:4444 ..... Done.
Updating registration configuration on server
 opendj.example.com:4444 ..... Done.
Updating registration configuration on server
```

```
 opendj2.example.com:4444 ..... Done.
Updating replication configuration for baseDN cn=schema on server
 opendj.example.com:4444 ..... Done.
Updating replication configuration for baseDN cn=schema on server
 opendj2.example.com:4444 ..... Done.
Initializing registration information on server opendj2.example.com:4444 with
 the contents of server opendj.example.com:4444 ..... Done.
Initializing schema on server opendj2.example.com:4444 with the contents of
 server opendj.example.com:4444 ..... Done.

Replication has been successfully enabled.  Note that for replication to
 work you must initialize the contents of the base DN's that are being
  replicated (use dsreplication initialize to do so).

See
/var/.../opends-replication-7958637258600693490.log
for a detailed log of this operation.
```

To enable secure connections for replication use the `--secureReplication1` and `--secureReplication2` options, which are equivalent to selecting Configure as Secure in the replication topology options screen of the setup wizard.

As you see in the command output, replication is set up to function once enabled. You must however initialize replication in order to start the process.

Tip

When scripting the configuration to set up multiple replicas in quick succession, use the same initial replication server each time you run the command. In other words, pass the same `--host1`, `--port1`, `--bindDN1`, `--bindPassword1`, and `--replicationPort1` options for each of the other replicas that you set up in your script.

If you need to add another OpenDJ directory server to participate in replication, use the **dsreplication enable** with the new server as the second server.

9.3.2 Initializing Replicas

You can initialize replication between servers by performing initialization over the network after you have enabled replication, or by importing the same LDIF data on all servers and then enabling replication. You can also add a new server by restoring a backup from an existing replica onto the new server and then enabling replication with an existing replica.

The alternatives are described step-by-step in the following procedures.

- Procedure 9.1, “To Initialize Replication Over the Network”

- Procedure 9.2, “To Initialize All Servers From the Same LDIF”
- Procedure 9.3, “To Create a New Replica From Existing Backup”

Procedure 9.1. To Initialize Replication Over the Network

Initialization over the network while the server is online works well when you have no initial data, or when your network bandwidth is large compared to the initial amount of data to replicate.

1. Enable replication on all servers.

   See Section 9.3.1, “Enabling Replication” for instructions.

2. Start replication with the **dsreplication initialize-all** command.

```
$ dsreplication
 initialize-all
 --adminUID admin
 --adminPassword password
 --baseDN dc=example,dc=com
 --hostname opendj.example.com
 --port 4444
 --trustAll
 --no-prompt

Initializing base DN dc=example,dc=com with the contents from
 opendj.example.com:4444: 160 entries processed (100 % complete).
Base DN initialized successfully.

See
/var/.../opends-replication-5020375834904394170.log
for a detailed log of this operation.
```

Procedure 9.2. To Initialize All Servers From the Same LDIF

This procedure can be useful when you are starting with a large amount of directory data that is available locally to all directory servers.

1. Import the same LDIF on all servers as described in the procedure, *To Import LDIF Data*.

   Do not yet accept updates to the directory data. Section 9.3.6, “Read-Only Replicas” shows how to prevent replicas from accepting updates from clients.

2. Enable replication for all servers.

   See Section 9.3.1, “Enabling Replication” for instructions.

3. Allow updates to the directory data by setting `writability-mode:enabled` using a command like the one you found in Section 9.3.6, “Read-Only Replicas”.

Procedure 9.3. To Create a New Replica From Existing Backup

You can create a new replica from a backup of a server in the existing topology.

1. Install a new server to use as the new replica.

2. Backup the database on an existing server as described in *Backing Up Directory Data*.

   At this point, other servers in the topology can continue to process updates.

3. Enable replication on the new replica.

```
$ dsreplication
 enable
 --adminUID admin
 --adminPassword password
 --baseDN dc=example,dc=com
 --host1 opendj.example.com
 --port1 4444
 --bindDN1 "cn=Directory Manager"
 --bindPassword1 password
 --replicationPort1 8989
 --host2 opendj3.example.com
 --port2 4444
 --bindDN2 "cn=Directory Manager"
 --bindPassword2 password
 --replicationPort2 8989
 --trustAll
 --no-prompt

Establishing connections ..... Done.
Checking registration information ..... Done.
Updating remote references on server opendj.example.com:4444 ..... Done.
Configuring Replication port on server opendj3.example.com:4444 ..... Done.
Updating replication configuration for baseDN dc=example,dc=com on server
 opendj.example.com:4444 ..... Done.
Updating replication configuration for baseDN dc=example,dc=com on server
 opendj3.example.com:4444 ..... Done.
Updating replication configuration for baseDN dc=example,dc=com on server
 opendj2.example.com:4444 ..... Done.
Updating remote references on server opendj2.example.com:4444 ..... Done.
Updating registration configuration on server
 opendj.example.com:4444 ..... Done.
Updating registration configuration on server
 opendj3.example.com:4444 ..... Done.
Updating registration configuration on server
 opendj2.example.com:4444 ..... Done.
Updating replication configuration for baseDN cn=schema on server
 opendj.example.com:4444 ..... Done.
Updating replication configuration for baseDN cn=schema on server
 opendj3.example.com:4444 ..... Done.
Updating replication configuration for baseDN cn=schema on server
 opendj2.example.com:4444 ..... Done.
Initializing registration information on server opendj3.example.com:4444 with
 the contents of server opendj.example.com:4444 ..... Done.
```

```
Replication has been successfully enabled.  Note that for replication to
 work you must initialize the contents of the base DN's that are being
 replicated (use dsreplication initialize to do so).

See
/var/.../opends-replication-1672058070147419978.log
for a detailed log of this operation.
```

Contrary to the message from the command, you do not need to use the **dsreplication initialize** command at this point.

4. On the new server, restore the database from the backup archive as described in the procedure, *To Restore a Replica*.

   As long as you restore the database on the new replica before the replication purge delay runs out, updates processed by other servers after you created the backup are replicated to the new server after you restore the data.

9.3.3 Stopping Replication

How you stop replication depends on whether the change is meant to be temporary or permanent.

Procedure 9.4. To Stop Replication Temporarily For a Replica

If you need to stop a server from replicating temporarily, you can do so using **dsconfig** command.

Warning

Do not allow modifications on the replica for which replication is disabled, as no record of such changes is kept, and the changes cause replication to diverge.

1. Disable the multimaster synchronization provider.

```
$ dsconfig
 set-synchronization-provider-prop
 --port 4444
 --hostname opendj2.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --set enabled:false
 --trustAll
 --no-prompt
```

2. (Optional) When you are ready to resume replication, enable the multimaster synchronization provider.

```
$ dsconfig
 set-synchronization-provider-prop
 --port 4444
 --hostname opendj2.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --set enabled:true
 --trustAll
 --no-prompt
```

Procedure 9.5. To Stop Replication Permanently For a Replica

If you need to stop a server from replicating permanently, for example in preparation to remove a server, you can do so with the **dsreplication disable** command.

1. Stop replication using the **dsreplication disable** command.

```
$ dsreplication
 disable
 --disableAll
 --port 4444
 --hostname opendj2.example.com
 --bindDN "cn=Directory Manager"
 --adminPassword password
 --trustAll
 --no-prompt
Establishing connections ..... Done.
Disabling replication on base DN cn=admin data of server
 opendj2.example.com:4444 ..... Done.
Disabling replication on base DN dc=example,dc=com of server
 opendj2.example.com:4444 ..... Done.
Disabling replication on base DN cn=schema of server
 opendj2.example.com:4444 ..... Done.
Disabling replication port 8989 of server
 opendj2.example.com:4444 ..... Done.
Removing registration information ..... Done.
Removing truststore information ..... Done.

See
/var/.../opends-replication-125248191132797765.log
for a detailed log of this operation.
```

   The **dsreplication disable** as shown completely removes the replication configuration information from the server.

2. (Optional) If you want to restart replication for the server, you need to run the **dsreplication enable** and **dsreplication initialize** commands again.

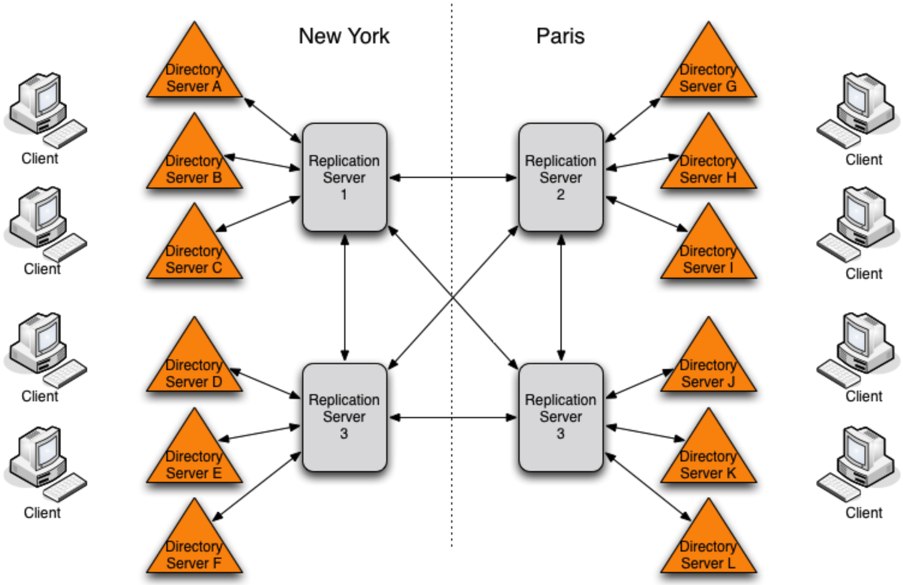
9.3.4 Stand-alone Replication Servers

Replication in OpenDJ is designed to be both easy to implement in environments with a few servers, and also scalable in environments with many servers. You can enable the replication service on each OpenDJ directory server in your deployment, for example, to limit the number of servers you deploy. Yet in a large deployment, you can use stand-alone replication servers — OpenDJ servers that do nothing but relay replication messages — to configure (and troubleshoot) the replication service separately from the directory service. You only need a few stand-alone replication servers publishing changes to serve many directory servers subscribed to the changes. Furthermore, replication is designed such that you need only connect a directory server to the nearest replication server for the directory server to replicate with all others in your topology. Yet only the stand-alone replication servers participate in fully-meshed replication.

All replication servers in a topology are connected to all other replication servers. Directory servers are connected only to one replication server at a time, and their connections should be to replication servers on the same LAN. Therefore the total number of replication connections, $\mathrm{Total}_{\mathrm{conn}}$ is expressed as follows.

$$\mathrm{Total}_{\mathrm{conn}} = (\mathrm{N}_{\mathrm{RS}} * \mathrm{N}_{\mathrm{RS}}\text{-}1)/2 + \mathrm{N}_{\mathrm{DS}} \tag{9.1}$$

Here, N_{RS} is the number of replication servers, and N_{DS} is the number of stand-alone directory servers. In other words, if you have only 3 servers, then $\mathrm{Total}_{\mathrm{conn}}$ is 3 with no stand-alone servers. However, if you have two data centers, and need 12 directory servers, then with no stand-alone directory servers $\mathrm{Total}_{\mathrm{conn}}$ is (12 * 11)/2 or 66. Yet, with 4 stand-alone replication servers, and 12 stand-alone directory servers, $\mathrm{Total}_{\mathrm{conn}}$ is (4 * 3)/2 + 12, or 18, with only four of those connections needing to go over the WAN. (By running four directory servers that also run replication servers and eight stand-alone directory servers, you reduce the number of replication connections to 14 for 12 replicas.)



Tip

If you set up OpenDJ directory server to replicate by using the Quick Setup wizard, then the wizard activated the replication service for that server. You can turn off the replication service on OpenDJ directory server, and then configure the server to work with a separate, stand-alone replication server instead. Start by using the **dsreplication disable --disableReplicationServer** command to turn off the replication service on the server.

Procedure 9.6. To Set Up a Stand-alone Replication Server

This example sets up a stand-alone replication server to handle the replication traffic between two directory servers that do not handle replication themselves.

Here the replication server is `rs.example.com`. The directory servers are `opendj.example.com` and `opendj2.example.com`.

In a real deployment, you would have more replication servers to avoid a single point of failure.

1. Setup the replication server as a directory server that has no database.

2. Setup the directory servers as stand-alone directory servers.

3. Enable replication with the appropriate `--noReplicationServer` and `--onlyReplicationServer` options.

```
$ dsreplication
 enable
 --adminUID admin
 --adminPassword password
 --baseDN dc=example,dc=com
 --host1 opendj.example.com
 --port1 4444
 --bindDN1 "cn=Directory Manager"
 --bindPassword1 password
 --noReplicationServer1
 --host2 rs.example.com
 --port2 4444
 --bindDN2 "cn=Directory Manager"
 --bindPassword2 password
 --replicationPort2 8989
 --onlyReplicationServer2
 --trustAll
 --no-prompt
Establishing connections ..... Done.
Only one replication server will be defined for the following base DN's:
dc=example,dc=com
It is recommended to have at least two replication servers (two changelogs) to
avoid a single point of failure in the replication topology.

Checking registration information ..... Done.
Configuring Replication port on server rs.example.com:4444 ..... Done.
Updating replication configuration for baseDN dc=example,dc=com on server
 opendj.example.com:4444 ..... Done.
Updating registration configuration on server
 opendj.example.com:4444 ..... Done.
Updating registration configuration on server
 rs.example.com:4444 ..... Done.
Updating replication configuration for baseDN cn=schema on server
 opendj.example.com:4444 ..... Done.
Initializing registration information on server rs.example.com:4444 with
 the contents of server opendj.example.com:4444 ..... Done.

Replication has been successfully enabled.  Note that for replication to work
 you must initialize the contents of the base DN's that are being
 replicated (use dsreplication initialize to do so).

See
/var/.../opends-replication-1720959352638609971.log
for a detailed log of this operation.

$ dsreplication
 enable
 --adminUID admin
 --adminPassword password
 --baseDN dc=example,dc=com
 --host1 opendj2.example.com
 --port1 4444
 --bindDN1 "cn=Directory Manager"
 --bindPassword1 password
 --noReplicationServer1
 --host2 rs.example.com
 --port2 4444
 --bindDN2 "cn=Directory Manager"
 --bindPassword2 password
 --replicationPort2 8989
 --onlyReplicationServer2
 --trustAll
 --no-prompt

Establishing connections ..... Done.
Only one replication server will be defined for the following base DN's:
dc=example,dc=com
It is recommended to have at least two replication servers (two changelogs) to
avoid a single point of failure in the replication topology.

Checking registration information ..... Done.
Updating remote references on server rs.example.com:4444 ..... Done.
Updating replication configuration for baseDN dc=example,dc=com on server
 opendj2.example.com:4444 ..... Done.
Updating replication configuration for baseDN dc=example,dc=com on server
 opendj.example.com:4444 ..... Done.
Updating registration configuration on server
 opendj2.example.com:4444 ..... Done.
Updating registration configuration on server
 rs.example.com:4444 ..... Done.
Updating registration configuration on server
 opendj.example.com:4444 ..... Done.
Updating replication configuration for baseDN cn=schema on server
 opendj2.example.com:4444 ..... Done.
Updating replication configuration for baseDN cn=schema on server
 opendj.example.com:4444 ..... Done.
Initializing registration information on server opendj2.example.com:4444 with
 the contents of server rs.example.com:4444 ..... Done.

Replication has been successfully enabled.  Note that for replication to work
 you must initialize the contents of the base DN's that are being
 replicated (use dsreplication initialize to do so).

See
/var/folders/.../opends-replication-5893037538856033562.log
for a detailed log of this operation.
```

4. Initialize replication from one of the directory servers.

```
$ dsreplication
 initialize-all
 --adminUID admin
 --adminPassword password
 --baseDN dc=example,dc=com
 --hostname opendj.example.com
 --port 4444
 --trustAll
 --no-prompt

Initializing base DN dc=example,dc=com with the contents from
 opendj.example.com:4444: 160 entries processed (100 % complete).
Base DN initialized successfully.

See
/var/.../opends-replication-7677303986403997574.log
for a detailed log of this operation.
```

9.3.5 Replication Groups

Replication lets you define groups so that replicas communicate first with replication servers in the group before going to replication servers outside the group. Groups are identified with unique numeric group IDs.

Replication groups are designed for deployments across multiple data centers, where you aim to focus replication traffic on the LAN rather than the WAN. In multi-data center deployments, group nearby servers together.

Procedure 9.7. To Set Up Replication Groups

For each group, set the appropriate group ID for the topology on both the replication servers and the directory servers.

The example commands in this procedure set up two replication groups, each with a replication server and a directory server. The directory servers are `opendj.example.com` and `opendj2.example.com`. The replication servers are `rs.example.com` and `rs2.example.com`. In a full-scale deployment, you would have multiple servers of each type in each group, such as all the replicas and replication servers in each data center being in the same group.

1. Pick a group ID for each group.

   The default group ID is 1.

2. Set the group ID for each group by replication domain on the directory servers.

```
$ dsconfig
 set-replication-domain-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --domain-name "dc=example,dc=com"
 --set group-id:1
 --trustAll
 --no-prompt

$ dsconfig
 set-replication-domain-prop
 --port 4444
 --hostname opendj2.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --domain-name "dc=example,dc=com"
 --set group-id:2
 --trustAll
 --no-prompt
```

3. Set the group ID for each group on the replication servers.

```
$ dsconfig
 set-replication-server-prop
 --port 4444
 --hostname rs.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --set group-id:1
 --trustAll
 --no-prompt
$ dsconfig
 set-replication-server-prop
 --port 4444
 --hostname rs2.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --set group-id:2
 --trustAll
 --no-prompt
```

9.3.6 Read-Only Replicas

By default all directory servers in a replication topology are read-write. You can however choose to make replicas take updates only from the replication protocol, and refuse updates from client applications.

```
$ dsconfig
 set-global-configuration-prop
 --port 4444
 --hostname opendj2.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set writability-mode:internal-only
 --trustAll
 --no-prompt
```

9.3.7 Assured Replication

In standard replication, when a client requests an update operation the directory server performs the update and, if the update is successful, sends information about the update to the replication service, and sends a result code to the client application right away. As a result, the client application can conclude that the update was successful, *but only on the replica that handled the update*.

Assured replication lets you force the replica performing the initial update to wait for confirmation that the update has been received elsewhere in the topology before sending a result code to the client application. You can configure assured replication either to wait for one or more replication servers to acknowledge having received the update, or to wait for all directory servers to have replayed the update.

As you might imagine, assured replication is theoretically safer than standard replication, yet it is also slower, potentially waiting for a timeout before failing when the network or other servers are down.

Procedure 9.8. To Ensure Updates Reach Replication Servers

Safe data mode requires the update be sent to `assured-sd-level` replication servers before acknowledgement is returned to the client application.

- For each directory server, set safe data mode for the replication domain, and also set the safe data level.

```
$ dsconfig
 set-replication-domain-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --domain-name "dc=example,dc=com"
 --set assured-type:safe-data
 --set assured-sd-level:1
 --trustAll
 --no-prompt

$ dsconfig
 set-replication-domain-prop
 --port 4444
 --hostname opendj2.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --domain-name "dc=example,dc=com"
 --set assured-type:safe-data
 --set assured-sd-level:1
 --trustAll
 --no-prompt
```

Procedure 9.9. To Ensure Updates Are Replayed Everywhere

Safe read mode requires the update be replayed on all directory servers before acknowledgement is returned to the client application.

- For each directory server, set safe read mode for the replication domain.

```
$ dsconfig
 set-replication-domain-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --domain-name "dc=example,dc=com"
 --set assured-type:safe-read
 --trustAll
 --no-prompt

$ dsconfig
 set-replication-domain-prop
 --port 4444
 --hostname opendj2.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --domain-name "dc=example,dc=com"
 --set assured-type:safe-read
 --trustAll
 --no-prompt
```

When working with assured replication, the replication server property degraded-status-threshold (default: 5000), sets the number of operations allowed to build up in the replication queue before the server is assigned degraded status. When a replication server has degraded status, assured replication ceases to have an effect.

9.3.8 Subtree Replication

OpenDJ can perform subtree replication, for example replicating ou=People, dc=example,dc=com, but not the rest of dc=example,dc=com, by putting the subtree in a separate backend from the rest of the suffix.

For example, in this case you might have a userRoot backend containing everything in dc=example,dc=com except ou=People,dc=example,dc=com, and a separate peopleRoot backend for ou=People,dc=example,dc=com. Then you replicate ou=People,dc=example,dc=com in its own topology.

9.3.9 Fractional Replication

OpenDJ can perform fractional replication, whereby you specify the attributes to include in or to exclude from the replication process.

You set fractional replication configuration as fractional-include or fractional-exclude properties for a replication domain. When you include attributes, the attributes that are required on the relevant object classes are also included, whether you specify them or not. When you exclude attributes, the excluded attributes must be optional attributes for the relevant object classes. Fractional replicas still respect schema definitions.

Fractional replication works by filtering objects at the replication server. Initialize replication as you would normally. Of course you cannot create a full replica from a replica with only a subset of the data. If you must prevent data from being replicated across a national boundary, split the replication server handling the updates from the directory servers receiving the updates as described in Procedure 9.6, “To Set Up a Stand-alone Replication Server”.

For example, you might configure an externally facing fractional replica to include only some inetOrgPerson attributes.

```
$ dsconfig
 set-replication-domain-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --domain-name "dc=example,dc=com"
 --trustAll
 --no-prompt
 --set
 fractional-include:inetorgperson:cn,givenname,mail,mobile,sn,telephonenumber
```

As another example, you might exclude a custom attribute called `sessionToken` from being replicated.

```
$ dsconfig
 set-replication-domain-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --domain-name "dc=example,dc=com"
 --set fractional-exclude:*:sessionToken
 --trustAll
 --no-prompt
```

This last example only works if you first define a `sessionToken` attribute in the directory server schema.

9.4 Change Notification For Your Applications

Some applications require notification when directory data updates occur. For example, an application might need to sync directory data with another database, or the application might need to kick off other processing when certain updates occur.

In addition to supporting persistent search operations, OpenDJ provides an external change log mechanism to allow applications to be notified of changes to directory data.

Procedure 9.10. To Enable the External Change Log

OpenDJ directory servers without replication cannot expose an external change log. The OpenDJ server that exposes the change log must function both as a directory server, and also as a replication server for the suffix whose changes you want logged.

- Enable replication without using the `--noReplicationServer` or `--onlyReplicationServer` options.

With replication enabled, the changelog data can be accessed under `cn=changelog`. For example, the following search shows the publicly visible data available before any changes have been made.

```
$ ldapsearch --baseDN cn=changelog --port 1389 "(objectclass=*)" \* +
dn: cn=changelog
cn: changelog
objectClass: top
objectClass: container
subschemaSubentry: cn=schema
hasSubordinates: false
entryDN: cn=changelog
```

Procedure 9.11. To Use the External Change Log

You read the external change log over LDAP. In addition, when you poll the change log periodically, you can get the list of updates that happened since your last request.

The external change log mechanism uses an LDAP control with OID `1.3.6.1.4.1.26027.1.5.4` to allow the exchange of cookies for the client application to bookmark the last changes seen, and then start reading the next set of changes from where it left off on the previous request.

This procedure shows the client reading the change log as `cn=Directory Manager`. Make sure your client application reads the changes with sufficient access to view all the changes it needs to see.

1. Send an initial search request using the LDAP control with no cookie value.

   Notice the value of the `changeLogCookie` attribute for the last of the two changes.

```
$ ldapsearch
 --baseDN cn=changelog
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --control "1.3.6.1.4.1.26027.1.5.4:false"
 "(objectclass=*)"
 \* +
dn: cn=changelog
cn: changelog
objectClass: top
objectClass: container
subschemaSubentry: cn=schema
hasSubordinates: true
entryDN: cn=changelog

# Public changelog exchange control(1.3.6.1.4.1.26027.1.5.4):
 dc=example,dc=com:0000013087cbc28212d100000001;
dn: replicationCSN=0000013087cbc28212d100000001,dc=example,dc=com,cn=changelog
targetDN: cn=arsene lupin,ou=special users,dc=example,dc=com
changeNumber: 0
```

```
changes:: b2JqZWN0Q2xhc3M6IHBlcnNvbgpvYmplY3RDbGFzczogdG9wCmNuOiBBcnNlbmUgTHVwaW
 4KdGVsZXBob25lTnVtYmVyOiArMzMgMSAyMyA0NSA2NyA4OQpzbjogTHVwaW4KZW50cnlVVUlEOiA5M
 GM3MTRmNy00ODZiLTRkNDctOTQwOS1iNDRkMTlkZWEzMWUKY3JlYXRlVGltZXN0YW1wOiAyMDExMDYx
 MzA2NTg1NVoKY3JlYXRvcnNOYW1lOiBjbj1EaXJlY3RvcnkgTWFuYWdlcixjbj1Sb290IEROcyxjbj1
 jb25maWcK
changeType: add
changeTime: 20110613065855Z
objectClass: top
objectClass: changeLogEntry
targetEntryUUID: 90c714f7-486b-4d47-9409-b44d19dea31e
replicationCSN: 0000013087cbc28212d100000001
numSubordinates: 0
replicaIdentifier: 4817
changeLogCookie: dc=example,dc=com:0000013087cbc28212d100000001;
changeInitiatorsName: cn=Directory Manager,cn=Root DNs,cn=config
subschemaSubentry: cn=schema
hasSubordinates: false
entryDN: replicationCSN=0000013087cbc28212d100000001,dc=example,dc=com,cn=change
 log

# Public changelog exchange control(1.3.6.1.4.1.26027.1.5.4):
 dc=example,dc=com:0000013087cbc34a12d100000002;
dn: replicationCSN=0000013087cbc34a12d100000002,dc=example,dc=com,cn=changelog
targetDN: cn=horace velmont,ou=special users,dc=example,dc=com
changeNumber: 0
changes:: b2JqZWN0Q2xhc3M6IHBlcnNvbgpvYmplY3RDbGFzczogdG9wCmNuOiBIb3JhY2UgVmVsbW
 9udAp0ZWxlcGhvbmVOdW1iZXI6ICszMyAxIDEyIDIzIDM0IDQ1CnNuOiBWZWxtb250CmVudHJ5VVVJR
 DogNmIyMjQ0MGEtNzZkMC00MDMxLTk0YjctMzViMWQ4NmYwNjdlCmNyZWF0ZVRpbWVzdGFtcDogMjAx
 MTA2MTMwNjU4NTVaCmNyZWF0b3JzTmFtZTogY249RGlyZWN0b3J5IE1hbmFnZXIsY249Um9vdCBETnM
 sY249Y29uZmlnCg==
changeType: add
changeTime: 20110613065855Z
objectClass: top
objectClass: changeLogEntry
targetEntryUUID: 6b22440a-76d0-4031-94b7-35b1d86f067e
replicationCSN: 0000013087cbc34a12d100000002
numSubordinates: 0
replicaIdentifier: 4817
changeLogCookie: dc=example,dc=com:0000013087cbc34a12d100000002;
changeInitiatorsName: cn=Directory Manager,cn=Root DNs,cn=config
subschemaSubentry: cn=schema
hasSubordinates: false
entryDN: replicationCSN=0000013087cbc34a12d100000002,dc=example,dc=com,cn=change
 log
```

In this example, two new users were added to another replica before the change log request was made.

Here the changes are base64 encoded, so you can decode them using the **base64** command.

```
$ base64 decode --encodedData b2JqZW...ZmlnCg==
objectClass: person
objectClass: top
cn: Horace Velmont
telephoneNumber: +33 1 12 23 34 45
sn: Velmont
entryUUID: 6b22440a-76d0-4031-94b7-35b1d86f067e
createTimestamp: 20110613065855Z
creatorsName: cn=Directory Manager,cn=Root DNs,cn=config

```

2. For the next search, provide the cookie to start reading where you left off last time.

   In this example, a description was added to Babs Jensen's entry.

```
$ ldapsearch
 --baseDN cn=changelog
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --control "1.3.6.1.4.1.26027.1.5.4:false:
 dc=example,dc=com:0000013087cbc34a12d100000002;"
 "(objectclass=*)"
 \* +
dn: cn=changelog
cn: changelog
objectClass: top
objectClass: container
subschemaSubentry: cn=schema
hasSubordinates: true
entryDN: cn=changelog

# Public changelog exchange control(1.3.6.1.4.1.26027.1.5.4):
 dc=example,dc=com:0000013087d7e27f12d100000003;
dn: replicationCSN=0000013087d7e27f12d100000003,dc=example,dc=com,cn=changelog
targetDN: uid=bjensen,ou=people,dc=example,dc=com
changeNumber: 0
changes:: YWRkOiBkZXNjcmlwdGlvbgpkZXNjcmlwdGlvbjogQSB0aGlyZCBjaGFuZ2UKLQpyZXBsYW
 NlOiBtb2RpZmllcnNOYW1lCm1vZGlmaWVyc05hbWU6IGNuPURpcmVjdG9yeSBNYW5hZ2VyLGNuPVJvb
 3QgRE5zLGNuPWNvbmZpZwotCnJlcGxhY2U6IG1vZGlmeVRpbWVzdGFtcAptb2RpZnlUaW1lc3RhbXA6
 IDIwMTEwNjEzMDcxMjEwWgotCg==
changeType: modify
changeTime: 20110613071210Z
objectClass: top
objectClass: changeLogEntry
targetEntryUUID: fc252fd9-b982-3ed6-b42a-c76d2546312c
replicationCSN: 0000013087d7e27f12d100000003
numSubordinates: 0
replicaIdentifier: 4817
changeLogCookie: dc=example,dc=com:0000013087d7e27f12d100000003;
changeInitiatorsName: cn=Directory Manager,cn=Root DNs,cn=config
subschemaSubentry: cn=schema
hasSubordinates: false
entryDN: replicationCSN=0000013087d7e27f12d100000003,dc=example,dc=com,cn=change
 log

```

If we base64-decode the changes, we see the following.

```
$ base64 decode --encodedData YWRkO...gotCg==
add: description
description: A third change
-
replace: modifiersName
modifiersName: cn=Directory Manager,cn=Root DNs,cn=config
-
replace: modifyTimestamp
modifyTimestamp: 20110613071210Z
-
```

3. If for some reason you lose the cookie, you can start over from the earliest available change by sending a search request with no value for the cookie.

Procedure 9.12. To Include Unchanged Attributes in the External Change Log

As shown above, the changes returned from a search on the external change log include only what was actually changed. If you have applications that need additional attributes published with every change log entry, regardless of whether or not the attribute itself has changed, then specify those using `ecl-include` and `ecl-include-for-deletes`.

1. Set the attributes to include for all update operations with `ecl-include`.

```
$ dsconfig
 set-external-changelog-domain-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --domain-name dc=example,dc=com
 --set ecl-include:"@person"
 --trustAll
 --no-prompt
```

2. Set the attributes to include for deletes with `ecl-include-for-deletes`.

```
$ dsconfig
 set-external-changelog-domain-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --domain-name dc=example,dc=com
 --add ecl-include-for-deletes:"*"
 --add ecl-include-for-deletes:"+"
 --trustAll
 --no-prompt
```

Procedure 9.13. To Limit External Change Log Content

You can limit external change log content by disabling the domain for a base DN. By default, `cn=schema` and `cn=admin data` are not enabled.

- Prevent OpenDJ from logging changes by disabling the domain.

```
$ dsconfig
 set-external-changelog-domain-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --domain-name dc=example,dc=com
 --set enabled:false
 --trustAll
 --no-prompt
```

The external change log can also work for applications that follow the Internet-Draft: Definition of an Object Class to Hold LDAP Change Records. Nothing special is required to get the objects specified for this legacy format. Such applications cannot however use the change log cookies that are shared across the replication topology, and therefore can continue to be used after failover to another replica in a multi-master replication environment.

Chapter 10

Backing Up & Restoring Data

OpenDJ lets you backup and restore your data either in compressed, binary format, or in LDAP Data Interchange Format. This chapter shows you how to backup and to restore OpenDJ data from archives, and explains portability of backup archives, as well as backing up server configuration information.

10.1 Backing Up Directory Data

A `bak/` directory is provided when you install OpenDJ, as a location to save binary backups. When you create a backup, the `bak/backup.info` contains information about the archive.

Archives produced by the **backup** command contain backups only of the directory data. Backups of server configuration are found in `config/archived-configs/`.

Procedure 10.1. To Back Up Data Immediately

To perform online backup, you start backup as a task by connecting to the administrative port and authenticating as a user with the `backend-backup` privilege, and also setting a start time for the task by using the `--start` option.

To perform offline backup when OpenDJ is stopped, you run the **backup** command without connecting to the server, authenticating, or requesting a backup task.

- Use one of the following alternatives.

- Back up only the database for Example.com, where the data is stored in the backend named `userRoot`.

  The following example requests an online backup task that starts immediately, backing up only the `userRoot` backend.

  ```
  $ backup
   --port 4444
   --bindDN "cn=Directory Manager"
   --bindPassword password
   --backendID userRoot
   --backupDirectory /path/to/opendj/bak
   --start 0
  Backup task 20110613143715983 scheduled to start Jun 13, 2011 2:37:15 PM CEST
  ```

- Stop the server to back up Example.com data offline.

  The following example stops OpenDJ, runs offline backup, and starts the server after backup has completed.

  ```
  $ stop-ds
  Stopping Server...

  [13/Jun/2011:14:31:00 +0200] category=BACKEND severity=NOTICE msgID=9896306
   msg=The backend userRoot is now taken offline
  [13/Jun/2011:14:31:00 +0200] category=CORE severity=NOTICE msgID=458955
   msg=The Directory Server is now stopped
  $ backup --backendID userRoot -d /path/to/opendj/bak
  [13/Jun/2011:14:33:48 +0200] category=TOOLS severity=NOTICE msgID=10944792
   msg=Starting backup for backend userRoot
  [13/Jun/2011:14:33:48 +0200] category=JEB severity=NOTICE msgID=8847446
   msg=Archived: 00000000.jdb
  [13/Jun/2011:14:33:48 +0200] category=TOOLS severity=NOTICE msgID=10944795
   msg=The backup process completed successfully
  $ start-ds
  ... The Directory Server has started successfully
  ```

- Back up all user data on the server.

  The following example requests an online backup task that starts immediately, backing up all backends.

  ```
  $ backup
   --port 4444
   --bindDN "cn=Directory Manager"
   --bindPassword password
   --backUpAll
   --backupDirectory /path/to/opendj/bak
   --start 0
  Backup task 20110613143801866 scheduled to start Jun 13, 2011 2:38:01 PM CEST
  ```

Procedure 10.2. To Schedule Data Backup

You can schedule online data backup using **crontab** format.

- Back up all user data every night at 2 AM, and notify diradmin@example.com when finished, or on error.

```
$ backup
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --backUpAll
 --backupDirectory /path/to/opendj/bak
 --recurringTask "00 02 * * *"
 --completionNotify diradmin@example.com
 --errorNotify diradmin@example.com
Recurring Backup task BackupTask-988d6adf-4d65-44bf-8546-6ea74a2480b0
scheduled successfully
```

10.2 Restoring Directory Data From Backup

When you restore data, the procedure to follow depends on whether the OpenDJ directory server is replicated.

Procedure 10.3. To Restore a Stand-alone Server

To restore OpenDJ when the server is online, you start a restore task by connecting to the administrative port and authenticating as a user with the `backend-restore` privilege, and also setting a start time for the task by using the `--start` option.

To restore data when OpenDJ is stopped, you run the **restore** command without connecting to the server, authenticating, or requesting a restore task.

- Use one of the following alternatives.

    - Stop the server to restore data for Example.com.

      The following example stops OpenDJ, restores data offline from one of the available backups, and then starts the server after the restore is complete.

```
$ stop-ds
Stopping Server...

[13/Jun/2011:15:44:06 +0200] category=BACKEND severity=NOTICE msgID=9896306
 msg=The backend userRoot is now taken offline
[13/Jun/2011:15:44:06 +0200] category=CORE severity=NOTICE msgID=458955
 msg=The Directory Server is now stopped
$ restore --backupDirectory /path/to/opendj/bak --listBackups
Backup ID:          20110613080032
Backup Date:        13/Jun/2011:08:00:45 +0200
Is Incremental:     false
Is Compressed:      false
Is Encrypted:       false
Has Unsigned Hash:  false
Has Signed Hash:    false
Dependent Upon:     none
$ restore --backupDirectory /path/to/opendj/bak --backupID 20110613080032
[13/Jun/2011:15:47:41 +0200] category=JEB severity=NOTICE msgID=8847445
 msg=Restored: 00000000.jdb (size 341835)
$ start-ds
... The Directory Server has started successfully
```

- Schedule the restore as a task to begin immediately.

  The following example requests an online restore task, scheduled to start immediately.

```
$ restore
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --backupDirectory /path/to/opendj/bak
 --backupID 20110613080032
 --start 0
Restore task 20110613155052932 scheduled to start Jun 13, 2011 3:50:52 PM CEST
```

Procedure 10.4. To Restore a Replica

After you restore a replica from backup, replication brings the replica up to date with changes that happened after you created the backup. In order to bring the replica up to date, replication must apply changes that happened after the backup was made. Replication uses internal change log records to determine what changes to apply.

Internal change log records are not kept forever, though. Replication is configured to purge the change log of old changes, preventing the log from growing indefinitely. Yet, for replication to determine what changes to apply to a restored replica, it must find change log records dating back at least to the last change in the backup. In other words, replication can bring the restored replica up to date *as long as the change log records used to determine which changes to apply have not been purged*.

Therefore, when you restore a replicated server from backup, make sure the backup you use is newer than the last purge of the replication change log (default: 3 days). If all your backups are older than the replication purge delay, do not restore from a backup, but instead initialize a new replica as described in *Initializing Replicas*.

- Restore the server database from the backup archive that you are sure is newer than the last purge of the replication change log.

```
$ stop-ds
Stopping Server...

[13/Jun/2011:15:44:06 +0200] category=BACKEND severity=NOTICE msgID=9896306
 msg=The backend userRoot is now taken offline
[13/Jun/2011:15:44:06 +0200] category=CORE severity=NOTICE msgID=458955
 msg=The Directory Server is now stopped
$ restore --backupDirectory /path/to/opendj/bak --listBackups
Backup ID:          20110613080032
Backup Date:        13/Jun/2011:08:00:45 +0200
Is Incremental:     false
Is Compressed:      false
Is Encrypted:       false
Has Unsigned Hash:  false
Has Signed Hash:    false
Dependent Upon:     none
$ restore --backupDirectory /path/to/opendj/bak --backupID 20110613080032
[13/Jun/2011:15:47:41 +0200] category=JEB severity=NOTICE msgID=8847445
 msg=Restored: 00000000.jdb (size 341835)
$ start-ds
... The Directory Server has started successfully
```

Chapter 11

Configuring Password Policy

If you want to synchronize password policy across your organization and your applications go to the directory for authentication, then the directory can be a good place to enforce your password policy uniformly. Even if you do not depend on the directory for all your password policy, you no doubt still want to consider directory password policy if only to choose the appropriate password storage scheme.

This chapter covers password policy, including examples of how to configure password policies for common use cases.

11.1 About OpenDJ Password Policies

OpenDJ password policies govern not only passwords, but also account lockout, and how OpenDJ provides notification about account status.

OpenDJ supports password policies as part of the server configuration, and also subentry password policies as part of the (replicated) user data.

11.1.1 Server Based Password Policies

You manage server based password policies in the OpenDJ configuration by using the **dsconfig** command. As they are part of the server configuration, such password policies are not replicated. You must instead apply password policy configuration updates to each replica in your deployment.

By default, OpenDJ includes two password policy configurations, one default for all users, and another for directory root DN users, such as `cn=Directory Manager`. You can see all the default password policy settings using the **dsconfig** command as follows.

```
$ dsconfig
 get-password-policy-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --policy-name "Default Password Policy"
 --advanced
Property                                  : Value(s)
------------------------------------------:--------------------------
account-status-notification-handler       : -
allow-expired-password-changes            : false
allow-multiple-password-values            : false
allow-pre-encoded-passwords               : 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
skip-validation-for-administrators        : false
state-update-failure-policy               : reactive
```

See the *OpenDJ Configuration Reference* page on *Password Policy* for detailed descriptions of each property.

Here you notice that many capabilities are not set by default: no lockout, no password expiration, no multiple passwords, no password validator to check that passwords contain the appropriate mix of characters. This means that if you decide to use the directory to enforce password policy, you must configure at least the default password policy to meet your needs.

Yet a few basic protections are configured by default. When you import LDIF with `userPassword` values, OpenDJ hashes the values before storing them. When a user provides a password value during a bind for example, the server hashes the value provided to compared it with the stored value. Even the directory manager cannot see the plain text value of a user's password.

```
$ ldapsearch
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --baseDN dc=example,dc=com
 uid=bjensen
 userpassword
dn: uid=bjensen,ou=People,dc=example,dc=com
userpassword: {SSHA}QWAtw8ch/9850HNFRRqLNMIQc1YhxCnOoGmk1g==
```

In addition, users can change their passwords provided you have granted them access to do so. OpenDJ uses the `userPassword` attribute to store passwords by default, rather than the `authPassword` attribute, which is designed to store passwords hashed by the client application.

11.1.2 Subentry Based Password Policies

You manage subentry password policies by adding the subentries alongside the user data. Thus OpenDJ can replicate subentry password policies across servers.

Subentry password policies support the Internet-Draft Password Policy for LDAP Directories (version 09). A subentry password policy effectively overrides settings in the default password policy defined in the OpenDJ configuration. Settings not supported or not included in the subentry password policy are thus inherited from the default password policy.

As a result, the following Internet-Draft password policy attributes override the default password policy when you set them in the subentry.

- `pwdAllowUserChange`, corresponding to the OpenDJ password policy property `allow-user-password-changes`
- `pwdMustChange`, corresponding to the OpenDJ password policy property `force-change-on-reset`
- `pwdGraceAuthNLimit`, corresponding to the OpenDJ password policy property `grace-login-count`
- `pwdLockoutDuration`, corresponding to the OpenDJ password policy property `lockout-duration`
- `pwdMaxFailure`, corresponding to the OpenDJ password policy property `lockout-failure-count`

- `pwdFailureCountInterval`, corresponding to the OpenDJ password policy property `lockout-failure-expiration-interval`
- `pwdMaxAge`, corresponding to the OpenDJ password policy property `max-password-age`
- `pwdMinAge`, corresponding to the OpenDJ password policy property `min-password-age`
- `pwdAttribute`, corresponding to the OpenDJ password policy property `password-attribute`
- `pwdSafeModify`, corresponding to the OpenDJ password policy property `password-change-requires-current-password`
- `pwdExpireWarning`, corresponding to the OpenDJ password policy property `password-expiration-warning-interval`
- `pwdInHistory`, corresponding to the OpenDJ password policy property `password-history-count`

The following Internet-Draft password policy attributes are not taken into account by OpenDJ.

- `pwdCheckQuality`, as OpenDJ has password validators. You can set password validators to use in the default password policy.
- `pwdMinLength`, as this is handled by the Length Based Password Validator. You can configure this as part of the default password policy.
- `pwdLockout`, as OpenDJ can deduce whether lockout is configured based on the values of other lockout-related password policy attributes.

Values of the following properties are inherited from the default password policy for Internet-Draft based password policies.

- `account-status-notification-handlers`
- `allow-expired-password-changes`
- `allow-multiple-password-values`
- `allow-pre-encoded-passwords`
- `default-password-storage-schemes`
- `deprecated-password-storage-schemes`
- `expire-passwords-without-warning`
- `force-change-on-add`
- `idle-lockout-interval`

- `last-login-time-attribute`
- `last-login-time-format`
- `max-password-reset-age`
- `password-generator`
- `password-history-duration`
- `password-validators`
- `previous-last-login-time-formats`
- `require-change-by-time`
- `require-secure-authentication`
- `require-secure-password-changes`
- `skip-validation-for-administrators`
- `state-update-failure-policy`

11.1.3 Which Password Policy Applies

The password policy that applies to a user is identified by the operational attribute, `pwdPolicySubentry`.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com uid=bjensen pwdPolicySubentry
dn: uid=bjensen,ou=People,dc=example,dc=com
pwdPolicySubentry: cn=Default Password Policy,cn=Password Policies,cn=config
```

11.2 Configuring Password Policies

You configure server based password policies using the **dsconfig** command. Notice that server based password policies are part of the server configuration, and therefore not replicated. Alternatively, you can configure a subset of password policy features using subentry based password policies that are stored with the replicated server data. This section covers both server based and subentry based password policies.

Procedure 11.1. To Adjust the Default Password Policy

You can reconfigure the default password policy for example to enforce password expiration, check that passwords do not match dictionary words, and prevent password reuse. This default policy is a server based password policy.

1. Enable the appropriate password validator.

```
$ dsconfig
 set-password-validator-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --validator-name Dictionary
 --set enabled:true
 --set check-substrings:true
 --set min-substring-length:4
 --trustAll
 --no-prompt
```

2. Apply the changes to the default password policy.

```
$ dsconfig
 set-password-policy-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --policy-name "Default Password Policy"
 --set max-password-age:90d
 --set min-password-age:4w
 --set password-history-count:7
 --set password-validator:Dictionary
 --trustAll
 --no-prompt
```

3. Check your work.

```
$ dsconfig
 get-password-policy-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --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                          : 12 w 6 d
max-password-reset-age                    : 0 s
min-password-age                          : 4 w
password-attribute                        : userpassword
```

```
password-change-requires-current-password : false
password-expiration-warning-interval      : 5 d
password-generator                        : Random Password Generator
password-history-count                    : 7
password-history-duration                 : 0 s
password-validator                        : Dictionary
previous-last-login-time-format           : -
require-change-by-time                    : -
require-secure-authentication             : false
require-secure-password-changes           : false
```

Procedure 11.2. To Create a Server Based Password Policy

You can add a password policy for example for new users who have not yet used their credentials to bind.

1. Create the new password policy.

```
$ dsconfig
 create-password-policy
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --policy-name "New Account Password Policy"
 --set default-password-storage-scheme:"Salted SHA-1"
 --set force-change-on-add:true
 --set password-attribute:userPassword
 --type password-policy
 --trustAll
 --no-prompt
```

2. Check your work.

```
$ dsconfig
 get-password-policy-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --policy-name "New Account 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                       : true
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                         : -
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
```

If you use a password policy like this, you might want to change the user's policy again when the new user successfully updates the password.

Procedure 11.3. To Create a Subentry Based Password Policy

You can add a subentry to configure a password policy that applies to Directory Administrators.

1. Create the entry that specifies the password policy.

```
$ cat /path/to/subentry-pwp.ldif
dn: cn=Subentry Password Policy,dc=example,dc=com
objectClass: top
objectClass: subentry
objectClass: pwdPolicy
cn: Subentry Password Policy
pwdAttribute: userPassword
pwdLockout: TRUE
pwdMaxFailure: 3
pwdFailureCountInterval: 300
pwdLockoutDuration: 300
pwdAllowUserChange: TRUE
pwdSafeModify: TRUE
subtreeSpecification: {base "ou=people", specificationFilter
  "(isMemberOf=cn=Directory Administrators,ou=Groups,dc=example,dc=com)" }
```

2. Add the policy to the directory.

```
$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename /path/to/subentry-pwp.ldif
Processing ADD request for cn=Subentry Password Policy,dc=example,dc=com
ADD operation successful for DN cn=Subentry Password Policy,dc=example,dc=com
```

3. Check that the policy applies as specified.

   In the example, the policy should apply to a Directory Administrator, while a normal user has the default password policy. Here, Kirsten Vaughan is a member of the Directory Administrators group, and Babs Jensen is not a member.

```
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 uid=kvaughan
 pwdPolicySubentry
dn: uid=kvaughan,ou=People,dc=example,dc=com
pwdPolicySubentry: cn=Subentry Password Policy,dc=example,dc=com

$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 uid=bjensen
 pwdPolicySubentry
dn: uid=bjensen,ou=People,dc=example,dc=com
pwdPolicySubentry: cn=Default Password Policy,cn=Password Policies,cn=config
```

11.3 Assigning Password Policies

You assign subentry based password policies for a subtree of the DIT by adding the policy to an LDAP subentry whose immediate superior is the root of the subtree. In other words you can add the subtree based password policy under `ou=People,dc=example,dc=com`, to have it apply to all entries under `ou=People,dc=example,dc=com`. You can further use the capabilities of LDAP subentries to refine the scope of application.

You assign server based password policies by using the `ds-pwp-password-policy-dn` attribute.

Procedure 11.4. To Assign a Password Policy to a User

1. Prevent users from selecting their own password policy.

```
$ cat protectpwp.ldif
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (target ="ldap:///uid=*,ou=People,dc=example,dc=com")(targetattr =
 "ds-pwp-password-policy-dn")(version 3.0;acl "Cannot choose own pass
 word policy";deny (write)(userdn = "ldap:///self");)

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename protectpwp.ldif
Processing MODIFY request for ou=People,dc=example,dc=com
MODIFY operation successful for DN ou=People,dc=example,dc=com
```

2. Update the user's `ds-pwp-password-policy-dn` attribute.

```
$ cat newuser.ldif
dn: uid=newuser,ou=People,dc=example,dc=com
uid: newuser
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
cn: New User
sn: User
ou: People
mail: newuser@example.com
userPassword: changeme
ds-pwp-password-policy-dn: cn=New Account Password Policy,cn=Password Policies,
 cn=config

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename newuser.ldif
Processing ADD request for uid=newuser,ou=People,dc=example,dc=com
ADD operation successful for DN uid=newuser,ou=People,dc=example,dc=com
```

3. Check your work.

```
$ ldapsearch
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --baseDN dc=example,dc=com
 uid=newuser
 pwdPolicySubentry
dn: uid=newuser,ou=People,dc=example,dc=com
pwdPolicySubentry: cn=New Account Password Policy,cn=Password Policies,cn=config
```

Procedure 11.5. To Assign a Password Policy to a Group

1. Create a subentry defining the collective attribute that sets the `ds-pwp-password-policy-dn` attribute for group members' entries.

```
$ cat pwp-coll.ldif
dn: cn=Password Policy for Dir Admins,dc=example,dc=com
objectClass: collectiveAttributeSubentry
objectClass: extensibleObject
objectClass: subentry
objectClass: top
cn: Password Policy for Dir Admins
ds-pwp-password-policy-dn;collective: cn=Root Password Policy,cn=Pass
 word Policies,cn=config
subtreeSpecification: { base "ou=People", specificationFilter "(isMemberOf=
 cn=Directory Administrators,ou=Groups,dc=example,dc=com)"}

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename pwp-coll.ldif
Processing ADD request for cn=Password Policy for Dir Admins,dc=example,dc=com
ADD operation successful for DN cn=Password Policy for Dir
 Admins,dc=example,dc=com
```

2. Check your work.

```
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 uid=kvaughan
 pwdPolicySubentry
dn: uid=kvaughan,ou=People,dc=example,dc=com
pwdPolicySubentry: cn=Root Password Policy,cn=Password Policies,cn=config
```

11.4 Configuring Password Generation

Password generators are used by OpenDJ during the LDAP password modify extended operation to construct a new password for the user. In other words, a directory administrator resetting a user's password can have OpenDJ directory server generate the new password.

```
$ ldappasswordmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --authzID "u:bjensen"
The LDAP password modify operation was successful
Generated Password:  eak77qdi
```

The default password policy shown in Procedure 11.1, “To Adjust the Default Password Policy” uses the Random Password Generator.

```
$ dsconfig
 get-password-policy-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --policy-name "Default Password Policy"
 --property password-generator
Property           : Value(s)
-------------------:--------------------------
password-generator : Random Password Generator
$ dsconfig
 get-password-generator-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --generator-name "Random Password Generator"
 --property password-generator
 Property               : Value(s)
-----------------------:-----------------------------------------------------
enabled                : true
password-character-set : alpha:abcdefghijklmnopqrstuvwxyz, numeric:0123456789
password-format        : "alpha:3,numeric:2,alpha:3"
```

Notice that the default configuration for the Random Password Generator defines two `password-character-set` values, and then uses those definitions in the `password-format` so that generated passwords have eight characters: three from the `alpha` set, followed by two from the `numeric` set, followed by three from the `alpha` set. The `password-character-set` name must be ASCII.

To set the password generator that OpenDJ employs when constructing a new password for a user, set the `password-generator` property for the password policy that applies to the user.

The following example does not change the password policy, but instead changes the Random Password Generator configuration, and then demonstrates a password being generated upon reset.

```
$ dsconfig
 set-password-generator-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --generator-name "Random Password Generator"
 --remove password-character-set:alpha:abcdefghijklmnopqrstuvwxyz
 --add
  password-character-set:alpha:ABCDEFGHIJKLMNOPQRSTUVWabcdefghijklmnopqrstuvwxyz
 --add password-character-set:punct:,./\`!@#\$%^&*:\;[]\"\'\(\)+=-_~\\
 --set
  password-format:alpha:3,punct:1,numeric:2,punct:2,numeric:3,alpha:3,punct:2
 --no-prompt
$ ldappasswordmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --authzID "u:bjensen"
The LDAP password modify operation was successful
Generated Password:  pld^06:)529HTq$'
```

If you also set up a password validator in the password policy as shown in Procedure 11.1, "To Adjust the Default Password Policy" and further described in Section 11.6, "Configuring Password Validation", make sure the generated passwords are acceptable to the validator.

11.5 Configuring Password Storage

Password storage schemes encode new passwords provided by users so that they are stored in an encoded manner. This makes it difficult or impossible for someone to determine the clear-text passwords from the encoded values. Password storage schemes also determine whether a clear-text password provided by a client matches the encoded value stored in the server.

