



Red Hat Directory Server Red Hat Directory Server 9

Administration Guide

Updated for Directory Server 9.1.2

Red Hat Directory Server Red Hat Directory Server 9 Administration Guide

Updated for Directory Server 9.1.2

Marc Muehlfeld
Red Hat Customer Content Services
mmuehlfeld@redhat.com

Petr Bokoč
Red Hat Customer Content Services

Ella Deon Ballard
Red Hat Customer Content Services

Legal Notice

Copyright © 2017 Red Hat, Inc.

This document is licensed by Red Hat under the [Creative Commons Attribution-ShareAlike 3.0 Unported License](#). If you distribute this document, or a modified version of it, you must provide attribution to Red Hat, Inc. and provide a link to the original. If the document is modified, all Red Hat trademarks must be removed.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This guide covers both GUI and command-line procedures for managing Directory Server instances and databases. This documentation is no longer maintained. For details, see .

Table of Contents

DEPRECATED DOCUMENTATION	6
PREFACE	6
1. DIRECTORY SERVER OVERVIEW	6
2. EXAMPLES AND FORMATTING	7
3. ADDITIONAL READING	8
4. GIVING FEEDBACK	9
CHAPTER 1. BASIC RED HAT DIRECTORY SERVER SETTINGS	11
1.1. SYSTEM REQUIREMENTS	11
1.2. DIRECTORY SERVER FILE LOCATIONS	12
1.3. STARTING AND STOPPING SERVERS	14
1.4. STARTING THE CONSOLE	15
1.5. ENABLING LDAP	18
1.6. CHANGING DIRECTORY SERVER PORT NUMBERS	19
1.7. CREATING A NEW DIRECTORY SERVER INSTANCE	23
1.8. USING DIRECTORY SERVER PLUG-INS	24
1.9. MANAGING CORE SERVER ATTRIBUTES	26
1.10. MANAGING SELINUX WITH THE DIRECTORY SERVER	28
CHAPTER 2. CONFIGURING DIRECTORY DATABASES	35
2.1. CREATING AND MAINTAINING SUFFIXES	35
2.2. CREATING AND MAINTAINING DATABASES	46
2.3. CREATING AND MAINTAINING DATABASE LINKS	62
2.4. CONFIGURING CASCADING CHAINING	87
2.5. USING REFERRALS	98
CHAPTER 3. CREATING DIRECTORY ENTRIES	107
3.1. MANAGING ENTRIES FROM THE DIRECTORY CONSOLE	107
3.2. MANAGING ENTRIES FROM THE COMMAND LINE	122
3.3. USING LDIF UPDATE STATEMENTS TO CREATE OR MODIFY ENTRIES	128
3.4. RENAMING AND MOVING ENTRIES	135
3.5. TRACKING MODIFICATIONS TO DIRECTORY ENTRIES	141
3.6. MAINTAINING REFERENTIAL INTEGRITY	147
CHAPTER 4. POPULATING DIRECTORY DATABASES	156
4.1. IMPORTING DATA	156
4.2. EXPORTING DATA	165
4.3. BACKING UP AND RESTORING DATA	171
CHAPTER 5. MANAGING ATTRIBUTES AND VALUES	180
5.1. ENFORCING ATTRIBUTE UNIQUENESS	180
5.2. ASSIGNING CLASS OF SERVICE	186
5.3. LINKING ATTRIBUTES TO MANAGE ATTRIBUTE VALUES	208
5.4. ASSIGNING AND MANAGING UNIQUE NUMERIC ATTRIBUTE VALUES	213
CHAPTER 6. ORGANIZING AND GROUPING ENTRIES	225
6.1. USING GROUPS	225
6.2. USING ROLES	250
6.3. AUTOMATICALLY CREATING DUAL ENTRIES	266
6.4. USING VIEWS	275
CHAPTER 7. CONFIGURING SECURE CONNECTIONS	283
7.1. REQUIRING SECURE CONNECTIONS	283

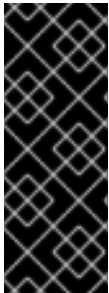
7.2. DISABLING SSL AND REQUIRING TLS	284
7.3. MANAGING CERTIFICATES USED BY THE DIRECTORY SERVER	284
7.4. SETTING UP TLS/SSL	298
7.5. COMMAND-LINE FUNCTIONS FOR START TLS	307
7.6. USING CERTUTIL	308
7.7. UPDATING ATTRIBUTE ENCRYPTION FOR NEW SSL/TLS CERTIFICATES	313
7.8. USING HARDWARE SECURITY MODULES	314
7.9. SETTING ENCRYPTION CIPHERS	316
7.10. USING CLIENT (CERTIFICATE-BASED) AUTHENTICATION	318
7.11. SETTING UP SASL IDENTITY MAPPING	328
7.12. USING KERBEROS GSS-API WITH SASL	334
7.13. DISABLING SASL MECHANISMS	339
7.14. USING SASL WITH LDAP CLIENTS	339
CHAPTER 8. MANAGING THE DIRECTORY SCHEMA	341
8.1. OVERVIEW OF SCHEMA	341
8.2. MANAGING OBJECT IDENTIFIERS	345
8.3. DIRECTORY SERVER ATTRIBUTE SYNTAXES	345
8.4. MANAGING CUSTOM SCHEMA IN THE CONSOLE	348
8.5. MANAGING SCHEMA USING LDAPMODIFY	355
8.6. CREATING CUSTOM SCHEMA FILES	357
8.7. DYNAMICALLY RELOADING SCHEMA	359
8.8. TURNING SCHEMA CHECKING ON AND OFF	362
8.9. USING SYNTAX VALIDATION	363
CHAPTER 9. MANAGING INDEXES	367
9.1. ABOUT INDEXES	367
9.2. CREATING STANDARD INDEXES	374
9.3. APPLYING NEW INDEXES TO EXISTING DATABASES	378
9.4. CREATING BROWSING (VLV) INDEXES	379
9.5. CHANGING THE INDEX SORT ORDER	384
9.6. CHANGING THE WIDTH FOR INDEXED SUBSTRING SEARCHES	385
9.7. DELETING INDEXES	386
CHAPTER 10. FINDING DIRECTORY ENTRIES	394
10.1. IMPROVING SEARCH PERFORMANCE THROUGH RESOURCE LIMITS	394
10.2. FINDING ENTRIES USING THE DIRECTORY SERVER CONSOLE	399
10.3. USING LDAPSEARCH	401
10.4. LDAP SEARCH FILTERS	405
10.5. EXAMPLES OF COMMON LDAPSEARCHES	419
10.6. USING PERSISTENT SEARCH	423
10.7. SEARCHING WITH SPECIFIED CONTROLS	424
CHAPTER 11. MANAGING REPLICATION	431
11.1. REPLICATION OVERVIEW	431
11.2. REPLICATION SCENARIOS	435
11.3. CREATING THE SUPPLIER BIND DN ENTRY	439
11.4. CONFIGURING SINGLE-MASTER REPLICATION	441
11.5. CONFIGURING MULTI-MASTER REPLICATION	452
11.6. CONFIGURING CASCADING REPLICATION	465
11.7. CONFIGURING REPLICATION FROM THE COMMAND LINE	479
11.8. TEMPORARILY SUSPENDING REPLICATION	493
11.9. MANAGING ATTRIBUTES WITHIN FRACTIONAL REPLICATION	494
11.10. MAKING A READ-ONLY REPLICA UPDATABLE	496

11.11. REMOVING A SUPPLIER FROM THE REPLICATION TOPOLOGY	497
11.12. MANAGING DELETED ENTRIES WITH REPLICATION	499
11.13. REMOVING THE CHANGELOG	500
11.14. TRIMMING THE REPLICATION CHANGELOG	501
11.15. INITIALIZING CONSUMERS	502
11.16. FORCING REPLICATION UPDATES	506
11.17. REPLICATION OVER SSL	509
11.18. SETTING REPLICATION TIMEOUT PERIODS	510
11.19. REPLICATING O=NETSCAPERROOT FOR ADMIN SERVER FAILOVER	511
11.20. REPLICATION WITH EARLIER RELEASES	513
11.21. USING THE RETRO CHANGELOG PLUG-IN	515
11.22. MONITORING REPLICATION STATUS	517
11.23. SETTING REPLICATION SESSION HOOKS	522
11.24. SOLVING COMMON REPLICATION CONFLICTS	523
11.25. TROUBLESHOOTING REPLICATION-RELATED PROBLEMS	531
CHAPTER 12. SYNCHRONIZING RED HAT DIRECTORY SERVER WITH MICROSOFT ACTIVE DIRECTORY	536
12.1. ABOUT WINDOWS SYNC	536
12.2. SUPPORTED ACTIVE DIRECTORY VERSIONS	539
12.3. STEPS FOR CONFIGURING WINDOWS SYNC	539
12.4. SYNCHRONIZING USERS	561
12.5. SYNCHRONIZING GROUPS	568
12.6. CONFIGURING UNI-DIRECTIONAL SYNCHRONIZATION	574
12.7. SYNCHRONIZING POSIX ATTRIBUTES FOR USERS AND GROUPS	575
12.8. DELETING AND RESURRECTING ENTRIES	577
12.9. SENDING SYNCHRONIZATION UPDATES	578
12.10. MODIFYING THE SYNC AGREEMENT	581
12.11. MANAGING THE PASSWORD SYNC SERVICE	590
12.12. TROUBLESHOOTING	593
CHAPTER 13. MANAGING ACCESS CONTROL	595
13.1. ACCESS CONTROL PRINCIPLES	595
13.2. DEFAULT ACIS	597
13.3. CREATING ACIS MANUALLY	598
13.4. BIND RULES	608
13.5. CREATING ACIS FROM THE CONSOLE	626
13.6. VIEWING ACIS	634
13.7. CHECKING ACCESS RIGHTS ON ENTRIES (GET EFFECTIVE RIGHTS)	635
13.8. LOGGING ACCESS CONTROL INFORMATION	647
13.9. ACCESS CONTROL USAGE EXAMPLES	648
13.10. ADVANCED ACCESS CONTROL: USING MACRO ACIS	673
13.11. SETTING ACCESS CONTROLS ON DIRECTORY MANAGER	678
13.12. ACCESS CONTROL AND REPLICATION	680
13.13. COMPATIBILITY WITH EARLIER RELEASES	680
CHAPTER 14. MANAGING USER AUTHENTICATION	682
14.1. MANAGING THE PASSWORD POLICY	682
14.2. MANAGING THE DIRECTORY MANAGER PASSWORD	700
14.3. CHECKING ACCOUNT AVAILABILITY FOR PASSWORDLESS ACCESS	703
14.4. CONFIGURING A PASSWORD-BASED ACCOUNT LOCKOUT POLICY	706
14.5. CONFIGURING TIME-BASED ACCOUNT LOCKOUT POLICIES	709
14.6. REPLICATING ACCOUNT LOCKOUT ATTRIBUTES	715
14.7. SYNCHRONIZING PASSWORDS	718

14.8. ENABLING DIFFERENT TYPES OF BINDS	719
14.9. USING PASS-THROUGH AUTHENTICATION	724
14.10. USING PAM FOR PASS-THROUGH AUTHENTICATION	733
14.11. MANUALLY INACTIVATING USERS AND ROLES	741
CHAPTER 15. MONITORING SERVER AND DATABASE ACTIVITY	745
15.1. TYPES OF DIRECTORY SERVER LOG FILES	745
15.2. VIEWING LOG FILES	745
15.3. CONFIGURING LOG FILES	746
15.4. GETTING ACCESS LOG STATISTICS	753
15.5. REPLACING LOG FILES WITH A NAMED PIPE	757
15.6. MONITORING THE LOCAL DISK FOR GRACEFUL SHUTDOWN	762
15.7. MONITORING SERVER ACTIVITY	764
15.8. MONITORING DATABASE ACTIVITY	771
15.9. MONITORING DATABASE LINK ACTIVITY	778
15.10. ENABLING AND DISABLING COUNTERS	779
CHAPTER 16. MONITORING DIRECTORY SERVER USING SNMP	781
16.1. ABOUT SNMP	781
16.2. CONFIGURING THE MASTER AGENT	781
16.3. CONFIGURING THE SUBAGENT	782
16.4. CONFIGURING SNMP TRAPS	784
16.5. CONFIGURING THE DIRECTORY SERVER FOR SNMP	784
16.6. USING THE MANAGEMENT INFORMATION BASE	785
CHAPTER 17. PLANNING FOR DISASTER	790
17.1. IDENTIFYING POTENTIAL SCENARIOS	790
17.2. DEFINING THE TYPE OF ROLLOVER	791
17.3. IDENTIFYING USEFUL DIRECTORY SERVER FEATURES FOR DISASTER RECOVERY	791
17.4. DEFINING THE RECOVERY PROCESS	793
17.5. BASIC EXAMPLE: PERFORMING A RECOVERY	793
APPENDIX A. USING LDAP CLIENT TOOLS	795
A.1. ENVIRONMENT VARIABLES USED WITH LDAP CLIENT TOOLS	795
A.2. USING SSL/TLS AND START TLS WITH LDAP CLIENT TOOLS	796
A.3. USING SASL WITH LDAP CLIENT TOOLS	797
A.4. RUNNING EXTENDED OPERATIONS	800
A.5. ADDING ENTRIES	801
A.6. COMPARING ENTRIES	802
A.7. CHANGING PASSWORDS	803
A.8. GENERATING LDAP URLS	805
APPENDIX B. LDAP DATA INTERCHANGE FORMAT	807
B.1. ABOUT THE LDIF FILE FORMAT	807
B.2. CONTINUING LINES IN LDIF	808
B.3. REPRESENTING BINARY DATA	809
B.4. SPECIFYING DIRECTORY ENTRIES USING LDIF	810
B.5. DEFINING DIRECTORIES USING LDIF	814
B.6. STORING INFORMATION IN MULTIPLE LANGUAGES	816
APPENDIX C. LDAP URLS	818
C.1. COMPONENTS OF AN LDAP URL	818
C.2. ESCAPING UNSAFE CHARACTERS	819
C.3. EXAMPLES OF LDAP URLS	820

APPENDIX D. INTERNATIONALIZATION	823
D.1. ABOUT LOCALES	823
D.2. SUPPORTED LOCALES	824
D.3. SUPPORTED LANGUAGE SUBTYPES	826
D.4. SEARCHING AN INTERNATIONALIZED DIRECTORY	828
D.5. TROUBLESHOOTING MATCHING RULES	833
APPENDIX E. MANAGING THE ADMIN SERVER	834
E.1. INTRODUCTION TO RED HAT ADMIN SERVER	834
E.2. ADMIN SERVER CONFIGURATION	835
APPENDIX F. USING ADMIN EXPRESS	868
F.1. MANAGING SERVERS IN ADMIN EXPRESS	868
F.2. CONFIGURING ADMIN EXPRESS	870
APPENDIX G. USING THE CONSOLE	878
G.1. OVERVIEW OF THE DIRECTORY SERVER CONSOLE	878
G.2. BASIC TASKS IN THE RED HAT CONSOLE	886
G.3. MANAGING SERVER INSTANCES	899
G.4. MANAGING DIRECTORY SERVER USERS AND GROUPS	904
G.5. SETTING ACCESS CONTROLS	919
GLOSSARY	924
INDEX	941
APPENDIX H. REVISION HISTORY	997

DEPRECATED DOCUMENTATION



IMPORTANT

Note that as of June 10, 2017, the support for Red Hat Directory Server 9 has ended. For details, see [“Red Hat Directory Server Life Cycle policy”](#). Red Hat recommends users of Directory Server 9 to update to the latest version.

Due to the end of the maintenance phase of this product, this documentation is no longer updated. Use it only as a reference!

PREFACE

Red Hat Directory Server (Directory Server) is a powerful and scalable distributed directory server based on the industry-standard Lightweight Directory Access Protocol (LDAP). Directory Server is the cornerstone for building a centralized and distributed data repository that can be used in your intranet, over your extranet with your trading partners, or over the public Internet to reach your customers.

This *Administrator's Guide* describes all of the administration tasks you need to perform to maintain Directory Server.

1. DIRECTORY SERVER OVERVIEW

Directory Server provides the following key features:

- **Multi-master replication** — Provides a highly available directory service for both read and write operations. Multi-master replication can be combined with simple and cascading replication scenarios to provide a highly flexible and scalable replication environment.
- **Chaining and referrals** — Increases the power of your directory by storing a complete logical view of your directory on a single server while maintaining data on a large number of Directory Servers transparently for clients.
- **Roles and classes of service** — Provides a flexible mechanism for grouping and sharing attributes between entries in a dynamic fashion.
- **Improved access control mechanisms** — Provides support for macros that dramatically reduce the number of access control statements used in the directory and increase the scalability of access control evaluation.
- **Resource-limits by bind DN** — Grants the power to control the amount of server resources allocated to search operations based on the bind DN of the client.
- **Multiple databases** — Provides a simple way of breaking down your directory data to simplify the implementation of replication and chaining in your directory service.
- **Password policy and account lockout** — Defines a set of rules that govern how passwords and user accounts are managed in the Directory Server.

- TLS and SSL — Provides secure authentication and communication over the network, using the Mozilla Network Security Services (NSS) libraries for cryptography.

The major components of Directory Server include the following:

- An LDAP server — The LDAP v3-compliant network daemon.
- Directory Server Console — A graphical management console that dramatically reduces the effort of setting up and maintaining your directory service.
- SNMP agent — Can monitor the Directory Server using the Simple Network Management Protocol (SNMP).

2. EXAMPLES AND FORMATTING

Each of the examples used in this guide, such as file locations and commands, have certain defined conventions.

2.1. Command and File Examples

All of the examples for Red Hat Directory Server commands, file locations, and other usage are given for Red Hat Enterprise Linux 6 (64-bit) systems. Be certain to use the appropriate commands and files for your platform.

Example 1. Example Command

To start the Red Hat Directory Server:

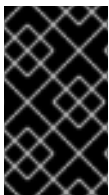
```
service dirsrv start
```

2.2. Brackets

Square brackets (`[]`) are used to indicate an alternative element in a name. For example, if a tool is available in `/usr/lib` on 32-bit systems and in `/usr/lib64` on 64-bit systems, then the tool location may be represented as `/usr/lib[64]`.

2.3. Client Tool Information

The tools for Red Hat Directory Server are located in the `/usr/bin` and the `/usr/sbin` directories.




IMPORTANT

The LDAP tools such as `ldapmodify` and `ldapsearch` from OpenLDAP use SASL connections by default. To perform a simple bind using a user name and password, use the `-x` argument to disable SASL.

2.4. Text Formatting and Styles

Certain words are represented in different fonts, styles, and weights. Different character formatting is used to indicate the function or purpose of the phrase being highlighted.

Formatting Style	Purpose
Monospace font	Monospace is used for commands, package names, files and directory paths, and any text displayed in a prompt.
 Monospace with a background	This type of formatting is used for anything entered or returned in a command prompt.
<i>Italicized text</i>	Any text which is italicized is a variable, such as <i>instance_name</i> or <i>hostname</i> . Occasionally, this is also used to emphasize a new term or other phrase.
Bolded text	Most phrases which are in bold are application names, such as Cygwin , or are fields or options in a user interface, such as a User Name Here: field or Save button.

Other formatting styles draw attention to important text.



NOTE

A note provides additional information that can help illustrate the behavior of the system or provide more detail for a specific issue.



IMPORTANT

Important information is necessary, but possibly unexpected, such as a configuration change that will not persist after a reboot.



WARNING

A warning indicates potential data loss, as may happen when tuning hardware for maximum performance.

3. ADDITIONAL READING

The *Red Hat Directory Server Deployment Guide* describes many of the basic directory and architectural concepts that you need to deploy, install, and administer a directory service successfully.

When you are familiar with Directory Server concepts and have done some preliminary planning for your directory service, install the Directory Server. The instructions for installing the various Directory Server components are contained in the *Red Hat Directory Server Installation Guide*. Many of the scripts and

commands used to install and administer the Directory Server are explained in detail in the *Red Hat Directory Server Configuration and Command-Line Tool Reference*.

The *Directory Server Administrator's Guide* describes how to set up, configure, and administer Red Hat Directory Server and its contents.

The document set for Directory Server contains the following guides:

- *Red Hat Directory Server Release Notes* contain important information on new features, fixed bugs, known issues and workarounds, and other important deployment information for this specific version of Directory Server.
- *Red Hat Directory Server Deployment Guide* provides an overview for planning a deployment of the Directory Server.
- *Red Hat Directory Server Administrator's Guide* contains procedures for the day-to-day maintenance of the directory service. Includes information on configuring server-side plug-ins.
- *Red Hat Directory Server Configuration and Command-Line Tool Reference* provides reference information on the command-line scripts, configuration attributes, schema elements, and log files shipped with Directory Server.
- *Red Hat Directory Server Installation Guide* contains procedures for installing your Directory Server as well as procedures for migrating from a previous installation of Directory Server.
- *Red Hat Directory Server Plug-in Programmer's Guide* describes how to write server plug-ins in order to customize and extend the capabilities of Directory Server.
- The *Red Hat Directory Server Performance Tuning Guide* contains features to monitor overall Directory Server and database performance, to tune attributes for specific operations, and to tune the server and database for optimum performance.

For the latest information about Directory Server, including current release notes, complete product documentation, technical notes, and deployment information, see the Red Hat Directory Server documentation site at https://access.redhat.com/site/documentation/Red_Hat_Directory_Server/.

4. GIVING FEEDBACK

If there is any error in this *Administrator's Guide* or there is any way to improve the documentation, please let us know. Bugs can be filed against the documentation for Red Hat Directory Server through Bugzilla, <http://bugzilla.redhat.com/bugzilla>. Make the bug report as specific as possible, so we can be more effective in correcting any issues:

1. Select the Red Hat Directory Server product.
2. Set the component to **Doc - administration-guide**.
3. Set the version number to 9.0.
4. For errors, give the page number (for the PDF) or URL (for the HTML), and give a succinct description of the problem, such as incorrect procedure or typo.

For enhancements, put in what information needs to be added and why.

5. Give a clear title for the bug. For example, "**Incorrect command example for setup script options**" is better than "**Bad example**".

We appreciate receiving any feedback — requests for new sections, corrections, improvements, enhancements, even new ways of delivering the documentation or new styles of docs. You are welcome to contact Red Hat Content Services directly at docs@redhat.com.

CHAPTER 1. BASIC RED HAT DIRECTORY SERVER SETTINGS

Red Hat Directory Server product includes a directory service, an administration server to manage multiple server instances, and a Java-based console to manage server instances through a graphical interface. This chapter provides an overview of the basic tasks for administering a directory service.

The Directory Server is a robust, scalable server designed to manage an enterprise-wide directory of users and resources. It is based on an open-systems server protocol called the Lightweight Directory Access Protocol (LDAP). Directory Server runs the **ns-slapd** daemon on the host machine. The server manages the directory databases and responds to client requests.