OpenDJ offers a variety of both reversible and one-way password storage schemes. Some schemes make it easy to recover the clear-text password, whereas others aim to make it computationally hard to do so.

```
$ dsconfig
 list-password-storage-schemes
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password

Password Storage Scheme : Type          : enabled
------------------------:---------------:--------
3DES                    : triple-des    : true
AES                     : aes           : true
Base64                  : base64        : true
Blowfish                : blowfish      : true
Clear                   : clear         : true
CRYPT                   : crypt         : true
MD5                     : md5           : true
PBKDF2                  : pbkdf2        : true
RC4                     : rc4           : true
Salted MD5              : salted-md5    : true
Salted SHA-1            : salted-sha1   : true
Salted SHA-256          : salted-sha256 : true
Salted SHA-384          : salted-sha384 : true
Salted SHA-512          : salted-sha512 : true
SHA-1                   : sha1          : true
```

As shown in Procedure 11.1, “To Adjust the Default Password Policy”, the default password storage scheme for users in Salted SHA-1. When you add users or import user entries with `userPassword` values in clear text, OpenDJ hashes them with the default password storage scheme. Root DN users have a different password policy by default, shown in Procedure 11.5, “To Assign a Password Policy to a Group”. The Root Password Policy uses Salted SHA-512 by default.

You change the default password policy storage scheme for users by changing the applicable password policy, as shown in the following example.

```
$ dsconfig
 set-password-policy-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --policy-name "Default Password Policy"
 --set default-password-storage-scheme:pbkdf2
 --no-prompt
```

Notice that the change in default password storage scheme does not cause OpenDJ to update any stored password values. By default, OpenDJ only stores a password with the new storage scheme the next time that the password is changed.

OpenDJ prefixes passwords with the scheme used to encode them, which means it is straightforward to see which password storage scheme is in use. After the default password storage scheme is changed to PBKDF2, old user passwords remain encoded with Salted SHA-1.

```
$ ldapsearch
 --port 1389
 --bindDN uid=bjensen,ou=people,dc=example,dc=com
 --bindPassword hifalutin
 --baseDN dc=example,dc=com
 "(uid=bjensen)" userPassword
dn: uid=bjensen,ou=People,dc=example,dc=com
userPassword: {SSHA}Rc3tkAj1qP5zGiRkwDIWDFxrxpGgO8Fwh3aibg==
```

When the password is changed, the new default password storage scheme takes effect, as shown in the following example.

```
$ ldappasswordmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --authzID "u:bjensen"
 --newPassword changeit
The LDAP password modify operation was successful
$ ldapsearch
 --port 1389
 --bindDN uid=bjensen,ou=people,dc=example,dc=com
 --bindPassword changeit
 --baseDN dc=example,dc=com
 "(uid=bjensen)" userPassword
dn: uid=bjensen,ou=People,dc=example,dc=com
userPassword: {PBKDF2}10000:O3V6G7y7n7AefOkRGNKQ5ukrMuO5uf+iEQ9ZLg==
```

When you change the password storage scheme for users, realize that the user passwords must change in order for OpenDJ to encode them with the chosen storage scheme. If you are changing the storage scheme because the old scheme was too weak, then you no doubt want users to change their passwords anyway.

If however the storage scheme change is not related to vulnerability, you can use the `deprecated-password-storage-scheme` property of the password policy to have OpenDJ store the password in the new format after successful authentication. This makes it possible to do password migration for active users without forcing users to change their passwords.

```
$ ldapsearch
 --port 1389
 --bindDN uid=kvaughan,ou=people,dc=example,dc=com
 --bindPassword bribery
 --baseDN dc=example,dc=com
 "(uid=kvaughan)" userPassword
dn: uid=kvaughan,ou=People,dc=example,dc=com
userPassword: {SSHA}hDgK44F2GhIIZj913b+29Ak7phb9oU3Lz4ogkg==

$ dsconfig
 set-password-policy-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --policy-name "Default Password Policy"
 --set deprecated-password-storage-scheme:"Salted SHA-1"
 --no-prompt
$ ldapsearch
 --port 1389
 --bindDN uid=kvaughan,ou=people,dc=example,dc=com
 --bindPassword bribery
 --baseDN dc=example,dc=com
 "(uid=kvaughan)" userPassword
dn: uid=kvaughan,ou=People,dc=example,dc=com
userPassword: {PBKDF2}10000:L4dCYqSsNnf47YZ3a6aC8K2E3DChhHHhpcoUzg==
```

Notice that with `deprecated-password-storage-scheme` set appropriately, Kirsten Vaughan's password was hashed again after she authenticated successfully.

11.6 Configuring Password Validation

Password validators are responsible for determining whether a proposed password is acceptable for use and can run checks like ensuring the password meets minimum length requirements, that it has an appropriate range of characters, or that it is not in the history. OpenDJ directory server provides a variety of password validators.

```
$ dsconfig
 list-password-validators
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password


Password Validator                   : Type                 : enabled
-------------------------------------:----------------------:--------
Attribute Value                      : attribute-value      : true
Character Set                        : character-set        : true
Dictionary                           : dictionary           : false
Length-Based Password Validator      : length-based         : true
Repeated Characters                  : repeated-characters  : true
Similarity-Based Password Validator  : similarity-based     : true
Unique Characters                    : unique-characters    : true
```

The password policy for a user specifies the set of password validators that should be used whenever that user provides a new password. By default no password validators are configured. You can see an example setting the Default Password Policy to use the Dictionary validator in Procedure 11.1, "To Adjust the Default Password Policy". The following example shows how to set up a custom password validator and assign it to the default password policy.

The custom password validator ensures passwords meet at least three of the following four criteria. Passwords are composed of:

- English lowercase characters (a through z)
- English uppercase characters (A through Z)
- Base 10 digits (0 through 9)
- Non-alphabetic characters (for example, !, $, #, %)

Notice how the `character-set` values are constructed. The initial `0:` means the set is optional, whereas `1:` would mean the set is required.

```
$ dsconfig
 create-password-validator
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --validator-name "Custom Character Set Password Validator"
 --set allow-unclassified-characters:true
 --set enabled:true
 --set character-set:0:abcdefghijklmnopqrstuvwxyz
 --set character-set:0:ABCDEFGHIJKLMNOPQRSTUVWXYZ
 --set character-set:0:0123456789
 --set character-set:0:!\"#\$%&\'\(\)*+,-./:\;\\<=\>?@[\\]^_\`{\|}~
 --set min-character-sets:3
 --type character-set
 --no-prompt

$ dsconfig
 set-password-policy-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --policy-name "Default Password Policy"
 --set password-validator:"Custom Character Set Password Validator"
 --no-prompt

$ ldappasswordmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --authzID "u:bjensen"
 --newPassword '!ABcd$%^'
```

In the preceding example, the character set of ASCII punctuation, `` !\"#\$%&\'\(\)*+,-./:\;\\<=\>?@[\\]^_\`{\|}~ ``, is hard to read because of all the escape characters. In practice it can be easier to enter sequences like that by using **dsconfig** in interactive mode, and letting it do the escaping for you. You can also use the `--commandFilePath {path}` option to save the result of your interactive session to a file for use in scripts later.

An attempt to set an invalid password fails as shown in the following example.

```
$ ldappasswordmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --authzID "u:bjensen"
 --newPassword hifalutin
 The LDAP password modify operation failed with result code 19
Error Message:  The provided new password failed the validation checks defined
in the server:  The provided password did not contain characters from at least
3 of the following character sets or ranges: 'ABCDEFGHIJKLMNOPQRSTUVWXYZ',
'!"#$%&'()*+,-./:;<=\>?@[\]^_`{|}~', '0123456789', 'abcdefghijklmnopqrstuvwxyz'
```

Validation does not affect existing passwords, but only takes effect when the password is updated.

Chapter 12

Implementing Account Lockout & Notification

OpenDJ directory server supports automatic account lockout. The aim of account lockout is not to punish users who mistype their passwords, but instead to protect the directory against attacks in which the attacker attempts to guess a user password, repeatedly attempting to bind until success is achieved.

Account lockout disables a user account after a specified number of successive authentication failures. When you implement account lockout, you can opt to have OpenDJ directory server unlock the account again after a specified interval, or you can leave the account locked until the password is reset.

Note

When you configure account lockout as part of password policy, OpenDJ locks an account after the specified number of consecutive authentication failures. Account lockout is not transactional across a replication topology, however. Under normal circumstances, replication nevertheless propagates lockout quickly. If ever replication is delayed, an attacker with direct access to multiple replicas could try to authenticate up to the specified number of times on each replica before being locked out on all replicas.

This chapter shows you how to set up account lockout policies, and how to intervene manually to lock and unlock accounts.

12.1 Configuring Account Lockout

Account lockout is configured as part of password policy. This section demonstrates configuring account lockout as part of the default password policy. Users are allowed three consecutive failures before being locked out for five minutes. Failures themselves also expire after five minutes.

Change the default password policy to activate lockout using the **dsconfig** command. As the password policy is part of the server configuration, you must manually apply the changes to each replica in a replication topology.

```
$ dsconfig
 set-password-policy-prop
 --port 4444
 --hostname `hostname`
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --policy-name "Default Password Policy"
 --set lockout-failure-count:3
 --set lockout-duration:5m
 --set lockout-failure-expiration-interval:5m
 --trustAll
 --no-prompt
```

Users having the default password policy are then locked out after three failed attempts in succession.

```
$ ldapsearch
 --port 1389
 --bindDN "uid=bjensen,ou=people,dc=example,dc=com"
 --bindPassword hifalutin
 --baseDN dc=example,dc=com
 uid=bjensen
 mail
dn: uid=bjensen,ou=People,dc=example,dc=com
mail: bjensen@example.com

$ ldapsearch
 --port 1389
 --bindDN "uid=bjensen,ou=people,dc=example,dc=com"
 --bindPassword fatfngrs
 --baseDN dc=example,dc=com
 uid=bjensen
 mail
The simple bind attempt failed
Result Code:  49 (Invalid Credentials)
$ ldapsearch
 --port 1389
 --bindDN "uid=bjensen,ou=people,dc=example,dc=com"
 --bindPassword fatfngrs
 --baseDN dc=example,dc=com
 uid=bjensen
 mail
The simple bind attempt failed
Result Code:  49 (Invalid Credentials)
$ ldapsearch
 --port 1389
 --bindDN "uid=bjensen,ou=people,dc=example,dc=com"
 --bindPassword fatfngrs
 --baseDN dc=example,dc=com
 uid=bjensen
 mail
The simple bind attempt failed
Result Code:  49 (Invalid Credentials)
$ ldapsearch
 --port 1389
 --bindDN "uid=bjensen,ou=people,dc=example,dc=com"
 --bindPassword hifalutin
 --baseDN dc=example,dc=com
 uid=bjensen
 mail
The simple bind attempt failed
Result Code:  49 (Invalid Credentials)
```

12.2 Managing Accounts Manually

This section covers disabling and enabling accounts by using the **manage-account** command. Password reset is covered in the chapter on performing LDAP operations.

For the following examples, the directory admin user, Kirsten Vaughan, has `ds-privilege-name: password-reset`, and the following ACI on `ou=People,dc=example,dc=com`.

```
(target="ldap:///ou=People,dc=example,dc=com") (targetattr ="*||+")(
version 3.0;acl "Admins can run amok"; allow(all) groupdn =
"ldap:///cn=Directory Administrators,ou=Groups,dc=example,dc=com";)
```

Procedure 12.1. To Disable an Account

- Set the account status to disabled with the **manage-account** command.

  ```
  $ manage-account
   set-account-is-disabled
   --port 4444
   --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
   --bindPassword bribery
   --operationValue true
   --targetDN uid=bjensen,ou=people,dc=example,dc=com
   --trustAll
  Account Is Disabled:  true
  ```

Procedure 12.2. To Activate a Disabled Account

- Clear the disabled status using the **manage-account** command.

```
$ manage-account
 clear-account-is-disabled
 --port 4444
 --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
 --bindPassword bribery
 --targetDN uid=bjensen,ou=people,dc=example,dc=com
 --trustAll
Account Is Disabled:  false
```

12.3 Managing Account Status Notification

OpenDJ can send mail about account status changes. OpenDJ needs an SMTP server to send messages, and needs templates for the mail it sends. By default, message templates are in English, under `/path/to/opendj/config/messages/`.

OpenDJ generates notifications only when OpenDJ writes to an entry or evaluates a user entry for authentication. OpenDJ generates account enabled and account disabled notifications when the user account is enabled or disabled with the **manage-account** command, which writes to the entry. OpenDJ generates password expiration notifications when a user tries to bind.

For example, if you set up OpenDJ to send a notification about password expiration, that notification gets triggered when the user authenticates during the password expiration warning interval. OpenDJ does not automatically scan entries to send password expiry notifications. OpenDJ does implement controls that you can pass in an LDAP search to determine whether a user's password is about to expire. See the appendix on *LDAP Controls* for a list. You can send notifications then based on the results of your search.

Procedure 12.3. To Mail Users About Account Status

The following steps demonstrate how to set up notifications. Whether OpenDJ sends notifications depends on the settings in the password policy, and on account activity as described above.

1. Identify the SMTP server to which OpenDJ sends messages.

```
$ dsconfig
 set-global-configuration-prop
 --port 4444
 --hostname `hostname`
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set smtp-server:smtp.example.com
 --trustAll
 --no-prompt
```

2. Set up OpenDJ to be able to mail users about account status.

```
$ dsconfig
 set-account-status-notification-handler-prop
 --port 4444
 --hostname `hostname`
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "SMTP Handler"
 --set enabled:true
 --set email-address-attribute-type:mail
 --trustAll
 --no-prompt
```

Notice that OpenDJ finds the user's mail address on the attribute on the user's entry, specified by `email-address-attribute-type`.

You can also configure the `message-subject` and `message-template-file` properties. Try interactive mode if you plan to do so.

You find templates for messages by default under the `config/messages` directory. You can edit the templates to suit your purposes.

3. Adjust applicable password policies to use the account status notification handler you configured.

```
$ dsconfig
 set-password-policy-prop
 --port 4444
 --hostname `hostname`
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --policy-name "Default Password Policy"
 --set account-status-notification-handler:"SMTP Handler"
 --trustAll
 --no-prompt
```

About Notification Message Templates

When editing the `config/messages` templates to suit your purposes, you can use the following tokens to have OpenDJ update the message text dynamically.

`%%notification-type%%`

This token is replaced with the name of the account status notification type for the notification.

`%%notification-message%%`

This token is replaced with the message for the account status notification.

`%%notification-user-dn%%`

This token is replaced with the string representation of the DN for the user that is the target of the account status notification.

`%%notification-user-attr:`*`attrname`*`%%`

This token is replaced with the value of the attribute specified by *`attrname`* from the user's entry. If the specified attribute has multiple values, then OpenDJ uses the first value encountered. If the specified attribute does not have any values, then OpenDJ replaces it with an emtpy string.

`%%notification-property:`*`propname`*`%%`

This token is replaced with the value of the specified notification property from the account status notification. If the specified property has multiple values, then OpenDJ uses the first value encountered. If the specified property does not have any values, then OpenDJ replaces it with an emtpy string. Valid *`propname`* values include the following.

- `account-unlock-time`
- `new-password`
- `old-password`
- `password-expiration-time`
- `password-policy-dn`
- `seconds-until-expiration`
- `seconds-until-unlock`
- `time-until-expiration`
- `time-until-unlock`

Chapter 13

Setting Resource Limits

This chapter shows you how to set resource limits that prevent directory clients from using an unfair share of system resources.

13.1 Limiting Search Resources

Well-written directory client applications limit the scope of their searches with filters that narrow the number of results returned. By default, OpenDJ also only allows users with appropriate privileges to perform unindexed searches.

You can further adjust additional limits on search operations, such as the following.

- The *lookthrough limit* defines the maximum number of candidate entries OpenDJ considers when processing a search.

  The default lookthrough limit, set by using the global server property `lookthrough-limit`, is 5000.

  You can override the limit for a particular user by changing the operational attribute, `ds-rlim-lookthrough-limit`, on the user's entry.

- The *size limit* sets the maximum number of entries returned for a search.

  The default size limit, set by using the global server property `size-limit`, is 1000.

You can override the limit for a particular user by changing the operational attribute, `ds-rlim-size-limit`, on the user's entry.

- The *time limit* defines the maximum processing time OpenDJ devotes to a search operation.

  The default time limit, set by using the global server property `time-limit`, is 1 minute.

  You can override the limit for a particular user by changing the operational attribute, `ds-rlim-time-limit`, on the user's entry. Times for `ds-rlim-time-limit` are expressed in seconds.

- The *idle time limit* defines how long OpenDJ allows idle connections to remain open.

  No default idle time limit is set. You can set an idle time limit by using the global server property `idle-time-limit`.

  You can override the limit for a particular user by changing the operational attribute, `ds-rlim-idle-time-limit`, on the user's entry. Times for `ds-rlim-idle-time-limit` are expressed in seconds.

- The maximum number of persistent searches can be set using the global server property `max-psearches`.

Procedure 13.1. To Set Search Limits For a User

- Change the user entry to set the limits to override.

```
$ cat limit.ldif
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: modify
add: ds-rlim-size-limit
ds-rlim-size-limit: 10

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename limit.ldif
Processing MODIFY request for uid=bjensen,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=bjensen,ou=People,dc=example,dc=com
```

Now when Babs Jensen performs a search returning more than 10 entries, she sees the following message.

```
Result Code:  4 (Size Limit Exceeded)
Additional Information:  This search operation has sent the maximum of
 10 entries to the client
```

Procedure 13.2. To Set Search Limits For a Group

1. Create an LDAP subentry to specify the limits using collective attributes.

```
$ cat grouplim.ldif
dn: cn=Remove Administrator Search Limits,dc=example,dc=com
objectClass: collectiveAttributeSubentry
objectClass: extensibleObject
objectClass: subentry
objectClass: top
cn: Remove Administrator Search Limits
ds-rlim-lookthrough-limit;collective: 0
ds-rlim-size-limit;collective: 0
ds-rlim-time-limit;collective: 0
subtreeSpecification: {base "ou=people", specificationFilter "
 (isMemberOf=cn=Directory Administrators,ou=Groups,dc=example,dc=com)" }

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename grouplim.ldif
Processing ADD request for
 cn=Remove Administrator Search Limits,dc=example,dc=com
ADD operation successful for DN
 cn=Remove Administrator Search Limits,dc=example,dc=com
```

2. Check the results.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com uid=kvaughan +|grep ds-rlim
ds-rlim-lookthrough-limit: 0
ds-rlim-time-limit: 0
ds-rlim-size-limit: 0
```

13.2 Limiting Idle Time

If you have applications that leave connections open for long periods, OpenDJ can end up devoting resources to maintaining connections that are no longer used. If your network does not drop such connections eventually, you can configure OpenDJ to drop them by setting the global configuration property, `idle-time-limit`. By default, no idle time limit is set.

Note

OpenDJ does not enforce idle timeout for persistent searches.

```
$ dsconfig
 set-global-configuration-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set idle-time-limit:24h
 --trustAll
 --no-prompt
```

The example shown sets the idle time limit to 24 hours.

13.3 Limiting Maximum Request Size

The default maximum request size of 5 MB, set using the advanced connection handler property `max-request-size`, is sufficient to satisfy most client requests. Yet, there are some cases where you might need to raise the request size limit. For example, if clients add groups with large numbers of members, those add requests can go beyond the 5 MB limit.

```
$ dsconfig
 set-connection-handler-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "LDAP Connection Handler"
 --set max-request-size:20mb
 --trustAll
 --no-prompt
```

The example shown sets the maximum request size on the LDAP connection handler to 20 MB.

Chapter 14

Working With Groups of Entries

OpenDJ supports several methods of grouping entries in the directory. Static groups list their members, whereas dynamic groups look up their membership based on an LDAP filter. OpenDJ also supports virtual static groups, which uses a dynamic group style definition, but allows applications to list group members as if the group were static.

When listing entries in static groups, you must also have a mechanism for removing entries from the list when they are deleted or modified in ways that end their membership. OpenDJ makes that possible with *referential integrity* functionality.

This chapter demonstrates how to work with groups.

Tip

The examples in this chapter assume that an `ou=Groups, dc=example,dc=com` entry already exists. If you imported data from Example.ldif, then you already have the entry. If you generated data during setup and did not create an organizational unit for groups yet, create the entry before you try the examples.

```
$ ldapmodify
 --defaultAdd
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
dn: ou=Groups,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: Groups

Processing ADD request for ou=Groups,dc=example,dc=com
ADD operation successful for DN ou=Groups,dc=example,dc=com
```

14.1 Creating Static Groups

A *static group* is expressed as an entry that enumerates all the entries that belong to the group. Static group entries grow as their membership increases.

Static group entries can take the standard object class `groupOfNames` where each `member` attribute value is a distinguished name of an entry, or `groupOfUniqueNames` where each `uniqueMember` attribute value has Name and Optional UID syntax.[1] Like other LDAP attributes, `member` and `uniqueMember` attributes take sets of unique values.

Static group entries can also have the object class `groupOfEntries`, which is like `groupOfNames` except that it is designed to allow groups not to have members.

When creating a group entry, use `groupOfNames` or `groupOfEntries` where possible.

To create a static group, add a group entry such as the following to the directory.

[1] Name and Optional UID syntax values are a DN optionally followed by *`#BitString`*. The *`BitString`*, such as `'0101111101'B`, serves to distinguish the entry from another entry having the same DN, which can occur when the original entry was deleted and a new entry created with the same DN.

```
$ cat static.ldif
dn: cn=My Static Group,ou=Groups,dc=example,dc=com
cn: My Static Group
objectClass: groupOfNames
objectClass: top
ou: Groups
member: uid=ahunter,ou=People,dc=example,dc=com
member: uid=bjensen,ou=People,dc=example,dc=com
member: uid=tmorris,ou=People,dc=example,dc=com

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename static.ldif
Processing ADD request for cn=My Static Group,ou=Groups,dc=example,dc=com
ADD operation successful for DN cn=My Static Group,ou=Groups,dc=example,dc=com
```

To change group membership, modify the values of the membership attribute.

```
$ cat add2grp.ldif
dn: cn=My Static Group,ou=Groups,dc=example,dc=com
changetype: modify
add: member
member: uid=scarter,ou=People,dc=example,dc=com

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename add2grp.ldif
Processing MODIFY request for cn=My Static Group,ou=Groups,dc=example,dc=com
MODIFY operation successful for DN
 cn=My Static Group,ou=Groups,dc=example,dc=com
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 "(cn=My Static Group)"
dn: cn=My Static Group,ou=Groups,dc=example,dc=com
ou: Groups
objectClass: groupOfNames
objectClass: top
member: uid=ahunter,ou=People,dc=example,dc=com
member: uid=bjensen,ou=People,dc=example,dc=com
member: uid=tmorris,ou=People,dc=example,dc=com
member: uid=scarter,ou=People,dc=example,dc=com
cn: My Static Group
```

RFC 4519 says a `groupOfNames` entry must have at least one member. Although OpenDJ allows you to create a `groupOfNames` without members, strictly speaking that behavior is not standard. Alternatively, you can use the `groupOfEntries` object class as shown in the following example.

```
$ cat group-of-entries.ldif
dn: cn=Initially Empty Static Group,ou=Groups,dc=example,dc=com
cn: Initially Empty Static Group
objectClass: groupOfEntries
```

```
objectClass: top
ou: Groups

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename group-of-entries.ldif
Processing ADD request for
 cn=Initially Empty Static Group,ou=Groups,dc=example,dc=com
ADD operation successful for DN
 cn=Initially Empty Static Group,ou=Groups,dc=example,dc=com
$ cat add-members.ldif
# Now add some members to the group.
dn: cn=Initially Empty Static Group,ou=Groups,dc=example,dc=com
changetype: modify
add: member
member: uid=ahunter,ou=People,dc=example,dc=com
member: uid=bjensen,ou=People,dc=example,dc=com
member: uid=tmorris,ou=People,dc=example,dc=com
member: uid=scarter,ou=People,dc=example,dc=com

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename add-members.ldif
Processing MODIFY request for
 cn=Initially Empty Static Group,ou=Groups,dc=example,dc=com
MODIFY operation successful for DN
 cn=Initially Empty Static Group,ou=Groups,dc=example,dc=com
```

14.2 Creating Dynamic Groups

A *dynamic group* specifies members using LDAP URLs. Dynamic groups entries can stay small even as their membership increases.

Dynamic group entries take the `groupOfURLs` object class, with one or more `memberURL` values specifying LDAP URLs to identify group members.

To create a dynamic group, add a group entry such as the following to the directory.

The following example builds a dynamic group of entries effectively matching the filter `"(l=Cupertino)"` (users whose location is Cupertino). Change the filter if your data is different, and so no entries have `l: Cupertino`.

```
$ cat dynamic.ldif
dn: cn=My Dynamic Group,ou=Groups,dc=example,dc=com
cn: My Dynamic Group
objectClass: top
objectClass: groupOfURLs
ou: Groups
memberURL: ldap:///ou=People,dc=example,dc=com??sub?l=Cupertino

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename dynamic.ldif
Processing ADD request for cn=My Dynamic Group,ou=Groups,dc=example,dc=com
ADD operation successful for DN cn=My Dynamic Group,ou=Groups,dc=example,dc=com
```

Group membership changes dynamically as entries change to match the `memberURL` values.

```
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 "(&(uid=*jensen)(isMemberOf=cn=My Dynamic Group,ou=Groups,dc=example,dc=com))"
 mail
dn: uid=bjensen,ou=People,dc=example,dc=com
mail: bjensen@example.com

dn: uid=rjensen,ou=People,dc=example,dc=com
mail: rjensen@example.com

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
dn: uid=ajensen,ou=People,dc=example,dc=com
changetype: modify
replace: l
l: Cupertino

Processing MODIFY request for uid=ajensen,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=ajensen,ou=People,dc=example,dc=com
^D
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 "(&(uid=*jensen)(isMemberOf=cn=My Dynamic Group,ou=Groups,dc=example,dc=com))"
 mail
dn: uid=ajensen,ou=People,dc=example,dc=com
mail: ajensen@example.com

dn: uid=bjensen,ou=People,dc=example,dc=com
mail: bjensen@example.com

dn: uid=rjensen,ou=People,dc=example,dc=com
mail: rjensen@example.com
```

14.3 Creating Virtual Static Groups

OpenDJ lets you create *virtual static groups*, which let applications see dynamic groups as what appear to be static groups.

The virtual static group takes auxiliary object class `ds-virtual-static-group`. Virtual static groups also take either the object class `groupOfNames`, or `groupOfUniqueNames`, but instead of having `member` or `uniqueMember` attributes, have `ds-target-group-dn` attributes pointing to other groups.

Generating the list of members can be resource intensive for large groups, so by default you cannot retrieve the list of members. You can change this with the **dsconfig** command by setting the `Virtual Static member` or `Virtual Static uniqueMember` property.

```
$ dsconfig
 set-virtual-attribute-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --name "Virtual Static member"
 --set allow-retrieving-membership:true
 --trustAll
 --no-prompt
```

The following example creates a virtual static group, and reads the group entry with all members.

```
$ cat virtual.ldif
dn: cn=Virtual Static,ou=Groups,dc=example,dc=com
cn: Virtual Static
objectclass: top
objectclass: groupOfNames
objectclass: ds-virtual-static-group
ds-target-group-dn: cn=My Dynamic Group,ou=Groups,dc=example,dc=com

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename virtual.ldif
Processing ADD request for cn=Virtual Static,ou=Groups,dc=example,dc=com
ADD operation successful for DN cn=Virtual Static,ou=Groups,dc=example,dc=com
$ ldapsearch --port 1389 --baseDN dc=example,dc=com "(cn=Virtual Static)"
dn: cn=Virtual Static,ou=Groups,dc=example,dc=com
objectClass: groupOfNames
objectClass: ds-virtual-static-group
objectClass: top
member: uid=jwalker,ou=People,dc=example,dc=com
member: uid=jmuffly,ou=People,dc=example,dc=com
member: uid=tlabonte,ou=People,dc=example,dc=com
member: uid=dakers,ou=People,dc=example,dc=com
member: uid=jreuter,ou=People,dc=example,dc=com
member: uid=rfisher,ou=People,dc=example,dc=com
```

```
member: uid=pshelton,ou=People,dc=example,dc=com
member: uid=rjensen,ou=People,dc=example,dc=com
member: uid=jcampaig,ou=People,dc=example,dc=com
member: uid=mjablons,ou=People,dc=example,dc=com
member: uid=mlangdon,ou=People,dc=example,dc=com
member: uid=aknutson,ou=People,dc=example,dc=com
member: uid=bplante,ou=People,dc=example,dc=com
member: uid=awalker,ou=People,dc=example,dc=com
member: uid=smason,ou=People,dc=example,dc=com
member: uid=ewalker,ou=People,dc=example,dc=com
member: uid=dthorud,ou=People,dc=example,dc=com
member: uid=btalbot,ou=People,dc=example,dc=com
member: uid=tcruse,ou=People,dc=example,dc=com
member: uid=kcarter,ou=People,dc=example,dc=com
member: uid=aworrell,ou=People,dc=example,dc=com
member: uid=bjensen,ou=People,dc=example,dc=com
member: uid=ajensen,ou=People,dc=example,dc=com
member: uid=cwallace,ou=People,dc=example,dc=com
member: uid=mwhite,ou=People,dc=example,dc=com
member: uid=kschmith,ou=People,dc=example,dc=com
member: uid=mtalbot,ou=People,dc=example,dc=com
member: uid=tschmith,ou=People,dc=example,dc=com
member: uid=gfarmer,ou=People,dc=example,dc=com
member: uid=speterso,ou=People,dc=example,dc=com
member: uid=prose,ou=People,dc=example,dc=com
member: uid=jbourke,ou=People,dc=example,dc=com
member: uid=mtyler,ou=People,dc=example,dc=com
member: uid=abergin,ou=People,dc=example,dc=com
member: uid=mschneid,ou=People,dc=example,dc=com
cn: Virtual Static
ds-target-group-dn: cn=My Dynamic Group,ou=Groups,dc=example,dc=com
```

14.4 Looking Up Group Membership

OpenDJ lets you look up which groups a user belongs to by using the `isMemberOf` attribute.

```
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 uid=bjensen
 isMemberOf
dn: uid=bjensen,ou=People,dc=example,dc=com
isMemberOf: cn=My Static Group,ou=Groups,dc=example,dc=com
isMemberOf: cn=Virtual Static,ou=Groups,dc=example,dc=com
isMemberOf: cn=My Dynamic Group,ou=Groups,dc=example,dc=com
```

You must request `isMemberOf` explicitly.

14.5 Configuring Referential Integrity

When you delete or rename an entry that belongs to static groups, that entry's DN must be removed or changed in the list of each group to which it belongs.

You can configure OpenDJ to resolve membership on your behalf after the change operation succeeds by enabling referential integrity.

Referential integrity functionality is implemented as a plugin. The referential integrity plugin is disabled by default. To enable the plugin, use the **dsconfig** command.

```
$ dsconfig
 set-plugin-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --plugin-name "Referential Integrity"
 --set enabled:true
 --trustAll --no-prompt
```

With the plugin enabled, you can see OpenDJ referential integrity resolving group membership automatically.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com "(cn=My Static Group)"
dn: cn=My Static Group,ou=Groups,dc=example,dc=com
ou: Groups
objectClass: groupOfNames
objectClass: top
member: uid=ahunter,ou=People,dc=example,dc=com
member: uid=bjensen,ou=People,dc=example,dc=com
member: uid=tmorris,ou=People,dc=example,dc=com
member: uid=scarter,ou=People,dc=example,dc=com
cn: My Static Group

$ ldapdelete
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 uid=scarter,ou=People,dc=example,dc=com
Processing DELETE request for uid=scarter,ou=People,dc=example,dc=com
DELETE operation successful for DN uid=scarter,ou=People,dc=example,dc=com
$ ldapsearch --port 1389 --baseDN dc=example,dc=com "(cn=My Static Group)"
dn: cn=My Static Group,ou=Groups,dc=example,dc=com
ou: Groups
objectClass: groupOfNames
objectClass: top
cn: My Static Group
member: uid=ahunter,ou=People,dc=example,dc=com
member: uid=bjensen,ou=People,dc=example,dc=com
member: uid=tmorris,ou=People,dc=example,dc=com
```

By default the referential integrity plugin is configured to manage `member` and `uniqueMember` attributes. These attributes take values that are DNs, and are indexed for equality by default. Before you add an additional attribute to manage, make sure that it has DN syntax and that it is indexed for equality. OpenDJ requires that the attribute be indexed because an unindexed search for integrity would potentially consume too many of the server's resources. Attribute syntax is explained in the chapter on *Managing Schema*. For instructions on indexing attributes, see the section on *Configuring & Rebuilding Indexes*.

You can also configure the referential integrity plugin to check that new entries added to groups actually exist in the directory by setting the `check-references` property to `true`. You can specify additional criteria once you have activated the check. To ensure that entries added must match a filter, set the `check-references-filter-criteria` to identify the attribute and the filter. For example, you can specify that group members must be person entries by setting `check-references-filter-criteria` to `member:(objectclass=person)`. To ensure that entries must be located in the same naming context, set `check-references-scope-criteria` to `naming-context`.

Chapter 15

Implementing Attribute Value Uniqueness

Some attribute values ought to remain unique. If you are using `uid` values as RDNs to distinguish between millions of user entries stored under `ou=People`, then you do not want your directory to contain two or more identical `uid` values. If your credit card or mobile number is stored as an attribute value on your directory entry, you certainly do not want to share that credit card or mobile number with another customer. The same is true for your email address.

The difficulty for you as directory administrator lies in implementing attribute value uniqueness without sacrificing the high availability that comes from using OpenDJ's loosely consistent, multi-master data replication. Indeed OpenDJ's replication model lets you maintain write access during network outages for directory applications. Yet, write access during a network outage can result in the same, theoretically unique attribute value getting assigned to two different entries at once. You do not notice the problem until the network outage goes away and replication resumes.

This chapter shows you how to set up attribute value uniqueness in your directory environment.

Procedure 15.1. To Enable Unique UIDs

OpenDJ provides a unique attribute plugin that you configure by using the **dsconfig** command. By default, the plugin is prepared to ensure attribute values are unique for `uid` attributes.

1. Set the base DN where `uid` should have unique values, and enable the plugin.

```
$ dsconfig
 set-plugin-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --plugin-name "UID Unique Attribute"
 --set base-dn:ou=people,dc=example,dc=com
 --set enabled:true
 --trustAll
 --no-prompt
```

   Alternatively, you can specify multiple base DNs for unique values across multiple suffixes.

```
$ dsconfig
 set-plugin-prop
 --port 4444
 --hostname opendj.example.com
 --bindDn "cn=Directory Manager"
 --bindPassword password
 --plugin-name "UID Unique Attribute"
 --set enabled:true
 --add base-dn:ou=people,dc=example,dc=com
 --add base-dn:ou=people,dc=example,dc=org
 --trustAll
 --no-prompt
```

2. Check that the plugin is working correctly.

```
$ cat bjensen.ldif
dn: uid=ajensen,ou=People,dc=example,dc=com
changetype: modify
add: uid
uid: bjensen

$ ldapmodify
 --defaultAdd
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename bjensen.ldif
Processing MODIFY request for uid=ajensen,ou=People,dc=example,dc=com
MODIFY operation failed
Result Code:  19 (Constraint Violation)
Additional Information:  A unique attribute conflict was detected for \
 attribute uid:  value bjensen already exists in entry
 uid=bjensen,ou=People,dc=example,dc=com
```

   If you have set up multiple suffixes, you might try something like this.

```
$ cat bjensen.ldif
dn: uid=bjensen,ou=People,dc=example,dc=org
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Babs
sn: Jensen
uid: bjensen

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename bjensen.ldif
Processing ADD request for uid=bjensen,ou=People,dc=example,dc=org
ADD operation failed
Result Code:  19 (Constraint Violation)
Additional Information:  A unique attribute conflict was detected for attribute
 uid:  value bjensen already exists in entry
 uid=bjensen,ou=People,dc=example,dc=com
```

Procedure 15.2. To Enable Unique Values For Other Attributes

You can also configure the unique attribute plugin for use with other attributes, such as `mail`, `mobile`, or attributes you define, for example `cardNumber`.

1. Before you set up the plugin, index the attribute for equality.

2. Set up the plugin configuration for your attribute.

```
$ dsconfig
 create-plugin
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --plugin-name "Unique mobile numbers"
 --type unique-attribute
 --set enabled:true
 --set base-dn:ou=people,dc=example,dc=com
 --set type:mobile
 --trustAll
 --no-prompt
```

3. Check that the plugin is working correctly.

```
$ cat mobile.ldif
dn: uid=ajensen,ou=People,dc=example,dc=com
changetype: modify
add: mobile
mobile: +1 828 555 1212

dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: modify
add: mobile
mobile: +1 828 555 1212

$ ldapmodify
 --defaultAdd
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename mobile.ldif
Processing MODIFY request for uid=ajensen,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=ajensen,ou=People,dc=example,dc=com
Processing MODIFY request for uid=bjensen,ou=People,dc=example,dc=com
MODIFY operation failed
Result Code:  19 (Constraint Violation)
Additional Information:  A unique attribute conflict was detected for
 attribute mobile:  value +1 828 555 1212 already exists in entry
 uid=ajensen,ou=People,dc=example,dc=com
```

Procedure 15.3. To Ensure Unique Attribute Values With Replication

The unique attribute plugin ensures unique attribute values on the directory server where the attribute value is updated. If client applications separately write the same attribute value at the same time on different directory replicas, it is possible that both servers consider the duplicate value unique, especially if the network is down between the replicas.

1. Enable the plugin identically on all replicas.
2. To avoid duplicate values where possible, try one of the following solutions.
   - Use a load balancer or proxy technology to direct all updates to the unique attribute to the same directory server.

     The drawback here is the need for an additional component to direct the updates to the same server, and to manage failover should that server go down.
   - Configure safe read mode assured replication between replicas storing the unique attribute.

     The drawbacks here are the cost of safe read assured replication, and the likelihood that assured replication can enter degraded mode during a network outage, thus continuing to allow updates during the outage.

Chapter 16
Managing Schema

Schema definitions describe the data, and especially the object classes and attribute types that can be stored in the directory. By default OpenDJ conforms strictly to LDAPv3 standards pertaining to schema definitions and attribute syntax checking, ensuring that data stored is valid and properly formed. Unless your data use only standard schema present in OpenDJ when you install, then you must add additional schema definitions to account the data your applications stored.

Out of the box, OpenDJ comes with many standard schema definitions. In addition you can update and extend schema definitions while OpenDJ is online. As a result you can add new applications requiring additional data without stopping your directory service.

This chapter demonstrates how to change and to extend OpenDJ schema. This chapter also identifies the standard schema definitions available when you install OpenDJ.

16.1 About Directory Schema

Directory schema, described in RFC 4512, define the kinds of information you find in the directory, and can define how the information are related. This chapter focuses primarily on two types of directory schema definitions.

- *Attribute type* definitions describe attributes of directory entries, such as `givenName` or `mail`.

  Here is an example of an attribute type definition.

  ```
  # Attribute type definition
  attributeTypes: ( 0.9.2342.19200300.100.1.3 NAME ( 'mail' 'rfc822Mailbox' )
    EQUALITY caseIgnoreIA5Match SUBSTR caseIgnoreIA5SubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256} X-ORIGIN 'RFC 4524' )
  ```

  Attribute type definitions start with an object identifier (OID), and generally a short name or names that are easier to remember than the OID. The attribute type definition can specify how attribute values should be collated for sorting, and what syntax they use. The X-ORIGIN is an extension to identify where the definition originated. When you define your one schema, you likely want to provide an X-ORIGIN to help you to track versions of definitions, and where the definitions came from.

- *Object class* definitions identify the attribute types that an entry must have, and may have. Examples of object classes include `person` and `organizationalUnit`.

  Here is an example of an object class definition.

  ```
  # Object class definition
  objectClasses: ( 2.5.6.6 NAME 'person' SUP top STRUCTURAL MUST ( sn $ cn )
    MAY ( userPassword $ telephoneNumber $ seeAlso $ description )
    X-ORIGIN 'RFC 4519' )
  ```

  Entries all have an attribute identifying their object classes, called `objectClass`.

  Object class definitions start with an object identifier (OID), and generally a short name that is easier to remember than the OID. The definition here says that the person object class inherits from the top object class, which is the top-level parent of all object classes. When you view the objectclass attribute values on an entry, you see the list of object classes that the entry takes. An entry can have one STRUCTURAL object class inheritance branch, such as `top` - `person` - `organizationalPerson` - `inetOrgPerson`. Yet entries can have multiple AUXILIARY object classes. The object class then defines the attribute types that must be included, and the attribute types that may be included on entries having the object class.

- An *attribute syntax* constrains what directory clients can store as attribute values.

  An attribute syntax is identified in an attribute type definition by its OID. String-based syntax OIDs are optionally followed by a number, set between braces, that represents a minimum upper bound on the number of characters in the attribute value. For example, in the attribute type definition shown above, the syntax is `1.3.6.1.4.1.1466.115.121.1.26{256}`. The syntax is an IA5 string (composed of characters from the international version of the ASCII character set) that can contain at least 256 characters.

You can find a table matching attribute syntax OIDs with their human-readable names in RFC 4517, Appendix A. Summary of Syntax Object Identifiers. The RFC describes attribute syntaxes in detail. Alternatively, you can see the attribute syntaxes that OpenDJ supports by opening the OpenDJ Control Panel and browsing to Schema > Manage Schema > Attribute Syntaxes. You can also list them by using the **dsconfig** command.

Although attribute syntaxes are often specified in attribute type definitions, directory servers do not always check that attribute values comply with attribute syntaxes. OpenDJ directory server does tend to enforce compliance by default, in particular for certificates, country strings, directory strings, JPEG photos, and telephone numbers. The aim is to avoid accumulating garbage in your directory data.

If you are trying unsuccessfully to import non-compliant data from a more lenient directory server, you can either clean the data before importing it, or if cleaning the data is not an option, read Section 16.3, “Relaxing Schema Checking to Import Legacy Data”.

When creating your own attribute type definitions, use existing attribute syntaxes where possible. If you must create your own attribute syntax, then consider the extensions in Extensions for Attribute Syntax Descriptions.

- Matching rules determine how the directory server compares attribute values to assertion values for LDAP search and LDAP compare operations.

  For example, suppose you search with the filter `(uid=bjensen)`. The assertion value in this case is `bjensen`.

  OpenDJ has the following schema definition for the user ID attribute.

  ```
  attributeTypes: ( 0.9.2342.19200300.100.1.1 NAME ( 'uid' 'userid' )
   EQUALITY caseIgnoreMatch SUBSTR caseIgnoreSubstringsMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} X-ORIGIN 'RFC 4519' )
  ```

  When finding an equality match for your search, OpenDJ uses the `caseIgnoreMatch` matching rule to check for user ID attribute values that equal `bjensen` without regard to case.

  You can see the matching rules that OpenDJ supports by opening the OpenDJ Control Panel and browsing to Schema > Manage Schema > Matching Rules. Notice that many matching rules support string collation in languages other than English. You can also list matching rules by using the **dsconfig** command.

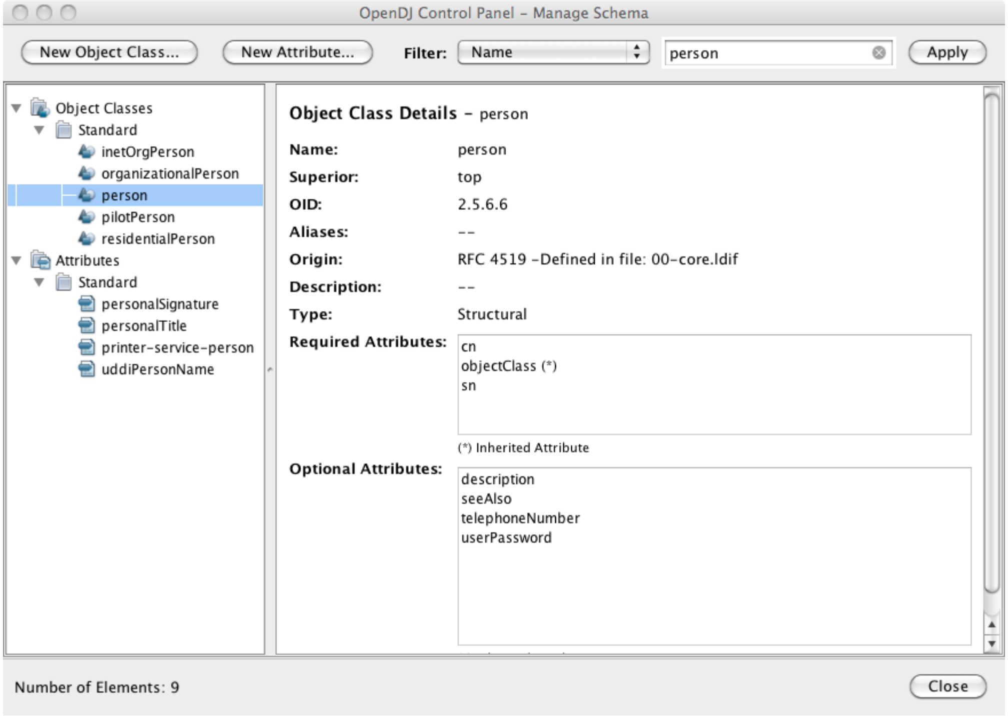
> As you can read in examples like, *Search: List Active Accounts*, OpenDJ matching rules enable directory clients to do more interesting searches than simply comparing strings. That example shows how to search for users who have authenticated in the last three months.

OpenDJ exposes schema over protocol through the `cn=schema` entry. OpenDJ stores the schema definitions corresponding to the entry in LDIF under the `config/schema/` directory. Many standard definitions and definitions pertaining to the server configuration are included at installation time.

16.2 Updating Directory Schema

OpenDJ directory server is designed to permit updating the list of directory schema definitions while the server is running. As a result you can add support for new applications that require new attributes or new kinds of entries without interrupting the directory service. OpenDJ also replicates schema definitions, so the schema you add on one replica are propagated to other replicas without you having to intervene manually.

As it is easy to introduce typos into schema definitions, the best way to start defining your own schema is with the OpenDJ Control Panel. Open the Control Panel > Schema > Manage Schema window to get started creating your custom object classes and attribute types.



Object Class Details – person

Name: person

Superior: top

OID: 2.5.6.6

Aliases: --

Origin: RFC 4519 -Defined in file: 00-core.ldif

Description: --

Type: Structural

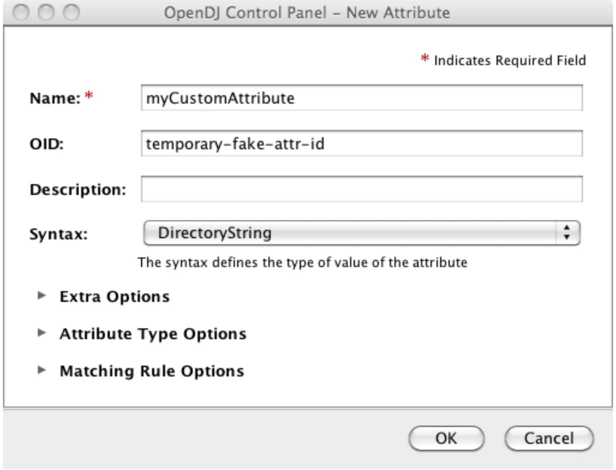
Required Attributes: cn, objectClass (*), sn

(*) Inherited Attribute

Optional Attributes: description, seeAlso, telephoneNumber, userPassword

As object classes reference attribute types, you first create custom attribute types, and then create the object class that references the attribute types.

Create a custom attribute type through the New Attribute window.


OpenDJ Control Panel – New Attribute
* Indicates Required Field
The syntax defines the type of value of the attribute
Extra Options
Attribute Type Options
Matching Rule Options
OK
Cancel

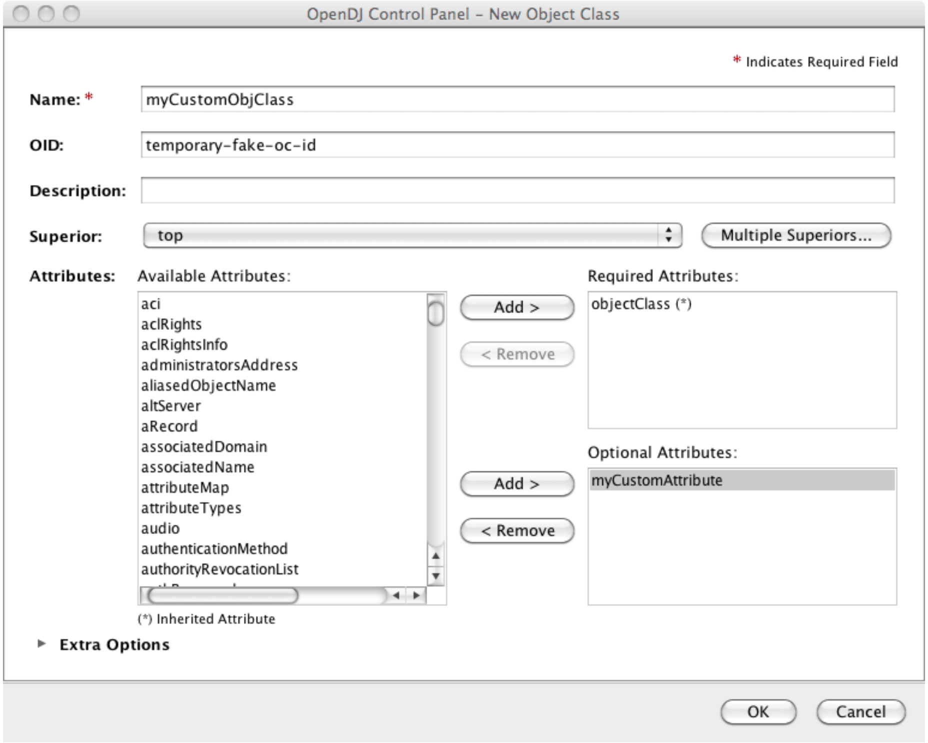
Name: * myCustomAttribute

OID: temporary-fake-attr-id

Description:

Syntax: DirectoryString

Using the New Object Class window, create an auxiliary object class that allows your new custom attribute type. You set the type to Auxiliary under Extra Options.



Name: * myCustomObjClass

OID: temporary-fake-oc-id

Description:

Superior: top

Attributes:

Available Attributes:

aci
aclRights
aclRightsInfo
administratorsAddress
aliasedObjectName
altServer
aRecord
associatedDomain
associatedName
attributeMap
attributeTypes
audio
authenticationMethod
authorityRevocationList

Required Attributes:

objectClass (*)

Optional Attributes:

myCustomAttribute

When you finish, the schema changes show up by default in the file `config/schema/99-user.ldif`. Notice that the file name starts with a number, 99. This number is larger than the numbers prefixing other schema file names. In fact, OpenDJ reads the schema files in sorted order, reading schema definitions as they occur. If OpenDJ reads a schema definition for an object class before it has read the definitions of the attribute types mentioned in the object class definition, then it displays an error. Therefore, when naming your schema file, make sure the name appears in the sorted list of file names *after* all the schema files containing definitions that your schema definitions depends on. The default file name for your schema, `99-user.ldif`, ensures that your definitions load only after all of the schema files installed by default.

You can create this file in the lab using the Control Panel, and then apply the definitions in production by adapting the content for use with the **ldapmodify** command, for example.

```
$ cat config/schema/99-user.ldif
dn: cn=schema
objectClass: top
objectClass: ldapSubentry
objectClass: subschema
cn: schema
attributeTypes: ( temporary-fake-attr-id NAME 'myCustomAttribute' EQUALITY case
 IgnoreMatch ORDERING caseIgnoreOrderingMatch SUBSTR caseIgnoreSubstrings
 Match SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE userApplications )
objectClasses: ( temporary-fake-oc-id NAME 'myCustomObjClass
 ' SUP top AUXILIARY MAY myCustomAttribute )
modifiersName: cn=Directory Manager,cn=Root DNs,cn=config
modifyTimestamp: 20110620095948Z
```

To test your schema definition, add the object class and attribute to an entry.

```
$ cat custom-attr.ldif
dn: uid=bjensen,ou=People,dc=example,dc=com
changetype: modify
add: objectClass
objectClass: myCustomObjClass
-
add: myCustomAttribute
myCustomAttribute: Testing 1, 2, 3...

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename custom-attr.ldif
Processing MODIFY request for uid=bjensen,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=bjensen,ou=People,dc=example,dc=com
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 uid=bjensen
 myCustomAttribute
dn: uid=bjensen,ou=People,dc=example,dc=com
myCustomAttribute: Testing 1, 2, 3...
```

In addition to supporting the standard schema definitions that are described in RFC 4512, section 4.1, OpenDJ also supports the following extensions that you can use when adding your own definitions.

Extensions for All Schema Definitions

`X-ORIGIN`

> Used to specify the origin of a schema element. Examples include `X-ORIGIN 'RFC 4519'`, `X-ORIGIN 'draft-ietf-ldup-subentry'`, and `X-ORIGIN 'OpenDJ Directory Server'`.

X-SCHEMA-FILE

Used to specify the relative path to the schema file containing the schema element such as X-SCHEMA-FILE '00-core.ldif'. Schema definitions are located by default in /path/to/opendj/config/schema/*.ldif files.

Extensions for Attribute Syntax Descriptions

X-ENUM

Used to define a syntax that is an enumeration of values. The following attribute syntax description defines a syntax allowing four possible attribute values for example.

```
ldapSyntaxes: ( security-label-syntax-oid DESC 'Security Label'
 X-ENUM ( 'top-secret' 'secret' 'confidential' 'unclassified' ) )
```

X-PATTERN

Used to define a syntax based on a regular expression pattern, where valid regular expressions are those defined for java.util.regex.Pattern. The following attribute syntax description defines a simple, lenient SIP phone URI syntax check.

```
ldapSyntaxes: ( simple-sip-uri-syntax-oid DESC 'Lenient SIP URI Syntax'
 X-PATTERN '^sip:[a-zA-Z0-9.]+@[a-zA-Z0-9.]+(:[0-9]+)?$' )
```

X-SUBST

Used as a fallback to substitute a defined syntax for one that OpenDJ does not implement. The following example substitutes Directory String syntax, which has OID 1.3.6.1.4.1.1466.115.121.1.15, for a syntax that OpenDJ does not implement.

```
ldapSyntaxes: ( non-implemented-syntax-oid DESC 'Not Implemented in OpenDJ'
 X-SUBST '1.3.6.1.4.1.1466.115.121.1.15' )
```

Extension for Attribute Type Descriptions

X-APPROX

X-APPROX is used to specify the approximate matching rule to use for a given attribute type when not using the default, which is the double metaphone approximate match.