Directory Server 9.0 is comprised of several components, which work in tandem:

- The *Directory Server* is the core LDAP server daemon. It is compliant with LDAP v3 standards. This component includes command-line server management and administration programs and scripts for common operations like export and backing up databases.
- The *Directory Server Console* is the user interface that simplifies managing users, groups, and other LDAP data for your enterprise. The Console is used for all aspects of server management, including making backups; configuring security, replication, and databases; adding entries; and monitoring servers and viewing statistics.
- The *Admin Server* is the management agent which administers Directory Server instances. It communicates with the Directory Server Console and performs operations on the Directory Server instances. It also provides a simple HTML interface and online help pages.

Most Directory Server administrative tasks are available through the Directory Server Console, but it is also possible to administer the Directory Server by manually editing the configuration files or by using command-line utilities.

1.1. SYSTEM REQUIREMENTS

This section contains information related to installing and upgrading Red Hat Directory Server 9.0, including prerequisites and hardware or platform requirements.

1.1.1. Required JRE

Red Hat Directory Server 9.0 requires the Oracle Java Runtime Environment (JRE) 1.8.0 or OpenJDK 1.8.0 for Red Hat Enterprise Linux 5 and 6.



IMPORTANT

It is not possible to manage instances of Directory Server older than 8.1 (which used JRE 1.5) with the 9.0 Directory Server Console because they are using different JRE versions.

1.1.2. Directory Server Supported Platforms

Directory Server 9.0 is supported on the following platforms:

- Red Hat Enterprise Linux 6 (32-bit)
- Red Hat Enterprise Linux 6 (64-bit)

**NOTE**

Red Hat Directory Server 9.0 is supported running on a virtual guest on a Red Hat Enterprise Linux virtual server.

1.1.3. Directory Server Console Supported Platforms

The Directory Server Console is supported on the following platforms:

- Red Hat Enterprise Linux 5 (32-bit)
- Red Hat Enterprise Linux 5 (64-bit)
- Red Hat Enterprise Linux 6 (32-bit)
- Red Hat Enterprise Linux 6 (64-bit)
- Microsoft Windows Server 2008 (32-bit)
- Microsoft Windows Server 2008 (64-bit)

1.1.4. Password Sync Service Platforms

The Password Sync Service works with these Microsoft Windows services:

- Active Directory on Microsoft Windows Server 2008 (32-bit)
- Active Directory on Microsoft Windows Server 2008 (64-bit)

1.1.5. Web Application Browser Support

Directory Server 9.0 supports the following browsers to access web-based interfaces, such as **Admin Express** and online help tools:

- Firefox 3.x
- Microsoft Internet Explorer 6.0 and higher

1.2. DIRECTORY SERVER FILE LOCATIONS

Red Hat Directory Server 9.0 conforms to the Filesystem Hierarchy Standards. For more information on FHS, see the FHS homepage, <http://www.pathname.com/fhs/>. The files and directories installed with Directory Server are listed in the tables below for each supported platform.

In the file locations listed in the following tables, *instance* is the server instance name that was given during setup. By default, this is the leftmost component of the fully-qualified host and domain name. For example, if the host name is **ldap.example.com**, the instance name is **ldap** by default.

The Admin Server directories are named the same as the Directory Server directories, only instead of the instance as a directory name, the Admin Server directories are named **admin-serv**. For any directory or folder named **slapd-*instance***, substitute **admin-serv**, such as **/etc/dirsrv/slapd-example** and **/etc/dirsrv/admin-serv**.

Table 1.1. Red Hat Enterprise Linux 5 (x86)

File or Directory	Location
Log files	/var/log/dirsrv/slapd-<i>instance</i>
Configuration files	/etc/dirsrv/slapd-<i>instance</i>
Instance directory	/usr/lib/dirsrv/slapd-<i>instance</i>
Certificate and key databases	<i>/etc/dirsrv/slapd-<i>instance</i></i>
Database files	/var/lib/dirsrv/slapd-<i>instance</i>
Runtime files	/var/lock/dirsrv/slapd-<i>instance</i> /var/run/dirsrv/slapd-<i>instance</i>
Initscripts	/etc/rc.d/init.d/dirsrv and /etc/sysconfig/dirsrv /etc/rc.d/init.d/dirsrv-admin and /etc/sysconfig/dirsrv-admin
Tools	/usr/bin/ /usr/sbin/

Table 1.2. Red Hat Enterprise Linux 5 and 6 (x86_64)

File or Directory	Location
Log files	/var/log/dirsrv/slapd-<i>instance</i>
Configuration files	/etc/dirsrv/slapd-<i>instance</i>
Instance directory	/usr/lib64/dirsrv/slapd-<i>instance</i>
Certificate and key databases	<i>/etc/dirsrv/slapd-<i>instance</i></i>
Database files	/var/lib/dirsrv/slapd-<i>instance</i>
Runtime files	/var/lock/dirsrv/slapd-<i>instance</i> /var/run/dirsrv/slapd-<i>instance</i>
Initscripts	/etc/rc.d/init.d/dirsrv and /etc/sysconfig/dirsrv /etc/rc.d/init.d/dirsrv-admin and /etc/sysconfig/dirsrv-admin

File or Directory	Location
Tools	<code>/usr/bin/</code> <code>/usr/sbin/</code>

1.3. STARTING AND STOPPING SERVERS

The Directory Server is running when the `setup-ds-admin.pl` script completes. Avoid stopping and starting the server to prevent interrupting replication, searches, and other server operations.

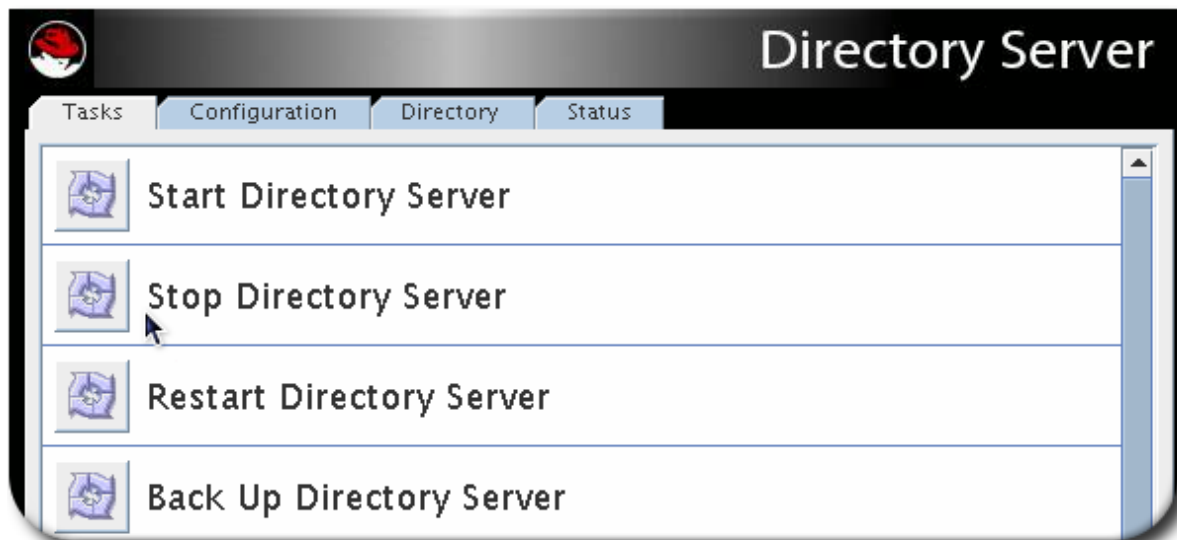
- If the Directory Server has SSL enabled, you cannot restart the server from the Console; you must use the command-line. It is possible to restart without being prompted for a password; see [Section 7.4.4, “Creating a Password File for the Directory Server”](#) for more information.
- Rebooting the host system can automatically start the `ns-slapd` process. The directory provides startup or run command (`rc`) scripts. Use the `chkconfig` command to enable the Directory Server and Admin Server to start on boot.

1.3.1. Starting and Stopping Directory Server from the Console

1. Start the Directory Server Console.

```
redhat-idm-console -a http://localhost:9830
```

2. In the **Tasks** tab, click **Start the Directory Server**, **Stop the Directory Server**, or **Restart the Directory Server**.



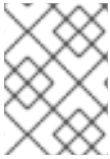
When the Directory Server is successfully started or stopped from the Directory Server Console, the server displays a message box stating that the server has either started or shut down.

1.3.2. Starting and Stopping Directory Server from the Command Line

The most common way to start and stop the Directory Server service is using system tools on Red Hat Enterprise Linux. For example, Linux uses the `service` tool:

```
service dirsrv {start|stop|restart} instance
```

Passing the instance name stops or starts only that instance; not giving any name starts or stops all instances.



NOTE

The service name for the Directory Server service on Red Hat Enterprise Linux is **dirsrv**.

The start/stop scripts are in the **/usr/sbin/** directory and are run similar to the **service** start/stop command:

```
/usr/sbin/{start|stop|restart}-dirsrv instance
```

If the instance name is not given, then the all instances are started or stopped.

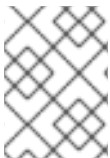
Alternatively, each instance has its own start and stop scripts that apply only to that instance.

```
/etc/dirsrv/slaped-instance_name/{start|stop|restart}-slaped
```

1.3.3. Starting and Stopping Admin Server

The Admin Server service is stopped and started using system tools on Red Hat Enterprise Linux. For example, on Red Hat Enterprise Linux 6 (64-bit), the command is **service**:

```
service dirsrv-admin {start|stop|restart}
```



NOTE

The service name for the Admin Server process on Red Hat Enterprise Linux is **dirsrv-admin**.

1.4. STARTING THE CONSOLE

1.4.1. Starting the Directory Server Console

There is a simple script to launch the Directory Server Console. The script is in the standard **/usr/bin** directory, so it can be run as follows:

```
redhat-idm-console
```



NOTE

Make sure that the correct Oracle Java Runtime Environment (JRE) or OpenJDK version is set in the **PATH** variable before launching the Console. To display the version and vendor information about Java on your system, enter:

```
java -version
```

The login screen prompts for the user name, password, and Admin Server location. It is possible to pass other information along with the Console command to supply the Admin Server URL, password, and user name. For example:

```
redhat-idm-console -a http://localhost:9830 -u "cn=Directory Manager" -w
secret
```

Table 1.3. redhat-idm-console Options

Option	Description
-a <i>adminURL</i>	Specifies a base URL for the instance of Admin Server to log into.
-f <i>fileName</i>	Writes errors and system messages to <i>fileName</i> .
-h	Prints out the help message for redhat-idm-console .
-s	Specifies the directory instance to access, either by specifying the DN of the server instance entry (SIE) or the instance name, such as slapd-example .
-u	Gives the user DN to use to log into the Console.
-w	Gives the password to use to log into the Console.
-w -	Reads the password from the standard output.
-x <i>options</i>	<p>Specifies extra options. There are three values for <i>extraOptions</i>:</p> <div> <p>nowinpos, which puts the Console window in the upper left corner of the screen</p> <p>nologo, which keeps the splash screen from being displayed and only opens the login dialog</p> <p>javalaaf, which uses the <i>Java look and feel</i> for the Console interface rather than the platform-specific styles</p> </div> <p>To use multiple options, separate them with a comma.</p>
-y <i>file</i>	Reads the password from the specified input file.

1.4.2. Logging into Directory Server

After starting the Directory Server Console, a login screen opens, requiring the user name and password for the user logging in and the URL for the Admin Server instance being access. The user logged in at

the Console is the user who is *binding* to Directory Server. This determines the access permissions granted and allowed operations while access the directory tree. The user account used to log into the Directory Server Console can make significant differences in the access; for example, the Directory Manager has access to every user and configuration entry in Directory Server, while the **admin** entry created during installation has access to only configuration entries, not user entries. Regular user accounts are more limited.

To bind to, or log into, the Directory Server, supply a user name and password at the login box.



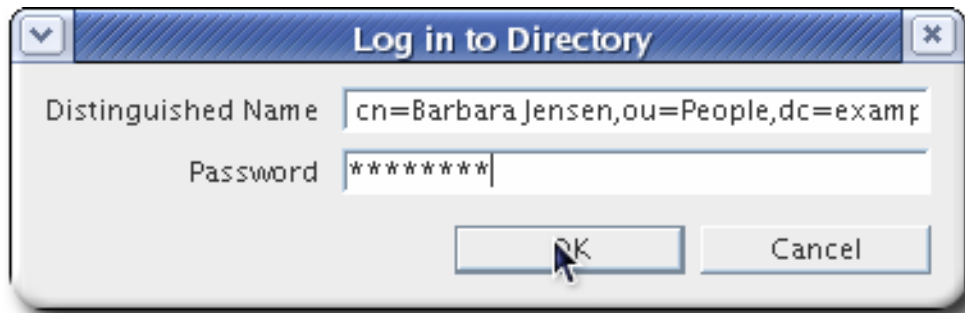
1.4.3. Changing the Login Identity

At any time during a session, you can log in as a different user, without having to restart the Console. To change the login identity:

1. In the Directory Server Console, select the **Tasks** tab.
2. Click **Log on to the Directory Server as a New User**.



3. A login dialog box appears.



Enter the full distinguished name of the entry with which to bind to the server. For example, to bind as user Barbara Jensen, enter her full DN in the login box:

```
cn=Barbara Jensen,ou=People,dc=example,dc=com
```

1.4.4. Viewing the Current Console Bind DN

To see the bind DN that is currently logged into the Directory Server Console, click the login icon in the lower-left corner of the window. The current bind DN appears next to the login icon.



Figure 1.1. Viewing the Bind DN

1.5. ENABLING LDAP

Inter-process communication (IPC) is a way for separate processes on a Unix machine or a network to communicate directly with each other. LDAP allows LDAP connections to run over IPC connections, meaning that LDAP operations can run over Unix sockets. These connections are much faster and more secure than regular LDAP connections.

LDAP is enabled through two configuration attributes:

- ***nsslapd-ldapi*** to enable LDAP for Directory Server
- ***nsslapd-ldapi*** to point to the Unix socket file

To enable LDAP:

1. Modify the ***nsslapd-ldapi*** to turn LDAP on and add the socket file attribute.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=config
changetype: modify
replace: nsslapd-ldapi
nsslapd-ldapi: on
-
add: nsslapd-ldapi
nsslapd-ldapi: /var/run/slapd-example.socket
```

- Restart the server to apply the new configuration.

```
service dirsrv restart example
```

1.6. CHANGING DIRECTORY SERVER PORT NUMBERS

The standard and secure LDAP port numbers used by Directory Server can be changed through the Directory Server Console or by changing the value of the *nsslapd-port* or *nsslapd-secureport* attribute under the *cn=config* entry in the *dse.ldif*.

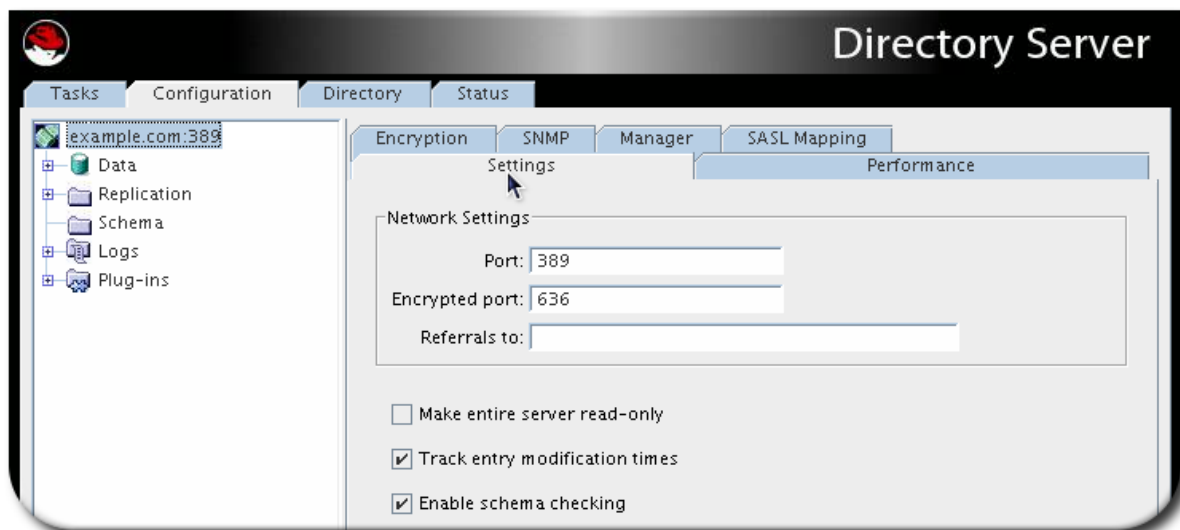


NOTE

Modifying the standard or secure port numbers for a Configuration Directory Server, which maintains the *o=NetScapeRoot* subtree, should be done through the Directory Server Console.

1.6.1. Changing Standard Port Numbers

- In the Directory Server Console, select the **Configuration** tab, and then select the top entry in the navigation tree in the left pane.
- Select the **Settings** tab in the right pane.



- Change the port numbers. The port number for the server to use for non-SSL communications in the **Port** field, with a default value of **389**.
- Click **Save**.
- The Console returns a warning, *You are about to change the port number for the Configuration Directory. This will affect all Administration Servers that use this directory and you'll need to update them with the new port number. Are you sure you want to change the port number?* Click **Yes**.
- Then a dialog appears, reading that the changes will not take effect until the server is restarted. Click **OK**.

**NOTE**

Do not restart the Directory Server at this point. If you do, you will not be able to make the necessary changes to the Admin Server through the Console.

7. Open the Admin Server Console.
8. In the **Configuration** tab, select the **Configuration DS** tab.



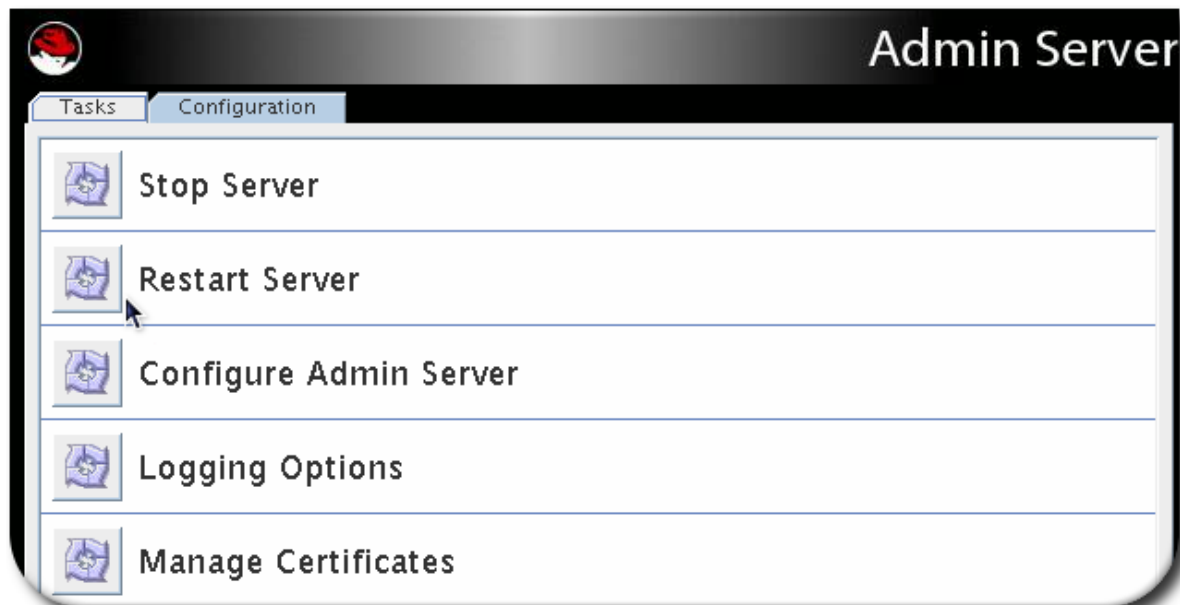
9. In the **LDAP Port** field, type in the new LDAP port number for your Directory Server instance.
10. Change the SELinux labels for the Directory Server ports so that the new port number is used in the Directory Server policies. By default, only port 389 is labeled. The process for labeling ports is covered in [Section 1.10.6, “Labeling SSL/TLS Ports”](#). For example:

```
/usr/sbin/semanage port -a -t ldap_port_t -p tcp 1389
```

**WARNING**

If the SELinux label is not reset, then the Directory Server will not be able to be restarted.

11. In the **Tasks** tab of the Directory Server Console, click **Restart Directory Server**. A dialog to confirm that you want to restart the server. Click **Yes**.



12. Open the **Configuration DS** tab of the Admin Server Console and select **Save**.

A dialog will appear, reading *The Directory Server setting has been modified. You must shutdown and restart your Admin Server and all the servers in the Server Group for the changes to take effect.* Click **OK**.

13. In the **Tasks** tab of the Admin Server Console, click **Restart Admin Server**. A dialog opens reading that the Admin Server has been successfully restarted. Click **Close**.



NOTE

You *must* close and reopen the Console before you can do anything else in the Console. Refresh may not update the Console, and, if you try to do anything, you will get a warning that reads *Unable to contact LDAP server*.

1.6.2. Changing SSL Port Numbers

Changing the configuration directory or user directory port or secure port numbers has the following repercussions:

- The Directory Server port number must also be updated in the Admin Server configuration.
- If there are other Directory Server instances that point to the configuration or user directory, update those servers to point to the new port number.

To modify the LDAPS port:

1. Make sure that the CA certificate used to issue the Directory Server instance's certificate is in the Admin Server certificate database. Importing CA certificates for the Admin Server is the same as the Directory Server process described in [Section 7.3.2, "Trusting the Certificate Authority"](#).
2. The secure port can be configured using the Directory Server Console, much like the process in [Section 1.6.1, "Changing Standard Port Numbers"](#) (only setting the value in the **Encrypted Port** field). However, in some circumstances, such as if there are multiple Directory Server instances on the same machine, where changing port numbers may not be possible through the Directory Server Console. It may be better to use `ldapmodify` to change the port number.

For example:

```
[root@server ~]# ldapmodify -x -h server.example.com -p 1389 -D
"cn=directory manager" -W
dn: cn=config
replace: nsslapd-securePort
nsslapd-securePort: 1636
```

3. Edit the corresponding port configuration for the Directory Server instance in the Admin Server configuration (**o=netscaperoot**).

First, search for the current configuration:

```
[root@server ~]# ldapsearch -x -h config-ds.example.com -p 389 -D
"cn=directory manager" -W -b "cn=slapd-ID,cn=389 Directory
Server,cn=Server
Group,cn=server.example.com,ou=example.com,o=NetscapeRoot" -s base "
(objectclass=*)"
nsSecureServerPort

dn: cn=slapd-ID,cn=389 Directory Server,cn=Server
Group,cn=server.example.com,ou=example.com,o=NetscapeRoot
nsSecureServerPort: 636
```

Then, edit the configuration:

```
[root@server ~]# ldapmodify -x -h config-ds.example.com -p 389 -D
"cn=directory manager" -WW

dn: cn=slapd-ID,cn=389 Directory Server,cn=Server
Group,cn=server.example.com,ou=example.com,o=NetscapeRoot
replace: nsSecureServerPort
nsSecureServerPort: 1636
```

4. Start the Directory Server Console for the instance and confirm that the new SSL port number is listed in the **Configuration** tab.
5. Optionally, select the **Use SSL in Console** check box.
6. Change the SELinux labels for the Directory Server ports so that the new port number is used in the Directory Server policies. By default, only port 389 is labeled. The process for labeling ports is covered in [Section 1.10.6, "Labeling SSL/TLS Ports"](#). For example:

```
/usr/sbin/semanage port -a -t ldap_port_t -p tcp 1636
```



WARNING

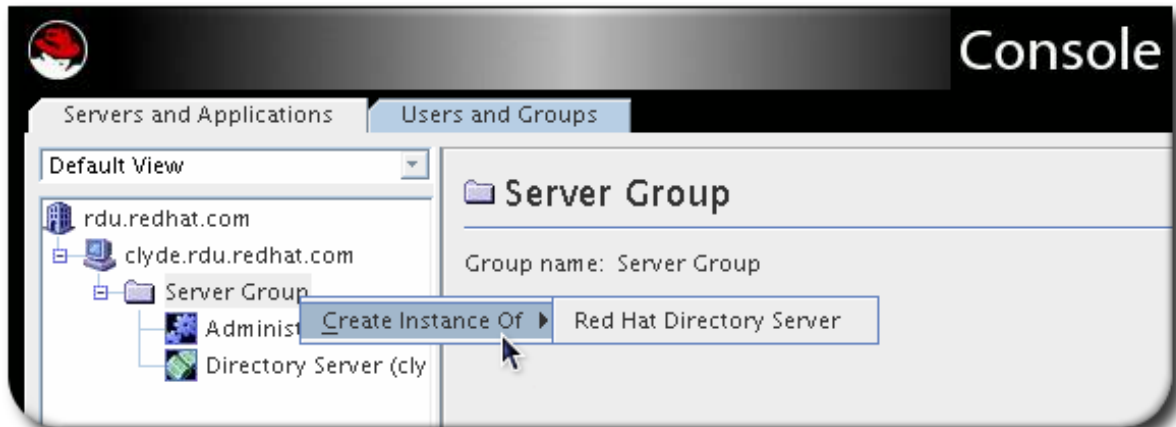
If the SELinux label is not reset, then the Directory Server will not be able to be restarted.

- Restart the Directory Server instance.

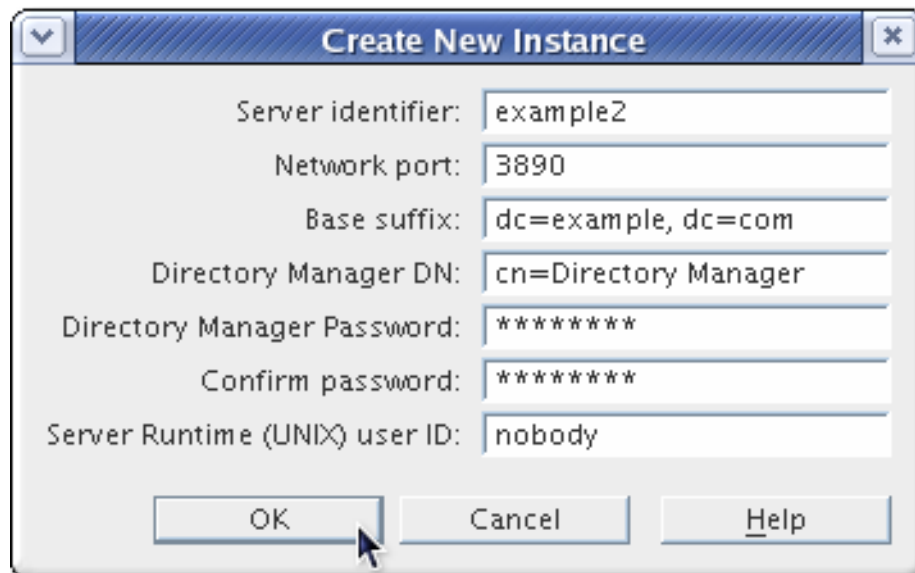
1.7. CREATING A NEW DIRECTORY SERVER INSTANCE

Additional instances can be created through the Directory Server Console or using the `setup-ds.pl` script. For information on using the `setup-ds.pl` script, see the *Directory Server Installation Guide*. To create an instance using the Directory Server Console:

- In the Red Hat Console window, select **Server Group** in the navigation tree, and then right-click.
- From the pop-up menu, select **Create Instance** and then **Directory Server**.



- Fill in the instance information.



- A unique name for the server. This name must only have alphanumeric characters, a dash (-), or an underscore (_).
- A port number for LDAP communications.
- The root suffix for the new Directory Server instance.
- A DN for the Directory Manager. This user has total access to every entry in the directory, without normal usage constraints (such as search timeouts).

- The password for the Directory Manager.
 - The user ID as which to run the Directory Server daemon.
4. Click **OK**.

A status box appears to confirm that the operation was successful. To dismiss it, click **OK**.

1.8. USING DIRECTORY SERVER PLUG-INS

Directory Server has a number of default plug-ins which configure core Directory Server functions, such as replication, classes of service, and even attribute syntaxes. Core plug-ins are enabled and completely configured by default.

Other default plug-ins extend the functionality of the Directory Server by providing consistent, but user-defined, behaviors, as with DNA, attribute uniqueness, and attribute linking. These plug-ins are available, but not enabled or configured by default.

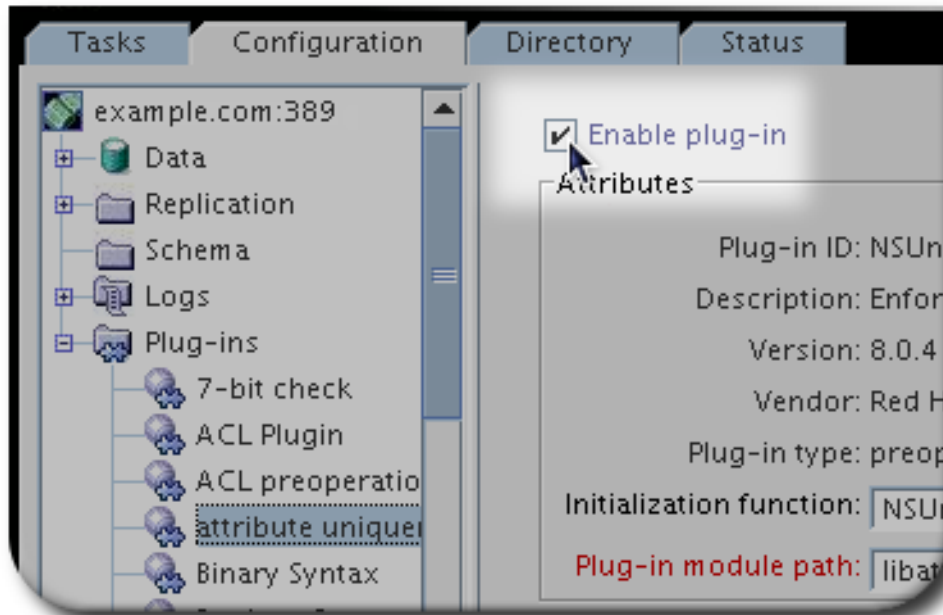
Using plug-ins also allows the Directory Server to be easily extended, so customers can write and deploy their own server plug-ins to perform whatever directory operations they need for their specific deployment.

The details of configuring and deploying plug-ins are covered in other guides (primarily the *Plug-in Programmer's Guide* and to some extent in the plug-in attribute reference in the *Configuration and Command-Line Tool Reference*). This section covers common administrative tasks for all plug-ins.

1.8.1. Enabling Plug-ins in the Directory Server Console

To enable and disable plug-ins over LDAP using the Directory Server Console:

1. In the Directory Server Console, select the **Configuration** tab.
2. Double-click the **Plugins** folder in the navigation tree.
3. Select the plug-in from the **Plugins** list.
4. To disable the plug-in, clear the **Enabled** check box. To enable the plug-in, check this check box.



5. Click **Save**.
6. Restart the Directory Server.

```
service dirsrv restart instance_name
```

NOTE

When a plug-in is disabled, all of the details about the plug-in — such as its version and its vendor — are not displayed in the Directory Server Console; all details fields show **NONE**.

Once a plug-in is enabled, those details will not be displayed in the Console until the Directory Server is restarted (loading the new plug-in configuration) and the Directory Server Console is refreshed.

1.8.2. Enabling Plug-ins in the Command Line

To disable or enable a plug-in through the command line, use the **ldapmodify** utility to edit the value of the **nsslapd-pluginEnabled** attribute. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=ACL Plugin,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
```

1.8.3. Setting the Plug-in Precedence

The plug-in precedence is the priority it has in the execution order of plug-ins. For pre- and post-operation plug-ins, this allows one plug-in to be executed and complete before the next plug-in is initiated, which lets the second plug-in take advantage of the first plug-in's results.

Plug-in precedence is configured in the ***nsslapd-pluginPrecedence*** attribute on the plug-in's configuration entry. This attribute has a value of 1 (highest priority) to 99 (lowest priority). If the attribute is not set, it has a default value of 50.

The ***nsslapd-pluginPrecedence*** attribute is set using the ***ldapmodify*** command. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=My Example Plugin,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginPrecedence
nsslapd-pluginPrecedence: 1
```



IMPORTANT

Don not set the plug-in precedence for the default Directory Server plug-ins unless told to do so by Red Hat support. The plug-in precedence attribute is primarily to govern the behavior of *custom* plug-ins, not to change the behavior of the core Directory Server plug-ins.

1.9. MANAGING CORE SERVER ATTRIBUTES

The Directory Server configuration itself is stored in the ***dse.ldif*** file, which contains the server configuration entries like ***cn=config***. The server entry itself is defined through a finite and strict set of attributes called *core server configuration attributes*. Although these attributes can be changed, no attributes can be added to the core server configuration and none can be deleted (except under very limited circumstances, as described in [Section 1.9.2, "Configuration Attributes Which Can Be Deleted"](#)).

This is described in more detail in the overview sections of the "Server Instance File Reference" chapter in the *Directory Server Configuration and Command-Line Tool Reference*.

This section provides details on how to check which core server attributes require that the server be restarted and how to check or change which core server configuration attributes can be deleted.

1.9.1. Configuration Attributes Requiring Server Restart

Some configuration attributes cannot be altered while the server is running. In these cases, for the changes to take effect, the server needs to be shut down and restarted. The modifications should be made either through the Directory Server Console or by manually editing the ***dse.ldif*** file when the ***dirsrv*** service is stopped.

Some of the attributes that require a server restart for any changes to take effect are listed below. This list is not exhaustive; to see a complete list, run ***ldapsearch*** and search for the ***nsslapd-requiresrestart*** attribute. For example:

```
ldapsearch -D "cn=directory manager" -W -p 389 -h server.example.com -b
"cn=config" -s sub -x "(objectclass=*)" | grep nsslapd-requiresrestart
```

<i>nsslapd-cachesize</i>	<i>nsslapd-certdir</i>	<i>nsslapd-dbcachesize</i>
<i>nsslapd-dbncache</i>	<i>nsslapd-plugin</i>	<i>nsslapd-changelogdir</i>

<i>nsslapd-changelogmaxage</i>	<i>nsslapd-changelogmaxentries</i>	<i>nsslapd-port</i>
<i>nsslapd-schemadir</i>	<i>nsslapd-saslpath</i>	<i>nsslapd-secureport</i>
<i>nsslapd-tmpdir</i>	<i>nsSSL2</i>	<i>nsSSL3</i>
<i>nsTLS1</i>	<i>nsSSLclientauth</i>	<i>nsSSLSessionTimeout</i>
<i>nsslapd-conntablesizesize</i>	<i>nsslapd-lockdir</i>	<i>nsslapd-maxdescriptors</i>
<i>nsslapd-reservedescriptors</i>	<i>nsslapd-listenhost</i>	<i>nsslapd-schema-ignore-trailing-spaces</i>
<i>nsslapd-securelistenhost</i>	<i>nsslapd-workingdir</i>	<i>nsslapd-return-exact-case</i>
<i>nsslapd-maxbersize</i> ^[a]	<i>nsslapd-allowed-to-delete-attrs</i>	
[a] Although this attribute requires a restart, it is not returned in the search.		

1.9.2. Configuration Attributes Which Can Be Deleted

Core server configuration attributes cannot be deleted, by default. All core configuration attributes are present, even if they are not written in the `dse.ldif` file, because they all have default values used by the server. Deleting any of those attributes is generally not allowed because the server requires that those attributes be present for it to run.

The *nsslapd-allowed-to-delete-attrs* parameter lists core configuration attributes which are *allowed* to be deleted from the configuration. Delete operations for those attributes will succeed.

The value of *nsslapd-allowed-to-delete-attrs* is a space-separated list of attribute names. By default, only two attributes are listed:

```
nsslapd-allowed-to-delete-attrs: nsslapd-listenhost nsslapd-securelistenhost
```

This can be changed using `ldapmodify` to add attributes to the list. Since this is a single-valued attribute, the *entire* list must be given in the modify statement; the modify operation overwrites the previous value, it does not append new values to it.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=config
changetype: modify
replace: nsslapd-allowed-to-delete-attrs
nsslapd-allowed-to-delete-attrs: nsslapd-listenhost nsslapd-securelistenhost nsslapd-rewrite-rfc1274
```

**WARNING**

Be extremely cautious about adding core server configuration attributes to the list of deletable attributes. Some attributes are critical for the server to operate, and deleting those attributes could cause the server not to run.

To return the list of attributes which can be deleted, use `grep`:

```
# egrep nsslapd-allowed-to-delete-attrs
/etc/dirsrv/slapd-instance_name/dse.ldif

nsslapd-allowed-to-delete-attrs: nsslapd-listenhost nsslapd-
securelistenhost nsslapd-rewrite-rfc1274
```

1.10. MANAGING SELINUX WITH THE DIRECTORY SERVER

SELinux is a security function in Linux that categorizes files, directories, ports, processes, users, and other objects on the server. Each object is placed in an appropriate security context to define how the object is allowed to behave on the server through its role, user, and security level. These roles for objects are grouped in domains, and SELinux rules define how the objects in one domain are allowed to interact with objects in another domain.

SELinux itself is much more complex to manage and implement than what is described here. This section is concerned only with giving the SELinux details for the Directory Server. Both the [Fedora project](#) and the [National Security Agency](#) have excellent resources for learning about SELinux.

**NOTE**

SELinux is a feature of Red Hat Enterprise Linux and, as such, is covered in the Red Hat Enterprise Linux *SELinux Guide* at https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Security-Enhanced_Linux/index.html.

1.10.1. SELinux Definitions for the Directory Server

SELinux has three different levels of enforcement: disabled (no SELinux), permissive (where the rules are lax), and enforcing (where all rules are strictly enforced). Red Hat Directory Server has defined SELinux policies that allow it to run as normal under strict SELinux enforcing mode, with a caveat. The Directory Server can run in different modes, one for normal operations and one for database operations like importing (Idif2db mode). The SELinux policies for the Directory Server only apply to normal mode.

By default, the Directory Server runs confined by SELinux policies.

The Directory Server processes are contained within the **dirsrv_t** domain. Ports used by the Directory Server instances are contained within the **ldap_port_t** domain.

[Table 1.4, “Summary of Directory Server SELinux Policies”](#) lists the security contexts and domains for the major components of the Directory Server.

Table 1.4. Summary of Directory Server SELinux Policies

File Path	Security Context	Description
dirsrv_t Domain		
/etc/dirsrv/*	dirsrv_config_t	Configuration files for the different instances.
/usr/sbin/ns-slapd	dirsrv_exec_t	The main server executable.
/usr/sbin/{start restart stop}-dirsrv	initrc_exec_t	The server start, restart, and stop scripts.
<div>/usr/lib/dirsrv/*</div> <div>/usr/lib64/dirsrv/*</div>	lib_t	The server and plug-in libraries.
/usr/share/dirsrv/*	dirsrv_share_t	The property files and templates for new instances.
/var/lib/dirsrv/*	dirsrv_var_lib_t	The default directories for database files, LDIF files, and backup files.
/var/lock/dirsrv/*	dirsrv_var_lock_t	Lock files.
/var/log/dirsrv/*	dirsrv_var_log_t	The server instance log files.
/var/run/dirsrv/*	dirsrv_var_run_t	The instance PID files and the SNMP statistics file.
ldap_t Domain		
Port 389 and 636 and any regular LDAP port configured for a Directory Server instance	ldap_port_t	The ports used by the Directory Server instances, including the default LDAP and LDAPS ports and whatever the configured LDAP port[a] for the Directory Server is
[a] Only the LDAP port is configured for the Directory Server when it is set up, so only this port is added to the SELinux configuration automatically. The LDAPS port must be added manually, as described in Section 1.10.6, “Labeling SSL/TLS Ports” .		

The Directory Server SELinux policies are configured when the server instance is set up (when **setup-ds-admin.pl** or **register-ds-admin.pl** are run). Each time a new instance is configured, the policies are updated with the appropriate information. These policies are automatically removed when the server instance is uninstalled.

1.10.2. SELinux Definitions for the SNMP Agent

The Directory Server runs an SNMP agent which can be used to configure traps and send alerts to an SNMP master agent, as described in [Chapter 16, Monitoring Directory Server Using SNMP](#). The SNMP sub-agent is contained within a separate domain, `dirsrv_snmp_t`.

The SNMP subagent runs as a process, `ldap-agent`. The process does not listen over any ports (the third-party SNMP master agent does), but the process does need to access some system files, such as PID and log files. The security context definitions for these files and process are listed in [Table 1.5, “Summary of Directory Server SELinux Policies”](#). All of these files are also covered by the Directory Server file contexts listed in [Table 1.4, “Summary of Directory Server SELinux Policies”](#).

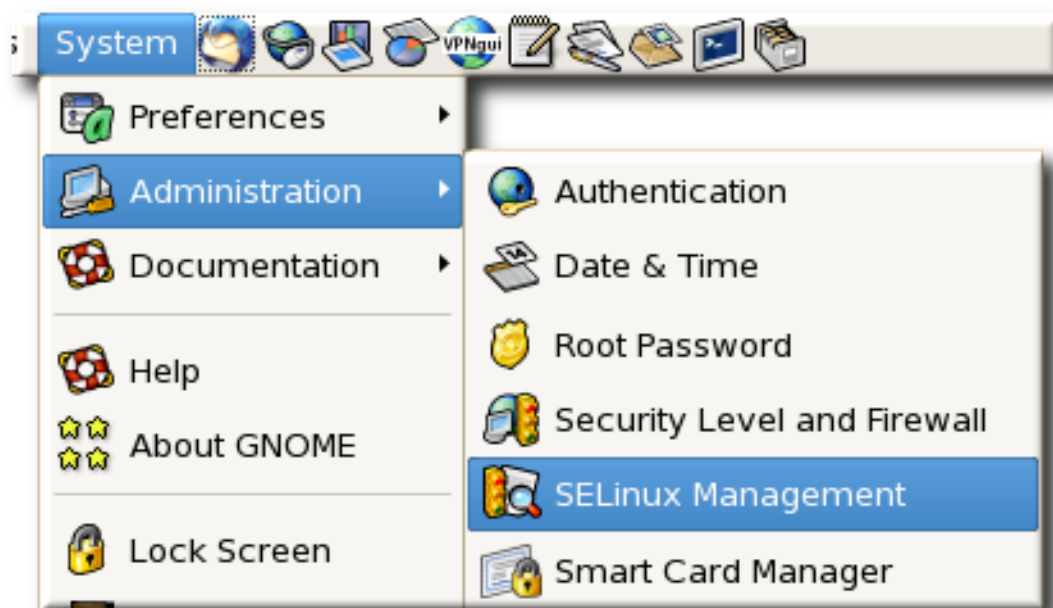
Table 1.5. Summary of Directory Server SELinux Policies

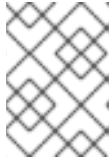
File Path	Security Context	Description
dirsrv_snmp_t Domain		
<code>/usr/sbin/ldap-agent-bin</code>	<code>dirsrv_snmp_exec_t</code>	The SNMP subagent daemon.
<code>/var/run/ldap-agent.pid</code>	<code>dirsrv_snmp_var_run_t</code>	The SNMP subagent PID file.
<code>/var/log/dirsrv/ldap-agent.log</code>	<code>dirsrv_snmp_var_log_t</code>	The SNMP subagent log file.

1.10.3. Viewing and Editing SELinux Policies for the Directory Server

The configured Directory Server and Admin Server policies can be viewed and edited using the SELinux Administration GUI. Much more information about editing SELinux policies and labels is in the [Red Hat Enterprise Linux Security-Enhanced Linux Guide](#).

1. Open the **Systems** menu.
2. Open the **Administration** menu, and select the **SELinux Management** item.



**NOTE**

You can launch the GUI from the command line using **system-config-selinux**.

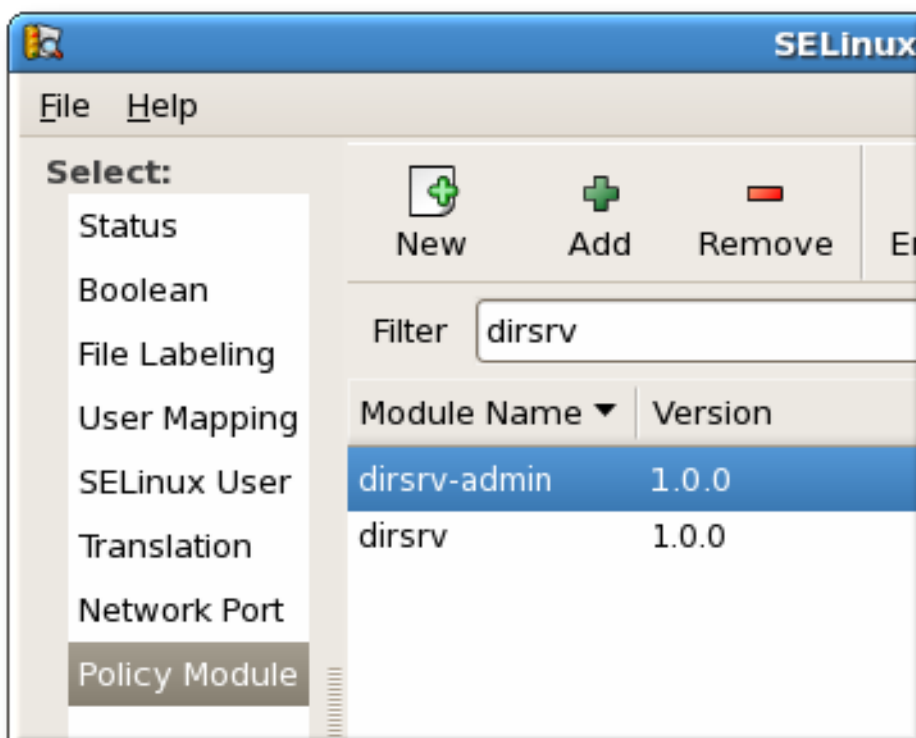
3. Open, add, or edit any file or port label or policy for Directory Server , as necessary.
4. After making any changes to the SELinux policies, run **restorecon** to load the changes to the labels or policies.

```
# restorecon -r -v [-f filename | directoryName]
```