16.3 Relaxing Schema Checking to Import Legacy Data

By default, OpenDJ accepts data that follows the standards in terms of what is allowed and what is rejected. You might have legacy data from a directory service that is more lenient, allowing non-standard constructions such as multiple structural object classes per entry, not checking attribute value syntax, or even not respecting schema definitions.

For example, when importing data with multiple structural object classes defined per entry, you can relax schema checking to warn rather than reject entries having this issue.

```
$ dsconfig
 set-global-configuration-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set single-structural-objectclass-behavior:warn
 --trustAll
 --no-prompt
```

You can allow attribute values that do not respect the defined syntax with the **dsconfig** command as well.

```
$ dsconfig
 set-global-configuration-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set invalid-attribute-syntax-behavior:warn
 --trustAll
 --no-prompt
```

You can even turn off schema checking altogether, although turning off schema checking only really makes sense when you are absolutely sure that the entries and attribute values respect the schema definitions, and you simply want to turn off schema checking temporarily to speed up import processing.

```
$ dsconfig
 set-global-configuration-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set check-schema:false
 --trustAll
 --no-prompt
```

16.4 Standard Schema Included With OpenDJ

The following files under `config/schema/` contain schema definitions out of the box.

`00-core.ldif`

This file contains a core set of attribute type and objectlass definitions from several standard LDAP documents, including draft-ietf-boreham-numsubordinates, draft-findlay-ldap-groupofentries, draft-furuseth-ldap-untypedobject, draft-good-ldap-changelog, draft-ietf-ldup-subentry, draft-wahl-ldap-adminaddr, RFC 1274, RFC 2079, RFC 2256, RFC 2798, RFC 3045, RFC 3296, RFC 3671, RFC 3672, RFC 4512, RFC 4519, RFC 4523, RFC 4524, RFC 4530, RFC 5020, and X.501.

`01-pwpolicy.ldif`

This file contains schema definitions from draft-behera-ldap-password-policy, which defines a mechanism for storing password policy information in an LDAP directory server.

`02-config.ldif`

This file contains the attribute type and objectclass definitions for use with the directory server configuration.

`03-changelog.ldif`

This file contains schema definitions from draft-good-ldap-changelog, which defines a mechanism for storing information about changes to directory server data.

`03-rfc2713.ldif`

This file contains schema definitions from RFC 2713, which defines a mechanism for storing serialized Java objects in the directory server.

`03-rfc2714.ldif`

This file contains schema definitions from RFC 2714, which defines a mechanism for storing CORBA objects in the directory server.

`03-rfc2739.ldif`

This file contains schema definitions from RFC 2739, which defines a mechanism for storing calendar and vCard objects in the directory server. Note that the definition in RFC 2739 contains a number of errors, and this schema file has been altered from the standard definition in order to fix a number of those problems.

`03-rfc2926.ldif`

This file contains schema definitions from RFC 2926, which defines a mechanism for mapping between Service Location Protocol (SLP) advertisements and LDAP.

`03-rfc3112.ldif`

This file contains schema definitions from RFC 3112, which defines the authentication password schema.

`03-rfc3712.ldif`

This file contains schema definitions from RFC 3712, which defines a mechanism for storing printer information in the directory server.

`03-uddiv3.ldif`

This file contains schema definitions from RFC 4403, which defines a mechanism for storing UDDIv3 information in the directory server.

`04-rfc2307bis.ldif`

This file contains schema definitions from the draft-howard-rfc2307bis specification, used to store naming service information in the directory server.

`05-rfc4876.ldif`

This file contains schema definitions from RFC 4876, which defines a schema for storing Directory User Agent (DUA) profiles and preferences in the directory server.

`05-samba.ldif`

This file contains schema definitions required when storing Samba user accounts in the directory server.

`05-solaris.ldif`

This file contains schema definitions required for Solaris and OpenSolaris LDAP naming services.

`06-compat.ldif`

This file contains the attribute type and objectclass definitions for use with the directory server configuration.

Chapter 17
Working With Referrals

Referrals point directory clients to another directory container, which can be another directory server running elsewhere, or another container on the same server. The client receiving a referral must then connect to the other container to complete the request.

Note

Some clients follow referrals on your behalf by default. The OpenDJ **ldapsearch** command does not follow referrals.

Referrals are used for example when a some directory data are temporarily unavailable due to maintenance. Referrals can also be used when a container holds only some of the directory data for a suffix and points to other containers for branches whose data is not available locally.

This chapter demonstrates how to add and remove referrals with the **ldapmodify** command. You can also use the Manage Entries window of the Control Panel to handle referrals.

17.1 About Referrals

Referrals are implemented as entries with LDAP URL `ref` attribute values that point elsewhere. The `ref` attribute type is required by the `referral` object class. The `referral` object class is structural, however, and therefore cannot by default be added to an entry that already has a structural object class defined. When adding a `ref` attribute type to an existing entry, you can use the `extensibleObject` auxiliary object class.

When a referral is set, OpenDJ returns the referral to client applications requesting the entry or child entries affected. Client applications must be capable of following the referral returned. When the directory server responds for example to your search with referrals to one or more LDAP URLs, your client then constructs new searches from the LDAP URLs returned, and tries again.

17.2 Managing Referrals

To create an LDAP referral either you create a referral entry, or you add the `extensibleObject` object class and the `ref` attribute with an LDAP URL to an existing entry. This section demonstrates use of the latter approach.

```
$ cat referral.ldif
dn: ou=People,dc=example,dc=com
changetype: modify
add: objectClass
objectClass: extensibleObject
-
add: ref
ref: ldap://opendj.example.com:2389/ou=People,dc=example,dc=com

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename referral.ldif
Processing MODIFY request for ou=People,dc=example,dc=com
MODIFY operation successful for DN ou=People,dc=example,dc=com
```

The example above adds a referral to `ou=People,dc=example,dc=com`. OpenDJ can now return a referral for operations under the People organizational unit.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com uid=bjensen description

SearchReference(referralURLs=
 {ldap://opendj.example.com:2389/ou=People,dc=example,dc=com??sub?})

$ ldapsearch --port 1389 --baseDN dc=example,dc=com ou=people

SearchReference(referralURLs=
 {ldap://opendj.example.com:2389/ou=People,dc=example,dc=com??sub?})
```

To access the entry instead of the referral, use the Manage DSAIT control.

```
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 --control ManageDSAIT:true
 ou=people
 ref
dn: ou=People,dc=example,dc=com
ref: ldap://opendj.example.com:2389/ou=People,dc=example,dc=com

$ cat people.ldif
dn: ou=People,dc=example,dc=com
changetype: modify
delete: ref
ref: ldap://opendj.example.com:2389/ou=People,dc=example,dc=com

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename people.ldif
Processing MODIFY request for ou=People,dc=example,dc=com
MODIFY operation successful for DN ou=People,dc=example,dc=com
A referral entry ou=People,dc=example,dc=com indicates that the operation must
 be processed at a different server
[ldap://opendj.example.com:2389/ou=People,dc=example,dc=com]
$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --control ManageDSAIT
 --filename people.ldif
Processing MODIFY request for ou=People,dc=example,dc=com
MODIFY operation successful for DN ou=People,dc=example,dc=com
$ ldapsearch --port 1389 --baseDN dc=example,dc=com ou=people
dn: ou=People,dc=example,dc=com
ou: People
objectClass: organizationalunit
objectClass: extensibleObject
objectClass: top
```

The example above shows how to remove the referral using the Manage DSAIT control with the **ldapmodify** command.

Chapter 18

Working With Virtual and Collective Attributes

OpenDJ supports virtual attributes with dynamically generated values. Virtual attributes are used by the server. You can also define your own. OpenDJ also supports standard collective attributes as described in RFC 3671, allowing entries to share common, read-only attribute values.

This chapter demonstrates how to define virtual and collective attributes, showing common solutions as examples of their use.

18.1 Virtual Attributes

OpenDJ defines a number of virtual attributes by default.

`entryDN`

The value is the DN of the entry.

`entryUUID`

Provides a universally unique identifier for the entry.

`etag`

Entity tag as defined in RFC 2616, useful for checking whether an entry has changed since you last read it from the directory.

`hasSubordinates`

Boolean. Whether the entry has children.

`numSubordinates`

Provides the number of direct child entries.

`isMemberOf`

Identifies groups the entry belongs to.

By default OpenDJ generates `isMemberOf` on user entries (entries that have the object class `person`), and on group entries (entries that have the object class `groupOfNames`, `groupOfUniqueNames`, or `groupOfEntries`). You can change this by editing the filter property of the `isMemberOf` virtual attribute configuration.

`member`

Generated for virtual static groups.

`uniqueMember`

Generated for virtual static groups.

`pwdPolicySubentry`

Identifies the password policy that applies to the entry.

By default OpenDJ assigns *root DN* users the password policy with DN `cn=Root Password Policy,cn=Password Policies,cn=config` and regular users the password policy with DN `cn=Default Password Policy,cn=Password Policies,cn=config`. See *Configuring Password Policy* for information on configuring and assigning password policies.

`subschemaSubentry`

References the schema definitions.

`collectiveAttributeSubentries`

References applicable collective attribute definitions.

`governingStructureRule`

References the rule on what type of subordinates the entry can have.

`structuralObjectClass`

References the structural object class for the entry.

These virtual attributes are typically operational, so you get them back from a search only when you request them.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com dc=example
dn: dc=example,dc=com
dc: example
objectClass: domain
objectClass: top

$ ldapsearch --port 1389 --baseDN dc=example,dc=com dc=example numSubordinates
dn: dc=example,dc=com
numSubordinates: 4
```

You can use the existing virtual attribute types to create your own virtual attributes, and you can also use the `user-defined` type to create your own. The virtual attribute is defined by the server configuration, which is not replicated.

```
$ dsconfig
 create-virtual-attribute
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --name "Served By Description"
 --type user-defined
 --set enabled:true
 --set attribute-type:description
 --set base-dn:dc=example,dc=com
 --set value:"Served by OpenDJ.Example.com"
 --trustAll
 --no-prompt
$ ldapsearch --port 1389 --baseDN dc=example,dc=com uid=bjensen description
dn: uid=bjensen,ou=People,dc=example,dc=com
description: Served by OpenDJ.Example.com
```

Collective attributes cover many use cases better than virtual attributes.

18.2 Collective Attributes

Collective attributes provide a standard mechanism for defining attributes that appear on all the entries in a subtree potentially filtered by object class. Standard collective attribute type names have the prefix `c-`.

OpenDJ extends collective attributes to make them easier to use. You can define any OpenDJ attribute as collective using the `;collective` attribute option. You can use LDAP filters in your subtree specification for fine-grained control over which entries have the collective attributes.

You can have entries inherit attributes from other entries using collective attributes. You establish the relationship between entries either by specifying another attribute of the entry that specifies the DN of the entry from which to inherit the attributes, or by specifying how to construct the RDN of the entry from which to inherit the attributes.

To Add Privileges For a Group of Administrators demonstrates setting administrative privileges in OpenDJ using collective attributes. The following examples demonstrate additional ways to use collective attributes in OpenDJ.

- Example 18.1, “Class of Service With Collective Attributes”
- Example 18.2, “Inheriting an Attribute From the Manager's Entry”
- Example 18.3, “Inheriting Attributes From the Locality”

Example 18.1. Class of Service With Collective Attributes

This example defines attributes that specify services available to a user depending on that user's service level.

Note

The following example depends on the `cos` object class, and the `classOfService` attribute type defined but commented out in the `Example.ldif` file imported as sample data. To try this example for yourself, add the attribute type and object class definitions in comments near the top of the file, and then uncomment the `objectClass: cos` and `classOfService` attribute lines in `Example.ldif` before importing the data into OpenDJ.

This example positions collective attributes that depend on the `classOfService` attribute values.

- For entries with `classOfService: bronze`, `mailQuota` is set to 1 GB, and `diskQuota` is set to 10 GB.
- For entries with `classOfService: silver`, `mailQuota` is set to 5 GB, and `diskQuota` is set to 50 GB.
- For entries with `classOfService: gold`, `mailQuota` is set to 10 GB, and `diskQuota` is set to 100 GB.

You define collective attributes in the user data using a subentry. In other words, collective attributes can be replicated. Collective attributes use attributes defined in the directory schema. First, add the `mailQuote` and `diskQuota` attributes, and adjust the definition of the `cos` object class to allow the two quota attributes.

```
$ cat quotas.ldif
dn: cn=schema
changetype: modify
add: attributeTypes
attributeTypes: ( example-class-of-service-attribute-type NAME 'classOfService
 ' EQUALITY caseIgnoreMatch ORDERING caseIgnoreOrderingMatch SUBSTR caseIgnore
 SubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE USAGE user
 Applications X-ORIGIN 'OpenDJ Documentation Examples' )
-
add: attributeTypes
attributeTypes: ( example-class-of-service-disk-quota NAME 'diskQuota
 ' EQUALITY caseIgnoreMatch ORDERING caseIgnoreOrderingMatch SUBSTR case
 IgnoreSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE user
 Applications X-ORIGIN 'OpenDJ Documentation Examples' )
-
add: attributeTypes
attributeTypes: ( example-class-of-service-mail-quota NAME 'mailQuota
 ' EQUALITY caseIgnoreMatch ORDERING caseIgnoreOrderingMatch SUBSTR case
 IgnoreSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE user
 Applications X-ORIGIN 'OpenDJ Documentation Examples' )
-
add: objectClasses
objectClasses: ( example-class-of-service-object-class NAME 'cos' SUP top AUX
 ILIARY MAY ( classOfService $ diskQuota $ mailQuota ) X-ORIGIN 'OpenDJ Doc
 umentation Examples' )

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename quotas.ldif
Processing MODIFY request for cn=schema
MODIFY operation successful for DN cn=schema
```

Use the following collective attribute definitions to set the quotas depending on class of service.

```
# cos.ldif: quotas by class of service
dn: cn=Bronze Class of Service,dc=example,dc=com
objectClass: collectiveAttributeSubentry
objectClass: extensibleObject
objectClass: subentry
objectClass: top
cn: Bronze Class of Service
diskQuota;collective: 10 GB
mailQuota;collective: 1 GB
subtreeSpecification: { base "ou=People", specificationFilter "(classOfService=
 bronze)" }

dn: cn=Silver Class of Service,dc=example,dc=com
objectClass: collectiveAttributeSubentry
objectClass: extensibleObject
objectClass: subentry
objectClass: top
cn: Silver Class of Service
diskQuota;collective: 50 GB
mailQuota;collective: 5 GB
subtreeSpecification: { base "ou=People", specificationFilter "(classOfService=
 silver)" }

dn: cn=Gold Class of Service,dc=example,dc=com
objectClass: collectiveAttributeSubentry
objectClass: extensibleObject
objectClass: subentry
objectClass: top
cn: Gold Class of Service
diskQuota;collective: 100 GB
```

```
mailQuota;collective: 10 GB
subtreeSpecification: { base "ou=People", specificationFilter "(classOfService=
 gold)" }
```

You can add the collective attribute subentries by using the **ldapmodify** command.

```
$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename cos.ldif
Processing ADD request for cn=Bronze Class of Service,dc=example,dc=com
ADD operation successful for DN cn=Bronze Class of Service,dc=example,dc=com
Processing ADD request for cn=Silver Class of Service,dc=example,dc=com
ADD operation successful for DN cn=Silver Class of Service,dc=example,dc=com
Processing ADD request for cn=Gold Class of Service,dc=example,dc=com
ADD operation successful for DN cn=Gold Class of Service,dc=example,dc=com
```

With the collective attributes defined, you can see the results on user entries.

```
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 uid=bjensen
 classOfService mailQuota diskQuota
dn: uid=bjensen,ou=People,dc=example,dc=com
mailQuota: 1 GB
classOfService: bronze
diskQuota: 10 GB

$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 uid=kvaughan
 classOfService mailQuota diskQuota
dn: uid=kvaughan,ou=People,dc=example,dc=com
mailQuota: 5 GB
classOfService: silver
diskQuota: 50 GB

$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 uid=scarter
 classOfService mailQuota diskQuota
dn: uid=scarter,ou=People,dc=example,dc=com
mailQuota: 10 GB
classOfService: gold
diskQuota: 100 GB
```

Example 18.2. Inheriting an Attribute From the Manager's Entry

This example demonstrates how to have OpenDJ set an employee's department number using the manager's department number. To try the example, first import `Example.ldif` into OpenDJ in order to load the appropriate sample data.

For this example the relationship between employee entries and manager entries is based on the manager attributes on employee entries. Each manager attribute on an employee's entry specifies the DN of the manager's entry. OpenDJ retrieves the department number from the manager's entry to populate the attribute on the employee's entry.

The collective attribute subentry that specifies the relationship looks like this:

```
dn: cn=Inherit Department Number From Manager,dc=example,dc=com
objectClass: top
objectClass: subentry
objectClass: inheritedCollectiveAttributeSubentry
objectClass: inheritedFromDNCollectiveAttributeSubentry
cn: Inherit Department Number From Manager
subtreeSpecification: { base "ou=People" }
inheritFromDNAttribute: manager
inheritAttribute: departmentNumber
```

This entry specifies that users inherit department number from their manager.

As seen in Example.ldif, Babs Jensen's manager is Torrey Rigden.

```
dn: uid=bjensen,ou=People,dc=example,dc=com
manager: uid=trigden, ou=People, dc=example,dc=com
```

Torrey's department number is 3001.

```
dn: uid=trigden,ou=People,dc=example,dc=com
departmentNumber: 3001
```

Babs inherits her department number from Torrey.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com uid=bjensen
 departmentNumber
dn: uid=bjensen,ou=People,dc=example,dc=com
departmentNumber: 3001
```

Example 18.3. Inheriting Attributes From the Locality

This example demonstrates how to have OpenDJ set a user's language preferences and street address based on locality. To try the example, first import Example.ldif into OpenDJ in order to load the appropriate sample data.

For this example the relationship between entries is based on locality. The collective attribute subentry specifies how to construct the RDN of the object holding the attribute values to inherit.

```
dn: cn=Inherit From Locality,dc=example,dc=com
objectClass: top
objectClass: subentry
objectClass: inheritedCollectiveAttributeSubentry
objectClass: inheritedFromRDNCollectiveAttributeSubentry
cn: Inherit From Locality
subtreeSpecification: { base "ou=People" }
inheritFromBaseRDN: ou=Locations
inheritFromRDNAttribute: l
inheritFromRDNType: l
inheritAttribute: preferredLanguage
inheritAttribute: street
collectiveConflictBehavior: real-overrides-virtual
```

This specifies that the RDN of the entry from which to inherit attributes is like `l=`*`localityName`*`,ou=Locations`, where *`localityName`* is the value of the `l` (`localityName`) attribute on the user's entry.

In other words, if the user's entry has `l: Bristol`, then the RDN of the entry from which to inherit attributes starts with `l=Bristol,ou=Locations`. The actual entry looks like this:

```
dn: l=Bristol,ou=Locations,dc=example,dc=com
objectClass: top
objectClass: locality
objectClass: extensibleObject
l: Bristol
street: 60 Queen Square
preferredLanguage: en-gb
```

The subentry also specifies two attributes to inherit for preferred language and street address.

The object class `extensibleObject` is added to allow the entry to take a preferred language.[1]

Notice the last line of the collective attribute subentry:

```
collectiveConflictBehavior: real-overrides-virtual
```

This line says that if a collective attribute clashes with a real attribute, the real value takes precedence over the virtual, collective value. You can also set `collectiveConflictBehavior` to `virtual-overrides-real` for the opposite precedence, or to `merge-real-and-virtual` to keep both sets of values.

Here, users can set their own language preferences. When users set language preferences manually, the collective attribute subentry is configured to give the

[1] The object class `extensibleObject` means, "Let me add whatever attributes I want." It is usually better practice to add your own auxiliary object class if you need to decorate an entry with more attributes. The shortcut is taken here as the focus of this example is not schema extension, but instead how to use collective attributes.

user's settings precedence over the locality-based setting, which is only a default guess.

Sam Carter is located in Bristol. Sam has specified no preferred languages.

```
dn: uid=scarter,ou=People,dc=example,dc=com
l: Bristol
```

Sam inherits both the street address and also preferred language from the Bristol locality.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com uid=scarter
 preferredLanguage street
dn: uid=scarter,ou=People,dc=example,dc=com
preferredLanguage: en-gb
street: 60 Queen Square
```

Babs's locality is San Francisco. Babs prefers English, but also knows Korean.

```
dn: uid=bjensen,ou=People,dc=example,dc=com
preferredLanguage: en, ko;q=0.8
l: San Francisco
```

Babs inherits the street address from the San Francisco locality, but keeps her language preferences.

```
$ ldapsearch --port 1389 --baseDN dc=example,dc=com uid=bjensen
 preferredLanguage street
dn: uid=bjensen,ou=People,dc=example,dc=com
preferredLanguage: en, ko;q=0.8
street: 500 3rd Street
```

Chapter 19

Configuring Pass Through Authentication

This chapter focuses on pass through authentication (PTA), whereby you configure another server to determine the response to an authentication request. A typical use case for pass through authentication involves passing authentication through to Active Directory for users coming from Microsoft Windows systems.

19.1 About Pass Through Authentication

You use *LDAP pass through authentication* when the credentials for authenticating are stored not in OpenDJ, but instead in a remote directory service. In effect OpenDJ redirects the bind operation against a remote LDAP server.

Exactly how OpenDJ redirects the bind depends on how the user entry in OpenDJ maps to the corresponding user entry in the remote directory.

OpenDJ provides you several choices to set up the mapping.

- When both the local entry in OpenDJ and the remote entry in the other server have the same DN, you do not have to set up the mapping at all. By default, OpenDJ redirects the bind with the original DN and password from the client application.

- When the local entry in OpenDJ has been provisioned with an attribute holding the DN of the remote entry, you can specify which attribute holds the DN, and OpenDJ redirects the bind on the remote server using the DN value.

- When you cannot get the remote bind DN directly, you need an attribute and value on the OpenDJ entry that corresponds to an identical attribute and value on the remote server in order to map the local entry to the remote entry. In this case you also need the bind credentials for a user who can search for the entry on the remote server. OpenDJ performs a search for the entry using the matching attribute and value, and then redirects the bind with the DN from the remote entry.

You configure pass through authentication as an authentication policy that you associate with a user's entry in the same way that you associate a password policy with a user's entry. Either a user has an authentication policy for pass through authentication, or the user has a local password policy.

19.2 Setting Up Pass Through Authentication

When setting up pass through authentication, you need to know to which remote server or servers to redirect binds, and you need to know how you map user entries in OpenDJ to user entries in the remote directory.

Procedure 19.1. To Set Up SSL Communication For Testing

When performing pass through authentication, you no doubt protect communications between OpenDJ and the server providing authentication. If you test using SSL with self-signed certificates, and you do not want the client blindly to trust the server, follow these steps to import the authentication server's certificate into the OpenDJ key store.

1. Export the server certificate from the authentication server.

   How you perform this step depends on the authentication directory server. With OpenDJ, you can export the certificate as shown here.

   ```
   $ cd /path/to/PTA-Server/config
   $ keytool
    -exportcert
    -rfc
    -alias server-cert
    -keystore keystore
    -storepass `cat keystore.pin`
    > /tmp/pta-srv-cert.pem
   ```

2. Make note of the host name used in the certificate.

You use the host name when configuring the SSL connection. With OpenDJ, you can view the certificate details as shown here.

```
$ keytool
 -list
 -v
 -alias server-cert
 -keystore keystore
 -storepass `cat keystore.pin`
Alias name: server-cert
Creation date: Sep 12, 2011
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=pta-server.example.com, O=OpenDJ Self-Signed Certificate
Issuer: CN=pta-server.example.com, O=OpenDJ Self-Signed Certificate
Serial number: 4e6dc429
Valid from: Mon Sep 12 10:34:49 CEST 2011 until: Wed Sep 11 10:34:49 CEST 2013
Certificate fingerprints:
  MD5:  B6:EE:1C:A0:71:12:EF:6F:21:24:B9:50:EF:8B:4E:6A
  SHA1: 7E:A1:C9:07:D2:86:56:31:24:14:F7:07:A8:6B:3E:A1:39:63:F4:0E
  Signature algorithm name: SHA1withRSA
  Version: 3
```

3. Import the authentication server certificate into OpenDJ's key store.

```
$ cd /path/to/opendj/config
$ keytool
 -importcert
 -alias pta-cert
 -keystore truststore
 -storepass `cat keystore.pin`
 -file /tmp/pta-srv-cert.pem
Owner: CN=pta-server.example.com, O=OpenDJ Self-Signed Certificate
Issuer: CN=pta-server.example.com, O=OpenDJ Self-Signed Certificate
Serial number: 4e6dc429
Valid from: Mon Sep 12 10:34:49 CEST 2011 until: Wed Sep 11 10:34:49 CEST 2013
Certificate fingerprints:
  MD5:  B6:EE:1C:A0:71:12:EF:6F:21:24:B9:50:EF:8B:4E:6A
  SHA1: 7E:A1:C9:07:D2:86:56:31:24:14:F7:07:A8:6B:3E:A1:39:63:F4:0E
  Signature algorithm name: SHA1withRSA
  Version: 3
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

Procedure 19.2. To Configure an LDAP Pass Through Authentication Policy

You configure authentication policies with the **dsconfig** command. Notice that authentication policies are part of the server configuration, and therefore not replicated.

1. Set up an authentication policy for pass through authentication to the authentication server.

```
$ dsconfig
 create-password-policy
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --type ldap-pass-through
 --policy-name "PTA Policy"
 --set primary-remote-ldap-server:pta-server.example.com:636
 --set mapped-attribute:uid
 --set mapped-search-base-dn:"dc=PTA Server,dc=com"
 --set mapping-policy:mapped-search
 --set use-ssl:true
 --set trust-manager-provider:JKS
 --trustAll
 --no-prompt
```

The policy shown here maps identities having this password policy to identities under `dc=PTA Server,dc=com`. Users must have the same `uid` values on both servers. The policy here also uses SSL between OpenDJ and the authentication server.

2. Check that your policy has been added to the list.

```
$ dsconfig
 list-password-policies
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --property use-ssl

Password Policy         : Type              : use-ssl
------------------------:-------------------:--------
Default Password Policy : password-policy   : -
PTA Policy              : ldap-pass-through : true
Root Password Policy    : password-policy   : -
```

Procedure 19.3. To Configure Pass Through Authentication To Active Directory

The steps below demonstrate setting up pass through authentication to Active Directory. Here is some background to help you make sense of the steps.

Entries on the OpenDJ side use `uid` as the naming attribute, and entries also have `cn` attributes. Active Directory entries use `cn` as the naming attribute. User entries on both sides share the same `cn` values. The mapping between entries therefore uses `cn`.

Consider the example where an OpenDJ account with `cn=LDAP PTA User` and DN `uid=ldapptauser,ou=People,dc=example,dc=com` corresponds to an Active Directory account with DN `CN=LDAP PTA User,CN=Users,DC=internal,DC=forgerock,DC=com`. The steps below enable the user with `cn=LDAP PTA User` on OpenDJ authenticate through to Active Directory.

```
$ ldapsearch
 --hostname opendj.example.com
 --baseDN dc=example,dc=com
 uid=ldapptauser
 cn
dn: uid=ldapptauser,ou=People,dc=example,dc=com
cn: LDAP PTA User

$ ldapsearch
 --hostname ad.example.com
 --baseDN "CN=Users,DC=internal,DC=forgerock,DC=com"
 --bindDN "cn=administrator,cn=Users,DC=internal,DC=forgerock,DC=com"
 --bindPassword password
 "(cn=LDAP PTA User)"
 cn
dn: CN=LDAP PTA User,CN=Users,DC=internal,DC=forgerock,DC=com
cn: LDAP PTA User
```

OpenDJ must map its `uid=ldapptauser,ou=People,dc=example,dc=com` entry to the Active Directory entry, `CN=LDAP PTA User,CN=Users,DC=internal,DC=forgerock,DC=com`. In order to do the mapping, OpenDJ has to perform a search for the user in Active Directory using the `cn` value it recovers from its own entry for the user. Active Directory does not allow anonymous searches, so part of the authentication policy configuration consists of the administrator DN and password OpenDJ uses to bind to Active Directory to be able to search.

Finally, before setting up the pass through authentication policy, make sure OpenDJ can connect to Active Directory over a secure connection to avoid sending passwords in the clear.

1. Export the certificate from the Windows server.

   a. Click start > All Programs > Administrative Tools > Certification Authority, then right-click the CA and select Properties.

   b. In the General tab, select the certificate and click View Certificate.

   c. In the Certificate dialog, click the Details tab, then click Copy to File...

   d. Use the Certificate Export Wizard to export the certificate into a file, such as `windows.cer`.

2. Copy the exported certificate to the system running OpenDJ.

3. Import the server certificate into OpenDJ's key store.

```
$ cd /path/to/opendj/config
$ keytool
 -importcert
 -alias ad-cert
 -keystore truststore
 -storepass `cat keystore.pin`
 -file ~/Downloads/windows.cer
Owner: CN=internal-ACTIVEDIRECTORY-CA, DC=internal, DC=forgerock, DC=com
```

```
Issuer: CN=internal-ACTIVEDIRECTORY-CA, DC=internal, DC=forgerock, DC=com
Serial number: 587465257200a7b14a6976cb47916b32
Valid from: Tue Sep 20 11:14:24 CEST 2011 until: Tue Sep 20 11:24:23 CEST 2016
Certificate fingerprints:
  MD5:  A3:D6:F1:8D:0D:F9:9C:76:00:BC:84:8A:14:55:28:38
  SHA1: 0F:BD:45:E6:21:DF:BD:6A:CA:8A:7C:1D:F9:DA:A1:8E:8A:0D:A4:BF
  Signature algorithm name: SHA1withRSA
  Version: 3

Extensions:

#1: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#2: ObjectId: 2.5.29.15 Criticality=false
KeyUsage [
  DigitalSignature
  Key_CertSign
  Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: A3 3E C0 E3 B2 76 15 DC   97 D0 B3 C0 2E 77 8A 11  .>...v.......w..
0010: 24 62 70 0A                                        $bp.
]
]

#4: ObjectId: 1.3.6.1.4.1.311.21.1 Criticality=false

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

At this point OpenDJ can connect to Active Directory over SSL.

4. Set up an authentication policy for OpenDJ users to authenticate to Active Directory.

```
$ dsconfig
 create-password-policy
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --type ldap-pass-through
 --policy-name "AD PTA Policy"
 --set primary-remote-ldap-server:ad.example.com:636
 --set mapped-attribute:cn
 --set mapped-search-base-dn:"CN=Users,DC=internal,DC=forgerock,DC=com"
 --set mapped-search-bind-dn:"cn=administrator,cn=Users,DC=internal,DC=forgerock
 ,DC=com"
 --set mapped-search-bind-password:password
 --set mapping-policy:mapped-search
 --set trust-manager-provider:JKS
 --set use-ssl:true
 --trustAll --no-prompt
```

5. Assign the authentication policy to a test user.

   ```
   $ ldapmodify
    --port 1389
    --bindDN "cn=Directory Manager"
    --bindPassword password
   dn: uid=ldapptauser,ou=People,dc=example,dc=com
   changetype: modify
   add: ds-pwp-password-policy-dn
   ds-pwp-password-policy-dn: cn=AD PTA Policy,cn=Password Policies,cn=config

   Processing MODIFY request for uid=ldapptauser,ou=People,dc=example,dc=com
   MODIFY operation successful for DN uid=ldapptauser,ou=People,dc=example,dc=com
   ```

6. Check that the user can bind using pass through authentication to Active Directory.

   ```
   $ ldapsearch
    --hostname opendj.example.com
    --port 1389
    --baseDN dc=example,dc=com
    --bindDN uid=ldapptauser,ou=People,dc=example,dc=com
    --bindPassword password
    "(cn=LDAP PTA User)"
    userpassword cn
   dn: uid=ldapptauser,ou=People,dc=example,dc=com
   cn: LDAP PTA User
   ```

   Notice that to complete the search, the user authenticated with a password to Active Directory, though no `userpassword` value is present on the entry on the OpenDJ side.

19.3 Assigning Pass Through Authentication Policies

You assign authentication policies in the same way as you assign password policies, by using the `ds-pwp-password-policy-dn` attribute.

Note

Although you assign the pass through authentication policy using the same attribute as for password policy, the authentication policy is not in fact a password policy. Therefore, the user with a pass through authentication policy does not have a value for the operational attribute `pwdPolicySubentry`.

```
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 uid=user.0
 pwdPolicySubentry
dn: uid=user.0,ou=People,dc=example,dc=com
```

Procedure 19.4. To Assign a Pass Through Authentication Policy To a User

Users depending on pass through authentication no longer need a local password policy, as they no longer authenticate locally.

Examples in the following procedure work for this user, whose entry on OpenDJ is as shown. Notice that the user has no password set. The user's password on the authentication server is `password`.

```
dn: uid=user.0,ou=People,dc=example,dc=com
cn: Aaccf Amar
description: This is the description for Aaccf Amar.
employeeNumber: 0
givenName: Aaccf
homePhone: +1 225 216 5900
initials: ASA
l: Panama City
mail: user.0@maildomain.net
mobile: +1 010 154 3228
objectClass: person
objectClass: inetorgperson
objectClass: organizationalperson
objectClass: top
pager: +1 779 041 6341
postalAddress: Aaccf Amar$01251 Chestnut Street$Panama City, DE  50369
postalCode: 50369
sn: Amar
st: DE
street: 01251 Chestnut Street
telephoneNumber: +1 685 622 6202
uid: user.0
```

This user's entry on the authentication server also has `uid=user.0`, and the pass through authentication policy performs the mapping to find the user entry in the authentication server.

1. Prevent users from changing their own password policies.

```
$ cat protect-pta.ldif
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (target ="ldap:///uid=*,ou=People,dc=example,dc=com")(targetattr =
 "ds-pwp-password-policy-dn")(version 3.0;acl "Cannot choose own pass
 word policy";deny (write)(userdn = "ldap:///self");)

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename protect-pta.ldif
Processing MODIFY request for ou=People,dc=example,dc=com
MODIFY operation successful for DN ou=People,dc=example,dc=com
```

2. Update the user's `ds-pwp-password-policy-dn` attribute.

```
$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
dn: uid=user.0,ou=People,dc=example,dc=com
changetype: modify
add: ds-pwp-password-policy-dn
ds-pwp-password-policy-dn: cn=PTA Policy,cn=Password Policies,cn=config

Processing MODIFY request for uid=user.0,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=user.0,ou=People,dc=example,dc=com
```

3. Check that the user can authenticate through to the authentication server.

```
$ ldapsearch
 --port 1389
 --baseDN dc=example,dc=com
 --bindDN uid=user.0,ou=People,dc=example,dc=com
 --bindPassword password
 uid=user.0
 cn sn
dn: uid=user.0,ou=People,dc=example,dc=com
cn: Aaccf Amar
sn: Amar
```

Procedure 19.5. To Assign a Pass Through Authentication Policy To a Group

Examples in the following steps use the pass through authentication policy as defined above. Kirsten Vaughan's entry has been reproduced on the authentication server under `dc=PTA Server,dc=com`.

1. Create a subentry to assign a collective attribute that sets the `ds-pwp-password-policy-dn` attribute for group members' entries.

```
$ cat pta-coll.ldif
dn: cn=PTA Policy for Dir Admins,dc=example,dc=com
objectClass: collectiveAttributeSubentry
objectClass: extensibleObject
objectClass: subentry
objectClass: top
cn: PTA Policy for Dir Admins
ds-pwp-password-policy-dn;collective: cn=PTA Policy,cn=Password Policies,
 cn=config
subtreeSpecification: { base "ou=People", specificationFilter "(isMemberOf=
 cn=Directory Administrators,ou=Groups,dc=example,dc=com)"}

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename pta-coll.ldif
Processing ADD request for cn=PTA Policy for Dir Admins,dc=example,dc=com
ADD operation successful for DN cn=PTA Policy for Dir Admins,dc=example,dc=com
```

2. Check that OpenDJ has applied the policy.

   a. Make sure you can bind as the user on the authentication server.

   ```
   $ ldapsearch
    --port 2389
    --bindDN "uid=kvaughan,ou=People,dc=PTA Server,dc=com"
    --bindPassword password
    --baseDN "dc=PTA Server,dc=com"
    uid=kvaughan
   dn: uid=kvaughan,ou=People,dc=PTA Server,dc=com
   objectClass: person
   objectClass: organizationalPerson
   objectClass: inetOrgPerson
   objectClass: top
   givenName: Kirsten
   uid: kvaughan
   cn: Kirsten Vaughan
   sn: Vaughan
   userPassword: {SSHA}x1BdtrJyRTw63kBSJFDvgvd4guzk66CV8L+t8w==
   ou: People
   mail: jvaughan@example.com
   ```

   b. Check that the user can authenticate through to the authentication server from OpenDJ.

   ```
   $ ldapsearch
    --port 1389
    --bindDN "uid=kvaughan,ou=people,dc=example,dc=com"
    --bindPassword password
    --baseDN dc=example,dc=com
    uid=kvaughan
    cn sn
   dn: uid=kvaughan,ou=People,dc=example,dc=com
   cn: Kirsten Vaughan
   sn: Vaughan
   ```

Chapter 20

Samba Password Synchronization

Samba, the Windows interoperability suite for Linux and UNIX, stores accounts because UNIX and Windows password storage management is not interoperable. The default account storage mechanism is designed to work well with relatively small numbers of accounts and configurations with one domain controller. For larger installations, you can configure Samba to use OpenDJ for storing Samba accounts. See the Samba documentation for your platform for instructions on how to configure an LDAP directory server such as OpenDJ as a Samba passdb backend.

The rest of this chapter focuses on how you keep passwords in sync when using OpenDJ for Samba account storage.

When you store Samba accounts in OpenDJ, Samba stores its own attributes as defined in the Samba schema. Samba does not use the LDAP standard `userPassword` attribute to store users' Samba passwords. You can configure Samba to apply changes to Samba passwords to LDAP passwords as well, too. Yet, if a user modifies her LDAP password directly without updating the Samba password, the LDAP and Samba passwords get out of sync.

The OpenDJ Samba Password plugin resolves this problem for you. The plugin intercepts password changes to Samba user profiles, synchronizing Samba password and LDAP password values. For an incoming Password Modify Extended Request or modify request changing the user password, the OpenDJ Samba Password plugin detects whether the user's entry reflects a Samba user profile (entry has object class `sambaSAMAccount`), hashes the incoming password value, and applies the password change to the appropriate password attribute, keeping the password values in sync. The OpenDJ Samba Password plugin can perform synchronization as long as new passwords values are provided in clear text in the modification request. If you configure Samba to synchronize LDAP passwords when it changes Samba passwords, then the plugin can ignore changes by the Samba user to avoid duplicate synchronization.

Procedure 20.1. To Set Up a Samba Administrator Account

The Samba Administrator synchronizes LDAP passwords after changing Samba passwords by issuing a Password Modify Extended Request. In Samba's `smb.conf` configuration file, the value of `ldap admin dn` is set to the DN of this account. When the Samba Administrator changes a user password, the plugin ignores the changes, so choose a distinct account different from Directory Manager and other administrators.

1. Create or choose an account for the Samba Administrator.

```
$ cat samba.ldif
dn: uid=samba-admin,ou=Special Users,dc=example,dc=com
cn: Samba Administrator
givenName: Samba
mail: samba@example.com
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: top
sn: Administrator
uid: samba-admin
userPassword: password

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
 --filename samba.ldif
Processing ADD request for uid=samba-admin,ou=Special Users,dc=example,dc=com
ADD operation successful for DN uid=samba-admin,ou=Special Users,
 dc=example,dc=com
```

2. Ensure the Samba Administrator can reset user passwords.

```
$ cat samba-rights.ldif
dn: uid=samba-admin,ou=Special Users,dc=example,dc=com
changetype: modify
add: ds-privilege-name
ds-privilege-name: password-reset

dn: dc=example,dc=com
changetype: modify
add: aci
aci: (target="ldap:///dc=example,dc=com") (targetattr ="*")(version 3.0; acl "
 Samba Admin user rights"; allow(all) groupdn ="ldap:///uid=samba-user,ou=
 Special Users,dc=example,dc=com";)

$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename samba-rights.ldif
Processing MODIFY request for uid=samba-admin,ou=Special Users,dc=example,dc=com
MODIFY operation successful for DN
 uid=samba-admin,ou=Special Users,dc=example,dc=com
Processing MODIFY request for dc=example,dc=com
MODIFY operation successful for DN dc=example,dc=com
```

Procedure 20.2. To Set Up the Samba Password Plugin

1. Determine whether the plugin must store passwords hashed like LanManager (`sync-lm-password`) or like Windows NT (`sync-nt-password`), based on how you set up Samba in your environment.

2. Enable the plugin.

   ```
   $ dsconfig
    create-plugin
    --port 4444
    --hostname opendj.example.com
    --bindDN "cn=Directory Manager"
    --bindPassword password
    --plugin-name "Samba Password Synchronisation"
    --type samba-password
    --set enabled:true
    --set pwd-sync-policy:sync-nt-password
    --set
    samba-administrator-dn:"uid=samba-admin,ou=Special Users,dc=example,dc=com"
    --trustAll
    --no-prompt
   ```

   At this point the Samba Password plugin is active.

3. (Optional) When troubleshooting Samba Password plugin issues, you can turn on debug logging as follows.

```
$ dsconfig
 create-debug-target
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --publisher-name "File-Based Debug Logger"
 --target-name org.opends.server.plugins.SambaPasswordPlugin
 --set debug-level:all
 --trustAll
 --no-prompt
$ dsconfig
 set-log-publisher-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --publisher-name "File-Based Debug Logger"
 --set enabled:true
 --trustAll
 --no-prompt
```

Chapter 21

Monitoring, Logging, & Alerts

This chapter describes the monitoring capabilities that OpenDJ implements, and shows how to configure them.

OpenDJ Control Panel provides basic monitoring capabilities under Monitoring > General Information, Monitoring > Connection Handler, and Monitoring > Manage Tasks. This chapter covers the other options for monitoring OpenDJ.

21.1 LDAP-Based Monitoring

OpenDJ exposes monitoring information over LDAP under the entry `cn=monitor`. Many different types of information are exposed. The following example shows monitoring information about the `userRoot` backend holding Example.com data.

Interface stability: Evolving

```
$ ldapsearch --port 1389 --baseDN cn=monitor "(cn=userRoot backend)"
dn: cn=userRoot backend,cn=Disk Space Monitor,cn=monitor
disk-state: normal
objectClass: top
objectClass: ds-monitor-entry
objectClass: extensibleObject
disk-dir: /path/to/opendj/db/userRoot
disk-free: 343039315968
cn: userRoot backend

dn: cn=userRoot Backend,cn=monitor
objectClass: top
objectClass: ds-monitor-entry
objectClass: ds-backend-monitor-entry
ds-backend-is-private: FALSE
ds-backend-writability-mode: enabled
cn: userRoot Backend
ds-backend-entry-count: 163
ds-backend-id: userRoot
ds-base-dn-entry-count: 163 dc=example,dc=com
ds-backend-base-dn: dc=example,dc=com
```

You can set global ACIs on the Access Control Handler if you want to limit read access under `cn=monitor`.

21.2 SNMP-Based Monitoring

OpenDJ lets you monitor the server over the Simple Network Management Protocol (SNMP), with support for the Management Information Base described in RFC 2605: Directory Server Monitoring MIB.

OpenDJ SNMP-based monitoring depends on OpenDMK, which you must download separately. OpenDJ directory server that you download from GitHub is built with OpenDMK, but due to licensing OpenDMK is not part of OpenDJ. SNMP is therefore not enabled by default.

To run the OpenDMK installer, use the self-extracting .jar.

```
$ java -jar ~/Downloads/opendmk-1.0-b02-*.jar
```

If you install under `/path/to`, then the runtime library needed for SNMP is `/path/to/OpenDMK-bin/lib/jdmkrt.jar`.

Once you have installed OpenDMK, you can set up a connection handler for SNMP by enabling the connection handler, and pointing OpenDJ to your installation of the OpenDMK `jdmkrt.jar` library.

```
$ dsconfig
 set-connection-handler-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "SNMP Connection Handler"
 --set enabled:true
 --set opendmk-jarfile:/path/to/OpenDMK-bin/lib/jdmkrt.jar
 --trustAll
 --no-prompt
```

By default, the SNMP Connection Handler listens on port 161 and uses port 162 for traps. On UNIX and Linux systems, only root can normally open these ports. Therefore if you install as a normal user, you might want to change the listen and trap ports.

```
$ dsconfig
 set-connection-handler-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "SNMP Connection Handler"
 --set listen-port:11161
 --set trap-port:11162
 --trustAll
 --no-prompt
```

Restart the SNMP Connection Handler to take the port number changes into account.

To restart the connection handler, you disable it, then enable it again.

```
$ dsconfig
 set-connection-handler-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "SNMP Connection Handler"
 --set enabled:false
 --trustAll
 --no-prompt
$ dsconfig
 set-connection-handler-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "SNMP Connection Handler"
 --set enabled:true
 --trustAll
 --no-prompt
```

Use a command such as **snmpwalk** to check that the SNMP listen port works.

```
$ snmpwalk -v 2c -c OpenDJ@OpenDJ localhost:11161
SNMPv2-SMI::mib-2.66.1.1.1.1 = STRING: "OpenDJ 4.8.1-SNAPSHOT..."
SNMPv2-SMI::mib-2.66.1.1.2.1 = STRING: "/path/to/opendj"
...
```

21.3 JMX-Based Monitoring

OpenDJ provides Java Management eXtensions (JMX) based monitoring. A number of tools support JMX, including **jconsole** and **jvisualvm**, which are bundled with the Sun/Oracle Java platform. JMX is not configured by default. Use the **dsconfig** command to configure the JMX connection handler.

Interface stability: Evolving

Configure the server to activate JMX access. The following example uses the reserved port number, 1689:

```
$ dsconfig
 set-connection-handler-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "JMX Connection Handler"
 --set enabled:true
 --trustAll
 --no-prompt
```

Add appropriate privileges to access JMX monitoring information. By default, no users have privileges to access the JMX connection. The following commands create a user with JMX privileges, who can authenticate over an insecure connection:

```
$ bin/dsconfig
   create-password-policy
   --policy-name "Allow insecure authentication"
   --type password-policy
   --set default-password-storage-scheme:PBKDF2-HMAC-SHA256
   --set password-attribute:userPassword
   --trustAll --no-prompt
   --hostname opendj.example.com
   --port 4444
   --bindDN "cn=Directory Manager"
   --bindPassword password
```

```
$ bin/ldapmodify --port 1389 --bindDN "cn=Directory Manager" --bindPassword password
   dn: uid=JMX Monitor,dc=example,dc=com
   objectClass: top
   objectClass: person
   objectClass: organizationalPerson
   objectClass: inetOrgPerson
   cn: JMX Monitor
   sn: User
   uid: JMX Monitor
   userPassword: password
   ds-privilege-name: monitor-read
   ds-privilege-name: jmx-notify
   ds-privilege-name: jmx-read
   ds-privilege-name: jmx-write
   ds-pwp-password-policy-dn: cn=Allow insecure authentication,cn=Password Policies,cn=config

   Processing ADD request for uid=JMX Monitor,dc=example,dc=com
   ADD operation successful for DN uid=JMX Monitor,dc=example,dc=com
   ^C
```

Connect remotely.

```
$ jconsole &
```

- Remote process:
  <term>service:jmx:rmi:///jndi/rmi://localhost:1689/org.opends.server.protocols.jmx.client-unknown</term>

- Username:
  <term>uid=JMX Monitor,dc=example,dc=com</term>

- Password:
  <term>password</term>

- Connect -> Insecure connection

21.4 Server Operation & Tasks

OpenDJ comes with two commands for monitoring server processes and tasks. The **status** command displays basic information about the local server, similar to what is seen in the default window of the Control Panel. The **manage-tasks** command lets you manage tasks scheduled on a server, such as nightly backup.

The **status** command takes administrative credentials to read the configuration, as does the Control Panel.

```
$ status --bindDN "cn=Directory Manager" --bindPassword password

          --- Server Status ---
Server Run Status:        Started
Open Connections:         1

          --- Server Details ---
Host Name:                localhost
Administrative Users:     cn=Directory Manager
Installation Path:        /path/to/opendj
Version:                  OpenDJ 4.8.1-SNAPSHOT
Java Version:             1.6.0_24
Administration Connector: Port 4444 (LDAPS)

          --- Connection Handlers ---
Address:Port : Protocol : State
-------------:----------:---------
--           : LDIF     : Disabled
0.0.0.0:636  : LDAPS    : Disabled
0.0.0.0:1389 : LDAP     : Enabled
0.0.0.0:1689 : JMX      : Disabled

          --- Data Sources ---
Base DN:     dc=example,dc=com
Backend ID:  userRoot
Entries:     163
Replication: Disabled
```

The **manage-tasks** command connects over the administration port, and so can connect to both local and remote servers.

```
$ manage-tasks
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --trustAll
 --no-prompt

ID                         Type    Status
--------------------------------------------------------
example                    Backup  Recurring
example-20110623030000000  Backup  Waiting on start time
```

21.5 Server Logs

By default OpenDJ stores access and errors logs as well as a server process ID file under the `logs/` directory. For the replication service, OpenDJ also keeps a replication log there. You can also configure a debug log. Furthermore, you can configure policies about how logs are rotated, and how they are retained. You configure logging using the **dsconfig** command.

- The *access log* traces the operations the server processes including timestamps, connection information, and information about the operation itself.

The access log can therefore grow quickly, as each client request results in at least one new log message.

The following access log excerpt shows a search operation from the local host, with the first three lines wrapped for readability.

```
[21/Jun/2011:08:01:53 +0200] CONNECT conn=4 from=127.0.0.1:49708
 to=127.0.0.1:1389 protocol=LDAP
[21/Jun/2011:08:01:53 +0200] SEARCH REQ conn=4 op=0 msgID=1
 base="dc=example,dc=com" scope=wholeSubtree filter="(uid=bjensen)" attrs="ALL"
[21/Jun/2011:08:01:53 +0200] SEARCH RES conn=4 op=0 msgID=1
 result=0 nentries=1 etime=3
[21/Jun/2011:08:01:53 +0200] UNBIND REQ conn=4 op=1 msgID=2
[21/Jun/2011:08:01:53 +0200] DISCONNECT conn=4 reason="Client Unbind"
```

- The *errors log* traces server events, error conditions, and warnings, categorized and identified by severity.

  The following errors log excerpt shows log entries about a backup task, with lines wrapped for readability.

  ```
  [22/Jun/2011:12:32:23 +0200] category=BACKEND severity=NOTICE msgID=9896349
   msg=Backup task 20110622123224088 started execution
  [22/Jun/2011:12:32:23 +0200] category=TOOLS severity=NOTICE msgID=10944792
   msg=Starting backup for backend userRoot
  [22/Jun/2011:12:32:24 +0200] category=JEB severity=NOTICE msgID=8847446
   msg=Archived: 00000000.jdb
  [22/Jun/2011:12:32:24 +0200] category=TOOLS severity=NOTICE msgID=10944795
   msg=The backup process completed successfully
  [22/Jun/2011:12:32:24 +0200] category=BACKEND severity=NOTICE msgID=9896350
   msg=Backup task 20110622123224088 finished execution
  ```

- If you use the HTTP Connection Handler, OpenDJ maintains a separate access log in `logs/http-access`. This access log, by default configured as the File Based HTTP Access Log Publisher, uses a different format than the LDAP access log. This HTTP access log uses Extended Log File Format with fields described in Microsoft's implementation as well. The following default fields are shown here in the order they occur in the log file.

  Interface stability: Evolving

  `cs-host`

  : Client host name

  `c-ip`

  : Client IP address

`cs-username`

Username used to authenticate

`x-datetime`

Completion timestamp for the HTTP request, which you can configure using the `log-record-time-format` property

`cs-method`

HTTP method requested by the client

`cs-uri-query`

Path and query string requested by the client

`cs-version`

HTTP version requested by the client

`sc-status`

HTTP status code for the operation

`cs(User-Agent)`

User-Agent identifier

`x-connection-id`

Connection ID used for OpenDJ internal operations

When using this field to match HTTP requests with internal operations in the LDAP access log, first set the access log advanced property, `suppress-internal-operations`, to `false`. By default, internal operations do not appear in the LDAP access log.

`x-etime`

Execution time in milliseconds needed by OpenDJ to service the HTTP request

Missing values are replaced with `-`. Tabs separate the fields, and if a field contains a tab character, then the field is surrounded with double quotes. OpenDJ then doubles double quotes in the field to escape them.

The following example shows an excerpt of an HTTP access log with the default configuration. Lines are folded and space reformatted for the printed page.

```
-  192.168.0.15  bjensen   22/May/2013:10:06:18 +0200
  GET  /users/bjensen?_prettyPrint=true                      HTTP/1.1    200
  curl/7.21.4  3    40
-  192.168.0.15  bjensen   22/May/2013:10:06:52 +0200
  GET  /groups/Directory%20Administrators?_prettyPrint=true  HTTP/1.1    200
  curl/7.21.4  4    41
-  192.168.0.12  bjensen   22/May/2013:10:07:07 +0200
  GET  /users/missing?_prettyPrint=true                      HTTP/1.1    200
  curl/7.21.4  5     9
-  192.168.0.12  -         22/May/2013:10:07:46 +0200
  GET  /users/missing?_prettyPrint=true                      HTTP/1.1    401
  curl/7.21.4  6     0
-  192.168.0.15  kvaughan  22/May/2013:10:09:10 +0200
  POST /users?_action=create&_prettyPrint=true               HTTP/1.1    200
  curl/7.21.4  7   120
```

You can configure the `log-format` for the access log using the **dsconfig** command. In addition to the default fields, the following standard fields are supported.