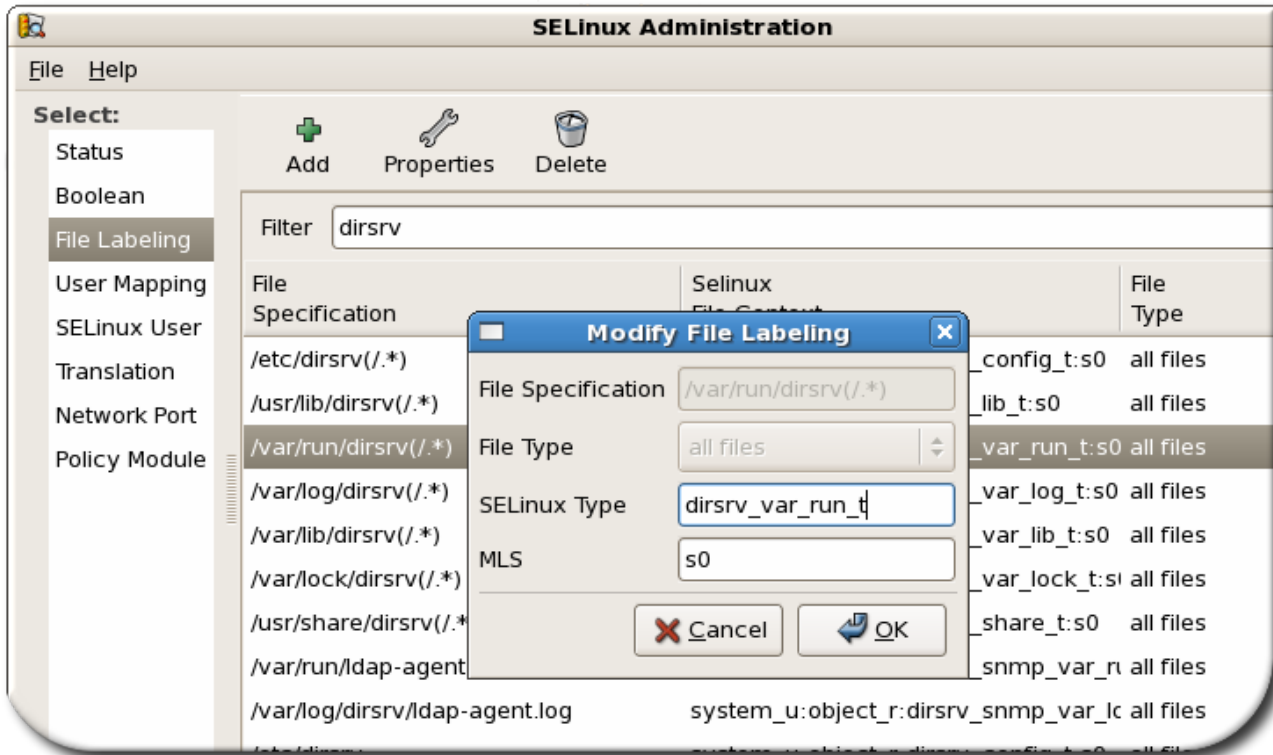
For example, if new policies were created for a custom LDIF directory:

```
# restorecon -r -v /myNewLdifDir
```

To check the version of the Directory Server SELinux policy installed, click the **Policy Module** link.



To view the policies set on the individual files and processes, click the **File Labeling** link. To view the policies for the port assignments for the server, click the **Network Port** link.



1.10.4. Starting the Directory Server Confined by SELinux

Three scripts control how the **ns-slapd** process transitions to the **dirsrv_t** domain when starting and stopping. All three of these scripts are in the **/usr/sbin/** directory:

- start-dirsrv
- stop-dirsrv
- restart-dirsrv

These scripts are run similar to the **service** commands used by Directory Server. A single instance can be specified using the instance name or the script can be run with no arguments and apply to all instances, as in [Section 1.3, “Starting and Stopping Servers”](#). For example:

```
/usr/sbin/start-dirsrv instance_name
```

Likewise, the SNMP subagent is started or stopped using the **service** command to run the **ldap-agent** process confined by SELinux policies. See [Section 16.3.2, “Starting the Subagent”](#) for more information.

```
service dirsrv-snmpp start
```

1.10.5. Managing SELinux Labels for Files Used by the Directory Server

There are a number of different files that the Directory Server has to access in normal operations, such as database, log, and index files. Many of these are configured in settings in **cn=config**, such as **nsslapd-dbdir**, **nsslapd-rundir**, and **nsslapd-ldapifilepath**. As long as these directory locations are left with their default settings, the confined **ns-slapd** process can access them just fine. However, if these file locations are moved, then the SELinux labels must be updated for the new locations so that the Directory Server process is allowed to access them.

**NOTE**

Do not change the default locations for Directory Server files and directories — such as the databases, run file, or LDAP configuration file — so that the SELinux policies do not have to be updated.

Most common files used by the Directory Server are covered by the SELinux policies by default. However, for some operations, the Directory Server must access *external* files, meaning files not directly created from Directory Server templates and maintained by the server. For example:

- *LDIF files for import and export.* If the import or export LDIF files are created in the default LDIF directory, `/var/lib/dirsrv/slapped-instance_name/ldif`, then the files will automatically be covered by the security context. If these are in a non-standard location, then the file labels must be changed for the Directory Server to access them.

These SELinux labels apply only to the LDIF *files* used for import/export operations. These contexts do not cover import or export operations, which are database operations and outside the purview of SELinux.

**IMPORTANT**

If you copy a file into the LDIF directory, then the command automatically relabels the copied files and everything is fine. If, however, a file is moved into the LDIF directory (`mv`), then it retains its original SELinux labels and will not be recognized by the `ns-slapd` process.

- *Custom plug-ins.* The SELinux file restrictions assume that any plug-in files used by the server are located in the default plug-in directory, `/usr/lib[64]/dirsrv/plugins` on Red Hat Enterprise Linux 6 (64-bit). Any `.so` files for custom plug-ins must be in that directory for the server to load and use them.

If the plug-in files must be stored in a non-default location for some reason, then add appropriate SELinux rules to allow the server to access the files. This is in [Section 1.10.3, “Viewing and Editing SELinux Policies for the Directory Server”](#) or using `semanage`.

- *SASL/GSS-API keytabs.* The Directory Server must be able to access the host keytab and `krb5.conf` configuration file for GSS-API authentication in SASL. (The host keytab is set in the `KRB5_KTNAME` directive in the `/etc/sysconfig/dirsrv` file.) For these files to be properly labeled in SELinux in the `dirsrv_config_t` context, they must be in the `/etc/dirsrv/` directory.

Only the host keytab and `krb5.conf` file must be in `/etc`. The user key tabs can still be in any directory.

Although import/export operations and SASL configurations are the most common situations when the Directory Server will access an external file, be sure to consider file labeling any time the Directory Server needs to access a file.

File labels can be added using the SELinux administrative interface ([Section 1.10.3, “Viewing and Editing SELinux Policies for the Directory Server”](#)) or using the `semanage` script. For details, see the `semanage(8)` man page.

1.10.6. Labeling SSL/TLS Ports

When the Directory Server is first set up, the given LDAP port is labeled for SELinux (the default is port 389). However, SSL/TLS is set up separately, after the Directory Server is already configured, so the LDAPS port for the Directory Server is not automatically labeled.

The default LDAP and LDAPS ports, 389 and 636, respectively, are already labeled as part of the policies in Red Hat Enterprise Linux. Any other LDAP port is added to those policies when the server is set up. If the Directory Server uses a secure port other than the defaults for its SSL/TLS connections, however, then an administrator must label the port manually. This can be done in the SELinux administrative interface shown in [Section 1.10.3, “Viewing and Editing SELinux Policies for the Directory Server”](#). It can also be done easily using the **semanage** script.

Use the **port** subcommand, the **-t** option to identify the security context, and the **-p** option to identify the port. The **-a** option adds the port label. For example:

```
/usr/sbin/semanage port -a -t ldap_port_t -p tcp 1636
```

To delete a port label, use the **-d** option. For example:

```
/usr/sbin/semanage port -d -t ldap_port_t -p tcp 1636
```

CHAPTER 2. CONFIGURING DIRECTORY DATABASES

The directory is made up of databases, and the directory tree is distributed across the databases. This chapter describes how to create *suffixes*, the branch points for the directory tree, and how to create the databases associated with each suffix. This chapter also describes how to create database links to reference databases on remote servers and how to use referrals to point clients to external sources of directory data.

For a discussion of concepts about distributing directory data, see the *Directory Server Deployment Guide*.

2.1. CREATING AND MAINTAINING SUFFIXES

Different pieces of the directory tree can be stored in different databases, and then these databases can be distributed across multiple servers. The directory tree contains branch points called *nodes*. These nodes may be associated with databases. A suffix is a node of the directory tree associated with a particular database. For example, a simple directory tree might appear as illustrated in [Figure 2.1, “A Directory Tree with One Root Suffix”](#).

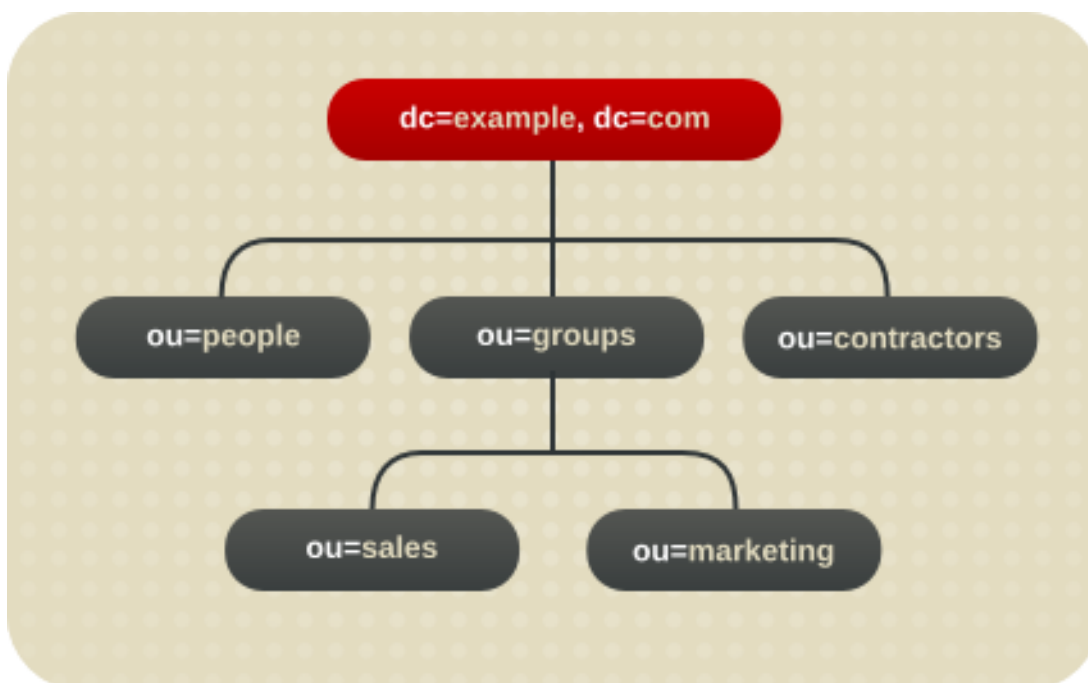


Figure 2.1. A Directory Tree with One Root Suffix

The **ou=people** suffix and all the entries and nodes below it might be stored in one database, the **ou=groups** suffix on another database, and the **ou=contractors** suffix on yet another database.