`c-port`

Client port number

`s-computername`

Server name where the access log was written

`s-ip`

Server IP address

`s-port`

Server port number

- The *replication log* traces replication events, with entries similar to the errors log. The following excerpt has lines wrapped for readability.

```
[22/Jun/2011:14:37:34 +0200] category=SYNC severity=NOTICE msgID=15139026
 msg=Finished total update: exported domain "dc=example,dc=com" from this
 directory server DS(24065) to all remote directory servers.
[22/Jun/2011:14:37:35 +0200] category=SYNC severity=MILD_WARNING msgID=14745663
 msg=Replication server RS(23947) at opendj.example.com/10.10.0.168:8989 has
 closed the connection to this directory server DS(24065). This directory
 server will now try to connect to another replication server in order to
 receive changes for the domain "dc=example,dc=com"
[22/Jun/2011:14:37:35 +0200] category=SYNC severity=NOTICE msgID=15138894
 msg=The generation ID for domain "dc=example,dc=com" has been reset to 3679640
```

Notice that the replication log does not trace replication operations. Use the external change log instead to get notifications about changes to directory data over protocol. You can alternatively configure an audit log, which is a type of access log that dumps changes in LDIF.

- A *debug log* traces details needed to troubleshoot a problem in the server. Debug logs can grow large quickly, and therefore no debug logs are enabled by default.

Each log depends on a *log publisher*, whose type corresponds to the type of log. OpenDJ uses file-based log publishers. The design allows for custom log publishers, however, which could publish the logs elsewhere besides a file.

For debug logging, you also set a *debug target* to control what gets logged.

21.5.1 Log Rotation & Retention

Each file-based log can be associated with a *log rotation policy*, and a *log retention policy*. The former can specify when, after how much time, or at what maximum size a log is rotated. The latter can specify a maximum number or size of logs to retain, or an amount of free disk space to maintain. The design allows for custom policies as well.

By default the file-based logs are subject to rotation and retention policies that you can list with **dsconfig list-log-rotation-policies** and **dsconfig list-log-retention-policies**.

For example, view the log rotation policies with the following command.

```
$ dsconfig
 list-log-rotation-policies
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password


Log Rotation Policy                 : Type       : file-size-limit : rotation-interval : time-of-day
------------------------------------:------------:-----------------:-------------------:------------
24 Hours Time Limit Rotation Policy : time-limit : -               : 1 d               : -
7 Days Time Limit Rotation Policy   : time-limit : -               : 1 w               : -
Fixed Time Rotation Policy          : fixed-time : -               : -                 : 2359
Size Limit Rotation Policy          : size-limit : 100 mb          : -                 : -
```

View the log retention policies with the following command.

```
$ dsconfig
 list-log-retention-policies
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password


Log Retention Policy             : Type            : disk-space-used : free-disk-space : number-of-files
---------------------------------:-----------------:-----------------:-----------------:----------------
File Count Retention Policy      : file-count      : -               : -               : 10
Free Disk Space Retention Policy : free-disk-space : -               : 500 mb          : -
Size Limit Retention Policy      : size-limit      : 500 mb          : -               : -
```

Use the **dsconfig get-log-publisher-prop** command to examine the policies that apply to a particular logger.

```
$ dsconfig
 get-log-publisher-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --publisher-name "File-Based Access Logger"
 --property retention-policy
 --property rotation-policy
Property         : Value(s)
-----------------:-------------------------------------------------------------
retention-policy : File Count Retention Policy
rotation-policy  : 24 Hours Time Limit Rotation Policy, Size Limit Rotation
                 : Policy
```

In other words, by default OpenDJ keeps 10 access log files, rotating the access log each day, or when the log size reaches 100 MB.

The **dsconfig** command offers a number of subcommands for creating and deleting log rotation and retention policies, and for setting policy properties. You can update which policies apply to a logger by using the **dsconfig set-log-publisher-prop** command.

21.5.2 Log Filtering

Each time a client application sends a request to OpenDJ, the server writes to its access log. As shown above, a simple search operation results in five messages written to the access log. This volume of logging gives you the information to analyze overall access patterns, or to audit access when you do not know in advance what you are looking for.

Yet when you do know what you are looking for, log filtering lets you limit what the server logs, and focus on what you want to see. You define the filter criteria, and also set the filtering policy.

You can filter both access and also audit logs.

Log filtering lets you define rules based these criteria.

- Client IP address, bind DN, group membership
- Port number
- Protocol used (such as LDAP, LDAPS, JMX)
- Response times
- Result codes (only log error results, for example)
- Search response criteria (number of entries returned, whether the search was indexed)
- Target DN
- Type of operation (connect, bind, add, delete, modify, rename, search, etc.)

The filtering policy in the log publisher configuration specifies whether to include or exclude log messages that match the criteria you define. OpenDJ does not filter logs until you update the log publisher configuration.

Example 21.1. Example: Exclude Control Panel-Related Messages

A common development troubleshooting technique consists of sending client requests while tailing the access log:

```
$ tail -f /path/to/opendj/logs/access
```

Trouble is, when OpenDJ Control Panel is running, or when you are also adapting your configuration using the **dsconfig** command, OpenDJ writes access log messages related to administration. These might prevent you from noticing the messages that interest you.

This example demonstrates how to filter out access log messages due to administrative connections over LDAPS on ports 1636 and 4444.

Create access log filtering criteria rules.

```
$ dsconfig
 create-access-log-filtering-criteria
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --publisher-name "File-Based Access Logger"
 --criteria-name "Exclude LDAPS on 1636 and 4444"
 --type generic
 --set connection-port-equal-to:1636
 --set connection-port-equal-to:4444
 --set connection-protocol-equal-to:ldaps
 --trustAll
 --no-prompt
```

Activate filtering to exclude messages from the default access log according to the criteria you specified.

```
$ dsconfig
 set-log-publisher-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --publisher-name "File-Based Access Logger"
 --set filtering-policy:exclusive
 --trustAll
 --no-prompt
```

At this point, OpenDJ filters out connections over LDAPS to ports 1636 and 4444. While performing operations in OpenDJ Control Panel, if you perform a simple **ldapsearch --port 1389 --baseDN dc=example,dc=com uid=bjensen cn**, then all you see in the access log is the effect of the **ldapsearch** command.

```
$ tail -f /path/to/opendj/logs/access
[19/Oct/2011:16:37:16 +0200] CONNECT conn=8 from=127.0.0.1:54165
 to=127.0.0.1:1389 protocol=LDAP
[19/Oct/2011:16:37:16 +0200] SEARCH REQ conn=8 op=0 msgID=1
 base="dc=example,dc=com" scope=wholeSubtree filter="(uid=bjensen)" attrs="cn"
[19/Oct/2011:16:37:16 +0200] SEARCH RES conn=8 op=0 msgID=1 result=0 nentries=1
 etime=14
[19/Oct/2011:16:37:16 +0200] UNBIND REQ conn=8 op=1 msgID=2
[19/Oct/2011:16:37:16 +0200] DISCONNECT conn=8 reason="Client Unbind"
```

In addition to the filtering policy, you can also adjust how OpenDJ writes log messages. By default, OpenDJ writes one log message for a request, and another for a response. You can set the log publisher property `log-format` to `combined` to have OpenDJ write a single message per operation. This can be helpful, for example, when evaluating response times. In addition, you can change the log message time stamps with `log-record-time-format`, and specify whether to log LDAP control OIDs for operations by setting `log-control-oids` to `true`.

21.6 Alert Notifications

OpenDJ can send alerts to provide notifications of significant server events. Yet alert notifications are not enabled by default. You can use the **dsconfig** command to enable alert notifications.

```
$ dsconfig
 set-alert-handler-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "JMX Alert Handler"
 --set enabled:true
 --trustAll
 --no-prompt
```

OpenDJ can also send mail over SMTP instead of JMX notifications. Before you set up the SMTP-based alert handler, you must identify an SMTP server to which OpenDJ sends messages.

```
$ dsconfig
 set-global-configuration-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set smtp-server:smtp.example.com
 --trustAll
 --no-prompt
$ dsconfig
 create-alert-handler
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --handler-name "SMTP Alert Handler"
 --type smtp
 --set enabled:true
 --set message-subject:"OpenDJ Alert, Type: %%alert-type%%, ID: %%alert-id%%"
 --set message-body:"%%alert-message%%"
 --set recipient-address:kvaughan@example.com
 --set sender-address:opendj@example.com
 --trustAll
 --no-prompt
```

Alert Types

OpenDJ directory server uses the following types when sending alerts. For alert types that indicate server problems, check `OpenDJ/logs/errors` for details.

`org.opends.server.AccessControlDisabled`

> The access control handler has been disabled.

`org.opends.server.AccessControlEnabled`

> The access control handler has been enabled.

`org.opends.server.authentiation.dseecompat.ACIParseFailed`

The dseecompat access control subsystem failed to correctly parse one or more ACI rules when the server first started.

`org.opends.server.BackendRunRecovery`

The JE backend has thrown a `RunRecoveryException`. The directory server needs to be restarted.

`org.opends.server.CannotCopySchemaFiles`

A problem has occurred while attempting to create copies of the existing schema configuration files before making a schema update, and the schema configuration has been left in a potentially inconsistent state.

`org.opends.server.CannotRenameCurrentTaskFile`

The directory server is unable to rename the current tasks backing file in the process of trying to write an updated version.

`org.opends.server.CannotRenameNewTaskFile`

The directory server is unable to rename the new tasks backing file into place.

`org.opends.server.CannotScheduleRecurringIteration`

The directory server is unable to schedule an iteration of a recurring task.

`org.opends.server.CannotWriteConfig`

The directory server is unable to write its updated configuration for some reason and therefore the server may not exhibit the new configuration if it is restarted.

`org.opends.server.CannotWriteNewSchemaFiles`

A problem has occurred while attempting to write new versions of the server schema configuration files, and the schema configuration has been left in a potentially inconsistent state.

`org.opends.server.CannotWriteTaskFile`

The directory server is unable to write an updated tasks backing file for some reason.

`org.opends.server.DirectoryServerShutdown`

The directory server has begun the process of shutting down.

`org.opends.server.DirectoryServerStarted`

The directory server has completed its startup process.

`org.opends.server.DiskFull`

Free disk space has reached the full threshold.

Default is 20 MB.

`org.opends.server.DiskSpaceLow`

Free disk space has reached the low threshold.

Default is 100 MB.

`org.opends.server.EnteringLockdownMode`

The directory server is entering lockdown mode, in which only root users are allowed to perform operations and only over the loopback address.

`org.opends.server.LDAPHandlerDisabledByConsecutiveFailures`

Consecutive failures have occurred in the LDAP connection handler and have caused it to become disabled.

`org.opends.server.LDAPHandlerUncaughtError`

Uncaught errors in the LDAP connection handler that have caused it to become disabled.

`org.opends.server.LDIFBackendCannotWriteUpdate`

An LDIF backend was unable to store an updated copy of the LDIF file after processing a write operation.

`org.opends.server.LDIFConnectionHandlerIOError`

The LDIF connection handler encountered an I/O error that prevented it from completing its processing.

`org.opends.server.LDIFConnectionHandlerParseError`

The LDIF connection handler encountered an unrecoverable error while attempting to parse an LDIF file.

`org.opends.server.LeavingLockdownMode`

The directory server is leaving lockdown mode.

`org.opends.server.ManualConfigEditHandled`

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 to another location.

`org.opends.server.ManualConfigEditLost`

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.

`org.opends.server.replication.UnresolvedConflict`

Multimaster replication cannot resolve a conflict automatically.

`org.opends.server.UncaughtException`

A directory server thread has encountered an uncaught exception that caused that thread to terminate abnormally. The impact that this problem has on the server depends on which thread was impacted and the nature of the exception.

`org.opends.server.UniqueAttributeSynchronizationConflict`

A unique attribute conflict has been detected during synchronization processing.

`org.opends.server.UniqueAttributeSynchronizationError`

An error occurred while attempting to perform unique attribute conflict detection during synchronization processing.

Chapter 22

Tuning Servers For Performance

Server tuning refers to the art of adjusting server, JVM, and system configuration to meet the service level performance requirements of directory clients. In the optimal case you achieve service level performance requirements without much tuning at all, perhaps only setting JVM runtime options when installing OpenDJ.

If you are reading this chapter, however, you are probably not facing an optimal situation. Instead you are looking for trade offs that maximize performance for clients given the constraints of your deployment. This chapter therefore aims to provide suggestions on how to measure and to improve directory service performance for better trade offs.

22.1 Defining Performance Requirements & Constraints

Your key performance requirement is most likely to satisfy your users or customers with the resources available to you. Before you can solve potential performance problems, define what those users or customers expect, and determine what resources you will have to satisfy their expectations.

22.1.1 Service-Level Agreements

Service-level agreement (SLA) is a formal name for what directory client applications and the people who run them expect from your service in terms of performance.

SLAs might cover many aspects of the directory service. Whether or not your SLA is formally defined, you ought to know what is expected, or at least what you provide, in the following four areas.

- Directory service *response times*

  Directory service response times range from less than a millisecond on average across a low latency connection on the same network to however long it takes your network to deliver the response. More important than average or best response times is the response time distribution, because applications set timeouts based on worst case scenarios. For example, a response time performance requirement might be defined as, "Directory response times must average less than 10 milliseconds for all operations except searches returning more than 10 entries, with 99.9% of response times under 40 milliseconds."

- Directory service *throughput*

  Directory service throughput can range up to many thousands of operations per second. In fact there is no upper limit for read operations such as searches, because only write operations must be replicated. To increase read throughput, simply add additional replicas. More important than average throughput is peak throughput. You might have peak write throughput in the middle of the night when batch jobs update entries in bulk, and peak binds for a special event or first thing Monday morning. For example, a throughput performance requirement might be expressed as, "The directory service must sustain a mix of 5,000 operations per second made up of 70% reads, 25% modifies, 3% adds, and 2% deletes."

  Even better is to mimic the behavior of key operations for performance testing, so that you understand the patterns of operations in the throughput you need to provide.

- Directory service *availability*

  OpenDJ is designed to let you build directory services that are basically available, including during maintenance and even upgrade of individual servers. Yet, in order to reach very high levels of availability, you must make sure not only that the software is designed for availability, but also that your operations execute in such a way as to preserve availability. Availability requirements can be as lax as best effort, or as stringent as 99.999% or more uptime.

Replication is the OpenDJ feature that allows you to build a highly available directory service.

- Directory service administrative support

  Do not forget to make sure you understand and set expectations about how you support your users when they run into trouble. Directory services can perhaps help you turn password management into a self-service visit to a web site, but some users no doubt still need to know what they can expect if they need your help.

Writing down the SLA, even if your first version consists of guesses, helps you reduce performance tuning from an open-ended project to a clear set of measurable goals for a manageable project with a definite outcome.

22.1.2 Available Resources

With your SLA in hand, take inventory of the server, networks, storage, people, and other resources at your disposal. Now is the time to estimate whether it is possible to meet the requirements at all.

If for example you are expected to serve more throughput than the network can transfer, maintain high availability with only one physical machine, store 100 GB of backups on a 50 GB partition, or provide 24/7 support all alone, no amount of tweaking available resources is likely to fix the problem.

When checking that the resources you have at least theoretically suffice to meet your requirements, do not forget that high availability in particular requires at least two of everything to avoid single points of failure. Be sure to list the resources you expect to have, when and how long you expect to have them, and why you need them. Also make note of what is missing and why.

22.1.2.1 Server Hardware Recommendations

Concerning server hardware, OpenDJ runs on systems with Java support, and is therefore quite portable. That said, OpenDJ tends to perform best on single-board, x86 systems due to low memory latency.

22.1.2.2 Storage Recommendations

OpenDJ is designed to work with local storage for the database, not for network file systems such as NFS.

High performance storage is essential if you need to handle high write throughput.

The Berkeley Java Edition DB works well with traditional disks as long as the database cache size allows the DB to stay fully cached in memory. This is the case because the database transaction log is append only. When the DB is too big to stay cached in memory, however, then cache misses lead to random disk access, slowing OpenDJ performance.

You might mitigate this effect by using solid-state disks for persistent storage, or for file system cache.

Regarding database size on disk, if you have sustained write traffic then the database grows to about twice its initial size on disk. This is normal, and due to the way the database manages its logs. The size on disk does not impact the DB cache size requirements.

22.2 Testing Performance

Even if you do not need high availability, you still need two of everything, because your test environment needs to mimic your production environment as closely as possible if you want to avoid nasty surprises.

In your test environment, you set up OpenDJ as you will later in production, and then conduct experiments to determine how best to meet the requirements defined in the SLA.

Use make-ldif to generate sample data that match what you expect to find in production.

The OpenDJ LDAP Toolkit provides three command-line tools to help with basic performance testing.

- The authrate command measures bind throughput and response time.
- The modrate command measures modification throughput and response time.
- The searchrate command measures search throughput and response time.

All three commands show you information about the response time distributions, and allow you to perform tests at specific levels of throughput.

If you need additional precision when evaluating response times, use the global configuration setting `etime-resolution` to change elapsed processing time resolution from milliseconds (default) to nanoseconds.

```
$ dsconfig
 set-global-configuration-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set etime-resolution:nanoseconds
 --no-prompt
```

For more extensive testing, try the SLAMD Distributed Load Generation Engine. SLAMD is built to test more than just directory, but is particularly well suited to test directory service performance, is well documented, and is available under the Sun Public License. SLAMD is designed both to offer an easy to used web-based interface, and also to allow you to customize jobs to match the access patterns you expect from client applications.

22.3 Tweaking OpenDJ Performance

When your tests show that OpenDJ performance is lacking even though you have the right underlying network, hardware, storage, and system resources in place, you can tweak OpenDJ performance in a number of ways. This section mentions the most common tweaks.

22.3.1 Java Settings

Default Java settings let you evaluate OpenDJ using limited system resources. If you need high performance for production system, test with the following JVM options. These apply to the Sun/Oracle JVM.

Tip

To apply JVM settings for your server, edit `config/java.properties`, and apply the changes with the **dsjavaproperties** command.

`-server`

Use the C2 compiler and optimizer.

`-d64`

To use a heap larger than about 3.5 GB on a 64-bit system, use this option.

`-Xms, -Xmx`

Set both minimum and maximum heap size to the same value to avoid resizing. Leave space for the entire DB cache and more.

`-Xmn`

Set the new generation size between 1-4 GB for high throughput deployments, but leave enough overall JVM heap to avoid overlaps with the space used for DB cache.

`-XX:MaxTenuringThreshold=1`

Force OpenDJ to create only objects that have either a short lifetime, or a long lifetime.

`-XX:+UseConcMarkSweepGC`

The CMS garbage collector tends to give the best performance characteristics. You might also consider the G1 garbage collector.

`-XX:+PrintGCDetails`
`-XX:+PrintGCTimeStamps`

Use these when diagnosing JVM tuning problems. You can turn them off when everything is running smoothly.

`-XX:+UseCompressedOops`

Java object pointers normally have the same size as native machine pointers. If you run a small, but 64-bit JVM, then compressed object pointers can save space. Set this option when you have a 64-bit JVM, `-Xmx` less than 32 GB, and Java SE 6u23 or later.

22.3.2 Data Storage Settings

By default, OpenDJ compressing attribute descriptions and object class sets to reduce data size. This is called compact encoding.

By default, OpenDJ does not however compress entries stored in its backend database. If your entries hold values that compress well — such as text, and not JPEG photos or MP3 audio — you can gain space by setting the local DB backend property `entries-compressed` to `true` before you (re-)import data from LDIF. With `entries-compressed: true` OpenDJ compresses entries before writing them to the database.[1]

[1] OpenDJ does not proactively rewrite all entries in the database after you change the settings. Instead, to force OpenDJ to compress all entries, import the data from LDIF.

```
$ dsconfig
 set-backend-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --backend-name userRoot
 --set entries-compressed:true
 --trustAll
 --no-prompt
$ import-ldif
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --ldifFile /path/to/Example.ldif
 --backendID userRoot
 --includeBranch dc=example,dc=com
 --start 0
Import task 20120917100628767 scheduled to start Sep 17, 2012 10:06:28 AM CEST
```

22.3.3 LDIF Import Settings

You can tweak OpenDJ to speed up import of large LDIF files.

By default, the temporary directory used for scratch files is `import-tmp` under the directory where you installed OpenDJ. Use **import-ldif** with the `--tmpdirectory` option to set this directory to a `tmpfs` file system, such as `/tmp`.

In some cases, you can improve performance by using the `--threadCount` option with the **import-ldif** command to set the thread count larger than the default, which is twice the number of CPUs.

If you are certain your LDIF contains only valid entries with correct syntax, because the LDIF was exported from OpenDJ with all checks active for example, you can skip schema and DN validation. Use the `--skipSchemaValidation` and `--skipDNValidation` options with the **import-ldif** command to skip validation.

22.3.4 Database Cache Settings

Database cache size is, by default, set as a percentage of the JVM heap, using the backend property `db-cache-percent`. Alternatively, you use the backend property `db-cache-size` to set the size. If you set up multiple database backends, the total percent of JVM heap used must remain less than 100, and must leave space for other uses. Default settings work for servers with one user data backend JVM heaps up to 2 GB. For heaps larger than 2 GB, you can allocate a larger percentage of heap space to DB cache.

Depending on the size of your database, you have a choice to make about database cache settings.

By caching the entire database in the JVM heap, you can get more deterministic response times and limit disk I/O. Yet, caching the whole DB can require a very large JVM, which you must pre-load on startup, and which can result in long garbage collections and a difficult-to-manage JVM. Test database pre-load on startup by setting the `preload-time-limit` for the backend.

```
$ dsconfig
 set-backend-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --backend-name userRoot
 --set preload-time-limit:30m
 --trustAll
 --no-prompt
```

Database pre-load is single-threaded, and loads each database one at a time.

By allowing file system cache to hold the portion of database that does not fit in DB cache, you trade less deterministic and slightly slower response times for not having to pre-load the DB and not having garbage collection pauses with large JVMs. How you configure the file system cache depends on your operating system.

22.3.5 Entry Cache Settings

OpenDJ implements an entry cache. The entry cache is not designed to cache every entry in your database, but is instead useful in cases where you have a few, typically large entries that are regularly used. For example, if you have a few large static groups and applications that regularly check group membership, you could cache your group entries.

```
$ dsconfig
 create-entry-cache
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --cache-name "Large Group Entry Cache"
 --type fifo
 --set cache-level:1
 --set include-filter:"(ou=Large Static Groups)"
 --set max-entries:10
 --set enabled:true
 --trustAll
 --no-prompt
```

You can use the global setting, `entry-cache-preload`, to force OpenDJ to load the entry cache as part of server startup.

```
$ dsconfig
 set-global-configuration-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set entry-cache-preload:true
 --no-prompt
```

By default, OpenDJ does not pre-load the entry cache.

22.3.6 Logging Settings

Debug logs trace the internal workings of OpenDJ, and therefore generally should be used sparingly, especially in high performance deployments.

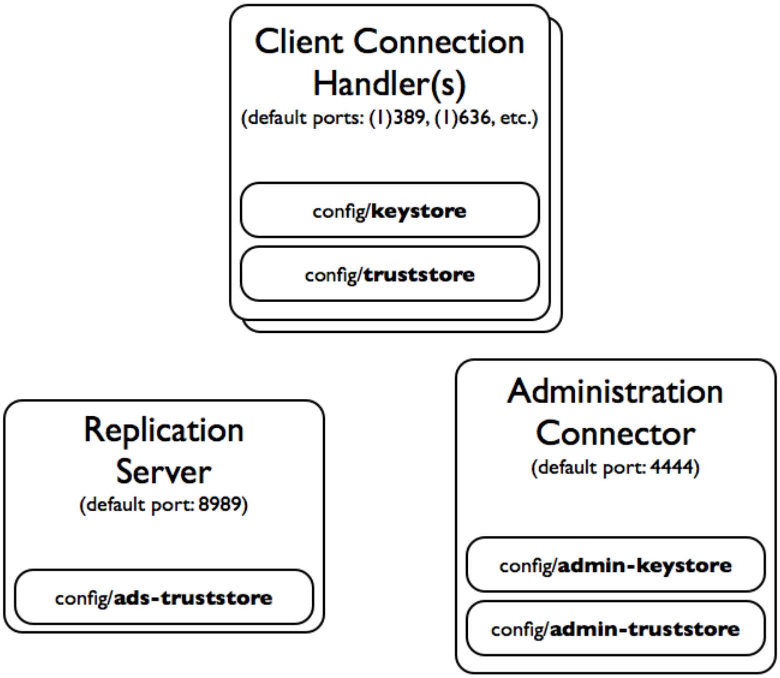
In general leave other logs active for production environments to help troubleshoot any issues that arise.

For OpenDJ servers handling very high throughput, however, such as 100,000 operations per second or more, the access log constitute a performance bottleneck, as each client request results in multiple access log messages. Consider disabling the access log in such cases.

```
$ dsconfig
 set-log-publisher-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --publisher-name "File-Based Access Logger"
 --set enabled:false
 --trustAll
 --no-prompt
```

Chapter 23
Changing Server Certificates

OpenDJ uses key stores (for private keys) and trust stores (for public, signed certificates). Up to three sets of key stores are used, as shown in the following illustration.



OpenDJ uses different sets of public and
private keys for different secure connections.

By default the key stores are located in the `/path/to/opendj/config` directory.

- The `keystore` and `truststore` hold keys for securing connections with client applications.
- The `admin-keystore` and `admin-truststore` hold keys for securing administrative connections, such as those used when connecting with the **dsconfig** command.
- The `ads-truststore` holds keys for securing replication connections with other OpenDJ servers in the replication topology.

Each key store has a specific purpose.

`admin-keystore`

> This Java Key Store holds the private key and administrative certificate for the server, `admin-cert`. This key pair is used to protect communications on the administration port. The password, stored in `admin-keystore.pin`, is also the key password for `admin-cert`.

`admin-truststore`

This Java Key Store holds a copy of the administrative certificate, `admin-cert`. The password is the same as for the `admin-keystore`, in other words the string in `admin-keystore.pin`.

`ads-truststore`

This Java Key Store holds public key certificates of all servers replicating with the current server. It also includes the `ads-certificate` key pair of the current server. The password is stored in `ads-truststore.pin`.

Do not change this key store directly.

`keystore`

This Java Key Store holds the private key and server certificate, `server-cert`, used to protect TLS/SSL communications with client applications. The password, stored in `keystore.pin`, is also the key password for `server-cert`.

`truststore`

This Java Key Store holds a copy of the `server-cert` certificate from the `keystore`. This is also where you import certificates of client applications if you want OpenDJ to recognize them. The password is the same as for the `keystore`, in other words the string in `keystore.pin`.

Tip

Examples in this chapter use self-signed certificates, but you can also use certificates signed by a Certificate Authority (CA).

When importing a certificate (**keytool -import**) signed by a well-known CA, use the `-trustcacerts` option to trust the CA certificates delivered with the Java runtime environment.

Procedure 23.1. To Replace a Server Key Pair

This procedure shows how to replace a server key pair in the `admin-keystore` and copy of the administrative certificate in `admin-truststore`.

The examples also apply when replacing a key pair in the `keystore` and copy of the server certificate in `truststore`. Just adapt the commands to use the correct key store, trust store, and PIN file names.

This procedure does not apply for replication key pairs. Instead, see Procedure 23.2, “To Replace the Key Pair Used for Replication”.

1. Check the alias of the key pair and certificate copy to replace.

   ```
   $ cd /path/to/opendj/config
   $ keytool -list -keystore admin-keystore -storepass `cat admin-keystore.pin`

   Keystore type: JKS
   Keystore provider: SUN

   Your keystore contains 1 entry

   admin-cert, Mar 15, 2013, PrivateKeyEntry,
   Certificate fingerprint (SHA1): 54:9F:C3:F8:7B:B6:...:0A:98:D0:17:8E
   $ keytool -list -keystore admin-truststore -storepass `cat admin-keystore.pin`

   Keystore type: JKS
   Keystore provider: SUN

   Your keystore contains 1 entry

   admin-cert, Mar 15, 2013, trustedCertEntry,
   Certificate fingerprint (SHA1): 54:9F:C3:F8:7B:B6:...:0A:98:D0:17:8E
   ```

   This alias is also stored in the server configuration.

2. Remove the key pair and certificate copy to replace.

   ```
   $ keytool
    -delete
    -alias admin-cert
    -keystore admin-keystore
    -storepass `cat admin-keystore.pin`
   $ keytool
    -delete
    -alias admin-cert
    -keystore admin-truststore
    -storepass `cat admin-keystore.pin`
   ```

3. Generate a new key pair in the key store.

   ```
   $ keytool
    -genkey
    -alias admin-cert
    -keyalg RSA
    -validity 7300
    -keysize 2048
    -dname "CN=opendj.example.com, O=Administration Connector Self-Signed Certificate"
    -keystore admin-keystore
    -storepass `cat admin-keystore.pin`
    -keypass `cat admin-keystore.pin`
   ```

   Notice that the `-alias` option takes the same alias as before. This is because the `ssl-cert-nickname` for the Administration Connector is configured as `admin-cert`. Also, the `-dname` option has a CN value corresponding to the fully-qualified domain name of the host where OpenDJ directory server is running.

4. Get the new key pair's certificate signed, using one of the following alternatives.

   - Self-sign the certificate.

     ```
     $ keytool
      -selfcert
      -alias admin-cert
      -keystore admin-keystore
      -storepass `cat admin-keystore.pin`
     ```

   - Create a certificate signing request, have it signed by a CA, and import the signed certificate from the CA reply.

     For examples of the **keytool** commands to use, see the procedure *To Request and Install a CA-Signed Certificate*.

5. Export a copy of the certificate from the key store.

   ```
   $ keytool
    -export
    -alias admin-cert
    -keystore admin-keystore
    -storepass `cat admin-keystore.pin`
    -file admin-cert.crt
   Certificate stored in file <admin-cert.crt>
   ```

6. Import the copy of the certificate into the trust store.

```
$ keytool
 -import
 -alias admin-cert
 -keystore admin-truststore
 -storepass `cat admin-keystore.pin`
 -file admin-cert.crt
Owner: CN=opendj.example.com, O=Administration Connector Self-Signed Certificate
Issuer: CN=opendj.example.com, O=Administration Connector Self-Signed Certificate
Serial number: 904fc2b
Valid from: Fri Mar 15 15:15:20 CET 2013 until: Thu Jun 13 16:15:20 CEST 2013
Certificate fingerprints:
  MD5:  DD:2A:A1:3A:39:87:DF:02:15:A4:8A:9D:77:89:F1:E4
  SHA1: E1:99:82:92:D7:9B:28:B7:93:D2:B5:5B:C9:DA:4E:D2:62:C2:E7:B0
  SHA256: C5:34:9C:04:E2:87:A9:B1:72:B5:...:99:86:3A:02:28:D0:AB:02:5F:F4:BE
  Signature algorithm name: SHA256withRSA
  Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: FE 33 69 67 FF E8 64 F6   D3 FB CD 14 1C D3 01 44  .3ig..d........D
0010: EE 62 40 DD                                        .b@.
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

7. Restart OpenDJ to make sure it reloads the key stores.

```
$ cd /path/to/opendj/bin
$ stop-ds --restart
```

8. If you have client applications trusting the self-signed certificate, have them import the new one (`admin-cert.crt` in this example).

Procedure 23.2. To Replace the Key Pair Used for Replication

Follow these steps to replace the key pair that is used to secure replication connections.

1. Generate a new key pair for the server.

   The changes you perform are replicated across the topology.

   OpenDJ has an `ads-certificate` and private key, which is a local copy of the key pair used to secure replication connections.

   To generate the new key pair, you remove the `ads-certificate` key pair, prompt OpenDJ to generate a new `ads-certificate` key pair, and then add a copy to the administrative data using the MD5 fingerprint of the certificate to define the RDN.

a. Delete the `ads-certificate` entry.

```
$ ldapmodify
 --port 1389
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
dn: ds-cfg-key-id=ads-certificate,cn=ads-truststore
changetype: delete

Processing DELETE request for ds-cfg-key-id=ads-certificate,cn=ads-truststore
DELETE operation successful for DN ds-cfg-key-id=ads-certificate,
 cn=ads-truststore
```

b. Prompt OpenDJ to generate a new, self-signed `ads-certificate` key pair.

You do this by adding an `ads-certificate` entry with object class `ds-cfg-self-signed-cert-request`.

```
$ ldapmodify
 --port 1389
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
dn: ds-cfg-key-id=ads-certificate,cn=ads-truststore
changetype: add
objectclass: ds-cfg-self-signed-cert-request

Processing ADD request for ds-cfg-key-id=ads-certificate,cn=ads-truststore
ADD operation successful for DN ds-cfg-key-id=ads-certificate,cn=ads-truststore
```

c. Retrieve the `ads-certificate` entry.

```
$ ldapsearch
 --port 1389
 --hostname opendj.example.com
 --baseDN cn=ads-truststore
 "(ds-cfg-key-id=ads-certificate)"
dn: ds-cfg-key-id=ads-certificate,cn=ads-truststore
ds-cfg-key-id: ads-certificate
ds-cfg-public-key-certificate;binary:: MIIB6zCCAVSgAwIBAgIEDKSUFjANBgkqhkiG9w0BA
 QUFADA6MRswGQYDVQQKExJPcGVuREogQ2VydGlmaWNhdGUxGzAZBgNVBAMTEm9wZW5hbS5leGFtcGxl
 LmNvbTAeFw0xMzAyMDcxMDMwMzNaFw0zMzAyMDIxMDMwMzNaMDoxGzAZBgNVBAoTEk9wZW5ESiBDZXJ
 0aWZpY2F0ZTEbMBkGA1UEAxMSb3BlbmFtLmV4YW1wbGUuY29tMIGfMA0GCSqGSIb3DQEBAQUAA4GNAD
 CBiQKBgQCfGLAiUOz4sC8CM9T5DPTk9V9ErNC8N59XwBt1aN7UjhQl4/JZZsetubtUrZBLS9cRrnYdZ
 cpFgLQNEmXifS+PdZ0DJkaLNFmd8ZX0spX8++fb4SkkggkmNRmi1fccDQ/DHMlwl7kk884lXummrzcD
 GbZ7p4vnY7y7GmD1vZSP+wIDAQABMA0GCSqGSIb3DQEBBQUAA4GBAJciUzUP8T8A9VV6dQB0SYCNG1o
 7IvpE7jGVZh6KvM0m5sBNX3wPbTVJQNij3TDm8nx6yhi6DUkpiAZfz/OBL5k+WSw80TjpIZ2+klhP1s
 srsST4Um4fHzDZXOXHR6NM83XxZBsR6MazYecL8CiGwnYW2AeBapzbAnGn1J831q1q
objectClass: top
objectClass: ds-cfg-instance-key
```

d. Retrieve the MD5 fingerprint of the `ads-certificate`.

In this example, the MD5 fingerprint is `07:35:80:D8:F3:CE:E1:39:9C:D0:73:DB:6C:FA:CC:1C`.

```
$ keytool
 -list
 -v
 -alias ads-certificate
 -keystore /path/to/opendj/config/ads-truststore
 -storepass `cat /path/to/opendj/config/ads-truststore.pin`
Alias name: ads-certificate
Creation date: Feb 7, 2013
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=opendj.example.com, O=OpenDJ Certificate
Issuer: CN=opendj.example.com, O=OpenDJ Certificate
Serial number: ca49416
Valid from: Thu Feb 07 11:30:33 CET 2013 until: Wed Feb 02 11:30:33 CET 2033
Certificate fingerprints:
  MD5:  07:35:80:D8:F3:CE:E1:39:9C:D0:73:DB:6C:FA:CC:1C
  SHA1: 56:30:F6:79:AA:C0:BD:61:88:3E:FB:38:38:9D:84:70:0B:E4:43:57
  SHA256: A8:4B:81:EE:30:2A:0C:09:2E:...:C1:41:F5:AB:19:C6:EE:AB:50:64
  Signature algorithm name: SHA1withRSA
  Version: 3
```

e. Using the MD5 fingerprint and the certificate entry, prepare LDIF to update `cn=admin data` with the new server certificate.

```
$ cat /path/to/update-server-cert.ldif
dn: ds-cfg-key-id=073580D8F3CEE1399CD073DB6CFACC1C,cn=instance keys,
 cn=admin data
changetype: add
ds-cfg-key-id: 073580D8F3CEE1399CD073DB6CFACC1C
ds-cfg-public-key-certificate;binary:: MIIB6zCCAVSgAwIBAgIEDKSUFjANBgkqhkiG9w0BA
 QUFADA6MRswGQYDVQQKExJPcGVuREogQ2VydGlmaWNhdGUxGzAZBgNVBAMTEm9wZW5hbS5leGFtcGxl
 LmNvbTAeFw0xMzAyMDcxMDMwMzNaFw0zMzAyMDIxMDMwMzNaMDoxGzAZBgNVBAoTEk9wZW5ESiBDZXJ
 0aWZpY2F0ZTEbMBkGA1UEAxMSb3BlbmFtLmV4YW1wbGUuY29tMIGfMA0GCSqGSIb3DQEBAQUAA4GNAD
 CBiQKBgQCfGLAiUOz4sC8CM9T5DPTk9V9ErNC8N59XwBt1aN7UjhQl4/JZZsetubtUrZBLS9cRrnYdZ
 cpFgLQNEmXifS+PdZ0DJkaLNFmd8ZX0spX8++fb4SkkggkmNRmi1fccDQ/DHMlwl7kk884lXummrzcD
 GbZ7p4vnY7y7GmD1vZSP+wIDAQABMA0GCSqGSIb3DQEBBQUAA4GBAJciUzUP8T8A9VV6dQB0SYCNG1o
 7IvpE7jGVZh6KvM0m5sBNX3wPbTVJQNij3TDm8nx6yhi6DUkpiAZfz/OBL5k+WSw80TjpIZ2+klhP1s
 srsST4Um4fHzDZXOXHR6NM83XxZBsR6MazYecL8CiGwnYW2AeBapzbAnGn1J831q1q
objectClass: top
objectClass: ds-cfg-instance-key

dn: cn=opendj.example.com:4444,cn=Servers,cn=admin data
changetype: modify
replace: ds-cfg-key-id
ds-cfg-key-id: 073580D8F3CEE1399CD073DB6CFACC1C
```

f. Update the administrative data, causing OpenDJ to create a copy of the new `ads-certificate` with its MD5 signature as the alias in the `ads-truststore`.

```
$ ldapmodify
 --port 1389
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --filename /path/to/update-server-cert.ldif
Processing ADD request for ds-cfg-key-id=073580D8F3CEE1399CD073DB6CFACC1C,
 cn=instance keys,cn=admin data
ADD operation successful for DN ds-cfg-key-id=073580D8F3CEE1399CD073DB6CFACC1C,
 cn=instance keys,cn=admin data
Processing MODIFY request for cn=opendj.example.com:4444,cn=Servers,
 cn=admin data
MODIFY operation successful for DN cn=opendj.example.com:4444,cn=Servers,
 cn=admin data
```

2. Force OpenDJ to reopen replication connections using the new key pair.

   Stop replication temporarily and then start it again as described in the *Administration Guide* section on *Configuring Replication*.

```
$ dsconfig
 set-synchronization-provider-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --set enabled:false
 --no-prompt
$ dsconfig
 set-synchronization-provider-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --provider-name "Multimaster Synchronization"
 --set enabled:true
 --no-prompt
```

Chapter 24

Moving Servers

When you change where OpenDJ is deployed, you must take host names, port numbers, and certificates into account. The changes can also affect your replication configuration. This chapter shows what to do when moving a server.

24.1 Overview

From time to time you might change server hardware, file system layout, or host names. At those times you move the services running on the system. You can move OpenDJ data between servers and operating systems. Most of the configuration is also portable.

Two aspects of the configuration are not portable.

- Server certificates contain the host name of the system. Even if you did not set up secure communications when you installed the server, the server still has a certificate used for secure communications on the administrative port.

  To resolve the issue with server certificates, you can change the server certificates during the move as described in this chapter.

- Replication configuration includes the host name and administrative port numbers.

You can work around the issue with replication configuration by disabling replication for the server before the move, and then enabling and initializing replication again after the move.

24.2 Before You Move

Take a moment to determine whether you find it quicker and easier to move your server, or instead to recreate a copy. To recreate a copy, install a new server, set up the new server configuration to match the old, and then copy only the data from the old server to the new server, initializing replication from existing data, or even from LDIF if your database is not too large.

After you decide to move a server, start by taking it out of service. Taking it out of service means directing client applications elsewhere, and then preventing updates from client applications, and finally disabling replication, too. Directing client applications elsewhere depends on your network configuration and possibly on your client application configuration. The other two steps can be completed with the **dsconfig** and **dsreplication** commands.

Procedure 24.1. To Take the Server Out of Service

1. Direct client applications to other servers.

   How you do this depends on your network and client application configurations.

2. Prevent the server from accepting updates from client applications.

   ```
   $ dsconfig
    set-global-configuration-prop
    --port 4444
    --hostname opendj2.example.com
    --bindDN "cn=Directory Manager"
    --bindPassword password
    --set writability-mode:internal-only
    --trustAll
    --no-prompt
   ```

3. Disable replication for the server.

```
$ dsreplication
 disable
 --disableAll
 --port 4444
 --hostname opendj2.example.com
 --adminUID admin
 --adminPassword password
 --trustAll
 --no-prompt
Establishing connections ..... Done.
Disabling replication on base DN dc=example,dc=com of server
 opendj2.example.com:4444 ..... Done.
Disabling replication on base DN cn=admin data of server
 opendj2.example.com:4444 ..... Done.
Disabling replication on base DN cn=schema of server
 opendj2.example.com:4444 ..... Done.
Disabling replication port 8989 of server opendj2.example.com:4444 ..... Done.
Removing registration information ..... Done.
Removing truststore information ..... Done.

See
/var/.../opends-replication-3173475478874782719.log
for a detailed log of this operation.
```

4. With the server no longer receiving traffic or accepting updates from clients, and no longer replicating to other servers, you can shut it down in preparation for the move.

```
$ stop-ds
Stopping Server...

... msg=The Directory Server is now stopped
```

5. (Optional) You might also choose to remove extra log files from the server `logs/` directory before moving the server.

24.3 Moving a Server

Now that you have decided to move your server, and prepared for the move, you must not only move the files but also fix the configuration and the server certificates, and then enable replication.

Procedure 24.2. To Move the Server

1. Move the contents of the server installation directory to the new location.

2. (Optional) If you must change port numbers, edit the port numbers in `config/config.ldif`, carefully avoiding changing any whitespace or other lines in the file.

3. Change server certificates as described in the chapter on *Changing Server Certificates*.

4. Start the server.

```
$ start-ds
... The Directory Server has started successfully
```

5. Enable and initialize replication.

```
$ dsreplication
 enable
 --adminUID admin
 --bindPassword password
 --baseDN dc=example,dc=com
 --host1 opendj.example.com
 --port1 4444
 --bindDN1 "cn=Directory Manager"
 --bindPassword1 password
 --replicationPort1 8989
 --host2 opendj2.example.com
 --port2 4444
 --bindDN2 "cn=Directory Manager"
 --bindPassword2 password
 --replicationPort2 8989
 --trustAll
 --no-prompt

Establishing connections ..... Done.
Checking registration information ..... Done.
Configuring Replication port on server opendj.example.com:4444 ..... Done.
Updating remote references on server opendj2.example.com:4444 ..... Done.
Updating replication configuration for baseDN dc=example,dc=com on server
 opendj.example.com:4444 ..... Done.
Updating replication configuration for baseDN dc=example,dc=com on server
 opendj2.example.com:4444 ..... Done.
Updating registration configuration on server
 opendj.example.com:4444 ..... Done.
Updating registration configuration on server
 opendj2.example.com:4444 ..... Done.
Updating replication configuration for baseDN cn=schema on server
 opendj.example.com:4444 ..... Done.
Updating replication configuration for baseDN cn=schema on server
 opendj2.example.com:4444 ..... Done.
Initializing registration information on server opendj.example.com:4444 with
 the contents of server opendj2.example.com:4444 ..... Done.
Initializing schema on server opendj2.example.com:4444 with the contents of
 server opendj.example.com:4444 ..... Done.

Replication has been successfully enabled.  Note that for replication to work
 you must initialize the contents of the base DN's that are being replicated
 (use dsreplication initialize to do so).

See /tmp/opends-replication-1476402020764482023.log for a detailed log of this
operation.

$ dsreplication
 pre-external-initialization
 --adminUID admin
 --bindPassword password
 --port 4444
 --baseDN dc=example,dc=com
 --trustAll
 --no-prompt

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 DN's on
 all the replicated servers.  You can use the command import-ldif or the binary
 copy to do so.  You must use the same LDIF file or binary copy on each server.

When the initialization is completed you must use the subcommand
 'post-external-initialization' for replication to work with the new base DN's
 contents.
$ dsreplication
 post-external-initialization
 --adminUID admin
 --bindPassword password
 --port 4444
 --baseDN dc=example,dc=com
 --trustAll
 --no-prompt

Updating replication information on base DN dc=example,dc=com ..... Done.

Post initialization procedure completed successfully.
```

6. Accept updates from client applications.

```
$ dsconfig
 set-global-configuration-prop
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set writability-mode:enabled
 --trustAll
 --no-prompt
```

7. Direct client applications to the server.

Chapter 25

Troubleshooting Server Problems

This chapter describes how to troubleshoot common server problems, and how to collect information necessary when seeking support help.

25.1 Identifying the Problem

In order to solve your problem methodically, save time by defining the problem clearly up front. In a replicated environment with multiple directory servers and many client applications, it can be particularly important to pin down not only the problem (difference in observed behavior compared to expected behavior), but also the circumstances and steps that lead to the problem occurring.

Answer the following questions.

- How do you reproduce the problem?
- What exactly is the problem? In other words, what is the behavior you expected? What is the behavior you observed?
- When did the problem start occurring? Under similar circumstances, when does the problem not occur?
- Is the problem permanent? Intermittent? Is it getting worse? Getting better? Staying the same?

Pinpointing the problem can sometimes indicate where you should start looking for solutions.

25.2 Troubleshooting Installation & Upgrade

Installation and upgrade procedures result in a log file tracing the operation. The log location differs by operating system, but look for lines in the command output of the following form.

```
See /var/....log for a detailed log of this operation.
```

25.3 Resetting Administrator Passwords

This section describes what to do if you forgot the password for Directory Manager or for the global (replication) administrator.

Procedure 25.1. Resetting the Directory Manager's Password

OpenDJ directory server stores the entry for Directory Manager in the LDIF representation of its configuration. You must be able to edit directory server files in order to reset Directory Manager's password.

1. Generate the encoded version of the new password using the OpenDJ **encode-password** command.

```
$ cd /path/to/opendj/bin/
$ ./encode-password --storageScheme SSHA512 --clearPassword password
Encoded Password:  "{SSHA512}yWqHnYV4a5llPvE7WHLe5jzK27oZQWLIlVcs9gySu4TyZJMg
 NQNRtnR/Xx2xces1wu1dVLI9jVVtl1W4BVsmOKjyjr0rWrHt"
```

2. Stop OpenDJ directory server while you edit the configuration.

```
$ ./stop-ds
```

3. Find Directory Manager's entry, which has DN `cn=Directory Manager,cn=Root DNs,cn=config`, in `/path/to/opendj/config/config.ldif`, and carefully replace the `userpassword` attribute value with the encoded version of the new password, taking care not to leave any whitespace at the end of the line.

```
dn: cn=Directory Manager,cn=Root DNs,cn=config
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: ds-cfg-root-dn-user
objectClass: top
userpassword: {SSHA512}yWqHnYV4a5llPvE7WHLe5jzK27oZQWLIlVcs9gySu4TyZJMg
 NQNRtnR/Xx2xces1wu1dVLI9jVVtl1W4BVsmOKjyjr0rWrHt
givenName: Directory
cn: Directory Manager
ds-cfg-alternate-bind-dn: cn=Directory Manager
sn: Manager
ds-pwp-password-policy-dn: cn=Root Password Policy,cn=Password Policies
 ,cn=config
ds-rlim-time-limit: 0
ds-rlim-lookthrough-limit: 0
ds-rlim-idle-time-limit: 0
ds-rlim-size-limit: 0
```

4. Start OpenDJ directory server again.

```
$ ./start-ds
```

5. Verify that you can administer the server as Directory Manager using the new password.

```
$ ./dsconfig -p 4444 -h `hostname` -D "cn=Directory Manager" -w password


>>>> OpenDJ configuration console main menu

What do you want to configure?

...

Enter choice: q
```

Procedure 25.2. To Reset the Global Administrator's Password

When you enable replication, part of the process involves creating a global administrator and setting that user's password. This user is present on all replicas. If you chose default values, this user has DN `cn=admin, cn=Administrators,cn=admin data`. You reset the password as you would for any other user, though you do so as Directory Manager.

1. Use the **ldappasswordmodify** command to reset the global administrator's password

```
$ cd /path/to/opendj/bin/
$ ./ldappasswordmodify
 --useStartTLS
 --port 1389
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --authzID "cn=admin,cn=Administrators,cn=admin data"
 --newPassword password
The LDAP password modify operation was successful
```

2. Let replication copy the password change to other replicas.

25.4 Enabling Debug Logging

OpenDJ can write debug information and stack traces to the server debug log. What is logged depends both on debug targets that you create, and also on the debug level that you choose.

Procedure 25.3. To Configure Debug Logging

1. Enable the debug log, `opendj/logs/debug`, which is not enabled by default.

```
$ dsconfig
 set-log-publisher-prop
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --publisher-name "File-Based Debug Logger"
 --set enabled:true
 --set default-debug-level:all
 --no-prompt
 --trustAll
```

   You can set `default-debug-level` to a less verbose level if necessary.

2. Create a debug target or targets.

   No debug targets are enabled by default.

```
$ dsconfig
 list-debug-targets
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --publisher-name "File-Based Debug Logger"
 --no-prompt
 --trustAll

Debug Target : debug-level : debug-category
-------------:-------------:---------------

$
```

A debug target specifies a fully-qualified OpenDJ Java package, class, or method for which to log debug messages at the level you specify.

```
$ dsconfig
 create-debug-target
 --hostname opendj.example.com
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --publisher-name "File-Based Debug Logger"
 --type generic
 --target-name org.opends.server.api
 --set debug-level:all
 --no-prompt
 --trustAll
```

3. Restart OpenDJ to see debug messages in the log.

```
$ /path/to/opendj/bin/stop-ds --restart
...
$ tail -f /path/to/opendj/logs/debug
...
```

If you have set `debug-level:all`, OpenDJ generates a great deal of output in the debug log file. Use debug logging very sparingly on production systems.

25.5 Preventing Access While You Fix Issues

Misconfiguration can potentially put OpenDJ in a state where you must intervene, and where you need to prevent users and applications from accessing the directory until you are done fixing the problem.

OpenDJ provides a *lockdown mode* that allows connections only on the loopback address, and allows only operations requested by root users, such as `cn=Directory Manager`. You can use lockdown mode to prevent all but administrative access to OpenDJ in order to repair the server.

To put OpenDJ into lockdown mode, the server must be running. You cause the server to enter lockdown mode by using a task. Notice that the modify operation is performed over the loopback address (accessing OpenDJ on the local host).

```
$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
dn: ds-task-id=Enter Lockdown Mode,cn=Scheduled Tasks,cn=tasks
objectClass: top
objectClass: ds-task
ds-task-id: Enter Lockdown Mode
ds-task-class-name: org.opends.server.tasks.EnterLockdownModeTask

Processing ADD request for
 ds-task-id=Enter Lockdown Mode,cn=Scheduled Tasks,cn=tasks
ADD operation successful for DN
 ds-task-id=Enter Lockdown Mode,cn=Scheduled Tasks,cn=tasks
```

OpenDJ logs a notice message in `logs/errors` when lockdown mode takes effect.

```
[30/Jan/2012:17:04:32 +0100] category=BACKEND severity=NOTICE msgID=9896350
 msg=Lockdown task Enter Lockdown Mode finished execution
```

Client applications that request operations get a message concerning lockdown mode.

```
$ ldapsearch --port 1389 --baseDN "" --searchScope base "(objectclass=*)" +
SEARCH operation failed
Result Code:  53 (Unwilling to Perform)
Additional Information:  Rejecting the requested operation because the server
 is in lockdown mode and will only accept requests from root users over
 loopback connections
```

You also leave lockdown mode by using a task.

```
$ ldapmodify
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --defaultAdd
dn: ds-task-id=Leave Lockdown Mode,cn=Scheduled Tasks,cn=tasks
objectClass: top
objectClass: ds-task
ds-task-id: Leave Lockdown Mode
ds-task-class-name: org.opends.server.tasks.LeaveLockdownModeTask

Processing ADD request for
 ds-task-id=Leave Lockdown Mode,cn=Scheduled Tasks,cn=tasks
ADD operation successful for DN
 ds-task-id=Leave Lockdown Mode,cn=Scheduled Tasks,cn=tasks
```

OpenDJ also logs a notice message when leaving lockdown.

```
[30/Jan/2012:17:13:05 +0100] category=BACKEND severity=NOTICE msgID=9896350
 msg=Leave Lockdown task Leave Lockdown Mode finished execution
```

25.6 Troubleshooting LDIF Import

By default OpenDJ requires that LDIF data you import respect standards. In particular, OpenDJ is set to check that entries to import match the schema defined for the server. You can temporarily bypass this check by using the `--skipSchemaValidation` with the **import-ldif** command.

OpenDJ also ensures by default that entries have only one structural object class. You can relax this behavior by using the advanced global configuration property, `single-structural-objectclass-behavior`. This can be useful when importing data exported from Sun Directory Server. For example, to warn when entries have more than one structural object class instead of reject such entries being added, set `single-structural-objectclass-behavior:warn` as follows.

```
$ dsconfig
 set-global-configuration-prop
 --port 4444
 --hostname `hostname`
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --set single-structural-objectclass-behavior:warn
 --trustAll
 --no-prompt
```

By default, OpenDJ also checks syntax for a number of attribute types. You can relax this behavior as well by using the **dsconfig set-attribute-syntax-prop** command. See the list of attribute syntaxes and use the `--help` option for further information.

When running **import-ldif**, you can use the `-R` *`rejectFile`* option to capture entries that could not be imported, and the `--countRejects` option to return the number of rejected entries as the **import-ldif** exit code.

Once you work through the issues with your LDIF data, reinstate the default behavior to ensure automated checking.

25.7 Troubleshooting TLS/SSL Connections

In order to trust the server certificate, client applications usually compare the signature on certificates with those of the Certificate Authorities (CAs) whose certificates are distributed with the client software. For example, the Java environment is distributed with a key store holding many CA certificates.

```
$ keytool -list -keystore $JAVA_HOME/lib/security/cacerts -storepass changeit
 | wc -l
     334
```

The self-signed server certificates that can be configured during OpenDJ setup are not recognized as being signed by any CAs. Your software therefore is configured not to trust the self-signed certificates by default. You must either configure the client applications to accept the self-signed certificates, or else use certificates signed by recognized CAs.

You can further debug the network traffic by collecting debug traces. To see the traffic going over TLS/SSL in debug mode, configure OpenDJ to dump debug traces from `javax.net.debug` into the `logs/server.out` file.

```
OPENDJ_JAVA_ARGS="-Djavax.net.debug=all" start-ds
```

25.7.1 Troubleshooting Certificates & SSL Authentication

Replication uses SSL to protect directory data on the network. In some configurations, replica can fail to connect to each other due to SSL handshake errors. This leads to error log messages such as the following.

```
[21/Nov/2011:13:03:20 -0600] category=SYNC severity=NOTICE
 msgID=15138921 msg=SSL connection attempt from myserver (123.456.789.012)
 failed: Remote host closed connection during handshake
```

Notice these problem characteristics in the message above.

- The host name, `myserver`, is not fully qualified.

  You should not see non fully qualified host names in the error logs. Non fully qualified host names are a sign that an OpenDJ server has not been configured properly.

  Always install and configure OpenDJ using fully-qualified host names. The OpenDJ administration connector, which is used by the **dsconfig** command, and also replication depend upon SSL and, more specifically, self-signed certificates for establishing SSL connections. If the host name used for connection establishment does not correspond to the host name stored in the SSL certificate then the SSL handshake can fail. For the purposes of establishing the SSL connection, a host name like `myserver` does not match `myserver.example.com`, and vice versa.

- The connection succeeded, but the SSL handshake failed, suggesting a problem with authentication or with the cipher or protocol negotiation. As most deployments use the same Java Virtual Machine, and the same JVM configuration for each replica, the problem is likely not related to SSL cipher or protocol negotiation, but instead lies with authentication.

Follow these steps on each OpenDJ server to check whether the problem lies with the host name configuration.

1. Make sure each OpenDJ server uses only fully qualified host names in the replication configuration. You can obtain a quick summary by running the following command against each server's configuration.

   ```
   $ grep ds-cfg-replication-server: config/config.ldif | sort | uniq
   ```

2. Make sure that the host names in OpenDJ certificates also contain fully qualified host names, and correspond to the host names found in the previous step.

   ```
   # Examine the certificates used for the administration connector.
   $ keytool -list -v -keystore config/admin-truststore
    -storepass `cat config/admin-keystore.pin` |grep "^Owner:"

   # Examine the certificates used for replication.
   $ keytool -list -v -keystore config/ads-truststore
    -storepass `cat config/ads-truststore.pin`| grep "^Owner:"
   ```

Sample output for a server on host `opendj.example.com` follows.

```
$ grep ds-cfg-replication-server: config/config.ldif |sort | uniq
ds-cfg-replication-server: opendj.example.com:8989
ds-cfg-replication-server: opendj.example.com:9989

$ keytool -list -v -keystore config/admin-truststore
-storepass `cat config/admin-keystore.pin` | grep "^Owner:"
Owner: CN=opendj.example.com, O=Administration Connector Self-Signed Certificate

$ keytool -list -v -keystore config/ads-truststore
 -storepass `cat config/ads-truststore.pin`| grep "^Owner:"
Owner: CN=opendj.example.com, O=OpenDJ Certificate
Owner: CN=opendj.example.com, O=OpenDJ Certificate
Owner: CN=opendj.example.com, O=OpenDJ Certificate
```

Unfortunately there is no easy solution to badly configured host names. It is often easier and quicker simply to reinstall your OpenDJ servers remembering to use fully qualified host names everywhere.

- When using the **setup** tool to install and configure a server ensure that the `-h` option is included, and that it specifies the fully qualified host name. Make sure you include this option even if you are not enabling SSL/StartTLS LDAP connections (see OPENDJ-363).

  If you are using the GUI installer, then make sure you specify the fully qualified host name on the first page of the wizard.

- When using the **dsreplication** tool to enable replication make sure that any `--host` options include the fully qualified host name.

If you cannot reinstall the server, follow these steps.

1. Disable replication in each replica.

```
$ dsreplication
 disable
 --disableAll
 --port adminPort
 --hostname hostName
 --bindDN "cn=Directory Manager"
 --adminPassword password
 --trustAll
 --no-prompt
```

2. Stop and restart each server in order to clear the in-memory ADS trust store backend.

3. Enable replication making certain that fully qualified host names are used throughout

```
$ dsreplication
 enable
 --adminUID admin
 --adminPassword password
 --baseDN dc=example,dc=com
 --host1 hostName1
 --port1 adminPort1
 --bindDN1 "cn=Directory Manager"
 --bindPassword1 password
 --replicationPort1 replPort1
 --host2 hostName2
 --port2 adminPort2
 --bindDN2 "cn=Directory Manager"
 --bindPassword2 password
 --replicationPort2 replPort2
 --trustAll
 --no-prompt
```

4. Repeat the previous step for each remaining replica. In other words, host1 with host2, host1 with host3, host1 with host4, ..., host1 with hostN.

5. Initialize all remaining replica with the data from host1.

```
$ dsreplication
 initialize-all
 --adminUID admin
 --adminPassword password
 --baseDN dc=example,dc=com
 --hostname hostName1
 --port 4444
 --trustAll
 --no-prompt
```

6. Check that the host names are correct in the configuration and in the key stores by following the steps you used to check for host name problems. The

only broken host name remaining should be in the key and trust stores for the administration connector.

```
$ keytool -list -v -keystore config/admin-truststore
 -storepass `cat config/admin-keystore.pin` |grep "^Owner:"
```

7. Stop each server, and then fix the remaining admin connector certificate as described here in the procedure *To Replace a Server Key Pair*.

25.7.2 Handling Compromised Keys

As explained in *Changing Server Certificates*, OpenDJ directory server has different keys and key stores for different purposes. The public keys used for replication are also used to encrypt shared secret symmetric keys for example to encrypt and to sign back ups. This section looks at what to do if either a key pair or secret key is compromised.

How you deal with the problem depends on which key was compromised.

- For a key pair used for a client connection handler and with a certificate signed by a certificate authority (CA), contact the CA for help. The CA might choose to publish a certificate revocation list (CRL) that identifies the certificate of the compromised key pair.

  Also make sure you replace the key pair. See *To Replace a Server Key Pair* for specific steps.

- For a key pair used for a client connection handler and that has a self-signed certificate, follow the steps in *To Replace a Server Key Pair*, and make sure the clients remove the compromised certificate from their trust stores, updating those trust stores with the new certificate.

- For a key pair that is used for replication, mark the key as compromised as described below, and replace the key pair. See *To Replace a Server Key Pair* for specific steps.

  To mark the key pair as compromised, follow these steps.

  1. Identity the key entry by searching administrative data on the server whose key was compromised.

     The server in this example is installed on `opendj.example.com` with administration port `4444`.

```
$ ldapsearch
 --port 1389
 --hostname opendj.example.com
 --baseDN "cn=admin data"
 "(cn=opendj.example.com:4444)" ds-cfg-key-id
dn: cn=opendj.example.com:4444,cn=Servers,cn=admin data
ds-cfg-key-id: 4F2F97979A7C05162CF64C9F73AF66ED
```

The key ID, `4F2F97979A7C05162CF64C9F73AF66ED`, is the RDN of the key entry.

2. Mark the key as compromised by adding the attribute, `ds-cfg-key-compromised-time`, to the key entry.

   The attribute has generalized time syntax, and so takes as its value the time at which the key was compromised expressed in generalized time. In the following example, the key pair was compromised at 8:34 AM UTC on March 21, 2013.

```
$ ldapmodify
 --port 1389
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword password
dn: ds-cfg-key-id=4F2F97979A7C05162CF64C9F73AF66ED,cn=instance keys,cn=admin data
changetype: modify
add: ds-cfg-key-compromised-time
ds-cfg-key-compromised-time: 201303210834Z

Processing MODIFY request for ds-cfg-key-id=4F2F97979A7C05162CF64C9F73AF66ED,
 cn=instance keys,cn=admin data
MODIFY operation successful for DN ds-cfg-key-id=4F2F97979A7C05162CF64C9F73AF66ED
 ,cn=instance keys,cn=admin data
```

3. If the server uses encrypted or signed data, then the shared secret keys used for encryption or signing and associated with the compromised key pair should also be considered compromised. Therefore, mark all shared secret keys encrypted with the instance key as compromised.

   To identify the shared secret keys, find the list of secret keys in the administrative data whose `ds-cfg-symmetric-key` starts with the key ID of the compromised key.

```
$ ldapsearch
 --port 1389
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --baseDN "cn=secret keys,cn=admin data"
 "(ds-cfg-symmetric-key=4F2F97979A7C05162CF64C9F73AF66ED*)" dn
dn: ds-cfg-key-id=fba16e59-2ce1-4619-96e7-8caf33f916c8,cn=secret keys,cn=admin d
 ata

dn: ds-cfg-key-id=57bd8b8b-9cc6-4a29-b42f-fb7a9e48d713,cn=secret keys,cn=admin d
 ata

dn: ds-cfg-key-id=f05e2e6a-5c4b-44d0-b2e8-67a36d304f3a,cn=secret keys,cn=admin d
 ata
```

For each such key, mark the entry with `ds-cfg-key-compromised-time` as shown above for the instance key.

Changes to administration data are replicated to other OpenDJ servers in the replication topology.

- For a shared secret key used for data encryption that has been compromised, mark the key entry with `ds-cfg-key-compromised-time` as shown in the example above that demonstrates marking the instance key as compromised.

  Again, changes to administration data are replicated to other OpenDJ servers in the replication topology.

25.8 Troubleshooting Client Operations

By default OpenDJ logs information about all LDAP client operations in `logs/access`, and all HTTP client operations in `logs/http-access`. The following lines are wrapped for readability, showing a search for the entry with `uid=bjensen` as traced in the LDAP access log. In the access log itself, each line starts with a time stamp.

```
[27/Jun/2011:17:23:00 +0200] CONNECT conn=19 from=127.0.0.1:56641
 to=127.0.0.1:1389 protocol=LDAP
[27/Jun/2011:17:23:00 +0200] SEARCH REQ conn=19 op=0 msgID=1
 base="dc=example,dc=com" scope=wholeSubtree filter="(uid=bjensen)" attrs="ALL"
[27/Jun/2011:17:23:00 +0200] SEARCH RES conn=19 op=0 msgID=1
 result=0 nentries=1 etime=3
[27/Jun/2011:17:23:00 +0200] UNBIND REQ conn=19 op=1 msgID=2
[27/Jun/2011:17:23:00 +0200] DISCONNECT conn=19 reason="Client Unbind"
```

As you see, each client connection and set of LDAP operations are traced, starting with a time stamp and information about the operation performed, then including information about the connection, the operation number for the sequence of operations performed by the client, a message identification number, and additional information about the operation.

To match HTTP client operations with related internal server operations, first prevent OpenDJ from suppressing internal operations from the LDAP access log by using the **dsconfig** command to set the LDAP access log publisher `suppress-internal-operations` advanced property to `false`. Then match the values of the `x-connection-id` field in the HTTP access log with `conn=`*`id`* values in the LDAP access log.

For example, consider an HTTP GET request for the `_id` field of the user `newuser`, which is handled by connection 4 as shown in `logs/http-access`.

```
-  192.168.0.12  bjensen  22/May/2013:16:27:52 +0200
  GET  /users/newuser?_fields=_id  HTTP/1.1  200
  curl/7.21.4  4  12
```

With internal operations logged in `logs/access`, log lines for the related operations have `conn=4`.

```
[22/May/2013:16:27:52 +0200] CONNECT conn=4
  from=192.168.0.12:63593 to=192.168.0.12:8080 protocol=HTTP/1.1
[22/May/2013:16:27:52 +0200] SEARCH REQ conn=4
  op=0 msgID=0 base="ou=people,dc=example,dc=com" scope=wholeSubtree
   filter="(&(objectClass=inetOrgPerson)(uid=bjensen))" attrs="1.1"
[22/May/2013:16:27:52 +0200] SEARCH RES conn=4
  op=0 msgID=0 result=0 nentries=1 etime=5
[22/May/2013:16:27:52 +0200] BIND REQ conn=4
  op=1 msgID=1 version=3 type=SIMPLE
   dn="uid=bjensen,ou=People,dc=example,dc=com"
[22/May/2013:16:27:52 +0200] BIND RES conn=4
  op=1 msgID=1 result=0 authDN="uid=bjensen,ou=People,dc=example,dc=com"
   etime=3
[22/May/2013:16:27:52 +0200] SEARCH REQ conn=4
  op=2 msgID=2 base="uid=newuser,ou=people,dc=example,dc=com" scope=baseObject
   filter="(objectClass=*)" attrs="uid,etag"
[22/May/2013:16:27:52 +0200] SEARCH RES conn=4
   op=2 msgID=2 result=0 nentries=1 etime=4
[22/May/2013:16:27:52 +0200] UNBIND REQ conn=4
   op=3 msgID=3
[22/May/2013:16:27:52 +0200] DISCONNECT conn=4
   reason="Client Unbind"
```

To help diagnose errors due to access permissions, OpenDJ supports the get effective rights control. The control OID, `1.3.6.1.4.1.42.2.27.9.5.2`, is not allowed by the default global ACIs. You must therefore add access to use the get effective rights control when not using it as Directory Manager.

25.8.1 Clients Need Simple Paged Results Control

For Solaris and some versions of Linux you might see a message in the OpenDJ access logs such as the following.

```
The request control with Object Identifier (OID) "1.2.840.113556.1.4.319"
```

```
cannot be used due to insufficient access rights
```

This message means clients are trying to use the simple paged results control without authenticating. By default, OpenDJ includes a global ACI to allow only authenticated users to use the control.

```
$ dsconfig
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword "password"
 get-access-control-handler-prop

Property   : Value(s)
-----------:-------------------------------------------------------------------
enabled    : true
global-aci : (extop="1.3.6.1.4.1.26027.1.6.1 || 1.3.6.1.4.1.26027.1.6.3 ||
...
           : (targetcontrol="1.3.6.1.1.12 || 1.3.6.1.1.13.1 || 1.3.6.1.1.13.2
           : || 1.2.840.113556.1.4.319 || 1.2.826.0.1.3344810.2.3 ||
           : 2.16.840.1.113730.3.4.18 || 2.16.840.1.113730.3.4.9 ||
           : 1.2.840.113556.1.4.473 || 1.3.6.1.4.1.42.2.27.9.5.9") (version
           : 3.0; acl "Authenticated users control access"; allow(read)
           : userdn="ldap:///all";), (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 grant anonymous (unauthenticated) user access to the control, add the OID for the simple paged results control to the list of those in the `Anonymous control access` global ACI.

```
$ dsconfig
 --port 4444
 --hostname opendj.example.com
 --bindDN "cn=Directory Manager"
 --bindPassword "password"
 set-access-control-handler-prop
 --remove global-aci:"(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\";)"
 --add global-aci:"(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 || 1.2.840.113556.1.4.319\")
 (version 3.0; acl \"Anonymous control access\"; allow(read)
 userdn=\"ldap:///anyone\";)"
 --no-prompt
```

Alternatively, stop OpenDJ, edit the corresponding ACI carefully in `/path/to/opendj/config/config.ldif`, and restart OpenDJ. [1]

[1] Unlike the **dsconfig** command, the `config.ldif` file is not a public interface, so this alternative should not be used in production.

25.9 Troubleshooting Replication

Replication can generally recover from conflicts and transient issues. Replication does, however, require that update operations be copied from server to server. It is therefore possible to experience temporary delays while replicas converge, especially when the write operation load is heavy. OpenDJ's tolerance for temporary divergence between replicas is what allows OpenDJ to remain available to serve client applications even when networks linking the replicas go down.

In other words, the fact that directory services are loosely convergent rather than transactional is a feature, not a bug.

That said, you may encounter errors. Replication uses its own error log file, `logs/replication`. Error messages in the log file have `category=SYNC`. The messages have the following form. Here the line is folded for readability.

```
[27/Jun/2011:14:37:48 +0200] category=SYNC severity=INFORMATION msgID=14680169
 msg=Replication server accepted a connection from 10.10.0.10/10.10.0.10:52859
 to local address 0.0.0.0/0.0.0.0:8989 but the SSL handshake failed. This is
 probably benign, but may indicate a transient network outage or a
 misconfigured client application connecting to this replication server.
 The error was: Remote host closed connection during handshake
```

OpenDJ maintains historical information about changes in order to bring replicas up to date, and to resolve replication conflicts. To prevent historical information from growing without limit, OpenDJ purges historical information after a configurable delay (`replication-purge-delay`, default: 3 days). A replica can become irrevocably out of sync if you restore it from a backup archive older than the purge delay, or if you stop it for longer than the purge delay. If this happens to you, disable the replica, and then reinitialize it from a recent backup or from a server that is up to date.

25.10 Asking For Help

When you cannot resolve a problem yourself, and want to ask for help, clearly identify the problem and how you reproduce it, and also the version of OpenDJ you use to reproduce the problem. The version includes both a version number and also a build time stamp.

```
$ dsconfig --version
OpenDJ 4.8.1-SNAPSHOT
Build yyyymmddhhmmssZ
```

Be ready to provide additional information, too.

- The output from the **java -version** command.

- `access` and `errors` logs showing what the server was doing when the problem started occurring
- A copy of the server configuration file, `config/config.ldif`, in use when the problem started occurring
- Other relevant logs or output, such as those from client applications experiencing the problem
- A description of the environment where OpenDJ is running, including system characteristics, host names, IP addresses, Java versions, storage characteristics, and network characteristics. This helps to understand the logs, and other information.

Tools Reference

You can find the tools under the `bin/` or `bat\` folder where you installed OpenDJ directory server. For example, `/path/to/opendj/bin`.

Table of Contents

backup

backup — back up OpenDJ directory data

backup

`backup` {options}

1 Description

This utility can be used to back up one or more directory server backends.

2 Options

The following options are supported.

`-a, --backUpAll`

Back up all backends in the server

`-A, --hash`

Generate a hash of the backup contents

`-B, --incrementalBaseID {backupID}`

Backup ID of the source archive for an incremental backup.

`-c, --compress`

Compress the backup content

`-d, --backupDirectory {backupDir}`

Path to the target directory for the backup file(s)

`-i, --incremental`

Perform an incremental backup rather than a full backup

`-I, --backupID {backupID}`

Use the provided identifier for the backup

`-n, --backendID {backendName}`

Backend ID for the backend to archive

`-s, --signHash`

Sign the hash of the backup contents

`-y, --encrypt`

Encrypt the backup contents

2.1 Task Backend Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server administration port number

Default value: 4444

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

Use `-w -` to have the command prompt for the password, rather than enter the password on the command line.

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

2.2 Task Scheduling Options

`--completionNotify {emailAddress}`

Email address of a recipient to be notified when the task completes. This option may be specified more than once.

`--dependency {taskID}`

ID of a task upon which this task depends. A task will not start execution until all its dependencies have completed execution.

`--errorNotify {emailAddress}`

Email address of a recipient to be notified if an error occurs when this task executes. This option may be specified more than once.

`--failedDependencyAction {action}`

Action this task will take should one if its dependent tasks fail. The value must be one of PROCESS, CANCEL, DISABLE. If not specified defaults to CANCEL.

`--recurringTask {schedulePattern}`

Indicates the task is recurring and will be scheduled according to the value argument expressed in crontab(5) compatible time/date pattern.

`-t, --start {startTime}`

Indicates the date/time at which this operation will start when scheduled as a server task expressed in YYYYMMDDhhmmssZ format for UTC time or YYYYMMDDhhmmss for local time. A value of '0' will cause the task to be scheduled for immediate execution. When this option is specified the operation will be scheduled to start at the specified time after which this utility will exit immediately.

2.3 Utility Input/Output Options

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

2.4 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

1

An error occurred while parsing the command-line arguments.

4 Examples

The following example backs up all user data while the server is online.

```
$ backup -p 4444 -D "cn=Directory Manager" -w password
 -a -d /path/to/opendj/bak -t 0
Backup task 20110613143801866 scheduled to start Jun 13, 2011 2:38:01 PM CEST
```

The following example schedules back up of all user data every night at 2 AM when the server is online, and notifies diradmin@example.com when finished, or on error.

```
$ backup -p 4444 -D "cn=Directory Manager" -w password -a
 -d /path/to/opendj/bak --recurringTask "00 02 * * *"
 --completionNotify diradmin@example.com --errorNotify diradmin@example.com
Recurring Backup task BackupTask-988d6adf-4d65-44bf-8546-6ea74a2480b0
scheduled successfully
```

The following example backs up all user data while the server is offline.

```
$ /path/to/opendj/bin/stop-ds
Stopping Server...
...
$ /path/to/opendj/bin/backup --backupAll --backupDirectory /path/to/opendj/bak
...
[28/Sep/2012:12:14:22 +0200] ... msg=The backup process completed successfully
$ /path/to/opendj/bin/start-ds
[28/Sep/2012:12:15:48 +0200] ... The Directory Server has started successfully
```

base64

base64 — encode and decode base64 strings

base64

`base64` *`subcommand`* {options}

1 Description

This utility can be used to encode and decode information using base64.

2 Subcommands

The following subcommands are supported.

base64 decode

Decode base64-encoded information into raw data

When no options are specified, this subcommand reads from standard input and writes to standard output.

Decode Options

`-d, --encodedData {data}`

The base64-encoded data to be decoded

`-f, --encodedDataFile {path}`

The path to a file containing the base64-encoded data to be decoded

`-o, --toRawFile {path}`

The path to a file to which the raw base64-decoded data should be written

base64 encode

Encode raw data using base64

When no options are specified, this subcommand reads from standard input and writes to standard output.

Decode Options

`-d, --rawData {data}`

The raw data to be base64 encoded

`-f, --rawDataFile {path}`

The path to a file containing the raw data to be base64 encoded

`-o, --toEncodedFile {path}`

The path to a file to which the base64-encoded data should be written

3 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

4 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

5 Examples

The following command shows the changes from the external change log in human-readable format.

```
$ base64 decode -d YWRkOiBkZXNjcmlwdGlvbgpkZXNjcmlwdGlvbjogQSB0aGlyZCBjaGFuZ2UK
LQpyZXBsYWNlOiBtb2RpZmllcnNOYW1lCm1vZGlmaWVyc05hbWU6IGNuPURpcmVjdG9yeSBNYW5hZ2V
yLGNuPVJvb3QgRE5zLGNuPWNvbmZpZwotCnJlcGxhY2U6IG1vZGlmeVRpbWVzdGFtcApb2RpZnlUaW
1lc3RhbXA6IDIwMTEwNjEzMDcxMjEwWgotCg==
add: description
description: A third change
-
replace: modifiersName
modifiersName: cn=Directory Manager,cn=Root DNs,cn=config
-
replace: modifyTimestamp
modifyTimestamp: 20110613071210Z
-
```

control-panel

control-panel — start the OpenDJ graphical admin interface

control-panel

`control-panel` {options}

1 Description

This utility can be used to display the Control Panel window which displays basic server information and allows to do some basic administration tasks on the server.

2 Options

The following options are supported.

`-r, --remote`

Connect to a remote server

2.1 LDAP Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --adminPasswordFile {bindPasswordFile}`

Global administrator password file

`-p, --port {port}`

Directory server administration port number

Default value: 4444

`-w, --adminPassword {bindPassword}`

Password for the global administrator

`-X, --trustAll`

Trust all server SSL certificates

2.2 General Options

`--version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

The following example starts the Control Panel on a remote host.

```
$ control-panel -r -h opendj.example.com -p 4444 &
```

create-rc-script

create-rc-script — script to manage OpenDJ as a service on UNIX

create-rc-script

`create-rc-script` {options}

1 Description

Create an RC script that may be used to start, stop, and restart the directory server on UNIX-based systems.

2 Options

The following options are supported.

`-f, --outputFile {path}`

The path to the output file to create.

`-j, --javaHome {path}`

The path to the Java installation that should be used to run the server.

`-J, --javaArgs {args}`

A set of arguments that should be passed to the JVM when running the server.

`-u, --userName {userName}`

The name of the user account under which the server should run.

2.1 General Options

`--version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

The following example adds a script to start OpenDJ at boot time on a Debian-based system, and then updates the runlevel system to use the script.

```
$ sudo create-rc-script -f /etc/init.d/opendj -u mark
$ sudo update-rc.d opendj
```

dbtest

dbtest — gather OpenDJ JE database debugging information

dbtest

`dbtest` `subcommand` [options]

1 Description

This utility can be used to debug the JE database.

2 Subcommands

The following subcommands are supported.

dbtest dump-database-container

Dump records from a database container

dbtest list-database-containers

List the database containers for an entry container

dbtest list-entry-containers

List the entry containers for a root container

dbtest list-index-status

List the status of indexes in an entry container

dbtest list-root-containers

List the root containers used by all JE backends

3 Options

The following global options are supported.

For other options, see **dbtest *`subcommand`* --help**.

`--version`

Display version information

`-?, -H, --help`

Display usage information

4 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

5 Examples

The following example displays debugging information about the equality index for `sudoUser`.

```
$ dbtest dump-database-container -n userRoot -b dc=example,dc=com
 -d sudoUser.equality
Indexed Value (6 bytes): %admin
Entry ID List (8 bytes): 165

Indexed Value (5 bytes): %sudo
Entry ID List (8 bytes): 164

Indexed Value (4 bytes): root
Entry ID List (8 bytes): 163


Total Records: 3
Total / Average Key Size: 13 bytes / 4 bytes
Total / Average Data Size: 24 bytes / 8 bytes
```

dsconfig

dsconfig — manage OpenDJ directory server configuration

dsconfig

`dsconfig [subcommand]` [options]

1 Description

This utility serves to configure a running directory server.

The **dsconfig** command is the primary command-line tool for viewing and editing OpenDJ configuration. When started without arguments, **dsconfig** prompts you for administration connection information, including the host name, administration port number, administrator bind DN and administrator password. The **dsconfig** command then connects securely to the directory server over the administration port. Once connected it presents you with a menu-driven interface to the server configuration.

When you pass connection information, subcommands, and additional options to **dsconfig**, the command runs in script mode and so is not interactive, though it can prompt you to ask whether to apply changes and whether to trust certificates (unless you use the `--no-prompt` and `--trustAll` options, respectively).

You can prepare **dsconfig** batch scripts by running the tool with the `--commandFilePath` option in interactive mode, then reading from the batch file with the `--batchFile` option in script mode. Batch files can be useful when you have many **dsconfig** commands to run and want to avoid starting the JVM and setting up a new connection for each command.

The **dsconfig** command categorizes directory server configuration into *components*, also called *managed objects*. Actual components often inherit from a parent component type. For example, one component is a Connection Handler. An LDAP Connection Handler is a type of Connection Handler. You configure the LDAP Connection Handler component to specify how OpenDJ directory server handles LDAP connections coming from client applications.

Configuration components have *properties*. For example, the LDAP Connection Handler component has properties such as `listen-port` and `allow-start-tls`. You can set the component's `listen-port` property to `389` to use the default LDAP port number. You can set the component's `allow-start-tls` property to `true` to permit LDAP client applications to use StartTLS. Much of the configuration you do with **dsconfig** involves setting component properties. The *OpenDJ Configuration Reference* covers all **dsconfig** component properties in detail, drawing on the documentation you also view when getting help through the **dsconfig** command.

2 Getting Help

The **dsconfig** command provides many subcommands. Use the following options to view help for subcommands.

See *dsconfig Subcommands* for details of individual subcommands.

dsconfig --help-all

Display all subcommands

dsconfig --help-core-server

Display subcommands relating to core server

dsconfig --help-database

Display subcommands relating to caching and back-ends

dsconfig --help-logging

Display subcommands relating to logging

dsconfig --help-replication

Display subcommands relating to replication

dsconfig --help-security

Display subcommands relating to authentication and authorization

dsconfig --help-user-management

Display subcommands relating to user management

For help with individual subcommands, either use **dsconfig *subcommand* --help**, or start **dsconfig** in interactive mode, without specifying a subcommand.

To view component properties, use the **dsconfig list-properties** command.

3 Generally Applicable Options

The following options are supported for all **dsconfig** subcommands.

`--advanced`

Allows the configuration of advanced components and properties

3.1 LDAP Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-I, --adminUID {adminUID}`

User ID of the global administrator to use to bind to the server. For the **enable** subcommand, if no global administrator was defined previously for any servers, the global administrator will be created using the UID provided.

Default value: admin

`-j, --adminPasswordFile {bindPasswordFile}`

Global administrator password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server administration port number

Default value: 4444

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

Default value: /path/to/opendj/config/admin-truststore

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --adminPassword {bindPassword}`

Password for the global administrator

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

3.2 Utility Input/Output Options

`--commandFilePath {path}`

The full path to the file where the equivalent non-interactive commands will be written when this command is run in interactive mode.

`--displayCommand`

Display the equivalent non-interactive option on standard output when this command is run in interactive mode.

`-F, --batchFilePath {batchFilePath}`

Path to a batch file containing a set of dsconfig commands to be executed

`-n, --no-prompt`

Use non-interactive mode. If data in the command is missing, the user is not prompted and the command exits with an error.

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

`-Q, --quiet`

Do not write progress information to standard output

`-s, --script-friendly`

Use script-friendly mode

`-v, --verbose`

Use verbose mode

3.3 General Options

`--version`

Display version information

`-?, -H, --help`

Display usage information

4 dsconfig Subcommands

This section covers individual **dsconfig** subcommands.

Subcommands let you create, list, and delete entire configuration components, and also let you get and set component properties. Subcommands therefore have names that reflect these five actions.

- create-*component*
- list-*components*
- delete-*component*
- get-*component*-prop
- set-*component*-prop

Component properties for the **dsconfig** command are covered in the *OpenDJ Configuration Reference*.

Many subcommands let you set property values. Notice in the reference for the subcommands below that specific options are available for handling multi-valued properties. Whereas you can assign a single property value using the `--set` option, you assign multiple values to a multi-valued property using the `--add` option. You can reset the values of the multi-valued property using the `--reset` option.

Some property values take a time duration. Durations are expressed as numbers followed by units. For example `1 s` means one second, and `2 w` means two weeks. Some durations have minimum granularity or maximum units, so you cannot necessary specify every duration in milliseconds or weeks for example. Some durations allow you to use a special value to mean unlimited. Units are specified as follows.

- `ms`: milliseconds
- `s`: seconds
- `m`: minutes
- `h`: hours
- `d`: days
- `w`: weeks

4.1 dsconfig create-access-log-filtering-criteria

Creates Access Log Filtering Criteria

`--publisher-name {name}`

The name of the Access Log Publisher

`--criteria-name {name}`

The name of the new Access Log Filtering Criteria

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Access Log Filtering Criteria which should be created (Default: generic). The value for TYPE can be one of: generic

4.2 dsconfig create-account-status-notification-handler

Creates Account Status Notification Handlers

`--handler-name {name}`

The name of the new Account Status Notification Handler

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Account Status Notification Handler which should be created. The value for TYPE can be one of: custom | error-log | smtp

4.3 dsconfig create-alert-handler

Creates Alert Handlers

`--handler-name {name}`

The name of the new Alert Handler

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Alert Handler which should be created. The value for TYPE can be one of: custom | jmx | smtp

4.4 dsconfig create-attribute-syntax

Creates Attribute Syntaxes

`--syntax-name {name}`

The name of the new Attribute Syntax

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Attribute Syntax which should be created (Default: generic). The value for TYPE can be one of: attribute-type-description | certificate | country-string | directory-string | generic | jpeg | telephone-number

4.5 dsconfig create-backend

Creates Backends

`--backend-name {STRING}`

The name of the new Backend which will also be used as the value of the "backend-id" property: Specifies a name to identify the associated backend.

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Backend which should be created. The value for TYPE can be one of: backup | config-file-handler | custom | ldif | local-db | memory | monitor | null | schema | task | trust-store

4.6 dsconfig create-certificate-mapper

Creates Certificate Mappers

`--mapper-name {name}`

The name of the new Certificate Mapper

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Certificate Mapper which should be created. The value for TYPE can be one of: custom | fingerprint | subject-attribute-to-user-attribute | subject-dn-to-user-attribute | subject-equals-dn

4.7 dsconfig create-connection-handler

Creates Connection Handlers

`--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 VALUE 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 | http | jmx | ldap | ldif | snmp

4.8 dsconfig create-debug-target

Creates Debug Targets

`--publisher-name {name}`

The name of the Debug Log Publisher

`--target-name {STRING}`

The name of the new Debug Target which will also be used as the value of the "debug-scope" property: Specifies the fully-qualified OpenDJ Java package, class, or method affected by the settings in this target definition. Use the number character (#) to separate the class name and the method name (that is, org.opends.server.core.DirectoryServer#startUp).

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Debug Target which should be created (Default: generic). The value for TYPE can be one of: generic

4.9 dsconfig create-entry-cache

Creates Entry Caches

`--cache-name {name}`

The name of the new Entry Cache

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Entry Cache which should be created. The value for TYPE can be one of: custom | fifo | file-system | soft-reference

4.10 dsconfig create-extended-operation-handler

Creates Extended Operation Handlers

`--handler-name {name}`

The name of the new Extended Operation Handler

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Extended Operation Handler which should be created. The value for TYPE can be one of: cancel | custom | get-connection-id | get-symmetric-key | password-modify | password-policy-state | start-tls | who-am-i

4.11 dsconfig create-group-implementation

Creates Group Implementations

`--implementation-name {name}`

The name of the new Group Implementation

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Group Implementation which should be created. The value for TYPE can be one of: custom | dynamic | static | virtual-static

4.12 dsconfig create-identity-mapper

Creates Identity Mappers

```
--mapper-name {name}
```

The name of the new Identity Mapper

```
--set {PROP:VALUE}
```

Assigns a value to a property where PROP is the name of the property and VALUE 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 Identity Mapper which should be created. The value for TYPE can be one of: custom | exact-match | regular-expression

4.13 dsconfig create-key-manager-provider

Creates Key Manager Providers

```
--provider-name {name}
```

The name of the new Key Manager Provider

```
--set {PROP:VALUE}
```

Assigns a value to a property where PROP is the name of the property and VALUE 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 Key Manager Provider which should be created. The value for TYPE can be one of: custom | file-based | pkcs11

4.14 dsconfig create-local-db-index

Creates Local DB Indexes

`--backend-name {name}`

The name of the Local DB Backend

`--index-name {OID}`

The name of the new Local DB Index which will also be used as the value of the "attribute" property: Specifies the name of the attribute for which the index is to be maintained.

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Local DB Index which should be created (Default: generic). The value for TYPE can be one of: generic

4.15 dsconfig create-local-db-vlv-index

Creates Local DB VLV Indexes

`--backend-name {name}`

The name of the Local DB Backend

`--index-name {STRING}`

The name of the new Local DB VLV Index which will also be used as the value of the "name" property: Specifies a unique name for this VLV index.

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Local DB VLV Index which should be created (Default: generic). The value for TYPE can be one of: generic

4.16 dsconfig create-log-publisher

Creates Log Publishers

`--publisher-name {name}`

The name of the new Log Publisher

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Log Publisher which should be created. The value for TYPE can be one of: custom-access | custom-debug | custom-error | custom-http-access | file-based-access | file-based-audit | file-based-debug | file-based-error | file-based-http-access

4.17 dsconfig create-log-retention-policy

Creates Log Retention Policies

`--policy-name {name}`

The name of the new Log Retention Policy

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Log Retention Policy which should be created. The value for TYPE can be one of: custom | file-count | free-disk-space | size-limit

4.18 dsconfig create-log-rotation-policy

Creates Log Rotation Policies

`--policy-name {name}`

The name of the new Log Rotation Policy

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Log Rotation Policy which should be created. The value for TYPE can be one of: custom | fixed-time | size-limit | time-limit

4.19 dsconfig create-matching-rule

Creates Matching Rules

`--rule-name {name}`

The name of the new Matching Rule

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Matching Rule which should be created (Default: generic). The value for TYPE can be one of: collation | generic

4.20 dsconfig create-monitor-provider

Creates Monitor Providers

`--provider-name {name}`

The name of the new Monitor Provider

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Monitor Provider which should be created. The value for TYPE can be one of: client-connection | custom | entry-cache | memory-usage | stack-trace | system-info | version

4.21 dsconfig create-password-generator

Creates Password Generators

`--generator-name {name}`

The name of the new Password Generator

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Password Generator which should be created. The value for TYPE can be one of: custom | random

4.22 dsconfig create-password-policy

Creates Authentication Policies

`--policy-name {name}`

The name of the new Authentication Policy

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Authentication Policy which should be created. The value for TYPE can be one of: ldap-pass-through | password-policy

4.23 dsconfig create-password-storage-scheme

Creates Password Storage Schemes

`--scheme-name {name}`

The name of the new Password Storage Scheme

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Password Storage Scheme which should be created. The value for TYPE can be one of: aes | base64 | blowfish | clear | crypt | custom | md5 | pbkdf2 | rc4 | salted-md5 | salted-sha1 | salted-sha256 | salted-sha384 | salted-sha512 | sha1 | triple-des

4.24 dsconfig create-password-validator

Creates Password Validators

`--validator-name {name}`

The name of the new Password Validator

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Password Validator which should be created. The value for TYPE can be one of: attribute-value | character-set | custom | dictionary | length-based | repeated-characters | similarity-based | unique-characters

4.25 dsconfig create-plugin

Creates Plugins

`--plugin-name {name}`

The name of the new Plugin

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Plugin which should be created. The value for TYPE can be one of: attribute-cleanup | change-number-control | custom | entry-uuid | fractional-ldif-import | last-mod | ldap-attribute-description-list | password-policy-import | profiler | referential-integrity | samba-password | seven-bit-clean | unique-attribute

4.26 dsconfig create-replication-domain

Creates Replication Domains

`--provider-name {name}`

The name of the Replication Synchronization Provider

`--domain-name {name}`

The name of the new Replication Domain

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Replication Domain which should be created (Default: generic). The value for TYPE can be one of: generic

4.27 dsconfig create-replication-server

Creates Replication Servers

`--provider-name {name}`

The name of the Replication Synchronization Provider

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Replication Server which should be created (Default: generic). The value for TYPE can be one of: generic

4.28 dsconfig create-sasl-mechanism-handler

Creates SASL Mechanism Handlers

`--handler-name {name}`

The name of the new SASL Mechanism Handler

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 SASL Mechanism Handler which should be created. The value for TYPE can be one of: anonymous | cram-md5 | custom | digest-md5 | external | gssapi | plain

4.29 dsconfig create-synchronization-provider

Creates Synchronization Providers

`--provider-name {name}`

The name of the new Synchronization Provider

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Synchronization Provider which should be created. The value for TYPE can be one of: custom | replication

4.30 dsconfig create-trust-manager-provider

Creates Trust Manager Providers

`--provider-name {name}`

The name of the new Trust Manager Provider

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Trust Manager Provider which should be created. The value for TYPE can be one of: blind | custom | file-based

4.31 dsconfig create-virtual-attribute

Creates Virtual Attributes

`--name {name}`

The name of the new Virtual Attribute

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE 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 Virtual Attribute which should be created. The value for TYPE can be one of: collective-attribute-subentries | custom | entity-tag | entry-dn | entry-uuid | governing-structure-rule | has-subordinates | is-member-of | member | num-subordinates | password-expiration-time | password-policy-subentry | structural-object-class | subschema-subentry | user-defined

4.32 dsconfig delete-access-log-filtering-criteria

Deletes Access Log Filtering Criteria

`--publisher-name {name}`

The name of the Access Log Publisher

`--criteria-name {name}`

The name of the Access Log Filtering Criteria

`-f | --force`

Ignore non-existent Access Log Filtering Criteria

4.33 dsconfig delete-account-status-notification-handler

Deletes Account Status Notification Handlers

`--handler-name {name}`

The name of the Account Status Notification Handler

`-f | --force`

Ignore non-existent Account Status Notification Handlers

4.34 dsconfig delete-alert-handler

Deletes Alert Handlers

`--handler-name {name}`

The name of the Alert Handler

`-f | --force`

Ignore non-existent Alert Handlers

4.35 dsconfig delete-attribute-syntax

Deletes Attribute Syntaxes

`--syntax-name {name}`

The name of the Attribute Syntax

`-f | --force`

Ignore non-existent Attribute Syntaxes

4.36 dsconfig delete-backend

Deletes Backends

`--backend-name {name}`

The name of the Backend

`-f | --force`

Ignore non-existent Backends

4.37 dsconfig delete-certificate-mapper

Deletes Certificate Mappers

`--mapper-name {name}`

The name of the Certificate Mapper

`-f | --force`

Ignore non-existent Certificate Mappers

4.38 dsconfig delete-connection-handler

Deletes Connection Handlers

`--handler-name {name}`

The name of the Connection Handler

`-f | --force`

Ignore non-existent Connection Handlers

4.39 dsconfig delete-debug-target

Deletes Debug Targets

`--publisher-name {name}`

The name of the Debug Log Publisher

`--target-name {name}`

The name of the Debug Target

`-f | --force`

Ignore non-existent Debug Targets

4.40 dsconfig delete-entry-cache

Deletes Entry Caches

`--cache-name {name}`

The name of the Entry Cache

`-f | --force`

Ignore non-existent Entry Caches

4.41 dsconfig delete-extended-operation-handler

Deletes Extended Operation Handlers

`--handler-name {name}`

The name of the Extended Operation Handler

`-f | --force`

Ignore non-existent Extended Operation Handlers

4.42 dsconfig delete-group-implementation

Deletes Group Implementations

`--implementation-name {name}`

The name of the Group Implementation

`-f | --force`

Ignore non-existent Group Implementations

4.43 dsconfig delete-identity-mapper

Deletes Identity Mappers

`--mapper-name {name}`

The name of the Identity Mapper

`-f | --force`

Ignore non-existent Identity Mappers

4.44 dsconfig delete-key-manager-provider

Deletes Key Manager Providers

`--provider-name {name}`

The name of the Key Manager Provider

`-f | --force`

Ignore non-existent Key Manager Providers

4.45 dsconfig delete-local-db-index

Deletes Local DB Indexes

`--backend-name {name}`

The name of the Local DB Backend

`--index-name {name}`

The name of the Local DB Index

`-f | --force`

Ignore non-existent Local DB Indexes

4.46 dsconfig delete-local-db-vlv-index

Deletes Local DB VLV Indexes

`--backend-name {name}`

The name of the Local DB Backend

`--index-name {name}`

The name of the Local DB VLV Index

`-f | --force`

Ignore non-existent Local DB VLV Indexes

4.47 dsconfig delete-log-publisher

Deletes Log Publishers

`--publisher-name {name}`

The name of the Log Publisher

`-f | --force`

Ignore non-existent Log Publishers

4.48 dsconfig delete-log-retention-policy

Deletes Log Retention Policies

`--policy-name {name}`

The name of the Log Retention Policy

`-f | --force`

Ignore non-existent Log Retention Policies

4.49 dsconfig delete-log-rotation-policy

Deletes Log Rotation Policies

`--policy-name {name}`

The name of the Log Rotation Policy

`-f | --force`

Ignore non-existent Log Rotation Policies

4.50 dsconfig delete-matching-rule

Deletes Matching Rules

`--rule-name {name}`

The name of the Matching Rule

`-f | --force`

Ignore non-existent Matching Rules

4.51 dsconfig delete-monitor-provider

Deletes Monitor Providers

`--provider-name {name}`

The name of the Monitor Provider

`-f | --force`

Ignore non-existent Monitor Providers

4.52 dsconfig delete-password-generator

Deletes Password Generators

`--generator-name {name}`

The name of the Password Generator

`-f | --force`

Ignore non-existent Password Generators

4.53 dsconfig delete-password-policy

Deletes Authentication Policies

`--policy-name {name}`

The name of the Authentication Policy

`-f | --force`

Ignore non-existent Authentication Policies

4.54 dsconfig delete-password-storage-scheme

Deletes Password Storage Schemes

`--scheme-name {name}`

The name of the Password Storage Scheme

`-f | --force`

Ignore non-existent Password Storage Schemes

4.55 dsconfig delete-password-validator

Deletes Password Validators

`--validator-name {name}`

The name of the Password Validator

`-f | --force`

Ignore non-existent Password Validators

4.56 dsconfig delete-plugin

Deletes Plugins

`--plugin-name {name}`

The name of the Plugin

`-f | --force`

Ignore non-existent Plugins

4.57 dsconfig delete-replication-domain

Deletes Replication Domains

`--provider-name {name}`

The name of the Replication Synchronization Provider

`--domain-name {name}`

The name of the Replication Domain

`-f | --force`

Ignore non-existent Replication Domains

4.58 dsconfig delete-replication-server

Deletes Replication Servers

`--provider-name {name}`

The name of the Replication Synchronization Provider

`-f | --force`

Ignore non-existent Replication Servers

4.59 dsconfig delete-sasl-mechanism-handler

Deletes SASL Mechanism Handlers

`--handler-name {name}`

The name of the SASL Mechanism Handler

`-f | --force`

Ignore non-existent SASL Mechanism Handlers

4.60 dsconfig delete-synchronization-provider

Deletes Synchronization Providers

`--provider-name {name}`

The name of the Synchronization Provider

`-f | --force`

Ignore non-existent Synchronization Providers

4.61 dsconfig delete-trust-manager-provider

Deletes Trust Manager Providers

`--provider-name {name}`

The name of the Trust Manager Provider

`-f | --force`

Ignore non-existent Trust Manager Providers

4.62 dsconfig delete-virtual-attribute

Deletes Virtual Attributes

`--name {name}`

The name of the Virtual Attribute

`-f | --force`

Ignore non-existent Virtual Attributes

4.63 dsconfig get-access-control-handler-prop

Shows Access Control Handler properties

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.64 dsconfig get-access-log-filtering-criteria-prop

Shows Access Log Filtering Criteria properties

`--publisher-name {name}`

The name of the Access Log Publisher

`--criteria-name {name}`

The name of the Access Log Filtering Criteria

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.65 dsconfig get-account-status-notification-handler-prop

Shows Account Status Notification Handler properties

`--handler-name {name}`

The name of the Account Status Notification Handler

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.66 dsconfig get-administration-connector-prop

Shows Administration Connector properties

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.67 dsconfig get-alert-handler-prop

Shows Alert Handler properties

`--handler-name {name}`

The name of the Alert Handler

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.68 dsconfig get-attribute-syntax-prop

Shows Attribute Syntax properties

`--syntax-name {name}`

The name of the Attribute Syntax

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.69 dsconfig get-backend-prop

Shows Backend properties

`--backend-name {name}`

The name of the Backend

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.70 dsconfig get-certificate-mapper-prop

Shows Certificate Mapper properties

`--mapper-name {name}`

The name of the Certificate Mapper

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.71 dsconfig get-connection-handler-prop

Shows Connection Handler properties

`--handler-name {name}`

The name of the Connection Handler

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.72 dsconfig get-crypto-manager-prop

Shows Crypto Manager properties

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.73 dsconfig get-debug-target-prop

Shows Debug Target properties

`--publisher-name {name}`

The name of the Debug Log Publisher

`--target-name {name}`

The name of the Debug Target

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.74 dsconfig get-entry-cache-prop

Shows Entry Cache properties

`--cache-name {name}`

The name of the Entry Cache

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.75 dsconfig get-extended-operation-handler-prop

Shows Extended Operation Handler properties

`--handler-name {name}`

The name of the Extended Operation Handler

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.76 dsconfig get-external-changelog-domain-prop

Shows External Changelog Domain properties

`--provider-name {name}`

The name of the Replication Synchronization Provider

`--domain-name {name}`

The name of the Replication Domain

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.77 dsconfig get-global-configuration-prop

Shows Global Configuration properties

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.78 dsconfig get-group-implementation-prop

Shows Group Implementation properties

`--implementation-name {name}`

The name of the Group Implementation

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.79 dsconfig get-identity-mapper-prop

Shows Identity Mapper properties

`--mapper-name {name}`

The name of the Identity Mapper

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.80 dsconfig get-key-manager-provider-prop

Shows Key Manager Provider properties

`--provider-name {name}`

The name of the Key Manager Provider

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.81 dsconfig get-local-db-index-prop

Shows Local DB Index properties

`--backend-name {name}`

The name of the Local DB Backend

`--index-name {name}`

The name of the Local DB Index

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.82 dsconfig get-local-db-vlv-index-prop

Shows Local DB VLV Index properties

`--backend-name {name}`

The name of the Local DB Backend

`--index-name {name}`

The name of the Local DB VLV Index

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.83 dsconfig get-log-publisher-prop

Shows Log Publisher properties

`--publisher-name {name}`

The name of the Log Publisher

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.84 dsconfig get-log-retention-policy-prop

Shows Log Retention Policy properties

`--policy-name {name}`

The name of the Log Retention Policy

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.85 dsconfig get-log-rotation-policy-prop

Shows Log Rotation Policy properties

`--policy-name {name}`

The name of the Log Rotation Policy

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.86 dsconfig get-matching-rule-prop

Shows Matching Rule properties

`--rule-name {name}`

The name of the Matching Rule

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.87 dsconfig get-monitor-provider-prop

Shows Monitor Provider properties

`--provider-name {name}`

The name of the Monitor Provider

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.88 dsconfig get-password-generator-prop

Shows Password Generator properties

`--generator-name {name}`

The name of the Password Generator

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.89 dsconfig get-password-policy-prop

Shows Authentication Policy properties

`--policy-name {name}`

The name of the Authentication Policy

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.90 dsconfig get-password-storage-scheme-prop

Shows Password Storage Scheme properties

`--scheme-name {name}`

The name of the Password Storage Scheme

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.91 dsconfig get-password-validator-prop

Shows Password Validator properties

`--validator-name {name}`

The name of the Password Validator

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.92 dsconfig get-plugin-prop

Shows Plugin properties

`--plugin-name {name}`

The name of the Plugin

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.93 dsconfig get-plugin-root-prop

Shows Plugin Root properties

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.94 dsconfig get-replication-domain-prop

Shows Replication Domain properties

`--provider-name {name}`

The name of the Replication Synchronization Provider

`--domain-name {name}`

The name of the Replication Domain

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.95 dsconfig get-replication-server-prop

Shows Replication Server properties

`--provider-name {name}`

The name of the Replication Synchronization Provider

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.96 dsconfig get-root-dn-prop

Shows Root DN properties

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.97 dsconfig get-root-dse-backend-prop

Shows Root DSE Backend properties

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.98 dsconfig get-sasl-mechanism-handler-prop

Shows SASL Mechanism Handler properties

`--handler-name {name}`

The name of the SASL Mechanism Handler

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.99 dsconfig get-synchronization-provider-prop

Shows Synchronization Provider properties

`--provider-name {name}`

The name of the Synchronization Provider

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.100 dsconfig get-trust-manager-provider-prop

Shows Trust Manager Provider properties

`--provider-name {name}`

The name of the Trust Manager Provider

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.101 dsconfig get-virtual-attribute-prop

Shows Virtual Attribute properties

`--name {name}`

The name of the Virtual Attribute

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.102 dsconfig get-work-queue-prop

Shows Work Queue properties

`--property {property}`

The name of a property to be displayed

`-E | --record`

Modifies the display output to show one property value per line

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.103 dsconfig list-access-log-filtering-criteria

Lists existing Access Log Filtering Criteria

`--publisher-name {name}`

The name of the Access Log Publisher

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.104 dsconfig list-account-status-notification-handlers

Lists existing Account Status Notification Handlers

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.105 dsconfig list-alert-handlers

Lists existing Alert Handlers

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.106 dsconfig list-attribute-syntaxes

Lists existing Attribute Syntaxes

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.107 dsconfig list-backends

Lists existing Backends

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.108 dsconfig list-certificate-mappers

Lists existing Certificate Mappers

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.109 dsconfig list-connection-handlers

Lists existing Connection Handlers

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.110 dsconfig list-debug-targets

Lists existing Debug Targets

`--publisher-name {name}`

The name of the Debug Log Publisher

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

```
-m | --unit-time {unit}
```

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.111 dsconfig list-entry-caches

Lists existing Entry Caches

```
--property {property}
```

The name of a property to be displayed

```
-z | --unit-size {unit}
```

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

```
-m | --unit-time {unit}
```

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.112 dsconfig list-extended-operation-handlers

Lists existing Extended Operation Handlers

```
--property {property}
```

The name of a property to be displayed

```
-z | --unit-size {unit}
```

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

```
-m | --unit-time {unit}
```

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.113 dsconfig list-group-implementations

Lists existing Group Implementations

```
--property {property}
```

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.114 dsconfig list-identity-mappers

Lists existing Identity Mappers

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.115 dsconfig list-key-manager-providers

Lists existing Key Manager Providers

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.116 dsconfig list-local-db-indexes

Lists existing Local DB Indexes

`--backend-name {name}`

The name of the Local DB Backend

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.117 dsconfig list-local-db-vlv-indexes

Lists existing Local DB VLV Indexes

`--backend-name {name}`

The name of the Local DB Backend

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.118 dsconfig list-log-publishers

Lists existing Log Publishers

`--property {property}`

The name of a property to be displayed

```
-z | --unit-size {unit}
```

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

```
-m | --unit-time {unit}
```

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.119 dsconfig list-log-retention-policies

Lists existing Log Retention Policies

```
--property {property}
```

The name of a property to be displayed

```
-z | --unit-size {unit}
```

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

```
-m | --unit-time {unit}
```

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.120 dsconfig list-log-rotation-policies

Lists existing Log Rotation Policies

```
--property {property}
```

The name of a property to be displayed

```
-z | --unit-size {unit}
```

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

```
-m | --unit-time {unit}
```

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.121 dsconfig list-matching-rules

Lists existing Matching Rules

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.122 dsconfig list-monitor-providers

Lists existing Monitor Providers

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.123 dsconfig list-password-generators

Lists existing Password Generators

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.124 dsconfig list-password-policies

Lists existing Password Policies

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.125 dsconfig list-password-storage-schemes

Lists existing Password Storage Schemes

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.126 dsconfig list-password-validators

Lists existing Password Validators

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.127 dsconfig list-plugins

Lists existing Plugins

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.128 dsconfig list-properties

Describes managed objects and their properties

`-c | --category {category}`

The category of components whose properties should be described

`-t | --type {type}`

The type of components whose properties should be described. The value for TYPE must be one of the component types associated with the CATEGORY specified using the "--category" option

`--inherited`

Modifies the display output to show the inherited properties of components

`--property {property}`

The name of a property to be displayed

4.129 dsconfig list-replication-domains

Lists existing Replication Domains

`--provider-name {name}`

The name of the Replication Synchronization Provider

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.130 dsconfig list-replication-server

Lists existing Replication Server

`--provider-name {name}`

The name of the Replication Synchronization Provider

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.131 dsconfig list-sasl-mechanism-handlers

Lists existing SASL Mechanism Handlers

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.132 dsconfig list-synchronization-providers

Lists existing Synchronization Providers

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.133 dsconfig list-trust-manager-providers

Lists existing Trust Manager Providers

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.134 dsconfig list-virtual-attributes

Lists existing Virtual Attributes

`--property {property}`

The name of a property to be displayed

`-z | --unit-size {unit}`

Display size data using the specified unit. The value for UNIT can be one of b, kb, mb, gb, or tb (bytes, kilobytes, megabytes, gigabytes, or terabytes)

`-m | --unit-time {unit}`

Display time data using the specified unit. The value for UNIT can be one of ms, s, m, h, d, or w (milliseconds, seconds, minutes, hours, days, or weeks)

4.135 dsconfig set-access-control-handler-prop

Modifies Access Control Handler properties

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.136 dsconfig set-access-log-filtering-criteria-prop

Modifies Access Log Filtering Criteria properties

`--publisher-name {name}`

The name of the Access Log Publisher

`--criteria-name {name}`

The name of the Access Log Filtering Criteria

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.137 dsconfig set-account-status-notification-handler-prop

Modifies Account Status Notification Handler properties

`--handler-name {name}`

The name of the Account Status Notification Handler

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.138 dsconfig set-administration-connector-prop

Modifies Administration Connector properties

`--set {PROP:VALUE}`

> Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

> Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

> Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

> Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.139 dsconfig set-alert-handler-prop

Modifies Alert Handler properties

`--handler-name {name}`

> The name of the Alert Handler

`--set {PROP:VALUE}`

> Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

> Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

> Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.140 dsconfig set-attribute-syntax-prop

Modifies Attribute Syntax properties

`--syntax-name {name}`

The name of the Attribute Syntax

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.141 dsconfig set-backend-prop

Modifies Backend properties

`--backend-name {name}`

The name of the Backend

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.142 dsconfig set-certificate-mapper-prop

Modifies Certificate Mapper properties

`--mapper-name {name}`

The name of the Certificate Mapper

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.143 dsconfig set-connection-handler-prop

Modifies Connection Handler properties

`--handler-name {name}`

The name of the Connection Handler

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.144 dsconfig set-crypto-manager-prop

Modifies Crypto Manager properties

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.145 dsconfig set-debug-target-prop

Modifies Debug Target properties

`--publisher-name {name}`

The name of the Debug Log Publisher

`--target-name {name}`

The name of the Debug Target

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.146 dsconfig set-entry-cache-prop

Modifies Entry Cache properties

`--cache-name {name}`

The name of the Entry Cache

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.147 dsconfig set-extended-operation-handler-prop

Modifies Extended Operation Handler properties

`--handler-name {name}`

The name of the Extended Operation Handler

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.148 dsconfig set-external-changelog-domain-prop

Modifies External Changelog Domain properties

`--provider-name {name}`

The name of the Replication Synchronization Provider

`--domain-name {name}`

The name of the Replication Domain

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.149 dsconfig set-global-configuration-prop

Modifies Global Configuration properties

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.150 dsconfig set-group-implementation-prop

Modifies Group Implementation properties

`--implementation-name {name}`

> The name of the Group Implementation

`--set {PROP:VALUE}`

> Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

> Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

> Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

> Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.151 dsconfig set-identity-mapper-prop

Modifies Identity Mapper properties

`--mapper-name {name}`

> The name of the Identity Mapper

`--set {PROP:VALUE}`

> Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

> Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.152 dsconfig set-key-manager-provider-prop

Modifies Key Manager Provider properties

`--provider-name {name}`

The name of the Key Manager Provider

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.153 dsconfig set-local-db-index-prop

Modifies Local DB Index properties

`--backend-name {name}`

The name of the Local DB Backend

`--index-name {name}`

The name of the Local DB Index

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.154 dsconfig set-local-db-vlv-index-prop

Modifies Local DB VLV Index properties

`--backend-name {name}`

The name of the Local DB Backend

`--index-name {name}`

The name of the Local DB VLV Index

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.155 dsconfig set-log-publisher-prop

Modifies Log Publisher properties

`--publisher-name {name}`

The name of the Log Publisher

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.156 dsconfig set-log-retention-policy-prop

Modifies Log Retention Policy properties

`--policy-name {name}`

The name of the Log Retention Policy

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.157 dsconfig set-log-rotation-policy-prop

Modifies Log Rotation Policy properties

`--policy-name {name}`

The name of the Log Rotation Policy

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.158 dsconfig set-matching-rule-prop

Modifies Matching Rule properties

`--rule-name {name}`

The name of the Matching Rule

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.159 dsconfig set-monitor-provider-prop

Modifies Monitor Provider properties

`--provider-name {name}`

The name of the Monitor Provider

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.160 dsconfig set-password-generator-prop

Modifies Password Generator properties

`--generator-name {name}`

The name of the Password Generator

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.161 dsconfig set-password-policy-prop

Modifies Authentication Policy properties

`--policy-name {name}`

The name of the Authentication Policy

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.162 dsconfig set-password-storage-scheme-prop

Modifies Password Storage Scheme properties

`--scheme-name {name}`

The name of the Password Storage Scheme

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.163 dsconfig set-password-validator-prop

Modifies Password Validator properties

`--validator-name {name}`

The name of the Password Validator

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.164 dsconfig set-plugin-prop

Modifies Plugin properties

`--plugin-name {name}`

The name of the Plugin

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.165 dsconfig set-plugin-root-prop

Modifies Plugin Root properties

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.166 dsconfig set-replication-domain-prop

Modifies Replication Domain properties

`--provider-name {name}`

The name of the Replication Synchronization Provider

`--domain-name {name}`

The name of the Replication Domain

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.167 dsconfig set-replication-server-prop

Modifies Replication Server properties

`--provider-name {name}`

The name of the Replication Synchronization Provider

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.168 dsconfig set-root-dn-prop

Modifies Root DN properties

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.169 dsconfig set-root-dse-backend-prop

Modifies Root DSE Backend properties

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.170 dsconfig set-sasl-mechanism-handler-prop

Modifies SASL Mechanism Handler properties

`--handler-name {name}`

The name of the SASL Mechanism Handler

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.171 dsconfig set-synchronization-provider-prop

Modifies Synchronization Provider properties

`--provider-name {name}`

The name of the Synchronization Provider

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.172 dsconfig set-trust-manager-provider-prop

Modifies Trust Manager Provider properties

`--provider-name {name}`

The name of the Trust Manager Provider

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.173 dsconfig set-virtual-attribute-prop

Modifies Virtual Attribute properties

`--name {name}`

The name of the Virtual Attribute

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

4.174 dsconfig set-work-queue-prop

Modifies Work Queue properties

`--set {PROP:VALUE}`

Assigns a value to a property where PROP is the name of the property and VALUE is the single value to be assigned. Specify the same property multiple times in order to assign more than one value to it

`--reset {property}`

Resets a property back to its default values where PROP is the name of the property to be reset

`--add {PROP:VALUE}`

Adds a single value to a property where PROP is the name of the property and VALUE is the single value to be added

`--remove {PROP:VALUE}`

Removes a single value from a property where PROP is the name of the property and VALUE is the single value to be removed

5 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

6 Examples

Much of the *OpenDJ Administration Guide* consists of **dsconfig** examples with text in between. This section therefore remains short.

The following example starts **dsconfig** in interactive, menu-driven mode on the default port of the current host.

```
$ dsconfig -h `hostname` -p 4444 -D "cn=Directory Manager" -w password

>>>> OpenDJ configuration console main menu

What do you want to configure?

    1)   Access Control Handler               21)  Log Publisher
    2)   Access Log Filtering Criteria        22)  Log Retention Policy
    3)   Account Status Notification Handler  23)  Log Rotation Policy
    4)   Administration Connector             24)  Matching Rule
    5)   Alert Handler                        25)  Monitor Provider
    6)   Attribute Syntax                     26)  Password Generator
    7)   Backend                              27)  Password Policy
    8)   Certificate Mapper                   28)  Password Storage Scheme
    9)   Connection Handler                   29)  Password Validator
    10)  Crypto Manager                       30)  Plugin
    11)  Debug Target                         31)  Plugin Root
    12)  Entry Cache                          32)  Replication Domain
    13)  Extended Operation Handler           33)  Replication Server
    14)  External Changelog Domain            34)  Root DN
    15)  Global Configuration                 35)  Root DSE Backend
    16)  Group Implementation                 36)  SASL Mechanism Handler
    17)  Identity Mapper                      37)  Synchronization Provider
    18)  Key Manager Provider                 38)  Trust Manager Provider
    19)  Local DB Index                       39)  Virtual Attribute
    20)  Local DB VLV Index                   40)  Work Queue

    q)   quit

Enter choice:
```

The following examples demonstrates generating a batch file that corresponds to an interactive session enabling the debug log. The example then demonstates using a modified batch file to disable the debug log.

```
$ dsconfig
 --hostname `hostname`
 --port 4444
 --bindDN "cn=Directory Manager"
 --bindPassword password
 --commandFilePath ~/enable-debug-log.batch
 ...
$ cat ~/enable-debug-log.batch
# dsconfig session start date: 19/Oct/2011:08:52:22 +0000

# Session operation number: 1
# Operation date: 19/Oct/2011:08:55:06 +0000
dsconfig set-log-publisher-prop \
          --publisher-name File-Based\ Debug\ Logger \
          --set enabled:true \
          --hostname opendj.example.com \
          --port 4444 \
          --trustStorePath /path/to/opendj/config/admin-truststore \
          --bindDN cn=Directory\ Manager \
          --bindPassword ****** \
          --no-prompt

$ cp ~/enable-debug-log.batch ~/disable-debug-log.batch
$ vi ~/disable-debug-log.batch
$ cat ~/disable-debug-log.batch
set-log-publisher-prop \
          --publisher-name File-Based\ Debug\ Logger \
          --set enabled:false \
          --hostname opendj.example.com \
          --port 4444 \
          --trustStorePath /path/to/opendj/config/admin-truststore \
          --bindDN cn=Directory\ Manager \
          --bindPassword password \
          --no-prompt

$ dsconfig --batchFilePath ~/disable-debug-log.batch --no-prompt
set-log-publisher-prop
--publisher-name
File-Based Debug Logger
--set
enabled:false
--hostname
opendj.example.com
--port
4444
--trustStorePath
/path/to/opendj/config/admin-truststore
--bindDN
cn=Directory Manager
--bindPassword
password
--no-prompt

$
```

Notice that the original command file looks like a shell script with the bind password value replaced by asterisks. To pass the content as a batch file to **dsconfig**, strip `dsconfig` itself, and include the bind password for the administrative user (or replace that option with an alternative, such as reading the password from a file).

dsframework

dsframework — manage OpenDJ administration framework

dsframework

`dsframework` *`subcommand`* {options}

1 Description

This utility can be used to perform operations in the directory server administration framework.

This utility has no interactive mode. Consider using `-X` if you authenticate over a secure connection protected with a self-signed certificate.

2 Subcommands

The **dsconfig** command provides many subcommands. Use the following options to view help for subcommands.

`--help-all`

Display all subcommands

`--help-admin-user`

Display subcommands relating to admin-user

`--help-server`

Display subcommands relating to server

`--help-server-group`

Display subcommands relating to server-group

For help with individual subcommands, use **dsframework *`subcommand`* --help**.

3 Options

The following options are supported.

3.1 LDAP Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-I, --adminUID {adminUID}`

User ID of the global administrator to use to bind to the server. For the **enable** subcommand, if no global administrator was defined previously for any servers, the global administrator will be created using the UID provided.

Default value: admin

`-j, --adminPasswordFile {bindPasswordFile}`

Global administrator password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server administration port number

Default value: 4444

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

Default value: /path/to/opendj/config/admin-truststore

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --adminPassword {bindPassword}`

Password for the global administrator

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

3.2 Utility Input/Output Options

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

`-v, --verbose`

Use verbose mode

3.3 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

4 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

5 Examples

This example lists server properties.

```
$ dsframework -p 4444 -h opendj.example.com -D "cn=Directory Manager"
 -w password -X list-server-properties
Option Types:

 r -- Property value(s) are readable
 w -- Property value(s) are writable
 m -- The property is mandatory
 s -- The property is single-valued

Property                       Options  Syntax              Default value
-------------------------------------------------------------------------
ldapsport                      rw--     INTEGER             -
certificate                    rw-s     STRING              -
hostname                       r-ms     STRING              localhost
ldapport                       rwm-     INTEGER             389
jmxsEnabled                    rw-s     BOOLEAN             false
instancepath                   rw-s     STRING              -
ldapsEnabled                   rw-s     BOOLEAN             false
jmxsport                       rw--     INTEGER             -
os                             rw-s     STRING              -
ds-cfg-key-id                  rw-s     STRING              -
jmxport                        rw--     INTEGER             -
description                    rw-s     STRING              -
id                             rw-s     STRING              -
startTLSEnabled                rw-s     BOOLEAN             false
jmxEnabled                     rw-s     BOOLEAN             false
ds-cfg-public-key-certificate  rw-s     CERTIFICATE_BINARY  -
location                       rw-s     STRING              -
ldapEnabled                    rw-s     BOOLEAN             false
```

dsjavaproperties

dsjavaproperties — apply OpenDJ Java home and JVM settings

dsjavaproperties

`dsjavaproperties` [options]

1 Description

This utility can be used to change the Java arguments and Java home that are used by other server commands.

Before launching the command, edit the properties file located in `/path/to/opendj/config/java.properties` to specify the Java arguments and Java home. When you have edited the properties file, run this command for the changes to be taken into account.

Changes apply to the current server installation. No modifications are made to your environment variables.

2 Options

The following options are supported.

`-Q, --quiet`

Run the tool in quiet mode. Quiet mode will not output progress information to standard output.

`-V, --version`

Display version information.

`-?, -H, --help`

Display usage information.

3 Files

This command depends on the content of the `config/java.properties` file.

4 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

5 Examples

```
$ dsjavaproperties
The operation was successful.  The server commands will use the java arguments
 and java home specified in the properties file located in
 /path/to/opendj/config/java.properties
```

dsreplication

dsreplication — manage OpenDJ directory data replication

dsreplication

`dsreplication` *subcommand* [options]

1 Description

This utility can be used to configure replication between servers so that the data of the servers is synchronized. For replication to work you must first enable replication using the **enable** subcommand and then initialize the contents of one of the servers with the contents of the other using the **initialize** subcommand.

2 Subcommands

The following subcommands are supported.

disable

Disable replication on the specified server for the provided base DN and removes references in the other servers with which it is replicating data.

enable

Update the configuration of the servers to replicate the data under the specified base DN. If one of the specified servers is already replicating the data under the base DN with other servers, executing this subcommand will update the configuration of all the servers. Thus it is sufficient to execute the command line once for each server added to the replication topology.

initialize

Initialize the contents of the data under the specified base DN on the destination server with the contents on the source server. This operation is required after enabling replication in order replication to work. **initialize-all** can also be used for this purpose.

initialize-all

Initialize the contents of the data under the specified base DN on all the servers whose contents are being replicated with the contents on the specified server. This operation is required after enabling replication for replication to work. Run **initialize** for each server to achieve the same effect.

post-external-initialization

This subcommand must be called after initializing the contents of all the replicated servers using the **import-ldif** command, or by copying the database. You must specify the list of base DNs that have been initialized, and you must provide the credentials of any of the servers that are being replicated. See **pre-external-initialization --help** for more information.

pre-external-initialization

This subcommand must be called before initializing the contents of all the replicated servers using the **import-ldif** command, or by copying the database. You must specify the list of base DNs that have been initialized, and you must provide the credentials of any of the servers that are being replicated. After calling this subcommand, initialize the contents of all the servers in the topology, either by using the same LDIF file or by copying the database to each of the servers, then call the **post-external-initialization** subcommand.

purge-historical

Launch a purge processing of the historical information stored in the user entries by replication. Since this processing may take a while, you must specify a maximum duration.

status

Display a list with the basic replication configuration of the base DNs of the servers defined in the registration information. If no base DNs are specified as parameter, information for all base DNs is displayed.

3 Options

The following options are supported.

`--advanced`

Access advanced settings when running this command in interactive mode.

`-b, --baseDN {baseDN}`

Base DN of the data to be replicated, initialized or for which you want to disable replication. Multiple base DNs can be provided by using this option multiple times.

3.1 LDAP Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-I, --adminUID {adminUID}`

User ID of the global administrator to use to bind to the server. For the **enable** subcommand, if no global administrator was defined previously for any servers, the global administrator will be created using the UID provided.

Default value: admin

`-j, --adminPasswordFile {bindPasswordFile}`

Global administrator password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server administration port number

Default value: 4444

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

Default value: /path/to/opendj/config/admin-truststore

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --adminPassword {bindPassword}`

Password for the global administrator

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

3.2 Utility Input/Output Options

`--commandFilePath {path}`

The full path to the file where the equivalent non-interactive commands will be written when this command is run in interactive mode.

`--displayCommand`

Display the equivalent non-interactive option on standard output when this command is run in interactive mode.

`-n, --no-prompt`

Use non-interactive mode. If data in the command is missing, the user is not prompted and the command exits with an error.

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

`-Q, --quiet`

Do not write progress information to standard output

3.3 General Options

`--version`

Display version information

`-?, -H, --help`

Display usage information

4 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

5 Examples

The following example enables and then initializes replication for a new replica on `opendj2.example.com` from an existing replica on `opendj.example.com`.

```
$ dsreplication enable -I admin -w password -X -n -b dc=example,dc=com
 --host1 opendj.example.com --port1 4444 --bindDN1 "cn=Directory Manager"
 --bindPassword1 password --replicationPort1 8989
 --host2 opendj2.example.com --port2 4444 --bindDN2 "cn=Directory Manager"
 --bindPassword2 password --replicationPort2 8989

Establishing connections ..... Done.
Checking registration information ..... Done.
Updating remote references on server opendj.example.com:4444 ..... Done.
Configuring Replication port on server opendj2.example.com:4444 ..... Done.
Updating replication configuration for baseDN dc=example,dc=com on server
 opendj.example.com:4444 ..... Done.
Updating replication configuration for baseDN dc=example,dc=com on server
 opendj2.example.com:4444 ..... Done.
Updating registration configuration on server
 opendj.example.com:4444 ..... Done.
Updating registration configuration on server
 opendj2.example.com:4444 ..... Done.
Updating replication configuration for baseDN cn=schema on server
 opendj.example.com:4444 ..... Done.
Updating replication configuration for baseDN cn=schema on server
 opendj2.example.com:4444 ..... Done.
Initializing registration information on server opendj2.example.com:4444 with
```

```
 the contents of server opendj.example.com:4444 ..... Done.
Initializing schema on server opendj2.example.com:4444 with the contents of
 server opendj.example.com:4444 ..... Done.

Replication has been successfully enabled.  Note that for replication to
 work you must initialize the contents of the base DN's that are being
  replicated (use dsreplication initialize to do so).

See
/var/.../opends-replication-7958637258600693490.log
for a detailed log of this operation.
$ dsreplication initialize-all -I admin -w password -X -n -b dc=example,dc=com
 -h opendj.example.com -p 4444

Initializing base DN dc=example,dc=com with the contents from
 opendj.example.com:4444: 160 entries processed (100 % complete).
Base DN initialized successfully.

See
/var/.../opends-replication-5020375834904394170.log
for a detailed log of this operation.
```

encode-password

encode-password — encode a password with an OpenDJ storage scheme

encode-password

`encode-password` {options}

1 Description

This utility can be used to encode user passwords with a specified storage scheme, or to determine whether a given clear-text value matches a provided encoded password.

2 Options

The following global options are supported.

`-a, --authPasswordSyntax`

Use the authentication password syntax rather than the user password syntax.

`-c, --clearPassword {clearPW}`

Clear-text password to encode or to compare against an encoded password.

`-e, --encodedPassword {encodedPW}`

Encoded password to compare against the clear-text password.

`-E, --encodedPasswordFile {file}`

Encoded password file.

`-f, --clearPasswordFile {file}`

Clear-text password file.

`-i, --interactivePassword`

The password to encode or to compare against an encoded password is interactively asked to the user.

`-l, --listSchemes`

List available password storage schemes.

`-r, --useCompareResultCode`

Use the LDAP compare result as an exit code for the password comparison.

`-s, --storageScheme {scheme}`

Scheme to use for the encoded password.

`-V, --version`

Display version information.

`-?, -H, --help`

Display usage information.

3 Exit Codes

0

The command completed successfully.

5

The `-r` option was used, and the compare did not match.

6

The `-r` option was used, and the compare did match.

other

An error occurred.

4 Examples

The following example encodes a password, and also shows comparison of a password with the encoded value.

```
$ encode-password -l
3DES
AES
BASE64
BLOWFISH
CLEAR
CRYPT
MD5
RC4
SHA
SMD5
SSHA
SSHA256
SSHA384
SSHA512
$ encode-password -c secret12 -s CRYPT
Encoded Password:  "{CRYPT}ZulJ6Dy3TFnrE"
$ encode-password -c secret12 -s CRYPT -e "{CRYPT}ZulJ6Dy3TFnrE" -r
The provided clear-text and encoded passwords match
$ echo $?
6
```

export-ldif

export-ldif — export OpenDJ directory data in LDIF

export-ldif

`export-ldif` {options}

1 Description

This utility can be used to export data from a directory server backend in LDIF form.

2 Options

The following options are supported.

`-a, --appendToLDIF`

Append an existing LDIF file rather than overwriting it.

`-b, --includeBranch {branchDN}`

Base DN of a branch to include in the LDIF export.

`-B, --excludeBranch {branchDN}`

Base DN of a branch to exclude from the LDIF export.

`-c, --compress`

Compress the LDIF data as it is exported.

`-e, --excludeAttribute {attribute}`

Attribute to exclude from the LDIF export.

`-E, --excludeFilter {filter}`

Filter to identify entries to exclude from the LDIF export.

`-i, --includeAttribute {attribute}`

Attribute to include in the LDIF export.

`-I, --includeFilter {filter}`

Filter to identify entries to include in the LDIF export.

`-l, --ldifFile {ldifFile}`

Path to the LDIF file to be written.

`-n, --backendID {backendName}`

Backend ID for the backend to export.

`-O, --excludeOperational`

Exclude operational attributes from the LDIF export.

`--wrapColumn {wrapColumn}`

Column at which to wrap long lines (0 for no wrapping).

Default value: 0

2.1 Task Backend Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server administration port number

Default value: 4444

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

Use `-w -` to have the command prompt for the password, rather than enter the password on the command line.

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

2.2 Task Scheduling Options

`--completionNotify {emailAddress}`

Email address of a recipient to be notified when the task completes. This option may be specified more than once.

`--dependency {taskID}`

ID of a task upon which this task depends. A task will not start execution until all its dependencies have completed execution.

`--errorNotify {emailAddress}`

Email address of a recipient to be notified if an error occurs when this task executes. This option may be specified more than once.

`--failedDependencyAction {action}`

Action this task will take should one if its dependent tasks fail. The value must be one of PROCESS, CANCEL, DISABLE. If not specified defaults to CANCEL.

`--recurringTask {schedulePattern}`

Indicates the task is recurring and will be scheduled according to the value argument expressed in crontab(5) compatible time/date pattern.

`-t, --start {startTime}`

Indicates the date/time at which this operation will start when scheduled as a server task expressed in YYYYMMDDhhmmssZ format for UTC time or YYYYMMDDhhmmss for local time. A value of '0' will cause the task to be scheduled for immediate execution. When this option is specified the operation will be scheduled to start at the specified time after which this utility will exit immediately.

2.3 Utility Input/Output Options

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

2.4 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

The following example exports data to a file, `Example.ldif`, with the server offline.

```
$ export-ldif -b dc=example,dc=com -n userRoot -l ../ldif/Example.ldif
[21/Jun/2011:13:40:49 +0200] category=BACKEND severity=INFORMATION ...
...Exported 160 entries and skipped 0 in 0 seconds (average rate 1428.6/sec)
```

import-ldif

import-ldif — import OpenDJ directory data from LDIF

import-ldif

`import-ldif` {options}

1 Description

This utility can be used to import LDIF data into a directory server backend.

2 Options

The following options are supported.

`-a, --append`

Append to an existing database rather than overwriting it.

`-A, --templateFile {templateFile}`

Path to a MakeLDIF template to use to generate the import data.

`-b, --includeBranch {branchDN}`

Base DN of a branch to include in the LDIF import.

`-B, --excludeBranch {branchDN}`

Base DN of a branch to exclude from the LDIF import.

`-c, --isCompressed`

LDIF file is compressed.

`--countRejects`

Count the number of entries rejected by the server and return that value as the exit code (values > 255 will be reduced to 255 due to exit code restrictions).

`-e, --excludeAttribute {attribute}`

Attribute to exclude from the LDIF import.

`-E, --excludeFilter {filter}`

Filter to identify entries to exclude from the LDIF import.

`-F, --clearBackend`

Remove all entries for all base DNs in the backend before importing.

`-i, --includeAttribute {attribute}`

Attribute to include in the LDIF import.

`-I, --includeFilter {filter}`

Filter to identify entries to include in the LDIF import.

`-l, --ldifFile {ldifFile}`

Path to the LDIF file to be imported.

`-n, --backendID {backendName}`

Backend ID for the backend to import.

`-O, --overwrite`

Overwrite an existing rejects and/or skip file rather than appending to it.

`-r, --replaceExisting`

Replace existing entries when appending to the database.

`-R, --rejectFile {rejectFile}`

Write rejected entries to the specified file.

`-s, --randomSeed {seed}`

Seed for the MakeLDIF random number generator.

Default value: 0

`-S, --skipSchemaValidation`

Skip schema validation during the LDIF import.

`--skipDNValidation`

Perform DN validation during later part of LDIF import.

`--skipFile {skipFile}`

Write skipped entries to the specified file.

`--threadCount {count}`

Number of threads used to read LDIF file during import. Default value (0) equals: 2 x (number of CPUs).

`--tmpdirectory {directory}`

Path to temporary directory for index scratch files during LDIF import.

Default value: `import-tmp`

2.1 Task Backend Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server administration port number

Default value: 4444

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

Use `-w -` to have the command prompt for the password, rather than enter the password on the command line.

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

2.2 Task Scheduling Options

`--completionNotify {emailAddress}`

Email address of a recipient to be notified when the task completes. This option may be specified more than once.

`--dependency {taskID}`

ID of a task upon which this task depends. A task will not start execution until all its dependencies have completed execution.

`--errorNotify {emailAddress}`

Email address of a recipient to be notified if an error occurs when this task executes. This option may be specified more than once.

`--failedDependencyAction {action}`

Action this task will take should one if its dependent tasks fail. The value must be one of PROCESS, CANCEL, DISABLE. If not specified defaults to CANCEL.

`--recurringTask {schedulePattern}`

Indicates the task is recurring and will be scheduled according to the value argument expressed in crontab(5) compatible time/date pattern.

`-t, --start {startTime}`

Indicates the date/time at which this operation will start when scheduled as a server task expressed in YYYYMMDDhhmmssZ format for UTC time or YYYYMMDDhhmmss for local time. A value of '0' will cause the task to be scheduled for immediate execution. When this option is specified the operation will be scheduled to start at the specified time after which this utility will exit immediately.

2.3 Utility Input/Output Options

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

2.4 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

The following example imports the content of the LDIF file, `Example.ldif`, with the server offline.

```
$ import-ldif -b dc=example,dc=com -n userRoot -l /path/to/Example.ldif
[21/Jun/2011:13:38:03 +0200] category=RUNTIME_INFORMATION severity=NOTICE...
... msg=Import LDIF environment close took 0 seconds
```

ldapcompare

ldapcompare — perform LDAP compare operations

ldapcompare

`ldapcompare` {options} [[attribute] | [:] | [value]] [DN...]

1 Description

This utility can be used to perform LDAP compare operations in the directory.

2 Options

The following options are supported.

`--assertionFilter {filter}`

Use the LDAP assertion control with the provided filter

`-c, --continueOnError`

Continue processing even if there are errors

`-f, --filename {file}`

LDIF file containing one DN per line of entries to compare

`-J, --control {controloid[:criticality[:value|::b64value|:<filePath]]}`

Use a request control with the provided information

`-m, --useCompareResultCode`

Use the LDAP compare result as an exit code for the LDAP compare operations.

`-n, --dry-run`

Show what would be done but do not perform any operation

2.1 LDAP Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server port number

Default value: 389

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-q, --useStartTLS`

Use StartTLS to secure communication with the server

`-r, --useSASLExternal`

Use the SASL EXTERNAL authentication mechanism

`--trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-V, --ldapVersion {version}`

LDAP protocol version number

Default value: 3

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

`-Z, --useSSL`

Use SSL for secure communication with the server

2.2 Utility Input/Output Options

`-i, --encoding {encoding}`

Use the specified character set for command-line input

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

`-s, --script-friendly`

Use script-friendly mode

`-v, --verbose`

Use verbose mode

2.3 General Options

`--version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

5

The -m option was used, and at least one of the LDAP compare operations did not match.

6

The -m option was used, and all the LDAP compare operations did match.

`ldap-error`

An LDAP error occurred while processing the operation.

LDAP result codes are described in RFC 4511. Also see the additional information for details.

89

An error occurred while parsing the command-line arguments.

4 Files

You can use `~/.opendj/tools.properties` to set the defaults for bind DN, host name, and port number as in the following example.

```
hostname=directory.example.com
port=1389
bindDN=uid=kvaughan,ou=People,dc=example,dc=com

ldapcompare.port=1389
ldapdelete.port=1389
ldapmodify.port=1389
ldappasswordmodify.port=1389
ldapsearch.port=1389
```

5 Examples

The following examples demonstrate comparing Babs Jensen's UID.

The following example uses a matching UID value.

```
$ ldapcompare -p 1389 uid:bjensen uid=bjensen,ou=people,dc=example,dc=com
Comparing type uid with value bjensen in entry
uid=bjensen,ou=people,dc=example,dc=com
Compare operation returned true for entry
uid=bjensen,ou=people,dc=example,dc=com
```

The following example uses a UID value that does not match.

```
$ ldapcompare -p 1389 uid:beavis uid=bjensen,ou=people,dc=example,dc=com
Comparing type uid with value beavis in entry
uid=bjensen,ou=people,dc=example,dc=com
Compare operation returned false for entry
uid=bjensen,ou=people,dc=example,dc=com
```

ldapdelete

ldapdelete — perform LDAP delete operations

ldapdelete

`ldapdelete` {options} [*DN*]

1 Description

This utility can be used to perform LDAP delete operations in the directory.

2 Options

The following options are supported.

`--assertionFilter {filter}`

Use the LDAP assertion control with the provided filter

`-c, --continueOnError`

Continue processing even if there are errors

`-f, --filename {file}`

LDIF file containing the changes to apply

`-J, --control {controloid[:criticality[:value|::b64value|:<filePath]]}`

Use a request control with the provided information

`-n, --dry-run`

Show what would be done but do not perform any operation

`-x, --deleteSubtree`

Delete the specified entry and all entries below it

`-Y, --proxyAs {authzID}`

Use the proxied authorization control with the given authorization ID

2.1 LDAP Connection Options

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-E, --reportAuthzID`

Use the authorization identity control

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server port number

Default value: 389

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-q, --useStartTLS`

Use StartTLS to secure communication with the server

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`--usePasswordPolicyControl`

Use the password policy request control

`-V, --ldapVersion {version}`

LDAP protocol version number

Default value: 3

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

`-Z, --useSSL`

Use SSL for secure communication with the server

2.2 Utility Input/Output Options

`-i, --encoding {encoding}`

Use the specified character set for command-line input

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

`-v, --verbose`

Use verbose mode

2.3 General Options

`--version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

`ldap-error`

An LDAP error occurred while processing the operation.

LDAP result codes are described in RFC 4511. Also see the additional information for details.

89

An error occurred while parsing the command-line arguments.

4 Files

You can use `~/.opendj/tools.properties` to set the defaults for bind DN, host name, and port number as in the following example.

```
hostname=directory.example.com
port=1389
bindDN=uid=kvaughan,ou=People,dc=example,dc=com

ldapcompare.port=1389
ldapdelete.port=1389
ldapmodify.port=1389
ldappasswordmodify.port=1389
ldapsearch.port=1389
```

5 Examples

The following command deletes a user entry from the directory.

```
$ ldapdelete -p 1389 -D "cn=Directory Manager" -w password
 uid=bjensen,ou=people,dc=example,dc=com
Processing DELETE request for uid=bjensen,ou=people,dc=example,dc=com
DELETE operation successful for DN uid=bjensen,ou=people,dc=example,dc=com
```

The following command deletes the ou=Groups entry and all entries underneath ou=Groups.

```
$ ldapdelete -p 1389 -D "cn=Directory Manager" -w password -x
 ou=groups,dc=example,dc=com
Processing DELETE request for ou=groups,dc=example,dc=com
DELETE operation successful for DN ou=groups,dc=example,dc=com
```

ldapmodify

ldapmodify — perform LDAP modify, add, delete, mod DN operations

ldapmodify

`ldapmodify` {options}

1 Description

This utility can be used to perform LDAP modify, add, delete, and modify DN operations in the directory.

When not using a file to specify modifications, end your input with EOF (Ctrl+D on UNIX, Ctrl+Z on Windows).

2 Options

The following options are supported.

`-a, --defaultAdd`

Treat records with no changetype as add operations

`--assertionFilter {filter}`

Use the LDAP assertion control with the provided filter

`-c, --continueOnError`

Continue processing even if there are errors

`-f, --filename {file}`

LDIF file containing the changes to apply

`-J, --control {controloid[:criticality[:value|::b64value|:<filePath]]}`

Use a request control with the provided information

`-n, --dry-run`

Show what would be done but do not perform any operation

`--postReadAttributes {attrList}`

Use the LDAP ReadEntry post-read control

`--preReadAttributes {attrList}`

Use the LDAP ReadEntry pre-read control

`-Y, --proxyAs {authzID}`

Use the proxied authorization control with the given authorization ID

2.1 LDAP Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-E, --reportAuthzID`

Use the authorization identity control

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server port number

Default value: 389

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-q, --useStartTLS`

Use StartTLS to secure communication with the server

`-r, --useSASLExternal`

Use the SASL EXTERNAL authentication mechanism

`--trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-V, --ldapVersion {version}`

LDAP protocol version number

Default value: 3

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

`-Z, --useSSL`

Use SSL for secure communication with the server

2.2 Utility Input/Output Options

`-i, --encoding {encoding}`

Use the specified character set for command-line input

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

`-v, --verbose`

Use verbose mode

2.3 General Options

`--version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

`ldap-error`

An LDAP error occurred while processing the operation.

LDAP result codes are described in RFC 4511. Also see the additional information for details.

89

An error occurred while parsing the command-line arguments.

4 Files

You can use `~/.opendj/tools.properties` to set the defaults for bind DN, host name, and port number as in the following example.

```
hostname=directory.example.com
port=1389
bindDN=uid=kvaughan,ou=People,dc=example,dc=com

ldapcompare.port=1389
ldapdelete.port=1389
ldapmodify.port=1389
ldappasswordmodify.port=1389
ldapsearch.port=1389
```

5 Examples

The following example demonstrates use of the command to add an entry to the directory.

```
$ cat newuser.ldif
dn: uid=newuser,ou=People,dc=example,dc=com
uid: newuser
facsimileTelephoneNumber: +1 408 555 1213
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: top
givenName: New
cn: New User
cn: Real Name
telephoneNumber: +1 408 555 1212
sn: Jensen
roomNumber: 1234
homeDirectory: /home/newuser
uidNumber: 10389
mail: newuser@example.com
l: South Pole
ou: Product Development
ou: People
gidNumber: 10636

$ ldapmodify -p 1389 -a -f newuser.ldif
 -D uid=kvaughan,ou=people,dc=example,dc=com -w bribery
Processing ADD request for uid=newuser,ou=People,dc=example,dc=com
ADD operation successful for DN uid=newuser,ou=People,dc=example,dc=com
```

The following example demonstrates adding a Description attribute to the new user's entry.

```
$ cat newdesc.ldif
dn: uid=newuser,ou=People,dc=example,dc=com
changetype: modify
add: description
description: A new user's entry

$ ldapmodify -p 1389 -f newdesc.ldif
 -D uid=kvaughan,ou=people,dc=example,dc=com -w bribery
Processing MODIFY request for uid=newuser,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=newuser,ou=People,dc=example,dc=com
```

The following example demonstrates changing the Description attribute for the new user's entry.

```
$ cat moddesc.ldif
dn: uid=newuser,ou=People,dc=example,dc=com
changetype: modify
replace: description
description: Another description

$ ldapmodify -p 1389 -f moddesc.ldif
 -D uid=kvaughan,ou=people,dc=example,dc=com -w bribery
Processing MODIFY request for uid=newuser,ou=People,dc=example,dc=com
MODIFY operation successful for DN uid=newuser,ou=People,dc=example,dc=com
```

The following example demonstrates deleting the new user's entry.

```
$ cat deluser.ldif
dn: uid=newuser,ou=People,dc=example,dc=com
changetype: delete

$ ldapmodify -p 1389 -f deluser.ldif
 -D uid=kvaughan,ou=people,dc=example,dc=com -w bribery
Processing DELETE request for uid=newuser,ou=People,dc=example,dc=com
DELETE operation successful for DN uid=newuser,ou=People,dc=example,dc=com
```

ldappasswordmodify

ldappasswordmodify — perform LDAP password modifications

ldappasswordmodify

`ldappasswordmodify` {options}

1 Description

This utility can be used to perform LDAP password modify operations in the directory.

2 Options

The following options are supported.

`-a, --authzID {authzID}`

Authorization ID for the user entry whose password should be changed

The authorization ID is a string having either the prefix `dn:` followed by the user's distinguished name, or the prefix `u:` followed by a user identifier that depends on the identity mapping used to match the user identifier to an entry in the directory. Examples include `dn:uid=bjensen,ou=People,dc=example,dc=com`, and, if we assume that `bjensen` is mapped to Barbara Jensen's entry, `u:bjensen`.

`-A, --provideDNForAuthzID`

Use the bind DN as the authorization ID for the password modify operation

`-c, --currentPassword {currentPassword}`

Current password for the target user

`-C, --currentPasswordFile {file}`

Path to a file containing the current password for the target user

`-J, --control {controloid[:criticality[:value|::b64value|:<filePath]]}`

Use a request control with the provided information

`-n, --newPassword {newPassword}`

New password to provide for the target user

`-N, --newPasswordFile {file}`

Path to a file containing the new password to provide for the target user

2.1 LDAP Connection Options

`--certNickname {nickname}`

Nickname of certificate for SSL client authentication

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default: 30000

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-p, --port {port}`

Directory server port number

Default value: 389

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-q, --useStartTLS`

Use StartTLS to secure communication with the server

`--trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

`-Z, --useSSL`

Use SSL for secure communication with the server

2.2 Utility Input/Output Options

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

`-?, -H, --help`

Display usage information

`-V, --version`

Display directory server version information

2.3 General Options

`--version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

`ldap-error`

An LDAP error occurred while processing the operation.

LDAP result codes are described in RFC 4511. Also see the additional information for details.

89

An error occurred while parsing the command-line arguments.

4 Files

You can use `~/.opendj/tools.properties` to set the defaults for bind DN, host name, and port number as in the following example.

```
hostname=directory.example.com
port=1389
bindDN=uid=kvaughan,ou=People,dc=example,dc=com

ldapcompare.port=1389
ldapdelete.port=1389
ldapmodify.port=1389
ldappasswordmodify.port=1389
ldapsearch.port=1389
```

5 Examples

The following example demonstrates a user changing the password for her entry.

```
$ cat /tmp/currpwd.txt /tmp/newpwd.txt
bribery
secret12
$ ldappasswordmodify -p 1389 -C /tmp/currpwd.txt -N /tmp/newpwd.txt
-A -D uid=kvaughan,ou=people,dc=example,dc=com -w bribery
The LDAP password modify operation was successful
```

ldapsearch

ldapsearch — perform LDAP search operations

ldapsearch

`ldapsearch` {options} [filter] [attributes...]

1 Description

This utility can be used to perform LDAP search operations in the directory.

2 Options

The following options are supported.

`-a, --dereferencePolicy {dereferencePolicy}`

Alias dereference policy ('never', 'always', 'search', or 'find')

Default value: never

`-A, --typesOnly`

Only retrieve attribute names but not their values

`--assertionFilter {filter}`

Use the LDAP assertion control with the provided filter

`-b, --baseDN {baseDN}`

Base DN format string

`-c, --continueOnError`

Continue processing even if there are errors

`-C, --persistentSearch ps[:changetype[:changesonly[:entrychgcontrols]]]`

Use the persistent search control

`--countEntries`

Count the number of entries returned by the server

`-e, --getEffectiveRightsAttribute {attribute}`

Specifies geteffectiverights control specific attribute list

`-f, --filename {file}`

LDIF file containing the changes to apply

`-g, --getEffectiveRightsAuthzid {authzID}`

Use geteffectiverights control with the provided authzid

`-G, --virtualListView {before:after:index:count | before:after:value}`

Use the virtual list view control to retrieve the specified results page

`-J, --control {controloid[:criticality[:value|::b64value|:<filePath]]}`

Use a request control with the provided information

`-l, --timeLimit {timeLimit}`

Maximum length of time in seconds to allow for the search

Default value: 0

`--matchedValuesFilter {filter}`

Use the LDAP matched values control with the provided filter

`-n, --dry-run`

Show what would be done but do not perform any operation

`-s, --searchScope {searchScope}`

Search scope ('base', 'one', 'sub', or 'subordinate')

Default value: sub

`subordinate` is an LDAP extension that might not work with all LDAP servers.

`-S, --sortOrder {sortOrder}`

Sort the results using the provided sort order

`--simplePageSize {numEntries}`

Use the simple paged results control with the given page size

Default value: 1000

`--subEntries`

Use subentries control to specify that subentries are visible and normal entries are not

`-Y, --proxyAs {authzID}`

Use the proxied authorization control with the given authorization ID

`-z, --sizeLimit {sizeLimit}`

Maximum number of entries to return from the search

Default value: 0

2.1 LDAP Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-E, --reportAuthzID`

Use the authorization identity control

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server port number

Default value: 389

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-q, --useStartTLS`

Use StartTLS to secure communication with the server

`-r, --useSASLExternal`

Use the SASL EXTERNAL authentication mechanism

`--trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`--usePasswordPolicyControl`

Use the password policy request control

`-V, --ldapVersion {version}`

LDAP protocol version number

Default value: 3

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

`-Z, --useSSL`

Use SSL for secure communication with the server

2.2 Utility Input/Output Options

`-i, --encoding {encoding}`

Use the specified character set for command-line input

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

`-T, --dontWrap`

Do not wrap long lines

`-v, --verbose`

Use verbose mode

2.3 General Options

`--version`

Display version information

`-?, -H, --help`

Display usage information

3 Filter

The filter argument is a string representation of an LDAP search filter as in `(cn=Babs Jensen)`, `(&(objectClass=Person)(|(sn=Jensen)(cn=Babs J*)))`, or `(cn:caseExactMatch:=Fred Flintstone)`.

4 Attribute

The optional attribute list specifies the attributes to return in the entries found by the search. In addition to identifying attributes by name such as `cn sn mail` and so forth, you can use the following notations, too.

`*`

Return all user attributes such as `cn`, `sn`, and `mail`.

`+`

Return all operational attributes such as `etag` and `pwdPolicySubentry`.

`@objectclass`

Return all attributes of the specified object class, where *`objectclass`* is one of the object classes on the entries returned by the search.

5 Exit Codes

0

The command completed successfully.

`ldap-error`

An LDAP error occurred while processing the operation.

LDAP result codes are described in RFC 4511. Also see the additional information for details.

89

An error occurred while parsing the command-line arguments.

6 Files

You can use `~/.opendj/tools.properties` to set the defaults for bind DN, host name, and port number as in the following example.

```
hostname=directory.example.com
port=1389
bindDN=uid=kvaughan,ou=People,dc=example,dc=com

ldapcompare.port=1389
ldapdelete.port=1389
ldapmodify.port=1389
ldappasswordmodify.port=1389
ldapsearch.port=1389
```

7 Examples

The following example searches for entries with UID containing `jensen`, returning only DNs and uid values.

```
$ ldapsearch -p 1389 -b dc=example,dc=com "(uid=*jensen*)" uid
dn: uid=ajensen,ou=People,dc=example,dc=com
uid: ajensen

dn: uid=bjensen,ou=People,dc=example,dc=com
uid: bjensen

dn: uid=gjensen,ou=People,dc=example,dc=com
uid: gjensen

dn: uid=jjensen,ou=People,dc=example,dc=com
uid: jjensen

dn: uid=kjensen,ou=People,dc=example,dc=com
uid: kjensen

dn: uid=rjensen,ou=People,dc=example,dc=com
uid: rjensen

dn: uid=tjensen,ou=People,dc=example,dc=com
uid: tjensen


Result Code:  0 (Success)
```

You can also use `@`*`objectclass`* notation in the attribute list to return the attributes of a particular object class. The following example shows how to return attributes of the `inetOrgPerson` object class.

```
$ ldapsearch -p 1389 -b dc=example,dc=com "(uid=bjensen)" @inetorgperson
dn: uid=bjensen,ou=People,dc=example,dc=com
givenName: Barbara
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: top
uid: bjensen
cn: Barbara Jensen
cn: Babs Jensen
telephoneNumber: +1 408 555 1862
sn: Jensen
roomNumber: 0209
mail: bjensen@example.com
l: Cupertino
ou: Product Development
ou: People
facsimileTelephoneNumber: +1 408 555 1992
```

You can use `+` in the attribute list to return all operational attributes, as in the following example.

```
$ ldapsearch -p 1389 -b dc=example,dc=com "(uid=bjensen)" +
dn: uid=bjensen,ou=People,dc=example,dc=com
numSubordinates: 0
structuralObjectClass: inetOrgPerson
etag: 0000000073c29972
pwdPolicySubentry: cn=Default Password Policy,cn=Password Policies,cn=config
subschemaSubentry: cn=schema
hasSubordinates: false
entryDN: uid=bjensen,ou=people,dc=example,dc=com
entryUUID: fc252fd9-b982-3ed6-b42a-c76d2546312c
```

ldif-diff

ldif-diff — compare small LDIF files

ldif-diff

`ldif-diff` {options}

1 Description

This utility can be used to compare two LDIF files and report the differences in LDIF format.

2 Options

The following options are supported.

`-a, --ignoreAttrs {file}`

File containing a list of attributes to ignore when computing the difference.

`--checkSchema`

Takes into account the syntax of the attributes as defined in the schema to make the value comparison. The provided LDIF files must conform to the server schema.

`-e, --ignoreEntries {file}`

File containing a list of entries (DN) to ignore when computing the difference.

`-o, --outputLDIF {file}`

File to which the output should be written.

`-O, --overwriteExisting`

Any existing output file should be overwritten rather than appending to it.

`-r, --useCompareResultCode`

Use the LDAP compare result as an exit code for reporting differences between the two LDIF files.

`-s, --sourceLDIF {file}`

LDIF file to use as the source data.

`-S, --singleValueChanges`

Each attribute-level change should be written as a separate modification per attribute value rather than one modification per entry.

`-t, --targetLDIF {file}`

LDIF file to use as the target data.

`-V, --version`

Display version information.

`-?, -H, --help`

Display usage information.

3 Exit Codes

0

The command completed successfully.

5

The -r option was used, and no differences are reported.

6

The-r option was used, and differences are reported.

other

An error occurred.

4 Examples

The following example demonstrates use of the command with two small LDIF files.

```
$ cat /path/to/newuser.ldif
dn: uid=newuser,ou=People,dc=example,dc=com
uid: newuser
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
cn: New User
sn: User
ou: People
```

```
mail: newuser@example.com
userPassword: changeme

$ cat /path/to/neweruser.ldif
dn: uid=newuser,ou=People,dc=example,dc=com
uid: newuser
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
cn: New User
sn: User
ou: People
mail: newuser@example.com
userPassword: secret12
description: A new description.

$ ldif-diff -s /path/to/newuser.ldif -t /path/to/neweruser.ldif
dn: uid=newuser,ou=People,dc=example,dc=com
changetype: modify
add: userPassword
userPassword: secret12
-
delete: userPassword
userPassword: changeme
-
add: description
description: A new description.
```

ldifmodify

ldifmodify — apply LDIF changes to LDIF

ldifmodify

`ldifmodify` {options}

1 Description

This utility can be used to apply a set of modify, add, and delete operations against data in an LDIF file.

2 Options

The following options are supported.

`-m, --changesLDIF {ldifFile}`

LDIF file containing the changes to apply.

`-s, --sourceLDIF {ldifFile}`

LDIF file containing the data to be updated.

`-t, --targetLDIF {ldifFile}`

File to which the updated data should be written.

`-V, --version`

Display version information.

`-?, -H, --help`

Display usage information.

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

The following example demonstrates use of the command.

```
$ cat /path/to/newuser.ldif
dn: uid=newuser,ou=People,dc=example,dc=com
uid: newuser
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: top
cn: New User
sn: User
ou: People
mail: newuser@example.com
userPassword: changeme

$ cat /path/to/newdiff.ldif
dn: uid=newuser,ou=People,dc=example,dc=com
changetype: modify
add: userPassword
userPassword: secret12
-
delete: userPassword
userPassword: changeme
-
add: description
description: A new description.

$ ldifmodify -s /path/to/newuser.ldif -m /path/to/newdiff.ldif -t neweruser.ldif
$ cat neweruser.ldif
dn: uid=newuser,ou=People,dc=example,dc=com
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: top
uid: newuser
description: A new description.
cn: New User
sn: User
userPassword: secret12
mail: newuser@example.com
ou: People
```

ldifsearch

ldifsearch — search LDIF with LDAP filters

ldifsearch

`ldifsearch` {options} [filter] [attribute...]

1 Description

This utility can be used to perform search operations against data in an LDIF file.

2 Options

The following options are supported.

`-b, --baseDN {baseDN}`

The base DN for the search. Multiple base DNs may be specified by providing the option multiple times. If no base DN is provided, then the root DSE will be used.

`-f, --filterFile {filterFile}`

The path to the file containing the search filter(s) to use. If this is not provided, then the filter must be provided on the command line after all configuration options.

`-l, --ldifFile {ldifFile}`

LDIF file containing the data to search. Multiple files may be specified by providing the option multiple times. If no files are provided, the data will be read from standard input.

`-o, --outputFile {outputFile}`

The path to the output file to which the matching entries should be written. If this is not provided, then the data will be written to standard output.

`-O, --overwriteExisting`

Any existing output file should be overwritten rather than appending to it.

`-s, --searchScope {scope}`

The scope for the search. It must be one of 'base', 'one', 'sub', or 'subordinate'. If it is not provided, then 'sub' will be used.

`-t, --timeLimit {timeLimit}`

Maximum length of time (in seconds) to spend processing.

Default value: 0

`-T, --dontWrap`

Long lines should not be wrapped.

`-V, --version`

Display version information.

`-z, --sizeLimit {sizeLimit}`

Maximum number of matching entries to return.

Default value: 0

`-?, -H, --help`

Display usage information.

3 Filter

The filter argument is a string representation of an LDAP search filter as in `(cn=Babs Jensen)`, `(&(objectClass=Person)(|(sn=Jensen)(cn=Babs J*)))`, or `(cn:caseExactMatch:=Fred Flintstone)`.

4 Attribute

The optional attribute list specifies the attributes to return in the entries found by the search. In addition to identifying attributes by name such as `cn sn mail` and so forth, you can use the following notations, too.

`*`

Return all user attributes such as `cn`, `sn`, and `mail`.

`+`

Return all operational attributes such as `etag` and `pwdPolicySubentry`.

`@`*`objectclass`*

Return all attributes of the specified object class, where *`objectclass`* is one of the object classes on the entries returned by the search.

5 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

6 Examples

The following example demonstrates use of the command.

```
$ ldifsearch -l /path/to/Example.ldif -b dc=example,dc=com uid=bjensen
dn: uid=bjensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: top
uid: bjensen
userpassword: hifalutin
facsimiletelephonenumber: +1 408 555 1992
givenname: Barbara
cn: Barbara Jensen
cn: Babs Jensen
telephonenumber: +1 408 555 1862
sn: Jensen
roomnumber: 0209
homeDirectory: /home/bjensen
mail: bjensen@example.com
l: Cupertino
ou: Product Development
ou: People
uidNumber: 1076
gidNumber: 1000
```

You can also use `@objectclass` notation in the attribute list to return the attributes of a particular object class. The following example shows how to return attributes of the `posixAccount` object class.

```
$ ldifsearch --ldifFile /path/to/Example.ldif
 --baseDN dc=example,dc=com "(uid=bjensen)" @posixaccount
dn: uid=bjensen,ou=People,dc=example,dc=com
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: top
uid: bjensen
userpassword: hifalutin
cn: Barbara Jensen
cn: Babs Jensen
homeDirectory: /home/bjensen
uidNumber: 1076
gidNumber: 1000
```

list-backends

list-backends — list OpenDJ backends and base DNs

list-backends

`list-backends` [options]

1 Description

This utility can be used to list the backends and base DNs configured in the Directory Server.

2 Options

The following options are supported.

`-b, --baseDN {baseDN}`

Base DN for which to list the backend ID.

`-n, --backendID {backendName}`

Backend ID of the backend for which to list the base DNs.

`-V, --version`

Display version information.

`-?, -H, --help`

Display usage information.

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

```
$ list-backends
Backend ID         : Base DN
-------------------:----------------------
adminRoot          : cn=admin data
ads-truststore     : cn=ads-truststore
backup             : cn=backups
config             : cn=config
monitor            : cn=monitor
myCompanyRoot      : "dc=myCompany,dc=com"
myOrgRoot          : o=myOrg
replicationChanges : dc=replicationChanges
schema             : cn=schema
tasks              : cn=tasks
userRoot           : "dc=example,dc=com"
```

make-ldif

make-ldif — generate test LDIF

make-ldif

`make-ldif` {options}

1 Description

This utility can be used to generate LDIF data based on a definition in a template file.

2 Options

The following options are supported.

`-o, --ldifFile {file}`

The path to the LDIF file to be written.

`-s, --randomSeed {seed}`

The seed to use to initialize the random number generator.

Default value: 0

`-t, --templateFile {file}`

The path to the template file with information about the LDIF data to generate.

`-V, --version`

Display version information.

`-?, -H, --help`

Display usage information.

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

The following example uses the default template to generate LDIF.

```
$ make-ldif -t ../config/MakeLDIF/example.template -o ../ldif/generated.ldif
Processed 1000 entries
Processed 2000 entries
...
Processed 10000 entries
LDIF processing complete.  10003 entries written
```

5 See Also

make-ldif.template(5)

make-ldif.template

make-ldif.template — template file for the make-ldif command

make-ldif.template

```
# Comment lines start with #.

# Optionally include classes that define custom tags.
# Custom tag classes extend org.opends.server.tools.makeldif.Tag and
# must be on the class path when you run make-ldif.
#
include custom.makeldif.tag.ClassName
...

# Optionally define constants used in the template.
# To reference constants later, put brackets around the name: [constant-name]
#
define constant-name=value
...

# Define branches by suffix DN, such as the following:
#
#  dc=example,dc=com
#  ou=People,dc=example,dc=com
#  ou=Groups,dc=example,dc=com
#
# make-ldif generates the necessary object class definitions and RDNs.
#
# A branch can have subordinateTemplates that define templates to use for
# the branch entry.
#
# A branch can have additional attributes generated on the branch entry. See
# the Description below for more information on specifying attribute values.
#
branch: suffix-dn
[subordinateTemplate: template-name:number
...]
[attribute: attr-value
...]

...

# Define entries using templates.
#
# A template can extend another template.
# A template defines the RDN attribute(s) used for generated entries.
# A template can have a subordinateTemplate that defines a template to use for
# the generated entries.
#
# A template then defines attributes. See the Description below for more
# information on specifying attribute values.
#
template: template-name
[extends: template-name]
rdnAttr: attribute[+attribute ...]
[subordinateTemplate: template-name:number]
[attribute: attr-value
...]
```

```
...
```

1 Description

Template files specify how to build LDIF. They allow you to define variables, insert random values from other files, and generally build arbitrarily large LDIF files for testing purposes. You pass template files to the **make-ldif** command when generating LDIF.

The Synopsis above shows the layout for a **make-ldif** template file. This section focuses on what you can do to specify entry attribute values, called *attr-value* in the Synopsis section.

Specifying Attribute Values

When specifying attribute values in **make-ldif** templates, you can use static text and constants that you have defined, enclosing names for constants in brackets, [myConstant]. You can use more than one constant per line, as in the following example.

```
description: Description for [org] under [suffix]
```

You can also use two kinds of tags when specifying attribute values. One kind of tag gets replaced with the value of another attribute in the generated entry. Such tags are delimited with braces, { }. For example, if your template includes definitions for first name and last name attributes:

```
givenName: <first>
sn: <last>
```

Then you can define a mail attribute that uses the values of both attributes, and an initials attribute that takes the first character of each.

```
mail: {givenName}.{sn}@[myDomain]
initials: {givenName:1}{sn:1}
```

The other kind of tag is delimited with < and >, as shown above in the example with <first> and <last>. Tag names are not case sensitive. Many tags can take arguments separated by colons, :, from the tag names within the tag.

Use backslashes to escape literal start tag characters (< [{) as shown in the following example, and to escape literal end tag characters within tags (>] }).

```
scimMail: \{"emails": \[\{"value": "{mail}", "type": "work", "primary": true}]}
xml: \<id>{uid}\</id>
```

OpenDJ supports the following tags.

<DN>

The DN tag gets replaced by the distinguished name of the current entry. An optional integer argument specifies the subcomponents of the DN to generate. For example, if the DN of the entry is `uid=bjensen,ou=People,dc=example,dc=com` `<DN:1>` gets replaced by `uid=bjensen`, and `<DN:-2>` gets replaced by `dc=example,dc=com`.

<File>

The File tag gets replaced by a line from a text file you specify. The File tag takes a required argument, the path to the text file, and an optional second argument, either `random` or `sequential`. For the file argument, either you specify an absolute path to the file such as `<file:/path/to/myDescriptions>`, or you specify a path relative to the `/path/to/opendj/config/MakeLDIF/` directory such as `<file:streets>`. For the second argument, if you specify `sequential` then lines from the file are read in sequential order. Otherwise, lines from the file are read in random order.

<First>

The first name tag gets replaced by a random line from `/path/to/opendj/config/MakeLDIF/first.names`. Combinations of generated first and last names are unique, with integers appended to the name strings if not enough combinations are available.

<GUID>

The GUID tag gets replaced by a 128-bit, type 4 (random) universally unique identifier, such as `f47ac10b-58cc-4372-a567-0e02b2c3d479`.

<IfAbsent>

The IfAbsent tag takes as its first argument the name of another attribute, and optionally as its second argument a value to use. This tag causes the attribute to be generated only if the named attribute is not present on the generated entry. Use this tag when you have used `<Presence>` to define another attribute that is not always present on generated entries.

<IfPresent>

The IfPresent takes as its first argument the name of another attribute, and optionally as its second argument a value to use. This tag causes the attribute to be generated only if the named attribute is also present on the generated entry. Use this tag when you have used `<Presence>` to define another attribute that is sometimes present on generated entries.

<Last>

The last name tag gets replaced by a random line from `/path/to/opendj/config/MakeLDIF/last.names`. Combinations of generated first and last names are unique, with integers appended to the name strings if not enough combinations are available.

<List>

The List tag gets replaced by one of the values from the list of arguments you provide. For example, `<List:bronze:silver:gold>` gets replaced with `bronze`, `silver`, or `gold`.

You can weight arguments to ensure some arguments are selected more often than others. For example, if you want two bronze for one silver and one gold, use `<List:bronze;2:silver;1:gold;1>`.

<ParentDN>

The ParentDN tag gets replaced by the distinguished name of the parent entry. For example, if the DN of the entry is `uid=bjensen,ou=People,dc=example,dc=com`, `<ParentDN>` gets replaced by `ou=People,dc=example,dc=com`.

<Presence>

The Presence tag takes a percent argument. It does not get replaced by a value itself, but instead results in the attribute being generated on the percentage of entries you specify in the argument. For example, `description: <Presence:50>A description` generates `description: A description` on half the entries.

<Random>

The Random tag lets you generate a variety of random numbers and strings. The Random tag has the following subtypes, which you include as arguments, that is `<Random:`*`subtype`*`>`.

- `alpha:`*`length`*
- `alpha:`*`minlength`*`:`*`maxlength`*
- `numeric:`*`length`*
- `numeric:`*`minvalue`*`:`*`maxvalue`*
- `numeric:`*`minvalue`*`:`*`maxvalue`*`:`*`format`*, where *`format`* is a `java.text.DecimalFormat` pattern
- `alphanumeric:`*`length`*
- `alphanumeric:`*`minlength`*`:`*`maxlength`*
- `chars:`*`characters`*`:`*`length`*

- `chars:`*`characters`*`:`*`minlength`*`:`*`maxlength`*

- `hex:`*`length`*

- `hex:`*`minlength`*`:`*`maxlength`*

- `base64:`*`length`*

- `base64:`*`minlength`*`:`*`maxlength`*

- `month`

- `month:`*`maxlength`*

- `telephone`, a telephone number starting with the country code +1

<RDN>

The RDN tag gets replaced with the RDN of the entry. Use this in the template after you have specified `rdnAttr` so that the RDN has already been generated when this tag is replaced.

An optional integer argument specifies the subcomponents of the RDN to generate.

<Sequential>

The Sequential tag gets replaced by a sequentially increasing generated integer. The first optional integer argument specifies the starting number. The second optional boolean argument specifies whether to start over when generating entries for a new parent entry. For example, `<Sequential>:42:true` starts counting from 42, and starts over when the parent entry changes from `o=Engineering` to `o=Marketing`.

<_DN>

The _DN tag gets replaced by the DN of the current entry with underscores in the place of commas.

<_ParentDN>

The _ParentDN tag gets replaced by the DN the parent entry with underscores in the place of commas.

2 Examples

The following example generates 10 organization units, each containing 50 entries.

```
define suffix=dc=example,dc=com
define maildomain=example.com
define numusers=50
define numorgs=10

branch: [suffix]

branch: ou=People,[suffix]
subordinateTemplate: orgunit:[numorgs]
description: This is the People container
telephoneNumber: +33 00010002

template: orgunit
subordinateTemplate: person:[numusers]
rdnAttr: ou
ou: Org-<sequential:0>
objectClass: top
objectClass: organizationalUnit
description: This is the {ou} organizational unit

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}.
```

3 See Also

make-ldif(1), `/path/to/opendj/config/MakeLDIF/example.template`

manage-account

manage-account — manage state of directory server accounts

manage-account

`manage-account` *`subcommand`* {options}

1 Description

This utility can be used to retrieve and manipulate the values of password policy state variables.

2 Subcommands

The following subcommands are supported.

manage-account clear-account-is-disabled

Clear account disabled state information from the user account

manage-account get-account-expiration-time

Display when the user account will expire

manage-account get-account-is-disabled

Display information about whether the user account has been administratively disabled

manage-account get-all

Display all password policy state information for the user

manage-account get-authentication-failure-times

Display the authentication failure times for the user

manage-account get-grace-login-use-times

Display the grace login use times for the user

manage-account get-last-login-time

Display the time that the user last authenticated to the server

manage-account get-password-changed-by-required-time

Display the required password change time with which the user last complied

manage-account get-password-changed-time

Display the time that the user's password was last changed

manage-account get-password-expiration-warned-time

Display the time that the user first received an expiration warning notice

manage-account get-password-history

Display password history state values for the user

manage-account get-password-is-reset

Display information about whether the user will be required to change his or her password on the next successful authentication

manage-account get-password-policy-dn

Display the DN of the password policy for the user

manage-account get-remaining-authentication-failure-count

Display the number of remaining authentication failures until the user's account is locked

manage-account get-remaining-grace-login-count

Display the number of grace logins remaining for the user

manage-account get-seconds-until-account-expiration

Display the length of time in seconds until the user account expires

manage-account get-seconds-until-authentication-failure-unlock

Display the length of time in seconds until the authentication failure lockout expires

manage-account get-seconds-until-idle-lockout

Display the length of time in seconds until user's account is locked because it has remained idle for too long

manage-account get-seconds-until-password-expiration

Display length of time in seconds until the user's password expires

manage-account get-seconds-until-password-expiration-warning

Display the length of time in seconds until the user should start receiving password expiration warning notices

manage-account get-seconds-until-password-reset-lockout

Display the length of time in seconds until user's account is locked because the user failed to change the password in a timely manner after an administrative reset

manage-account get-seconds-until-required-change-time

Display the length of time in seconds that the user has remaining to change his or her password before the account becomes locked due to the required change time

manage-account set-account-is-disabled

Specify whether the user account has been administratively disabled

3 Global Options

The following global options are supported.

`-b, --targetDN {targetDN}`

The DN of the user entry for which to get and set password policy state information

3.1 LDAP Connection Options

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server administration port number

Default value: 4444

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

Use `-w -` to have the command prompt for the password, rather than enter the password on the command line.

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

3.2 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

4 Exit Codes

0

The command completed successfully.

89

An error occurred while parsing the command-line arguments.

5 Examples

For the following examples, the directory admin user, Kirsten Vaughan, has `ds-privilege-name: password-reset`, and the following ACI on `ou=People,dc=example,dc=com`.

```
(target="ldap:///ou=People,dc=example,dc=com") (targetattr ="*||+")(
 version 3.0;acl "Admins can run amok"; allow(all) groupdn =
 "ldap:///cn=Directory Administrators,ou=Groups,dc=example,dc=com";)
```

The following command locks a user account.

```
$ manage-account -p 4444 -D "uid=kvaughan,ou=people,dc=example,dc=com"
 -w bribery set-account-is-disabled -O true
 -b uid=bjensen,ou=people,dc=example,dc=com -X
Account Is Disabled:  true
```

The following command unlocks a user account.

```
$ manage-account -p 4444 -D "uid=kvaughan,ou=people,dc=example,dc=com"
 -w bribery clear-account-is-disabled
 -b uid=bjensen,ou=people,dc=example,dc=com -X
Account Is Disabled:  false
```

manage-tasks

manage-tasks — manage OpenDJ server administration tasks

manage-tasks

`manage-tasks` {options}

1 Description

This utility can be used to obtain a list of tasks scheduled to run within the directory server as well as information about individual tasks.

2 Options

The following options are supported.

`-c, --cancel {taskID}`

ID of a particular task to cancel

`-i, --info {taskID}`

ID of a particular task about which this tool will display information

`-s, --summary`

Print a summary of tasks

2.1 LDAP Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server administration port number

Default value: 4444

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

2.2 Utility Input/Output Options

`-n, --no-prompt`

Use non-interactive mode. If data in the command is missing, the user is not prompted and the tool will fail

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

2.3 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

The following example demonstrates use of the command with a server that does daily backups at 2:00 AM.

```
$ manage-tasks -p 4444 -h opendj.example.com -D "cn=Directory Manager"
 -w password -s

  ID                                 Type    Status
  ---------------------------------------------------------------
  example-backup                     Backup  Recurring
  example-backup-20110622020000000  Backup  Waiting on start time
```

rebuild-index

rebuild-index — rebuild index after configuration change

rebuild-index

`rebuild-index` {options}

1 Description

This utility can be used to rebuild index data within a backend based on the Berkeley DB Java Edition.

2 Options

The following options are supported.

`-b, --baseDN {baseDN}`

> Base DN of a backend supporting indexing. Rebuild is performed on indexes within the scope of the given base DN.

`--clearDegradedState`

> Indicates that indexes do not need rebuilding because they are known to be empty and forcefully marks them as valid. *This is an advanced option which must only be used in cases where a degraded index is known to be empty and does not therefore need rebuilding*. This situation typically arises when an index is created for an attribute which has just been added to the schema.

`-i, --index {index}`

> Names of index(es) to rebuild. For an attribute index this is simply an attribute name. At least one index must be specified for rebuild. Cannot be used with the `--rebuildAll` or `--rebuildDegraded` options.

`--rebuildAll`

> Rebuild all indexes, including any DN2ID, DN2URI, VLV and extensible indexes. Cannot be used with the `--index` or `--rebuildDegraded` options.

`--rebuildDegraded`

> Rebuild all degraded indexes, including any DN2ID, DN2URI, VLV and extensible indexes. Cannot be used with the `--index` or `--rebuildAll` options.

`--tmpdirectory {directory}`

> Path to temporary directory for index scratch files during index rebuilding.

Default value: import-tmp

2.1 Task Backend Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server administration port number

Default value: 4444

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

Use `-w -` to have the command prompt for the password, rather than enter the password on the command line.

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

2.2 Task Scheduling Options

`--completionNotify {emailAddress}`

Email address of a recipient to be notified when the task completes. This option may be specified more than once.

`--dependency {taskID}`

ID of a task upon which this task depends. A task will not start execution until all its dependencies have completed execution.

`--errorNotify {emailAddress}`

Email address of a recipient to be notified if an error occurs when this task executes. This option may be specified more than once.

`--failedDependencyAction {action}`

Action this task will take should one if its dependent tasks fail. The value must be one of PROCESS, CANCEL, DISABLE. If not specified defaults to CANCEL.

`--recurringTask {schedulePattern}`

Indicates the task is recurring and will be scheduled according to the value argument expressed in crontab(5) compatible time/date pattern.

`-t, --start {startTime}`

Indicates the date/time at which this operation will start when scheduled as a server task expressed in YYYYMMDDhhmmssZ format for UTC time or YYYYMMDDhhmmss for local time. A value of '0' will cause the task to be scheduled for immediate execution. When this option is specified the operation will be scheduled to start at the specified time after which this utility will exit immediately.

2.3 Utility Input/Output Options

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

2.4 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

1

An error occurred while parsing the command-line arguments.

4 Examples

The following example schedules a task to start immediately that rebuilds the `cn` (common name) index.

```
$ rebuild-index -p 4444 -h opendj.example.com -D "cn=Directory Manager"
 -w password -b dc=example,dc=com -i cn -t 0
Rebuild Index task 20110607160349596 scheduled to start Jun 7, 2011 4:03:49 PM
```

restore

restore — restore OpenDJ directory data backups

restore

`restore` {options}

1 Description

This utility can be used to restore a backup of a directory server backend.

2 Options

The following options are supported.

`-d, --backupDirectory {backupDir}`

Path to the target directory for the backup file(s)

`-I, --backupID {backupID}`

Use the provided identifier for the backup

`-l, --listBackups`

List available backups in the backup directory

`-n, --dry-run`

Verify the contents of the backup but do not restore it

2.1 Task Backend Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: localhost.localdomain

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server administration port number

Default value: 4444

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

Use `-w -` to have the command prompt for the password, rather than enter the password on the command line.

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

2.2 Task Scheduling Options

`--completionNotify {emailAddress}`

Email address of a recipient to be notified when the task completes. This option may be specified more than once.

`--dependency {taskID}`

ID of a task upon which this task depends. A task will not start execution until all its dependencies have completed execution.

`--errorNotify {emailAddress}`

Email address of a recipient to be notified if an error occurs when this task executes. This option may be specified more than once.

`--failedDependencyAction {action}`

Action this task will take should one if its dependent tasks fail. The value must be one of PROCESS, CANCEL, DISABLE. If not specified defaults to CANCEL.

`--recurringTask {schedulePattern}`

Indicates the task is recurring and will be scheduled according to the value argument expressed in crontab(5) compatible time/date pattern.

`-t, --start {startTime}`

Indicates the date/time at which this operation will start when scheduled as a server task expressed in YYYYMMDDhhmmssZ format for UTC time or YYYYMMDDhhmmss for local time. A value of '0' will cause the task to be scheduled for immediate execution. When this option is specified the operation will be scheduled to start at the specified time after which this utility will exit immediately.

2.3 Utility Input/Output Options

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

2.4 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

1

An error occurred while parsing the command-line arguments.

4 Examples

The following example schedules a restore as a task to begin immediately while OpenDJ directory server is online.

```
$ restore -p 4444 -D "cn=Directory Manager" -w password
 -d /path/to/opendj/bak -I 20110613080032 -t 0
Restore task 20110613155052932 scheduled to start Jun 13, 2011 3:50:52 PM CEST
```

The following example restores data while OpenDJ is offline.

```
$ /path/to/opendj/bin/stop-ds
Stopping Server...
...
$ /path/to/opendj/bin/restore --backupDirectory /path/to/opendj/bak/userRoot
 --listBackups
Backup ID:          20120928102414Z
Backup Date:        28/Sep/2012:12:24:17 +0200
Is Incremental:     false
Is Compressed:      false
Is Encrypted:       false
Has Unsigned Hash:  false
Has Signed Hash:    false
Dependent Upon:     none

$ /path/to/opendj/bin/restore --backupDirectory /path/to/opendj/bak/userRoot
 --backupID 20120928102414Z
[28/Sep/2012:12:26:20 +0200] ... msg=Restored: 00000000.jdb (size 355179)
$ /path/to/opendj/bin/start-ds
[28/Sep/2012:12:27:29 +0200] ... The Directory Server has started successfully
```

setup

setup — install OpenDJ directory server

setup

`setup` [options]

1 Description

This utility can be used to setup the directory server.

2 Options

The following options are supported.

`-a, --addBaseEntry`

Indicates whether to create the base entry in the directory server database

`--adminConnectorPort {port}`

Port on which the Administration Connector should listen for communication

Default value: 4444

`-b, --baseDN {baseDN}`

Base DN for user information in the directory server. Multiple base DNs may be provided by using this option multiple times

Default value: dc=example,dc=com

`-d, --sampleData {numEntries}`

Specifies that the database should be populated with the specified number of sample entries

Default value: 0

`-D, --rootUserDN {rootUserDN}`

DN for the initial root user for the directory server

Default value: cn=Directory Manager

`--generateSelfSignedCertificate`

Generate a self-signed certificate that the server should use when accepting SSL-based connections or performing StartTLS negotiation

`-h, --hostname {host}`

The fully-qualified directory server host name that will be used when generating self-signed certificates for LDAP SSL/StartTLS, the administration connector, and replication

Default value: localhost.localdomain

`-i, --cli`

Use the command line install. If not specified the graphical interface will be launched. The rest of the options (excluding help and version) will only be taken into account if this option is specified

`-j, --rootUserPasswordFile {rootUserPasswordFile}`

Path to a file containing the password for the initial root user for the directory server

`-l, --ldifFile {ldifFile}`

Path to an LDIF file containing data that should be added to the directory server database. Multiple LDIF files may be provided by using this option multiple times

`-N, --certNickname {nickname}`

Nickname of the certificate that the server should use when accepting SSL-based connections or performing StartTLS negotiation

`-O, --doNotStart`

Do not start the server when the configuration is completed

`-p, --ldapPort {port}`

Port on which the Directory Server should listen for LDAP communication

Default value: 389

`-q, --enableStartTLS`

Enable StartTLS to allow secure communication with the server using the LDAP port

`-R, --rejectFile {rejectFile}`

Write rejected entries to the specified file

`-S, --skipPortCheck`

Skip the check to determine whether the specified ports are usable

`--skipFile {skipFile}`

Write skipped entries to the specified file

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file. A PIN is required when you specify to use an existing certificate (JKS, JCEKS, PKCS#12 or PKCS#11) as server certificate

`--useJavaKeystore {keyStorePath}`

Path of a Java Key Store (JKS) containing a certificate to be used as the server certificate

`--useJCEKS {keyStorePath}`

Path of a JCEKS containing a certificate to be used as the server certificate

`--usePkcs11Keystore`

Use a certificate in a PKCS#11 token that the server should use when accepting SSL-based connections or performing StartTLS negotiation

`--usePkcs12keyStore {keyStorePath}`

Path of a PKCS#12 key store containing the certificate that the server should use when accepting SSL-based connections or performing StartTLS negotiation

`-w, --rootUserPassword {rootUserPassword}`

Password for the initial root user for the Directory Server

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN. A PIN is required when you specify to use an existing certificate (JKS, JCEKS, PKCS#12 or PKCS#11) as server certificate

`-x, --jmxPort {jmxPort}`

Port on which the Directory Server should listen for JMX communication

Default value: 1689

`-Z, --ldapsPort {port}`

Port on which the Directory Server should listen for LDAPS communication. The LDAPS port will be configured and SSL will be enabled only if this argument is explicitly specified

Default value: 636

2.1 Utility Input/Output Options

`-n, --no-prompt`

Use non-interactive mode. If data in the command is missing, the user is not prompted and the tool will fail

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

`-Q, --quiet`

Run setup in quiet mode. Quiet mode will not output progress information to standard output

`-v, --verbose`

Use verbose mode

2.2 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

The following command installs OpenDJ directory server, enabling StartTLS and importing 100 example entries without interaction.

```
$ ./opendj/setup --cli -b dc=example,dc=com -d 100 -D "cn=Directory Manager"
 -w password -h `hostname` -p 1389
 --generateSelfSignedCertificate --enableStartTLS -n

OpenDJ 4.8.1-SNAPSHOT
Please wait while the setup program initializes...

See /var/.../opends-setup-484...561.log for a detailed log of this operation.

Configuring Directory Server ..... Done.
Configuring Certificates ..... Done.
Importing Automatically-Generated Data (100 Entries) ......... Done.
Starting Directory Server .......... Done.

To see basic server configuration status and configuration you can launch
 /path/to/opendj/bin/status
```

start-ds

start-ds — start OpenDJ directory server

start-ds

`start-ds` [options]

1 Description

This utility can be used to start the directory server, as well as to obtain the server version and other forms of general server information.

2 Options

The following options are supported.

`-L, --useLastKnownGoodConfig`

Attempt to start using the configuration that was in place at the last successful startup (if it is available) rather than using the current active configuration.

`-N, --nodetach`

Do not detach from the terminal and continue running in the foreground. This option cannot be used with the -t, --timeout option.

`-Q, --quiet`

Use quiet mode.

`-s, --systemInfo`

Display general system information.

`-t, --timeout {seconds}`

Maximum time (in seconds) to wait before the command returns (the server continues the startup process, regardless). A value of '0' indicates an infinite timeout, which means that the command returns only when the server startup is completed. The default value is 60 seconds. This option cannot be used with the -N, --nodetach option.

Default value: 200

`-V, --version`

Display version information.

`-?, -H, --help`

Display usage information.

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

The following command starts the server without displaying information about the startup process.

```
$ start-ds -Q
```

status

status — display basic OpenDJ server information

status

`status` [options]

1 Description

This utility can be used to display basic server information.

2 Options

The following options are supported.

2.1 LDAP Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-D, --bindDN {bindDN}`

DN to use to bind to the server

Default value: cn=Directory Manager

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

2.2 Utility Input/Output Options

`-n, --no-prompt`

Use non-interactive mode. If data in the command is missing, the user is not prompted and the tool will fail

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`-r, --refresh {period}`

When this argument is specified, the status command will display its contents periodically. Used to specify the period (in seconds) between two status displays

`-s, --script-friendly`

Use script-friendly mode

2.3 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

```
$ status -D "cn=Directory Manager" -w password

          --- Server Status ---
Server Run Status:        Started
Open Connections:         1

          --- Server Details ---
Host Name:                localhost.localdomain
Administrative Users:     cn=Directory Manager
Installation Path:        /path/to/opendj
Version:                  OpenDJ 4.8.1-SNAPSHOT
Java Version:             1.6.0_29
Administration Connector: Port 4444 (LDAPS)

          --- Connection Handlers ---
Address:Port : Protocol    : State
-------------:-------------:---------
--           : LDIF        : Disabled
8989         : Replication : Enabled
0.0.0.0:161  : SNMP        : Disabled
0.0.0.0:636  : LDAPS       : Disabled
0.0.0.0:1389 : LDAP        : Enabled
0.0.0.0:1689 : JMX         : Disabled

          --- Data Sources ---
Base DN:                      dc=example,dc=com
Backend ID:                   userRoot
Entries:                      160
Replication:                  Enabled
Missing Changes:              0
Age of Oldest Missing Change: <not available>

Base DN:     dc=myCompany,dc=com
Backend ID:  myCompanyRoot
Entries:     3
Replication: Disabled

Base DN:     o=myOrg
Backend ID:  myOrgRoot
Entries:     3
Replication: Disabled
```

stop-ds

stop-ds — stop OpenDJ directory server

stop-ds

`stop-ds` [options]

1 Description

This utility can be used to request that the directory server stop running or perform a restart.

When run without connection options, **stop-ds** sends a signal to the OpenDJ process to stop the server. When run with connection options, the **stop-ds** connects to the OpenDJ administration port and creates a shutdown task to stop the server.

2 Options

The following options are supported.

`-r, --stopReason {stopReason}`

Reason the server is being stopped or restarted.

`-R, --restart`

Attempt to automatically restart the server once it has stopped.

`-t, --stopTime {stopTime}`

Indicates the date/time at which the shutdown operation will begin as a server task expressed in format YYYYMMDDhhmmssZ for UTC time or YYYYMMDDhhmmss for local time. A value of '0' will cause the shutdown to be scheduled for immediate execution. When this option is specified the operation will be scheduled to start at the specified time after which this utility will exit immediately.

`-Y, --proxyAs {authzID}`

Use the proxied authorization control with the given authorization ID.

2.1 LDAP Connection Options

`-D, --bindDN {bindDN}`

DN to use to bind to the server

`-h, --hostname {host}`

Directory server hostname or IP address

Default value: 127.0.0.1

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-p, --port {port}`

Directory server administration port number

Default value: 4444

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

Use `-w -` to have the command prompt for the password, rather than enter the password on the command line.

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

2.2 Utility Input/Output Options

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

`-Q, --quiet`

Use quiet mode

2.3 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

The following example restarts OpenDJ directory server.

```
$ stop-ds --restart
Stopping Server...

...The Directory Server has started successfully
```

uninstall

uninstall — remove OpenDJ directory server software

uninstall

`uninstall` [options]

1 Description

This utility can be used to uninstall the directory server.

2 Options

The following options are supported.

`-a, --remove-all`

Remove all components of the server (this option is not compatible with the rest of remove options)

`-b, --backup-files`

Remove backup files

`-c, --configuration-files`

Remove configuration files

`-d, --databases`

Remove database contents

`-e, --ldif-files`

Remove LDIF files

`-f, --forceOnError`

Specifies whether the uninstall should continue if there is an error updating references to this server in remote server instances or not. This option can only be used with the --no-prompt option.

`-i, --cli`

Specifies to use the command line install. If not specified the graphical interface will be launched. The rest of the options (excluding help and version) will only be taken into account if this option is specified

`-l, --server-libraries`

Remove Server Libraries and Administrative Tools

`-L, --log-files`

Remove log files

2.1 LDAP Connection Options

`--connectTimeout {timeout}`

Maximum length of time (in milliseconds) that can be taken to establish a connection. Use '0' to specify no time out.

Default value: 30000

`-h, --referencedHostName {host}`

The name of this host (or IP address) as it is referenced in remote servers for replication

Default value: localhost.localdomain

`-I, --adminUID {adminUID}`

User ID of the Global Administrator to use to bind to the server.

Default value: admin

`-j, --bindPasswordFile {bindPasswordFile}`

Bind password file

`-K, --keyStorePath {keyStorePath}`

Certificate key store path

`-N, --certNickname {nickname}`

Nickname of certificate for SSL client authentication

`-o, --saslOption {name=value}`

SASL bind options

`-P, --trustStorePath {trustStorePath}`

Certificate trust store path

`-T, --trustStorePassword {trustStorePassword}`

Certificate trust store PIN

`-u, --keyStorePasswordFile {keyStorePasswordFile}`

Certificate key store PIN file

`-U, --trustStorePasswordFile {path}`

Certificate trust store PIN file

`-w, --bindPassword {bindPassword}`

Password to use to bind to the server

`-W, --keyStorePassword {keyStorePassword}`

Certificate key store PIN

`-X, --trustAll`

Trust all server SSL certificates

2.2 Utility Input/Output Options

`-n, --no-prompt`

Use non-interactive mode. If data in the command is missing, the user is not prompted and the tool will fail

`--noPropertiesFile`

No properties file will be used to get default command line argument values

`--propertiesFilePath {propertiesFilePath}`

Path to the file containing default property values used for command line arguments

`-Q, --quiet`

Run setup in quiet mode. Quiet mode will not output progress information to standard output

`-v, --verbose`

Use verbose mode

2.3 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

> 0

An error occurred.

4 Examples

The following command removes OpenDJ directory server without interaction.

```
$ ./opendj/uninstall -a --cli -I admin -w password -n

Stopping Directory Server ..... Done.
Deleting Files under the Installation Path ..... Done.

The Uninstall Completed Successfully.
To complete the uninstallation, you must delete manually the following files
and directories:
/path/to/opendj/lib
See /var/.../opends-uninstall-3...0.log for a detailed log of this operation.
$ rm -rf opendj
```

upgrade

upgrade — upgrade OpenDJ configuration & application data

upgrade

`upgrade` {options}

1 Description

This utility upgrades OpenDJ configuration (schema, directory server configuration, and other configuration files) and application data (primarily directory data) so that it is compatible with the binary files and scripts that are installed.

The **upgrade** command thus performs only part of the upgrade process, which includes the following phases for a single server.

1. Get and unpack a newer version of OpenDJ directory server software.
2. Stop the current OpenDJ directory server.
3. Overwrite existing binary and script files with those of the newer version, and then run this utility, the **upgrade** command, before restarting OpenDJ.
4. Start the upgraded OpenDJ directory server.

> **Important**
>
> The **upgrade** command *does not back up OpenDJ before you upgrade, nor does it restore OpenDJ if the **upgrade** command fails*. In order to revert a failed upgrade, make sure you back up OpenDJ directory server before you overwrite existing binary and script files.

By default, the **upgrade** command requests confirmation before making important configuration changes. You can use the `--no-prompt` option to run the command non-interactively.

When using the `--no-prompt` option, if the **upgrade** command cannot complete because it requires confirmation for a potentially very long or critical task, then it exits with an error and a message about how to finish making the changes. You can add the `--force` option to force a non-interactive upgrade to continue in this case, also performing long running and critical tasks.

After upgrading, see the resulting `upgrade.log` file for a full list of operations performed.

2 Options

The following options are supported.

`--acceptLicense`

Automatically accepts the product license if there is one in the delivery.

`--force`

Forces a non-interactive upgrade to continue even if it requires user interaction. In particular, long running or critical upgrade tasks, such as re-indexing, which require user confirmation will be skipped. This option may only be used with the `--no-prompt` option.

`--ignoreErrors`

Ignores any errors which occur during the upgrade. This option should be used with caution and may be useful in automated deployments where potential errors are known in advance and resolved after the upgrade has completed.

2.1 Utility Input/Output Options

`-n, --no-prompt`

Use non-interactive mode. Prompt for any required information rather than fail.

`-Q, --quiet`

Use quiet mode.

`-v, --verbose`

Use verbose mode.

2.2 General Options

`-V, --version`

Display version information.

`-?, -H, --help`

Display usage information.

3 Exit Codes

0

The command completed successfully.

2

The command was run in non-interactive mode, but could not complete because confirmation was required to run a long or critical task.

See the error message or the log for details.

Other

An error occurred.

See the *Installation Guide* for an example upgrade process for OpenDJ directory server installed from the cross-platform (.zip) delivery, *Upgrading From OpenDJ 2.4.5*.

Native packages (.deb, .rpm) perform more of the upgrade process, stopping OpenDJ if it is running, overwriting older files with newer files, running this utility, and starting OpenDJ if it was running when you upgraded the package(s).

verify-index

verify-index — check index for consistency or errors

verify-index

`verify-index` {options}

1 Description

This utility can be used to ensure that index data is consistent within a backend based on the Berkeley DB Java Edition.

2 Options

The following options are supported.

`-b, --baseDN {baseDN}`

Base DN of a backend supporting indexing. Verification is performed on indexes within the scope of the given base DN.

`-c, --clean`

Specifies that a single index should be verified to ensure it is clean. An index is clean if each index value references only entries containing that value. Only one index at a time may be verified in this way.

`--countErrors`

Count the number of errors found during the verification and return that value as the exit code (values > 255 will be reduced to 255 due to exit code restrictions).

`-i, --index {index}`

Name of an index to be verified. For an attribute index this is simply an attribute name. Multiple indexes may be verified for completeness, or all indexes if no indexes are specified. An index is complete if each index value references all entries containing that value.

2.1 General Options

`-V, --version`

Display version information

`-?, -H, --help`

Display usage information

3 Exit Codes

0

The command completed successfully.

1

An error occurred while parsing the command-line arguments.

0-255

The number of errors in the index, as indicated for the `--countErrors` option.

4 Examples

The following example verifies the `cn` (common name) index for completeness and for errors.

```
$ verify-index -b dc=example,dc=com -i cn --clean --countErrors
[07/Jun/2011:16:06:50 +0200] category=BACKEND severity=INFORMATION
 msgID=9437595 msg=Local DB backend userRoot does not specify the number of
 lock tables: defaulting to 97
[07/Jun/2011:16:06:50 +0200] category=BACKEND severity=INFORMATION
 msgID=9437594 msg=Local DB backend userRoot does not specify the number of
 cleaner threads: defaulting to 24 threads
[07/Jun/2011:16:06:51 +0200] category=JEB severity=NOTICE msgID=8847461
 msg=Checked 1316 records and found 0 error(s) in 0 seconds
 (average rate 2506.7/sec)
[07/Jun/2011:16:06:51 +0200] category=JEB severity=INFORMATION
 msgID=8388710 msg=Number of records referencing more than one entry: 315
[07/Jun/2011:16:06:51 +0200] category=JEB severity=INFORMATION
 msgID=8388711 msg=Number of records that exceed the entry limit: 0
[07/Jun/2011:16:06:51 +0200] category=JEB severity=INFORMATION
 msgID=8388712 msg=Average number of entries referenced is 1.58/record
[07/Jun/2011:16:06:51 +0200] category=JEB severity=INFORMATION
 msgID=8388713 msg=Maximum number of entries referenced by any
 record is 32
```

OpenDJ Glossary

Abandon operation

LDAP operation to stop processing of a request in progress, after which the directory server drops the connection without a reply to the client application.

Access control

Control to grant or to deny access to a resource.

Access control instruction (ACI)

Instruction added as a directory entry attribute for fine-grained control over what a given user or group member is authorized to do in terms of LDAP operations and access to user data.

ACIs are implemented independently from privileges, which apply to administrative operations.
See Also Privilege.

Access control list (ACL)

An access control list connects a user or group of users to one or more security entitlements. For example, users in group "sales" are granted the entitlement "read-only" to some financial data.

`access` log

Directory server log tracing the operations the server processes including timestamps, connection information, and information about the operation itself.

Account lockout

The act of making an account temporarily or permanently inactive after successive authentication failures.

Active user

A user that has the ability to authenticate and use the services, having valid credentials.

Add operation

LDAP operation to add a new entry or entries to the directory.

Anonymous

A user that does not need to authenticate, and is unknown to the system.

Anonymous bind

A bind operation using simple authentication with an empty DN and an empty password, allowing "anonymous" access such as reading public information.

Approximate index

Index is used to match values that "sound like" those provided in the filter.

Attribute

Properties of a directory entry, stored as one or more key-value pairs. Typical examples include the common name (`cn`) to store the user's full name and variations of the name, user ID (`uid`) to store a unique identifier for the entry, and `mail` to store email addresses.

`audit` log

Type of access log that dumps changes in LDIF.

Authentication

The process of verifying who is requesting access to a resource; the act of confirming the identity of a principal.

Authorization

The process of determining whether access should be granted to an individual based on information about that individual; the act of determining whether to grant or to deny a principal access to a resource.

Backend

Repository that a directory server can access to store data. Different implementations with different capabilities exist.

Binary copy

Binary backup archive of one directory server that can be restored on another directory server.

Bind operation

LDAP authentication operation to determine the client's identity in LDAP terms, the identity which is later used by the server to authorize (or not) access to directory data that the client wants to lookup or change.

Collective attribute

A standard mechanism for defining attributes that appear on all the entries in a particular subtree.

Compare operation

LDAP operation to compare a specified attribute value with the value stored on an entry in the directory.

Control

Information added to an LDAP message to further specify how an LDAP operation should be processed. OpenDJ supports many LDAP controls.

Database cache

Memory space set aside to hold database content.

`debug` log

Directory server log tracing details needed to troubleshoot a problem in the server.

Delete operation

LDAP operation to remove an existing entry or entries from the directory.

Directory

A directory is a network service which lists participants in the network such as users, computers, printers, and groups. The directory provides a convenient, centralized, and robust mechanism for publishing and consuming information about network participants.

Directory hierarchy

A directory can be organized into a hierarchy in order to make it easier to browse or manage. Directory hierarchies normally represent something in the physical world, such as organizational hierarchies or physical locations. For example, the top level of a directory may represent a company, the next level down divisions, the next level down departments, and so on. Alternately, the top level may represent the world, the next level down countries, next states or provinces, next cities, and so on.

Directory manager

Default Root DN who has privileges to do full administration of the OpenDJ server, including bypassing access control evaluation, changing access controls, and changing administrative privileges.
See Also Root DN.

Directory object

A directory object is an item in a directory. Example objects include users, user groups, computers and more. Objects may be organized into a hierarchy and contain identifying attributes.
See Also Entry.

Directory server

Server application for centralizing information about network participants. A highly available directory service consists of multiple directory servers configured to replicate directory data.
See Also Directory, Replication.

Directory Services Markup Language (DSML)

Standard language to access directory services using XML. DMSL v1 defined an XML mapping of LDAP objects, while DSMLv2 maps the LDAP Protocol and data model to XML.

Distinguished name (DN)

Fully qualified name for a directory entry, such as `uid=bjensen,ou=People,dc=example,dc=com`, built by concatenating the entry RDN (`uid=bjensen`) with the DN of the parent entry (`ou=People,dc=example,dc=com`).

Dynamic group
Group that specifies members using LDAP URLs.

Entry
As generic and hierarchical data stores, directories always contain different kinds of entries, either nodes (or containers) or leaf entries. An entry is an object in the directory, defined by one of more object classes and their related attributes. At startup, OpenDJ reports the number of entries contained in each suffix.

Entry cache
Memory space set aside to hold frequently-accessed, large entries, such as static groups.

Equality index
Index used to match values that correspond exactly (though generally without case sensitivity) to the value provided in the search filter.

`errors` log
Directory server log tracing server events, error conditions, and warnings, categorized and identified by severity.

Export
Save directory data in an LDIF file.

Extended operation
Additional LDAP operation not included in the original standards. OpenDJ supports several standard LDAP extended operations.

Extensible match index
Index for a matching rule other than approximate, equality, ordering, presence, substring or VLV, such as an index for generalized time.

External user
An individual that accesses company resources or services but is not working for the company. Typically a customer or partner.

Filter
An LDAP search filter is an expression that the server uses to find entries that match a search request, such as `(mail=*@example.com)` to match all entries having an email address in the example.com domain.

Group
Entry identifying a set of members whose entries are also in the directory.

Idle time limit
Defines how long OpenDJ allows idle connections to remain open.

Import
Read in and index directory data from an LDIF file.

Inactive user

An entry in the directory that once represented a user but which is now no longer able to be authenticated.

Index

Directory server backend feature to allow quick lookup of entries based on their attribute values.
See Also Approximate index, Equality index, Extensible match index, Ordering index, Presence index, Substring index, Virtual list view (VLV) index, Index entry limit.

Index entry limit

When the number of entries that an index key points to exceeds the index entry limit, OpenDJ stops maintaining the list of entries for that index key.

Internal user

An individual who works within the company either as an employee or as a contractor.

LDAP Data Interchange Format (LDIF)

Standard, portable, text-based representation of directory content. See RFC 2849.

LDAP URL

LDAP Uniform Resource Locator such as `ldap://directory.example.com:389/dc=example,dc=com??sub?(uid=bjensen)`. See RFC 2255.

LDAPS

LDAP over SSL.

Lightweight Directory Access Protocol (LDAP)

A simple and standardized network protocol used by applications to connect to a directory, search for objects and add, edit or remove objects. See RFC 4510.

Lookthrough limit

Defines the maximum number of candidate entries OpenDJ considers when processing a search.

Matching rule

Defines rules for performing matching operations against assertion values. Matching rules are frequently associated with an attribute syntax and are used to compare values according to that syntax. For example, the `distinguishedNameEqualityMatch` matching rule can be used to determine whether two DNs are equal and can ignore unnecessary spaces around commas and equal signs, differences in capitalization in attribute names, and so on.

Modify DN operation

LDAP modification operation to request that the server change the distinguished name of an entry.

Modify operation

LDAP modification operation to request that the server change one or more attributes of an entry.

Naming context

Base DN under which client applications can look for user data.

Object class

Identifies entries that share certain characteristics. Most commonly, an entry's object classes define the attributes that must and may be present on the entry. Object classes are stored on entries as values of the `objectClass` attribute. Object classes are defined in the directory schema, and can be abstract (defining characteristics for other object classes to inherit), structural (defining the basic structure of an entry, one structural inheritance per entry), or auxiliary (for decorating entries already having a structural object class with other required and optional attributes).

Object identifier (OID)

String that uniquely identifies an object, such as `0.9.2342.19200300.100.1.1` for the user ID attribute or `1.3.6.1.4.1.1466.115.121.1.15` for `DirectoryString` syntax.

Operational attribute

An attribute that has a special (operational) meaning for the directory server, such as `pwdPolicySubentry` or `modifyTimestamp`.

Ordering index

Index used to match values for a filter that specifies a range.

Password policy

A set of rules regarding what sequence of characters constitutes an acceptable password. Acceptable passwords are generally those that would be too difficult for another user or an automated program to guess and thereby defeat the password mechanism. Password policies may require a minimum length, a mixture of different types of characters (lowercase, uppercase, digits, punctuation marks, and so forth), avoiding dictionary words or passwords based on the user's name, and so forth. Password policies may also require that users not reuse old passwords and that users change their passwords regularly.

Password reset

Password change performed by a user other than the user who owns the entry.

Password storage scheme

Mechanism for encoding user passwords stored on directory entries. OpenDJ implements a number of password storage schemes.

Password validator

Mechanism for determining whether a proposed password is acceptable for use. OpenDJ implements a number of password validators.

Presence index

Index used to match the fact that an attribute is present on the entry, regardless of the value.

Principal

Entity that can be authenticated, such as a user, a device, or an application.

Privilege

Server configuration settings controlling access to administrative operations such as exporting and importing data, restarting the server, performing password reset, and changing the server configuration.

Privileges are implemented independently from access control instructions (ACI), which apply to LDAP operations and user data.
See Also Access control instruction (ACI).

Referential integrity

Ensuring that group membership remains consistent following changes to member entries.

`referint` log

Directory server log tracing referential integrity events, with entries similar to the errors log.

Referral

Reference to another directory location, which can be another directory server running elsewhere or another container on the same server, where the current operation can be processed.

Relative distinguished name (RDN)

Initial portion of a DN that distinguishes the entry from all other entries at the same level, such as `uid=bjensen` in `uid=bjensen,ou=People,dc=example,dc=com`.

Replication

Data synchronization that ensures all directory servers participating eventually share a consistent set of directory data.

`replication` log

Directory server log tracing replication events, with entries similar to the errors log.

Root DN

A directory superuser, whose account is specific to a directory server under `cn=Root DNs,cn=config`.

The default Root DN is Directory Manager. You can create additional Root DN accounts, each with different administrative privileges.
See Also Directory manager, Privilege.

Root DSE

The directory entry with distinguished name "" (empty string), where DSE stands for DSA-Specific Entry. DSA stands for Directory Server Agent, a single directory server. The root DSE serves to expose information over LDAP about what the directory server supports in terms of LDAP controls, auth password schemes, SASL mechanisms, LDAP protocol versions, naming contexts, features, LDAP extended operations, and so forth.

Schema

LDAP schema defines the object classes, attributes types, attribute value syntaxes, matching rules and so on that constrain entries held by the directory server.

Search filter

See Filter.

Search operation

LDAP lookup operation where a client requests that the server return entries based on an LDAP filter and a base DN under which to search.

Simple authentication

Bind operation performed with a user's entry DN and user's password. Use simple authentication only if the network connection is secure.

Size limit

Sets the maximum number of entries returned for a search.

Static group

Group that enumerates member entries.

Subentry

An entry, such as a password policy entry, that resides with the user data but holds operational data, and is not visible in search results unless explicitly requested.

Substring index

Index used to match values specified with wildcards in the filter.

Task

Mechanism to provide remote access to directory server administrative functions. OpenDJ supports tasks to backup and restore backends, to import and export LDIF files, and to stop and restart the server.

Time limit

Defines the maximum processing time OpenDJ devotes to a search operation.

Unbind operation

LDAP operation to release resources at the end of a session.

Unindexed search

Search operation for which no matching index is available. If no indexes are applicable, then the directory server potentially has to go through all entries to look for candidate matches. For this reason, the `unindexed-search` privilege, allowing users to request searches for which no applicable index exists, is reserved for the directory manager by default.

User

An entry that represents an individual that can be authenticated through credentials contained or referenced by its attributes. A user may represent an internal user or an external user, and may be an active user or an inactive user.

User attribute

An attribute for storing user data on a directory entry such as `mail` or `givenname`.

Virtual attribute

An attribute with dynamically generated values that appear in entries but are not persistently stored in the backend.

Virtual directory

An application that exposes a consolidated view of multiple physical directories over an LDAP interface. Consumers of the directory information connect to the virtual directory's LDAP service. Behind the scenes, requests for information and updates to the directory are sent to one or more physical directories where the actual information resides. Virtual directories enable organizations to create a consolidated view of information that for legal or technical reasons cannot be consolidated into a single physical copy.

Virtual list view (VLV) index

Browsing index designed to help the directory server respond to client applications that need for example to browse through a long list of results a page at a time in a GUI.

Virtual static group

OpenDJ group that lets applications see dynamic groups as what appear to be static groups.

X.500

A family of standardized protocols for accessing, browsing and maintaining a directory. X.500 is functionally similar to LDAP, but is generally considered to be more complex, and has consequently not been widely adopted.

Appendix A. REST LDAP Configuration

OpenDJ offers two alternatives for RESTful access to directory data.

- OpenDJ directory server has an HTTP connection handler that exposes the RESTful API over HTTP (or HTTPS). You configure the mapping between JSON resources and LDAP entries by editing the configuration file for the HTTP connection handler, by default `/path/to/opendj/config/http-config.json`.
- The OpenDJ REST LDAP gateway runs as a Servlet independent from your directory service. You configure the gateway to access your directory service by editing `opendj-rest2ldap-servlet.json` where you deploy the gateway web application.

The JSON format configuration can hold the following configuration objects. Some of the configuration settings are available only in the REST LDAP gateway configuration. The order here is the order shown in the default configuration file.

Interface stability: Evolving

"ldapConnectionFactories" (required, gateway only)

> Configures how the gateway connects to LDAP servers. This entire configuration object applies only to the REST LDAP gateway.
>
> Configures at least a connection factory for unauthenticated connections that are used for bind requests. By default, also configures a factory for authenticated connections that are used for searches during authentication and for proxied authorization operations.

The default configuration is set to connect to a local directory server listening for LDAP connections on port 1389, authenticating as the root DN user `cn=Directory Manager`, with the password `password`.

"default"

Configures the unauthenticated connection factory for bind operations.

"connectionPoolSize" (optional)

The gateway creates connection pools to the primary and secondary LDAP servers that maintain up to `connectionPoolSize` connections to the servers.

Default: 24

```
"connectionPoolSize": 24
```

"connectionSecurity" (optional)

Whether connections to LDAP servers should be secured by using SSL or StartTLS. The following values are supported.

- "none" (default) means connections use plain LDAP and are not secured.
- "ssl" means connections are secured using LDAPS.
- "startTLS" means connections are secured using LDAP and StartTLS.

If you set "connectionSecurity", also review the "trustManager" and "fileBasedTrustManager*" settings.

"heartBeatIntervalSeconds" (optional)

The gateway tests its connections every `heartBeatIntervalSeconds` seconds to detect whether the connection is still alive.

Default: 30 (seconds)

```
"heartBeatIntervalSeconds": 30
```

"fileBasedTrustManagerFile" (optional)

If "trustManager" is set to "file", then this setting configures the location of the trust store file.

Default: "/path/to/truststore"

"fileBasedTrustManagerPassword" (optional)

If "trustManager" is set to "file", then this setting specifies the trust store password.

Default: "password"

"fileBasedTrustManagerType" (optional)

If "trustManager" is set to "file", then this setting configures the format for the data in the trust store file specified by the "fileBasedTrustManagerFile" setting. Formats include the following, though other implementations might be supported as well depending on the Java environment.

- "JKS" (default) specifies Java Key Store format.

- "PKCS12" specifies Public-Key Cryptography Standards 12 format.

"primaryLDAPServers" (required)

The gateway accesses this array of LDAP servers before failing over to the secondary LDAP servers. These might be LDAP servers in the same data center for example.

```
{
    "primaryLDAPServers": [
        {
            "hostname": "local1.example.com",
            "port": 1389
        },
        {
            "hostname": "local2.example.com",
            "port": 1389
        }
    ]
}
```

By default, the gateway connects to the directory server listening on port 1389 on the local host.

"secondaryLDAPServers" (optional)

The gateway accesses this array of LDAP servers if primary LDAP servers cannot be contacted. These might be LDAP servers in the same data center for example.

```
{
    "secondaryLDAPServers": [
        {
            "hostname": "remote1.example.com",
            "port": 1389
        },
        {
            "hostname": "remote2.example.com",
            "port": 1389
        }
    ]
}
```

No secondary LDAP servers are configured by default.

"trustManager" (optional)

If "connectionSecurity" is set to "ssl" or "startTLS", then this setting configures how the LDAP servers are trusted. This setting is ignored if "connectionSecurity" is set to "none".

- "file" means trust the LDAP server certificate if it is signed by a Certificate Authority (CA) trusted according to the file-based trust store configured with the "fileBasedTrustManager*" settings.
- "jvm" means trust the LDAP server certificate if it is signed by a CA trusted by the Java environment.
- "trustAll" (default) means blindly trust all LDAP server certificates.

"root"

Configures the authenticated connection factory.

"inheritFrom" (optional)

Identifies the unauthenticated connection factory from which to inherit settings. If this connection factory does not inherit from another configuration object, then you must specify the configuration here.

Default: "default"

"authentication" (required)

The gateway authenticates by simple bind using the credentials specified.

```
{
    "authentication": {
        "bindDN": "cn=Directory Manager",
        "password": "password"
    }
}
```

"authenticationFilter" (required)

Configures the REST LDAP authentication filter. If the configuration is not present, the filter is disabled.

The default configuration allows HTTP Basic authentication where user entries are `inetOrgPerson` entries expected to have `uid=`*`username`*, and to be found under `ou=people,dc=example,dc=com`. The default configuration also allows alternative, HTTP header based authentication in the style of OpenIDM.

By default, authentication is required both for the gateway and for the HTTP connection handler. When the HTTP connection handler property `authentication-required` is set to `false` (default: `true`), the HTTP connection handler accepts both authenticated and unauthenticated requests. All requests are subject to access control and resource limit settings in the same way as LDAP client requests to the directory server. The `authentication-required` setting can be overridden by the global configuration property `reject-unauthenticated-requests` (default: `false`), described in the section on *Restricting Client Access*.

To protect passwords, configure HTTPS for the HTTP connection handler or for the container where the REST LDAP gateway runs.

The filter has the following configuration fields.

"supportHTTPBasicAuthentication"

Whether to support HTTP Basic authentication. If this is set to `true`, then the entry corresponding to the user name is found using the "searchBaseDN", "searchScope", and "searchFilterTemplate" settings.

Default: `true`

"supportAltAuthentication"

Whether to allow alternative, HTTP header based authentication. If this is set to `true`, then the headers to use are specified in the "altAuthenticationUsernameHeader" and "altAuthenticationPasswordHeader" values, and the bind DN is resolved using the "searchFilterTemplate" value.

Default: `true`

"altAuthenticationUsernameHeader"

Specifies the HTTP header containing the username for authentication when alternative, HTTP-header based authentication is allowed.

Default: "X-OpenIDM-Username"

"altAuthenticationPasswordHeader"

Specifies the HTTP header containing the password for authentication when alternative, HTTP-header based authentication is allowed.

Default: "X-OpenIDM-Password"

"reuseAuthenticatedConnection" (gateway only)

Whether to use authenticated LDAP connections for subsequent LDAP operations. If this is set to `true`, the gateway does not need its own connection factory, nor does it need to use proxied authorization for LDAP operations. Instead, it performs the operations as the user on the authenticated connection.

Default: `true`

"method" (gateway only)

Specifies the authentication method used by the gateway. The following values are supported.

- "search-simple" (default) means the user name is resolved to an LDAP bind DN by a search using the "searchFilterTemplate" value.

- "sasl-plain" means the user name is resolved to an authorization ID (authzid) using the "saslAuthzIdTemplate" value.

- "simple" means the user name is the LDAP bind DN.

"bindLDAPConnectionFactory" (gateway only)

Identifies the factory providing connections used for bind operations to authenticate users to LDAP servers.

Default: "default"

"saslAuthzIdTemplate" (gateway only)

Sets how to resolve the authorization ID when the authentication "method" is set to "sasl-plain", substituting `%s` in the template with the user name provided. The user name provided by is DN escaped before the value is returned.

Default: "dn:uid=%s,ou=people,dc=example,dc=com" "searchLDAPConnectionFactory" (gateway only)

Identifies the factory providing connections used to find user entries in the directory server when the "method" is set to "search-simple".

Default: "root"

"searchBaseDN"

Sets the base DN to search for user entries. For the gateway, this applies when the "method" is set to "search-simple". This always applies for the HTTP connection handler.

Default: "ou=people,dc=example,dc=com"

"searchScope"

Sets the search scope below the base DN such as "sub" (subtree search) or "one" (one-level search) to search for user entries. For the gateway, this applies when the "method" is set to "search-simple". This always applies for the HTTP connection handler.

Default: "sub"

"searchFilterTemplate"

Sets the search filter used to find the user entry, substituting `%s` in the template with the user name provided. The user name provided by is DN escaped before the value is returned. For the gateway, this applies when the "method" is set to "search-simple". This always applies for the HTTP connection handler.

Default: "(&(objectClass=inetOrgPerson)(uid=%s))"

"servlet" (required)

Configures how HTTP resources map to LDAP entries, and for the gateway how to connect to LDAP servers and how to use proxied authorization.

The default gateway configuration tries to reuse authenticated connections for LDAP operations, falling back to a connection authenticated as root DN using proxied authorization for LDAP operations.

"ldapConnectionFactory" (gateway only)

Specifies the connection factory used by the gateway to perform LDAP operations if an authenticated connection is not passed from the authentication filter according to the setting for "reuseAuthenticatedConnection".

Default: "root"

"authorizationPolicy" (gateway only)

Specifies how to handle LDAP authorization. The following values are supported.

- "proxy" (default) means use proxied authorization when no authenticated connection is provided for reuse, resolving the authorization ID according to the setting for "proxyAuthzIdTemplate".
- "none" means do not use proxied authorization and do not reuse authenticated connections, but instead use connections from the factory specified in "ldapConnectionFactory".
- "reuse" means reuse an authenticated connection passed by the filter, and fail if no connection was passed by the filter.

"proxyAuthzIdTemplate" (gateway only)

Specifies the template to derive the authorization ID from the security context created during authentication. Use `{dn}` to indicate the user's bind DN or `{id}` to indicate the user name provided for authentication.

Default: "dn:{dn}"

"mappings"

For each collection URI such as `/users` and `/groups`, you configure a mapping between the JSON resource returned over HTTP, and the LDAP entry returned by the directory service.

Each mapping has a number of configuration elements.

"baseDN" (required)

The base DN where LDAP entries are found for this mapping.

"readOnUpdatePolicy" (optional)

The policy used to read an entry before it is deleted, or to read an entry after it is added or modified. One of the following.

- "controls": (default) use RFC 4527 read-entry controls to reflect the state of the resource at the time the update was performed.

  The directory service must support RFC 4527.

- "disabled": do not read the entry or return the resource on update.

- "search": perform an LDAP search to retrieve the entry before deletion or after it is added or modified.

  The JSON resource returned might differ from the LDAP entry that was updated.

"useSubtreeDelete" (required)

Whether to use the LDAP Subtree Delete Request Control (OID: `1.2.840.113556.1.4.805`) for LDAP delete operations resulting from delete operations on resources.

Default: `false`. The default configuration uses `false`.

Set this to `true` if you want this behavior, if your directory server supports the control, and if clients that request delete operations have access to use the control.

"usePermissiveModify" (required)

Whether to use the LDAP Permissive Modify Request Control (OID: `1.2.840.113556.1.4.1413`) for LDAP modify operations resulting from patch and update operations on resources.

Default: `false`. The default configuration uses `true`.

Set this to `false` when using the gateway if your directory server does not support the control.

"etagAttribute" (optional)

The LDAP attribute to use for multi-version concurrency control (MVCC).

Default: "etag"

"namingStrategy" (required)

The approach used to map LDAP entry names to JSON resources. The following naming strategies are supported.

- RDN and resource ID are both derived from a single user attribute in the LDAP entry, as in the following example, where the `uid` attribute is the RDN and its value is the JSON resource ID.

```
{
    "namingStrategy": {
        "strategy": "clientDNNaming",
        "dnAttribute": "uid"
    }
}
```

- RDN and resource ID are derived from separate user attributes in the LDAP entry, as in the following example where the RDN attribute is `uid` but the JSON resource ID is the value of the `mail` attribute.

```
{
    "namingStrategy": {
        "strategy": "clientNaming",
        "dnAttribute": "uid",
        "idAttribute": "mail"
    }
}
```

- RDN is derived from a user attribute and the resource ID from an operational attribute in the LDAP entry, as in the following example, where the RDN attribute is `uid` but the JSON resource ID is the value of the `entryUUID` operational attribute.

```
{
    "namingStrategy": {
        "strategy": "serverNaming",
        "dnAttribute": "uid",
        "idAttribute": "entryUUID"
    }
}
```

"additionalLDAPAttributes" (optional, but necessary)

LDAP attributes to include during LDAP add operations as an array of type-value lists, such as the following example.

```
{
    "additionalLDAPAttributes": [
        {
            "type": "objectClass",
            "values": [
                "top",
                "person",
                "organizationalPerson",
                "inetOrgPerson"
            ]
        }
    ]
}
```

This configuration element is useful to set LDAP object classes for example, which are not present in JSON resources.

"attributes" (required)

How the JSON resource fields map to attributes on LDAP entries, each taking the form "*field-name*": *mapping-object*. A number of *mapping-object*s are supported.

"constant"

Maps a single JSON attribute to a fixed value.

This can be useful as in the default case where each JSON resource "schemas" takes the SCIM URN, and so the value is not related to the underlying LDAP entries.

```
{
    "schemas": {
        "constant": [
            "urn:scim:schemas:core:1.0"
        ]
    }
}
```

"simple"

Maps a JSON field to an LDAP attribute.

Simple mappings are used where the correspondence between JSON fields and LDAP attributes is one-to-one.

```
{
    "userName": {
        "simple": {
            "ldapAttribute": "mail",
            "isSingleValued": true,
            "writability": "readOnly"
        }
    }
}
```

Simple mappings can take a number of fields.

- (Required) "ldapAttribute": the name of LDAP attribute.
- (Optional) "defaultJSONValue": the JSON value if no LDAP attribute is available on the entry.

  No default is set if this is omitted.
- (Optional) "isBinary": true means the LDAP attribute is binary and the JSON field gets the base64-encoded value.

  Default: `false`
- (Optional) "isRequired": true means the LDAP attribute is mandatory and must be provided to create the resource; false means it is optional.

  Default: `false`

- (Optional) "isSingleValued": true means represent a possibly multi-valued LDAP attribute as a single value; false means represent it as an array of values.

  Default: determine the representation based on the LDAP schema, so SINGLE-VALUE attributes take single values, and multi-valued attributes take arrays.

- (Optional) "writability": indicates whether the LDAP attribute supports updates. This field can take the following values.

  - "createOnly": This attribute can be set only when the entry is created. Attempts to update this attribute thereafter result in errors.

  - "createOnlyDiscardWrites": This attribute can be set only when the entry is created. Attempts to update this attribute thereafter do not result in errors. Instead the update value is discarded.

  - "readOnly": This attribute cannot be written. Attempts to write this attribute result in errors.

  - "readOnlyDiscardWrites": This attribute cannot be written. Attempts to write this attribute do not result in errors. Instead the value to write is discarded.

  - "readWrite": (default) This attribute can be set at creation and updated thereafter.

"object"

Maps a JSON object to LDAP attributes.

This mapping lets you create JSON objects whose fields themselves have mappings to LDAP attributes.

"reference"

Maps a JSON field to an LDAP entry found by reference.

This mapping works for LDAP attributes whose values reference other entries. This is shown in the following example from the default configuration. The LDAP `manager` attribute values are user entry DNs. Here, the JSON `manager` field takes the user ID and name from the entry referenced by the LDAP attribute. On updates, changes to the JSON manager `_id` affect which manager entry is referenced, yet any changes to the manager's name are discarded, because changing managers only affects which user entry to point to, not the referenced user's name.

```
{
    "manager": {
        "reference": {
            "ldapAttribute": "manager",
            "baseDN": "ou=people,dc=example,dc=com",
            "primaryKey": "uid",
            "mapper": {
                "object": {
                    "_id": {
                        "simple": {
                            "ldapAttribute": "uid",
                            "isSingleValued": true,
                            "isRequired": true
                        }
                    },
                    "displayName": {
                        "simple": {
                            "ldapAttribute": "cn",
                            "isSingleValued": true,
                            "writability": "readOnlyDiscardWrites"
                        }
                    }
                }
            }
        }
    }
}
```

Babs Jensen's manager in the sample LDAP data is Torrey Rigden, who has user ID `trigden`. Babs's entry has `manager: uid=trigden,ou=People,dc=example,dc=com`. With this mapping, the resulting JSON field is the following.

```
{
    "manager": [
        {
            "_id": "trigden",
            "displayName": "Torrey Rigden"
        }
    ]
}
```

Reference mapping objects have the following fields.

- (Required) "baseDN": indicates the base LDAP DN under which to find entries referenced by the JSON resource.
- (Required) "ldapAttribute": specifies the LDAP attribute in the entry underlying the JSON resource whose value points to the referenced entry.
- (Required) "mapper": describes how the referenced entry content maps to the content of this JSON field.
- (Required) "primaryKey": indicates which LDAP attribute in the mapper holds the primary key to the referenced entry.

- (Optional) "isRequired": true means the LDAP attribute is mandatory and must be provided to create the resource; false means it is optional.

  Default: `false`

- (Optional) "isSingleValued": true means represent a possibly multi-valued LDAP attribute as a single value; false means represent it as an array of values.

  Default: `false`

- (Optional) "searchFilter": specifies the LDAP filter to use to search for the referenced entry. The default is `"(objectClass=*)"`.

- (Optional) "writability": indicates whether the mapping supports updates, as described above for the simple mapping. The default is "readWrite".

The default mappings expose a SCIM view of user and group data.

```
{
    "/users": {
        "baseDN": "ou=people,dc=example,dc=com",
        "readOnUpdatePolicy": "controls",
        "useSubtreeDelete": false,
        "usePermissiveModify": true,
        "etagAttribute": "etag",
        "namingStrategy": {
            "strategy": "clientDNNaming",
            "dnAttribute": "uid"
        },
        "additionalLDAPAttributes": [
            {
                "type": "objectClass",
                "values": [
                    "top",
                    "person",
                    "organizationalPerson",
                    "inetOrgPerson"
                ]
            }
        ],
        "attributes": {
            "schemas": {
                "constant": [
                    "urn:scim:schemas:core:1.0"
                ]
            },
            "_id": {
                "simple": {
                    "ldapAttribute": "uid",
                    "isSingleValued": true,
                    "isRequired": true,
                    "writability": "createOnly"
                }
            },
            "_rev": {
                "simple": {
                    "ldapAttribute": "etag",
                    "isSingleValued": true,
                    "writability": "readOnly"
                }
            },
            "userName": {
                "simple": {
                    "ldapAttribute": "mail",
                    "isSingleValued": true,
                    "writability": "readOnly"
                }
            },
            "displayName": {
                "simple": {
                    "ldapAttribute": "cn",
                    "isSingleValued": true,
                    "isRequired": true
                }
            },
            "name": {
                "object": {
                    "givenName": {
                        "simple": {
                            "ldapAttribute": "givenName",
                            "isSingleValued": true
                        }
                    },
                    "familyName": {
                        "simple": {
                            "ldapAttribute": "sn",
                            "isSingleValued": true,
                            "isRequired": true
                        }
                    }
                }
            },
            "manager": {
                "reference": {
                    "ldapAttribute": "manager",
                    "baseDN": "ou=people,dc=example,dc=com",
                    "primaryKey": "uid",
                    "mapper": {
                        "object": {
                            "_id": {
                                "simple": {
                                    "ldapAttribute": "uid",
                                    "isSingleValued": true,
                                    "isRequired": true
                                }
                            },
                            "displayName": {
                                "simple": {
                                    "ldapAttribute": "cn",
                                    "isSingleValued": true,
                                    "writability": "readOnlyDiscardWrites"
                                }
                            }
                        }
                    }
                }
            },
            "groups": {
                "reference": {
                    "ldapAttribute": "isMemberOf",
                    "baseDN": "ou=groups,dc=example,dc=com",
                    "writability": "readOnly",
                    "primaryKey": "cn",
                    "mapper": {
                        "object": {
                            "_id": {
                                "simple": {
                                    "ldapAttribute": "cn",
                                    "isSingleValued": true
                                }
                            }
                        }
                    }
                }
            },
            "contactInformation": {
                "object": {
                    "telephoneNumber": {
                        "simple": {
                            "ldapAttribute": "telephoneNumber",
                            "isSingleValued": true
                        }
                    },
                    "emailAddress": {
                        "simple": {
                            "ldapAttribute": "mail",
                            "isSingleValued": true
                        }
                    }
                }
            },
            "meta": {
                "object": {
                    "created": {
                        "simple": {
                            "ldapAttribute": "createTimestamp",
                            "isSingleValued": true,
                            "writability": "readOnly"
                        }
                    },
                    "lastModified": {
                        "simple": {
                            "ldapAttribute": "modifyTimestamp",
                            "isSingleValued": true,
                            "writability": "readOnly"
                        }
                    }
                }
            }
        }
    },
    "/groups": {
        "baseDN": "ou=groups,dc=example,dc=com",
        "readOnUpdatePolicy": "controls",
        "useSubtreeDelete": false,
        "usePermissiveModify": true,
        "etagAttribute": "etag",
        "namingStrategy": {
            "strategy": "clientDNNaming",
            "dnAttribute": "cn"
        },
        "additionalLDAPAttributes": [
            {
                "type": "objectClass",
                "values": [
                    "top",
                    "groupOfUniqueNames"
                ]
            }
        ],
        "attributes": {
            "schemas": {
                "constant": [
                    "urn:scim:schemas:core:1.0"
                ]
            },
            "_id": {
                "simple": {
                    "ldapAttribute": "cn",
                    "isSingleValued": true,
                    "isRequired": true,
                    "writability": "createOnly"
                }
            },
            "_rev": {
                "simple": {
                    "ldapAttribute": "etag",
                    "isSingleValued": true,
                    "writability": "readOnly"
                }
            },
            "displayName": {
                "simple": {
                    "ldapAttribute": "cn",
                    "isSingleValued": true,
                    "isRequired": true,
                    "writability": "readOnly"
                }
            },
            "members": {
                "reference": {
                    "ldapAttribute": "uniqueMember",
                    "baseDN": "dc=example,dc=com",
                    "primaryKey": "uid",
                    "mapper": {
                        "object": {
                            "_id": {
                                "simple": {
                                    "ldapAttribute": "uid",
                                    "isSingleValued": true,
                                    "isRequired": true
                                }
                            },
                            "displayName": {
                                "simple": {
                                    "ldapAttribute": "cn",
                                    "isSingleValued": true,
                                    "writability": "readOnlyDiscardWrites"
                                }
                            }
                        }
                    }
                }
            },
            "meta": {
                "object": {
                    "created": {
                        "simple": {
                            "ldapAttribute": "createTimestamp",
                            "isSingleValued": true,
                            "writability": "readOnly"
                        }
                    },
                    "lastModified": {
                        "simple": {
                            "ldapAttribute": "modifyTimestamp",
                            "isSingleValued": true,
                            "writability": "readOnly"
                        }
                    }
                }
            }
        }
    }
}
```

Appendix B. File Layout

OpenDJ software installs and creates the following files and directories. The following list is not necessarily exhaustive.

`QuickSetup.app`

Mac OS X GUI for installing OpenDJ

`Uninstall.app`

Mac OS X GUI for removing OpenDJ

`bak`

Directory for saving backup files

`bat`

Windows command-line tools and control panel

`bin`

UNIX/Linux/Mac OS X command-line tools and control panel

`changelogDb`

JE backend data for the external change log when using replication

`classes`

Directory added to the `CLASSPATH` for OpenDJ, permitting individual classes to be patched `config`

OpenDJ server configuration and schema, PKI stores, LDIF generation templates, resources for upgrade

`config/MakeLDIF`

Templates for use with the **make-ldif** LDIF generation tool

`config/config.ldif`

LDIF representation of current OpenDJ server config

Use the **dsconfig** command to edit OpenDJ server configuration.

`config/java.properties`

JVM settings for OpenDJ server and tools

`config/schema`

OpenDJ directory server LDAP schema definition files

`config/tasks.ldif`

Data used by task scheduler backend so that scheduled tasks and recurring tasks persist after server restart

`config/tools.properties`

Default settings for command-line tools

Use as a template when creating an `~/.opendj/tools.properties` file.

`config/upgrade`

Resources used by the upgrade command to move to the next version of OpenDJ

`config/wordlist.txt`

List of words used to check password strength

`db`

JE backend data for backends that you create

`example-plugin.zip`

Sample OpenDJ plugin code. Custom plugins are meant to be installed in `lib/extensions`.

`import-tmp`

Used when importing data into OpenDJ

`instance.loc`

Pointer to OpenDJ on the file system, useful in installations where the program files are separate from the server instance files

`ldif`

Directory for saving LDIF export files

`legal-notices`

License information

`lib`

Scripts and libraries needed by OpenDJ and added to the `CLASSPATH` for OpenDJ

`lib/extensions`

File system directory to hold your custom plugins

`locks`

Directory to hold lock files used when OpenDJ is running to prevent backends from accidentally being used by more than one server process

`logs`

Access, errors, audit, and replication logs

`logs/server.pid`

Contains the process ID for the server when OpenDJ is running

`setup`

UNIX setup utility

`setup.bat`

Windows setup utility

`uninstall`

UNIX utility for removing OpenDJ

`uninstall.bat`

Windows utility for removing OpenDJ

`upgrade`

UNIX utility for upgrading OpenDJ by pointing to the new .zip

`upgrade.bat`

Windows utility for upgrading OpenDJ by pointing to the new .zip

Appendix C. Ports Used

OpenDJ server software uses the following TCP/IP ports by default.

LDAP: 389 (1389)

> OpenDJ directory server listens for LDAP requests from client applications on port 389 by default. OpenDJ directory server uses port 1389 by default for users who cannot use privileged ports. LDAP is enabled by default.

LDAPS: 636 (1636)

> OpenDJ directory server listens for LDAPS requests from client applications on port 636 by default. OpenDJ directory server uses port 1636 by default for users who cannot use privileged ports. LDAPS is not enabled by default.

Administrative connections: 4444

> OpenDJ directory server listens for administrative traffic on port 4444 by default. The administration connector is enabled by default.

SNMP: 161, 162

> OpenDJ directory server listens for SNMP traffic on port 161 by default, and uses port 162 for traps. SNMP is not enabled by default.

JMX: 1689

> OpenDJ directory server listens for Java Management eXtension traffic on port 1689 by default. JMX is not enabled by default.

HTTP: 8080

OpenDJ directory server can listen for HTTP client requests to the RESTful API. The default port is 8080, but HTTP access is not enabled by default.

Replication: 8989

OpenDJ directory server listens for replication traffic on port 8989 by default. Replication is not enabled by default.

Appendix D. Standards, RFCs, & Internet-Drafts

OpenDJ 4.8.1-SNAPSHOT software implements the following RFCs, Internet-Drafts, and standards.

RFC 1274: The COSINE and Internet X.500 Schema

X.500 Directory Schema, or Naming Architecture, for use in the COSINE and Internet X.500 pilots.

RFC 1321: The MD5 Message-Digest Algorithm

MD5 message-digest algorithm that takes as input a message of arbitrary length and produces as output a 128-bit "fingerprint" or "message digest" of the input.

RFC 1777: Lightweight Directory Access Protocol (LDAPv2)

Provide access to the X.500 Directory while not incurring the resource requirements of the Directory Access Protocol.

Classified as an Historic document.

RFC 1778: The String Representation of Standard Attribute Syntaxes

Defines the requirements that must be satisfied by encoding rules used to render X.500 Directory attribute syntaxes into a form suitable for use in the LDAP, then defines the encoding rules for the standard set of attribute syntaxes.

Classified as an Historic document.

RFC 1779: A String Representation of Distinguished Names

Defines a string format for representing names, which is designed to give a clean representation of commonly used names, whilst being able to represent any distinguished name.

Classified as an Historic document.

RFC 2079: Definition of an X.500 Attribute Type and an Object Class to Hold Uniform Resource Identifiers (URIs)

Defines a new attribute type and an auxiliary object class to allow URIs, including URLs, to be stored in directory entries in a standard way.

RFC 2222: Simple Authentication and Security Layer (SASL)

Describes a method for adding authentication support to connection-based protocols.

RFC 2246: The TLS Protocol Version 1.0

Specifies Version 1.0 of the Transport Layer Security protocol.

RFC 2247: Using Domains in LDAP/X.500 Distinguished Names

Defines an algorithm by which a name registered with the Internet Domain Name Service can be represented as an LDAP distinguished name.

RFC 2251: Lightweight Directory Access Protocol (v3)

Describes a directory access protocol designed to provide access to directories supporting the X.500 models, while not incurring the resource requirements of the X.500 Directory Access Protocol.

RFC 2252: Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions

Defines a set of syntaxes for LDAPv3, and the rules by which attribute values of these syntaxes are represented as octet strings for transmission in the LDAP protocol.

RFC 2253: Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names

Defines a common UTF-8 format to represent distinguished names unambiguously.

RFC 2254: The String Representation of LDAP Search Filters

Defines the string format for representing names, which is designed to give a clean representation of commonly used distinguished names, while being able to represent any distinguished name.

RFC 2255: The LDAP URL Format

Describes a format for an LDAP Uniform Resource Locator.

RFC 2256: A Summary of the X.500(96) User Schema for use with LDAPv3

Provides an overview of the attribute types and object classes defined by the ISO and ITU-T committees in the X.500 documents, in particular those intended for use by directory clients.

RFC 2307: An Approach for Using LDAP as a Network Information Service

Describes an experimental mechanism for mapping entities related to TCP/IP and the UNIX system into X.500 entries so that they may be resolved with the Lightweight Directory Access Protocol.

RFC 2377: Naming Plan for Internet Directory-Enabled Applications

Proposes a new directory naming plan that leverages the strengths of the most popular and successful Internet naming schemes for naming objects in a hierarchical directory.

RFC 2696: LDAP Control Extension for Simple Paged Results Manipulation

Allows a client to control the rate at which an LDAP server returns the results of an LDAP search operation.

RFC 2713: Schema for Representing Java(tm) Objects in an LDAP Directory

Defines a common way for applications to store and retrieve Java objects from the directory.

RFC 2714: Schema for Representing CORBA Object References in an LDAP Directory

Define a common way for applications to store and retrieve CORBA object references from the directory.

RFC 2739: Calendar Attributes for vCard and LDAP

Defines a mechanism to locate a user calendar and free/busy time using the LDAP protocol.

RFC 2798: Definition of the inetOrgPerson LDAP Object Class

Define an object class called inetOrgPerson for use in LDAP and X.500 directory services that extends the X.521 standard organizationalPerson class.

RFC 2829: Authentication Methods for LDAP

Specifies particular combinations of security mechanisms which are required and recommended in LDAP implementations.

RFC 2830: Lightweight Directory Access Protocol (v3): Extension for Transport Layer Security

Defines the "Start Transport Layer Security (TLS) Operation" for LDAP.

RFC 2849: The LDAP Data Interchange Format (LDIF) - Technical Specification

Describes a file format suitable for describing directory information or modifications made to directory information.

RFC 2891: LDAP Control Extension for Server Side Sorting of Search Results

Describes two LDAPv3 control extensions for server side sorting of search results.

RFC 2926: Conversion of LDAP Schemas to and from SLP Templates

Describes a procedure for mapping between Service Location Protocol service advertisements and lightweight directory access protocol descriptions of services.

RFC 3045: Storing Vendor Information in the LDAP root DSE

Specifies two Lightweight Directory Access Protocol attributes, vendorName and vendorVersion that MAY be included in the root DSA-specific Entry (DSE) to advertise vendor-specific information.

RFC 3062: LDAP Password Modify Extended Operation

Describes an LDAP extended operation to allow modification of user passwords which is not dependent upon the form of the authentication identity nor the password storage mechanism used.

RFC 3112: LDAP Authentication Password Schema

Describes schema in support of user/password authentication in a LDAP directory including the authPassword attribute type. This attribute type holds values derived from the user's password(s) (commonly using cryptographic strength one-way hash).

RFC 3377: Lightweight Directory Access Protocol (v3): Technical Specification

Specifies the set of RFCs comprising the Lightweight Directory Access Protocol Version 3 (LDAPv3), and addresses the "IESG Note" attached to RFCs 2251 through 2256.

RFC 3383: Internet Assigned Numbers Authority (IANA) Considerations for the Lightweight Directory Access Protocol (LDAP)

Provides procedures for registering extensible elements of the Lightweight Directory Access Protocol (LDAP).

RFC 3546: Transport Layer Security (TLS) Extensions

Describes extensions that may be used to add functionality to Transport Layer Security.

RFC 3671: Collective Attributes in the Lightweight Directory Access Protocol (LDAP)

Summarizes the X.500 information model for collective attributes and describes use of collective attributes in LDAP.

RFC 3672: Subentries in the Lightweight Directory Access Protocol (LDAP)

Adapts X.500 subentries mechanisms for use with the Lightweight Directory Access Protocol (LDAP).

RFC 3673: Lightweight Directory Access Protocol version 3 (LDAPv3): All Operational Attributes

Describes an LDAP extension which clients may use to request the return of all operational attributes.

RFC 3674: Feature Discovery in Lightweight Directory Access Protocol (LDAP)

Introduces a general mechanism for discovery of elective features and extensions which cannot be discovered using existing mechanisms.

RFC 3771: Lightweight Directory Access Protocol (LDAP) Intermediate Response Message

Defines and describes the IntermediateResponse message, a general mechanism for defining single-request/multiple-response operations in Lightweight Directory Access Protocol.

RFC 3829: Lightweight Directory Access Protocol (LDAP) Authorization Identity Request and Response Controls

Extends the Lightweight Directory Access Protocol bind operation with a mechanism for requesting and returning the authorization identity it establishes.

RFC 3876: Returning Matched Values with the Lightweight Directory Access Protocol version 3 (LDAPv3)

Describes a control for the Lightweight Directory Access Protocol version 3 that is used to return a subset of attribute values from an entry.

RFC 3909: Lightweight Directory Access Protocol (LDAP) Cancel Operation

Describes a Lightweight Directory Access Protocol extended operation to cancel (or abandon) an outstanding operation, with a response to indicate the outcome of the operation.

RFC 4346: The Transport Layer Security (TLS) Protocol Version 1.1

Specifies Version 1.1 of the Transport Layer Security protocol.

RFC 4370: Lightweight Directory Access Protocol (LDAP) Proxied Authorization Control

Defines the Proxy Authorization Control, that allows a client to request that an operation be processed under a provided authorization identity instead of under the current authorization identity associated with the connection.

RFC 4403: Lightweight Directory Access Protocol (LDAP) Schema for Universal Description, Discovery, and Integration version 3 (UDDIv3)

Defines the Lightweight Directory Access Protocol schema for representing Universal Description, Discovery, and Integration data types in an LDAP directory.

RFC 4422: Simple Authentication and Security Layer (SASL)

Describes a framework for providing authentication and data security services in connection-oriented protocols via replaceable mechanisms.

RFC 4505: Anonymous Simple Authentication and Security Layer (SASL) Mechanism

Describes a new way to provide anonymous login is needed within the context of the Simple Authentication and Security Layer framework.

RFC 4510: Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map

Provides a road map of the LDAP Technical Specification.

RFC 4511: Lightweight Directory Access Protocol (LDAP): The Protocol

Describes the protocol elements, along with their semantics and encodings, of the Lightweight Directory Access Protocol.

RFC 4512: Lightweight Directory Access Protocol (LDAP): Directory Information Models

Describes the X.500 Directory Information Models as used in LDAP.

RFC 4513: Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms

Describes authentication methods and security mechanisms of the Lightweight Directory Access Protocol.

RFC 4514: Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names

Defines the string representation used in the Lightweight Directory Access Protocol to transfer distinguished names.

RFC 4515: Lightweight Directory Access Protocol (LDAP): String Representation of Search Filters

Defines a human-readable string representation of LDAP search filters that is appropriate for use in LDAP URLs and in other applications.

RFC 4516: Lightweight Directory Access Protocol (LDAP): Uniform Resource Locator

Describes a format for a Lightweight Directory Access Protocol Uniform Resource Locator.

RFC 4517: Lightweight Directory Access Protocol (LDAP): Syntaxes and Matching Rules

Defines a base set of syntaxes and matching rules for use in defining attributes for LDAP directories.

RFC 4518: Lightweight Directory Access Protocol (LDAP): Internationalized String Preparation

Defines string preparation algorithms for character-based matching rules defined for use in LDAP.

RFC 4519: Lightweight Directory Access Protocol (LDAP): Schema for User Applications

Provides a technical specification of attribute types and object classes intended for use by LDAP directory clients for many directory services, such as White Pages.

RFC 4524: COSINE LDAP/X.500 Schema

Provides a collection of schema elements for use with the Lightweight Directory Access Protocol from the COSINE and Internet X.500 pilot projects.

RFC 4525: Lightweight Directory Access Protocol (LDAP) Modify-Increment Extension

Describes an extension to the Lightweight Directory Access Protocol Modify operation to support an increment capability.

RFC 4526: Lightweight Directory Access Protocol (LDAP) Absolute True and False Filters

Extends the Lightweight Directory Access Protocol to support absolute True and False filters based upon similar capabilities found in X.500 directory systems.

RFC 4527: Lightweight Directory Access Protocol (LDAP) Read Entry Controls

Specifies an extension to the Lightweight Directory Access Protocol to allow the client to read the target entry of an update operation.

RFC 4528: Lightweight Directory Access Protocol (LDAP) Assertion Control

Defines the Lightweight Directory Access Protocol Assertion Control, which allows a client to specify that a directory operation should only be processed if an assertion applied to the target entry of the operation is true.

RFC 4529: Requesting Attributes by Object Class in the Lightweight Directory Access Protocol (LDAP)

Extends LDAP to support a mechanism that LDAP clients may use to request the return of all attributes of an object class.

RFC 4530: Lightweight Directory Access Protocol (LDAP) entryUUID Operational Attribute

Describes the LDAP/X.500 'entryUUID' operational attribute and associated matching rules and syntax.

RFC 4532: Lightweight Directory Access Protocol (LDAP) "Who am I?" Operation

Provides a mechanism for Lightweight Directory Access Protocol clients to obtain the authorization identity the server has associated with the user or application entity.

RFC 4616: The PLAIN Simple Authentication and Security Layer (SASL) Mechanism

Defines a simple clear-text user/password Simple Authentication and Security Layer mechanism called the PLAIN mechanism.

RFC 4634: US Secure Hash Algorithms (SHA and HMAC-SHA)

> Specifies Secure Hash Algorithms, SHA-256, SHA-384, and SHA-512, for computing a condensed representation of a message or a data file.

RFC 4752: The Kerberos V5 ("GSSAPI") Simple Authentication and Security Layer (SASL) Mechanism

> Describes the method for using the Generic Security Service Application Program Interface (GSS-API) Kerberos V5 in the Simple Authentication and Security Layer, called the GSSAPI mechanism.

RFC 4876: A Configuration Profile Schema for Lightweight Directory Access Protocol (LDAP)-Based Agents

> Defines a schema for storing a profile for agents that make use of the Lightweight Directory Access protocol (LDAP).

RFC 5020: The Lightweight Directory Access Protocol (LDAP) entryDN Operational Attribute

> Describes the Lightweight Directory Access Protocol (LDAP) / X.500 'entryDN' operational attribute, that provides a copy of the entry's distinguished name for use in attribute value assertions.

FIPS 180-1: Secure Hash Standard (SHA-1)

> Specifies a Secure Hash Algorithm, SHA-1, for computing a condensed representation of a message or a data file.

FIPS 180-2: Secure Hash Standard (SHA-1, SHA-256, SHA-384, SHA-512)

> Specifies four Secure Hash Algorithms for computing a condensed representation of electronic data.

DSMLv2: Directory Service Markup Language

> Provides a method for expressing directory queries and updates as XML documents.

JavaScript Object Notation

> A data-interchange format that aims to be both "easy for humans to read and write," and also "easy for machines to parse and generate."

Simple Cloud Identity Management: Core Schema 1.0

> Platform neutral schema and extension model for representing users and groups in JSON and XML formats. OpenDJ supports the JSON formats.

Appendix E. LDAP Controls

Controls provide a mechanism whereby the semantics and arguments of existing LDAP operations may be extended. One or more controls may be attached to a single LDAP message. A control only affects the semantics of the message it is attached to. Controls sent by clients are termed *request controls*, and those sent by servers are termed *response controls*.

OpenDJ software supports the following LDAP controls.

Account Usability Control

> Object Identifier: 1.3.6.1.4.1.42.2.27.9.5.8
>
> Control originally provided by Sun Microsystems, used to determine whether a user account can be used to authenticate to the directory.

Assertion Request Control

> Object Identifier: 1.3.6.1.1.12
>
> RFC: RFC 4528 - Lightweight Directory Access Protocol (LDAP) Assertion Control

Authorization Identity Request Control

> Object Identifier: 2.16.840.1.113730.3.4.16
>
> RFC: RFC 3829 - Lightweight Directory Access Protocol (LDAP) Authorization Identity Request and Response Controls

Authorization Identity Response Control

Object Identifier: 2.16.840.1.113730.3.4.15

RFC: RFC 3829 - Lightweight Directory Access Protocol (LDAP) Authorization Identity Request and Response Controls

Entry Change Notification Response Control

Object Identifier: 2.16.840.1.113730.3.4.7

Internet-Draft: draft-ietf-ldapext-psearch - Persistent Search: A Simple LDAP Change Notification Mechanism

Get Effective Rights Request Control

Object Identifier: 1.3.6.1.4.1.42.2.27.9.5.2

Internet-Draft: draft-ietf-ldapext-acl-model - Access Control Model for LDAPv3

Manage DSAIT Request Control

Object Identifier: 2.16.840.1.113730.3.4.2

RFC: RFC 3296 - Named Subordinate References in Lightweight Directory Access Protocol (LDAP) Directories

Matched Values Request Control

Object Identifier: 1.2.826.0.1.3344810.2.3

RFC: RFC 3876 - Returning Matched Values with the Lightweight Directory Access Protocol version 3 (LDAPv3)

No-Op Control

Object Identifier: 1.3.6.1.4.1.4203.1.10.2

Internet-Draft: draft-zeilenga-ldap-noop - LDAP No-Op Control

Password Expired Response Control

Object Identifier: 2.16.840.1.113730.3.4.4

Internet-Draft: draft-vchu-ldap-pwd-policy - Password Policy for LDAP Directories

Password Expiring Response Control

Object Identifier: 2.16.840.1.113730.3.4.5

Internet-Draft: draft-vchu-ldap-pwd-policy - Password Policy for LDAP Directories

Password Policy Response Control

Object Identifier: 1.3.6.1.4.1.42.2.27.8.5.1

Internet-Draft: draft-behera-ldap-password-policy - Password Policy for LDAP Directories

Permissive Modify Request Control

Object Identifier: 1.2.840.113556.1.4.1413

Microsoft defined this control that, "Allows an LDAP modify to work under less restrictive conditions. Without it, a delete will fail if an attribute done not exist, and an add will fail if an attribute already exists. No data is needed in this control." (source of quote)

Persistent Search Request Control

Object Identifier: 2.16.840.1.113730.3.4.3

Internet-Draft: draft-ietf-ldapext-psearch - Persistent Search: A Simple LDAP Change Notification Mechanism

Post-Read Request Control

Object Identifier: 1.3.6.1.1.13.2

RFC: RFC 4527 - Lightweight Directory Access Protocol (LDAP) Read Entry Controls

Post-Read Response Control

Object Identifier: 1.3.6.1.1.13.2

RFC: RFC 4527 - Lightweight Directory Access Protocol (LDAP) Read Entry Controls

Pre-Read Request Control

Object Identifier: 1.3.6.1.1.13.1

RFC: RFC 4527 - Lightweight Directory Access Protocol (LDAP) Read Entry Controls

Pre-Read Response Control

Object Identifier: 1.3.6.1.1.13.1

RFC: RFC 4527 - Lightweight Directory Access Protocol (LDAP) Read Entry Controls

Proxied Authorization v1 Request Control

Object Identifier: 2.16.840.1.113730.3.4.12

Internet-Draft: draft-weltman-ldapv3-proxy-04 - LDAP Proxied Authorization Control

Proxied Authorization v2 Request Control

Object Identifier: 2.16.840.1.113730.3.4.18

RFC: RFC 4370 - Lightweight Directory Access Protocol (LDAP) Proxied Authorization Control

Public Changelog Exchange Control

Object Identifier: 1.3.6.1.4.1.26027.1.5.4

OpenDJ specific, for using the bookmark cookie when reading the external change log.

Server Side Sort Request Control

Object Identifier: 1.2.840.113556.1.4.473

RFC: RFC 2891 - LDAP Control Extension for Server Side Sorting of Search Results

Server Side Sort Response Control

Object Identifier: 1.2.840.113556.1.4.474

RFC: RFC 2891 - LDAP Control Extension for Server Side Sorting of Search Results

Simple Paged Results Control

Object Identifier: 1.2.840.113556.1.4.319

RFC: RFC 2696 - LDAP Control Extension for Simple Paged Results Manipulation

Subentries Request Controls

Object Identifier: 1.3.6.1.4.1.4203.1.10.1

RFC: Subentries in the Lightweight Directory Access Protocol (LDAP)

Object Identifier: 1.3.6.1.4.1.7628.5.101.1

Internet-Draft: draft-ietf-ldup-subentry - LDAP Subentry Schema

Subtree Delete Request Control

Object Identifier: 1.2.840.113556.1.4.805

Internet-Draft: draft-armijo-ldap-treedelete - Tree Delete Control

Virtual List View Request Control

Object Identifier: 2.16.840.1.113730.3.4.9

Internet-Draft: draft-ietf-ldapext-ldapv3-vlv - LDAP Extensions for Scrolling View Browsing of Search Results

Virtual List View Response Control

Object Identifier: 2.16.840.1.113730.3.4.10

Internet-Draft: draft-ietf-ldapext-ldapv3-vlv - LDAP Extensions for Scrolling View Browsing of Search Results

The LDAP Relax Rules Control

Object Identifier: 1.3.6.1.4.1.4203.666.5.12

Internet-Draft: ddraft-zeilenga-ldap-relax-03 - The LDAP Relax Rules Control

Appendix F. LDAP Extended Operations

Extended operations allow additional operations to be defined for services not already available in the protocol

OpenDJ software supports the following LDAP extended operations.

Cancel Extended Request

> Object Identifier: 1.3.6.1.1.8
>
> RFC: RFC 3909 - Lightweight Directory Access Protocol (LDAP) Cancel Operation

Get Connection ID Extended Request

> Object Identifier: 1.3.6.1.4.1.26027.1.6.2
>
> OpenDJ extended operation to return the connection ID of the associated client connection. This extended operation is intended for OpenDJ internal use.

Password Modify Extended Request

> Object Identifier: 1.3.6.1.4.1.4203.1.11.1
>
> RFC: RFC 3062 - LDAP Password Modify Extended Operation

Password Policy State Extended Operation

> Object Identifier: 1.3.6.1.4.1.26027.1.6.1

OpenDJ extended operation to query and update password policy state for a given user entry. This extended operation is intended for OpenDJ internal use.

Start Transport Layer Security Extended Request

Object Identifier: 1.3.6.1.4.1.1466.20037

RFC: RFC 4511 - Lightweight Directory Access Protocol (LDAP): The Protocol

Who am I? Extended Request

Object Identifier: 1.3.6.1.4.1.4203.1.11.3

RFC: RFC 4532 - Lightweight Directory Access Protocol (LDAP) "Who am I?" Operation

Appendix G. Localization

OpenDJ software stores data in UTF-8 format. It enables you to store and to search for attribute values according to a variety of language specific locales. OpenDJ software is also itself localized for a smaller variety of languages.

G.1 OpenDJ Languages

OpenDJ 4.8.1-SNAPSHOT software has been localized in the following languages.

- French
- German
- Japanese
- Simplified Chinese
- Spanish

Note

Certain messages have also been translated into Catalan, Korean, Polish, and Traditional Chinese. Some error messages including messages labeled SEVERE and FATAL are provided only in English.

G.2 Directory Support For Locales and Language Subtypes

OpenDJ software supports the following locales, with their associated language and country codes, and their collation order object identifiers. Locale support depends on the underlying Java Virtual Machine.

Albanian

Code tag: sq

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.127.1

Arabic

Code tag: ar

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.3.1

Arabic (Algeria)

Code tag: ar-DZ

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.6.1

Arabic (Bahrain)

Code tag: ar-BH

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.5.1

Arabic (Egypt)

Code tag: ar-EG

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.7.1

Arabic (Iraq)

Code tag: ar-IQ

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.9.1

Arabic (Jordan)

Code tag: ar-JO

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.10.1

Arabic (Kuwait)

Code tag: ar-KW

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.11.1

Arabic (Lebanon)

Code tag: ar-LB

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.12.1

Arabic (Libya)

Code tag: ar-LY

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.13.1

Arabic (Morocco)

Code tag: ar-MA

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.14.1

Arabic (Oman)

Code tag: ar-OM

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.15.1

Arabic (Qatar)

Code tag: ar-QA

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.16.1

Arabic (Saudi Arabia)

Code tag: ar-SA

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.17.1

Arabic (Sudan)

Code tag: ar-SD

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.18.1

Arabic (Syria)

Code tag: ar-SY

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.19.1

Arabic (Tunisia)

Code tag: ar-TN

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.20.1

Arabic (United Arab Emirates)

Code tag: ar-AE

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.4.1

Arabic (Yemen)

Code tag: ar-YE

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.21.1

Belarusian

Code tag: be

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.22.1

Bulgarian

Code tag: bg

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.23.1

Catalan

Code tag: ca

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.25.1

Chinese

Code tag: zh

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.143.1

Chinese (Simplified) (China)

Code tag: zh-CN

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.144.1

Chinese (Traditional) (Hong Kong)

Code tag: zh-HK

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.145.1

Chinese (Traditional) (Taiwan)

Code tag: zh-TW

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.148.1

Croatian

Code tag: hr

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.87.1

Czech

Code tag: cs

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.26.1

Danish

Code tag: da

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.27.1

Dutch

Code tag: nl

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.105.1

Dutch (Belgium)

Code tag: nl-BE

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.106.1

Dutch (Netherlands)

Code tag: nl-NL

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.105.1

English

Code tag: en

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.34.1

English (Australia)

Code tag: en-AU

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.35.1

English (Canada)

Code tag: en-CA

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.36.1

English (Great Britain)

Code tag: en-GB

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.37.1

English (India)

Code tag: en-IN

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.40.1

English (Ireland)

Code tag: en-IE

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.39.1

English (New Zealand)

Code tag: en-NZ

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.42.1

English (South Africa)

Code tag: en-ZA

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.46.1

English (United States)

Code tag: en-US

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.34.1

Estonian

Code tag: et

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.69.1

Finnish

Code tag: fi

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.74.1

French

Code tag: fr

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.76.1

French (Belgium)

Code tag: fr-BE

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.77.1

French (Canada)

Code tag: fr-CA

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.78.1

French (France)

Code tag: fr-FR

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.76.1

French (Luxembourg)

Code tag: fr-LU

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.80.1

French (Switzerland)

Code tag: fr-CH

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.79.1

German

Code tag: de

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.28.1

German (Austria)

Code tag: de-AT

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.29.1

German (Germany)

Code tag: de-DE

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.28.1

German (Luxembourg)

Code tag: de-LU

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.32.1

German (Switzerland)

Code tag: de-CH

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.31.1

Greek

Code tag: el

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.33.1

Hebrew

Code tag: he

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.85.1

Hungarian

Code tag: hu

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.88.1

Icelandic

Code tag: is

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.91.1

Italian

Code tag: it

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.92.1

Italian (Switzerland)

Code tag: it-CH

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.93.1

Japanese

Code tag: ja

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.94.1

Korean

Code tag: ko

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.97.1

Latvian

Code tag: lv

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.101.1

Lithuanian

Code tag: lt

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.100.1

Macedonian

Code tag: mk

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.102.1

Norwegian

Code tag: no

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.107.1

Norwegian (Bokmål)

Code tag: no-NO

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.107.1

Norwegian (Nynorsk)

Code tag: no-NO-NY

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.108.1

Polish

Code tag: pl

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.114.1

Portuguese

Code tag: pt

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.115.1

Portuguese (Brazil)

Code tag: pt-BR

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.116.1

Portuguese (Portugal)

Code tag: pt-PT

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.115.1

Romanian

Code tag: ro

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.117.1

Russian

Code tag: ru

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.118.1

Russian (Russia)

Code tag: ru-RU

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.118.1

Serbian

Code tag: sr

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.128.1

Slovak

Code tag: sk

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.121.1

Slovenian

Code tag: sl

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.122.1

Spanish

Code tag: es

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.49.1

Spanish (Argentina)

Code tag: es-AR

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.50.1

Spanish (Bolivia)

Code tag: es-BO

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.51.1

Spanish (Chile)

Code tag: es-CL

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.52.1

Spanish (Colombia)

Code tag: es-CO

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.53.1

Spanish (Costa Rica)

Code tag: es-CR

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.54.1

Spanish (Dominican Republic)

Code tag: es-DO

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.55.1

Spanish (Ecuador)

Code tag: es-EC

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.56.1

Spanish (El Salvador)

Code tag: es-SV

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.65.1

Spanish (Guatemala)

Code tag: es-GT

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.57.1

Spanish (Honduras)

Code tag: es-HN

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.58.1

Spanish (Mexico)

Code tag: es-MX

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.59.1

Spanish (Nicaragua)

Code tag: es-NI

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.60.1

Spanish (Panama)

Code tag: es-PA

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.61.1

Spanish (Paraguay)

Code tag: es-PY

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.64.1

Spanish (Peru)

Code tag: es-PE

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.62.1

Spanish (Puerto Rico)

Code tag: es-PR

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.63.1

Spanish (Spain)

Code tag: es-ES

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.49.1

Spanish (Uruguay)

Code tag: es-UY

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.67.1

Spanish (Venezuela)

Code tag: es-VE

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.68.1

Swedish

Code tag: sv

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.129.1

Swedish (Sweden)

Code tag: sv-SE

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.129.1

Thai

Code tag: th

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.136.1

Turkish

Code tag: tr

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.140.1

Ukranian

Code tag: uk

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.141.1

Vietnamese

Code tag: vi

Collation order object identifier: 1.3.6.1.4.1.42.2.27.9.4.142.1

OpenDJ software supports the following language subtypes.

- Albanian, sq
- Arabic, ar
- Belarusian, be
- Bulgarian, bg
- Catalan, ca
- Chinese, zh
- Croatian, hr
- Czech, cs
- Danish, da
- Dutch, nl
- English, en
- Estonian, et
- Finnish, fi
- French, fr
- German, de

- Greek, el
- Hebrew, he
- Hungarian, hu
- Icelandic, is
- Italian, it
- Japanese, ja
- Korean, ko
- Latvian, lv
- Lithuanian, lt
- Macedonian, mk
- Norwegian, no
- Polish, pl
- Portuguese, pt
- Romanian, ro
- Russian, ru
- Serbian, sr
- Slovak, sk
- Slovenian, sl
- Spanish, es
- Swedish, sv
- Thai, th
- Turkish, tr
- Ukranian, uk
- Vietnamese, vi

Appendix H. Release Levels & Interface Stability

This appendix includes ForgeRock definitions for product release levels and interface stability.

In addition to the indications concerning interface stability that you find in the documentation, review the following information about OpenDJ user and application programming interfaces.

- Client tools — **ldap***, **ldif***, and ***rate** commands — are Evolving.
- The following classes, interfaces, and methods in the OpenDJ directory server APIs are Evolving.
  - `org.forgerock.opendj.ldap.Connections#newInternalConnection`
  - `org.forgerock.opendj.ldap.Connections#newInternalConnectionFactory`
  - `org.forgerock.opendj.ldap.Connections#newServerConnectionFactory`
  - `org.forgerock.opendj.ldap.FutureResult`
  - `org.forgerock.opendj.ldap.LDAPClientContext`
  - `org.forgerock.opendj.ldap.LDAPListener`
  - `org.forgerock.opendj.ldap.LDAPListenerOptions`
  - `org.forgerock.opendj.ldap.MemoryBackend`

  - `org.forgerock.opendj.ldap.RequestContext`
  - `org.forgerock.opendj.ldap.RequestHandler`
  - `org.forgerock.opendj.ldap.RequestHandlerFactory`
  - `org.forgerock.opendj.ldap.ServerConnection`
  - `org.forgerock.opendj.ldap.ServerConnectionFactory`

- The following classes and interfaces in the OpenDJ LDAP SDK APIs are Evolving.

  - `org.forgerock.opendj.ldap.ConnectionSecurityLayer`
  - `org.forgerock.opendj.ldap.LDAPUrl`
  - `org.forgerock.opendj.ldap.requests.BindRequest`, including sub-types and especially SASL sub-types
  - `org.forgerock.opendj.ldap.schema.MatchingRuleImpl`
  - `org.forgerock.opendj.ldap.schema.SchemaValidationPolicy`
  - `org.forgerock.opendj.ldap.schema.SyntaxImpl`

  The following methods are Deprecated.

  - `org.forgerock.opendj.ldap.LDAPListenerOptions#getTCPNIOTransport`
  - `org.forgerock.opendj.ldap.LDAPListenerOptions#setTCPNIOTransport`
  - `org.forgerock.opendj.ldap.LDAPOptions#getTCPNIOTransport`
  - `org.forgerock.opendj.ldap.LDAPOptions#setTCPNIOTransport`

  The class `org.forgerock.opendj.ldap.CoreMessages` is Internal.

- For all Java APIs, `com.*` packages are Internal.

- The configuration, user, and application programming interfaces for RESTful access over HTTP to directory data are Evolving. This includes interfaces exposed for the HTTP Connection Handler, its access log, and also the REST LDAP gateway.

- Text in log messages should be considered Internal. Log message IDs are Evolving.

- The default content of `cn=schema` (directory server LDAP schema) is Evolving.

- The monitoring interface `cn=monitor` for LDAP and the monitoring interface exposed by the JMX Connection Handler are Evolving.
- Newly Deprecated and Removed interfaces are identified in the *Release Notes* chapter, *OpenDJ Compatibility*.
- Interfaces that are not described in released product documentation should be considered Internal/Undocumented. For example, the LDIF representation of the server configuration, `config.ldif`, should be considered Internal.

H.1 Open Identity Platform Product Release Levels

Open Identity Platform defines Major, Minor, and Maintenance product release levels. The release level is reflected in the version number. The release level tells you what sort of compatibility changes to expect.

Major (version: x[.0.0], trailing 0s are optional)

> Major releases bring big new features. Major releases can include changes even to Stable interfaces. Major releases can remove previously Deprecated functionality, and in rare cases remove Evolving functionality that has not been explicitly Deprecated. Major releases also include the changes present in previous Minor and Maintenance releases.

Minor (version: x.y[.0], trailing 0s are optional)

> Minor releases might include new features, backwards-compatible changes to Stable interfaces in the same Major release, and incompatible changes to Evolving interfaces. Minor releases can remove previously Deprecated functionality. Minor releases also include the changes present in Maintenance releases.

Maintenance (version: x.y.z)

> Maintenance releases can include bug fixes. Maintenance releases are intended to be fully compatible with previous versions from the same Minor release.

H.2 Open Identity Platform Product Interface Stability

Open Identity Platform products support many protocols, APIs, GUIs, and command-line interfaces. Some of these interfaces are standard and very stable. Others offer new functionality that is continuing to evolve.

We realize that you invest in these interfaces, and therefore must know when and how Open Identity Platform Community expects them to change. For that reason, Open Identity Platform Community defines interface stability labels and uses these definitions in Open Identity Platform products.

Stable

> This documented interface is expected to undergo only backwards-compatible changes between major releases. Changes are announced at least one minor release before they take effect.

Evolving

> This documented interface is continuing to evolve and so is expected to change, potentially in backwards-incompatible ways even in a minor release. Changes are documented at the time of product release.
>
> While new protocols and APIs are still in the process of standardization, they are Evolving. This applies for example to recent Internet-Draft implementations, and also to newly developed functionality.

Deprecated

> This interface is deprecated and likely to be removed in a future release. For previously stable interfaces, the change was likely announced in a previous release. Deprecated interfaces will be removed from Open Identity Platform products.

Removed

> This interface was deprecated in a previous release and has now been removed from the product.

Internal/Undocumented

> Internal and undocumented interfaces can change without notice. If you depend on one of these interfaces, contact Open Identity Platform support to discuss your needs.

Index