- [Section 2.1.1, “Creating Suffixes”](#)
- [Section 2.1.2, “Maintaining Suffixes”](#)

2.1.1. Creating Suffixes

Both root and sub suffixes can be created to organize the contents of the directory tree. A *root* suffix is the parent of a sub suffix. It can be part of a larger tree designed for the Directory Server. A *sub suffix* is a branch underneath a root suffix. The data for root and sub suffixes are contained by databases.

A directory might contain more than one root suffix. For example, an ISP might host several websites,

one for **example.com** and one for **redhat.com**. The ISP would create two root suffixes, one corresponding to the **dc=example, dc=com** naming context and one corresponding to the **dc=redhat, dc=com** naming context, as shown in [Figure 2.2, “A Directory Tree with Two Root Suffixes”](#).

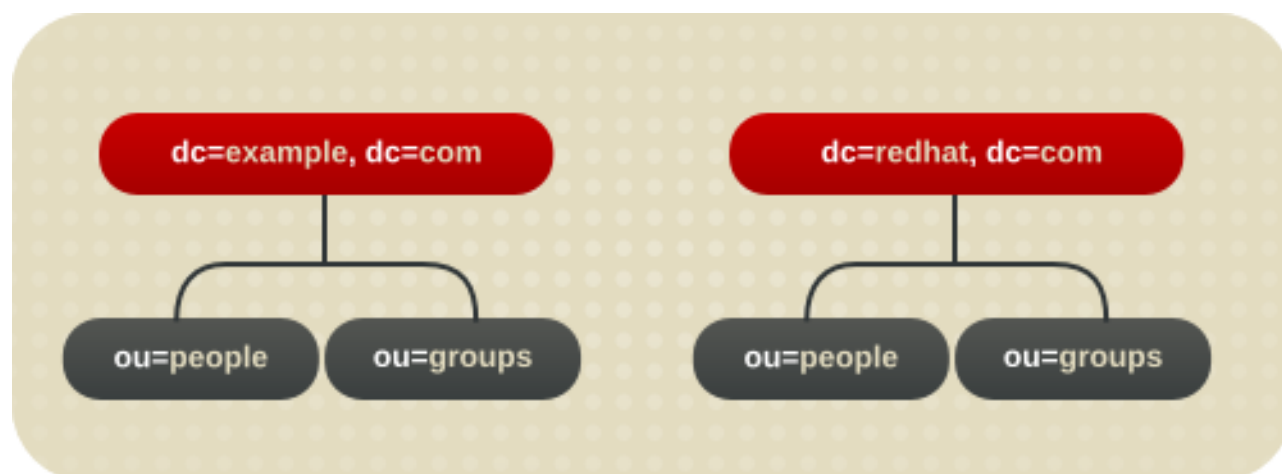


Figure 2.2. A Directory Tree with Two Root Suffixes

It is also possible to create root suffixes to exclude portions of the directory tree from search operations. For example, Example Corporation wants to exclude their European office from a search on the general Example Corporation directory. To do this, they create two root suffixes. One root suffix corresponds to the general Example Corporation directory tree, **dc=example, dc=com**, and one root suffix corresponds to the European branch of their directory tree, **l=europe, dc=example, dc=com**. From a client application's perspective, the directory tree looks as illustrated in [Figure 2.3, “A Directory Tree with a Root Suffix Off Limits to Search Operations”](#).

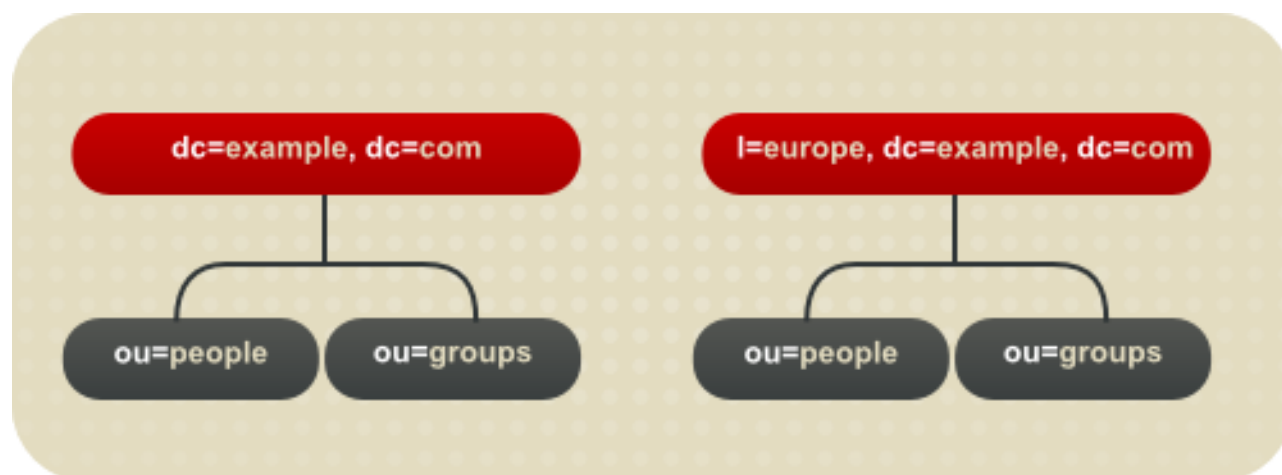


Figure 2.3. A Directory Tree with a Root Suffix Off Limits to Search Operations

Searches performed by client applications on the **dc=example, dc=com** branch of Example Corporation's directory will not return entries from the **l=europe, dc=example, dc=com** branch of the directory, as it is a separate root suffix.

If Example Corporation decides to include the entries in the European branch of their directory tree in general searches, they make the European branch a sub suffix of the general branch. To do this, they create a root suffix for Example Corporation, **dc=example, dc=com**, and then create a sub suffix beneath it for their European directory entries, **l=europe, dc=example, dc=com**. From a client application's perspective, the directory tree appears as illustrated in [Figure 2.4, “A Directory Tree with a Sub Suffix”](#).

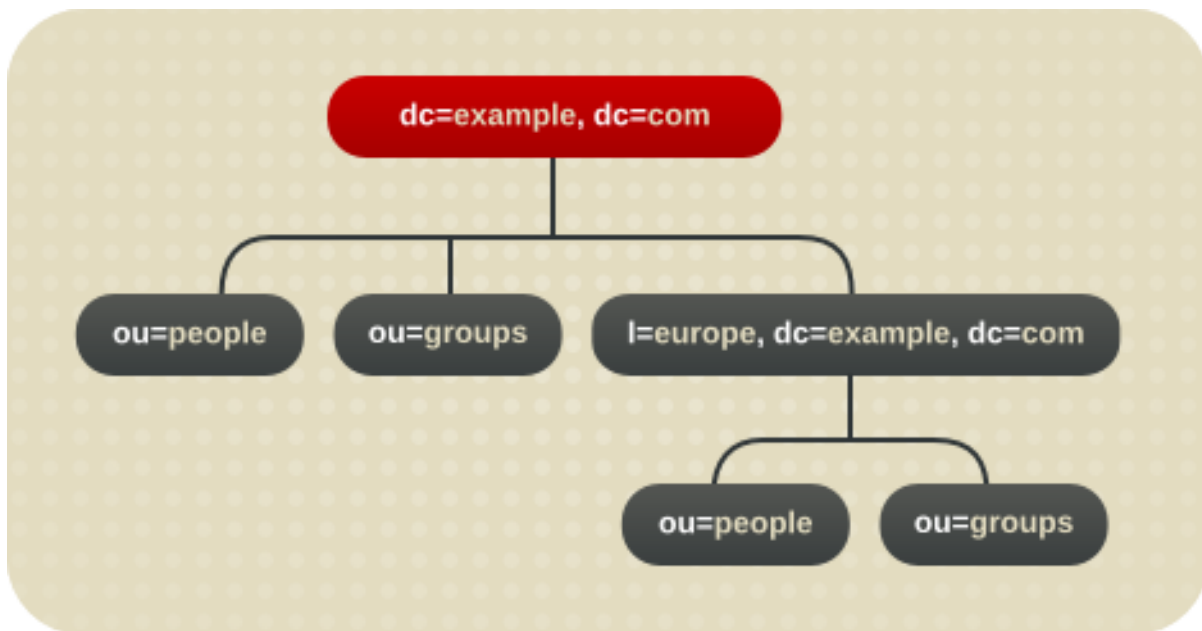


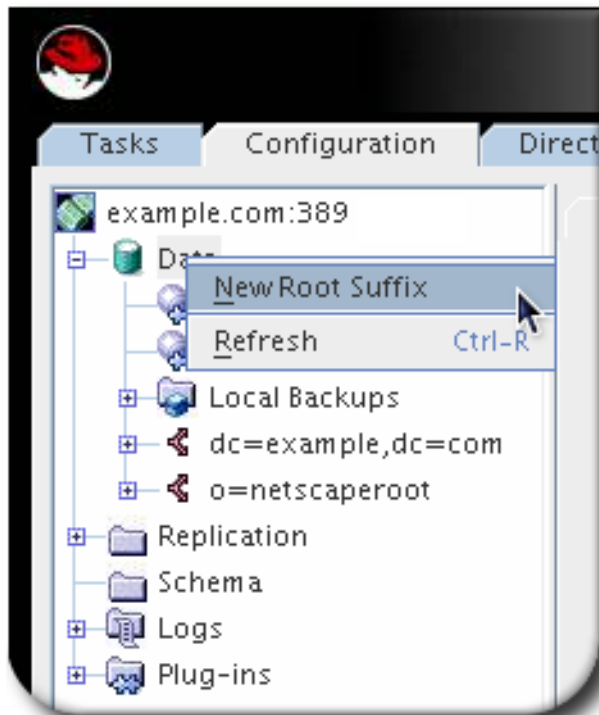
Figure 2.4. A Directory Tree with a Sub Suffix

This section describes creating root and sub suffixes for the directory using either the Directory Server Console or the command line.

- [Section 2.1.1.1, “Creating a New Root Suffix Using the Console”](#)
- [Section 2.1.1.2, “Creating a New Sub Suffix Using the Console”](#)
- [Section 2.1.1.3, “Creating Root and Sub Suffixes from the Command Line”](#)

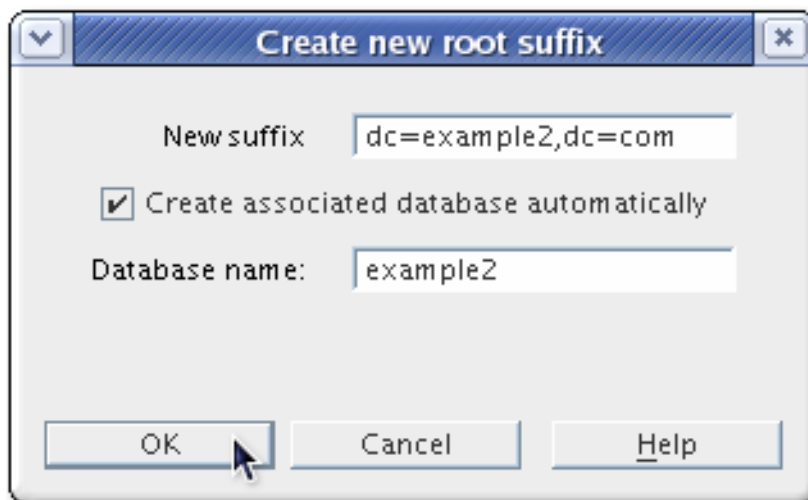
2.1.1.1. Creating a New Root Suffix Using the Console

1. In the Directory Server Console, select the **Configuration** tab.
2. Right-click **Data** in the left navigation pane, and select **New Root Suffix** from the pop-up menu.



3. Enter a unique suffix in the **New suffix** field.

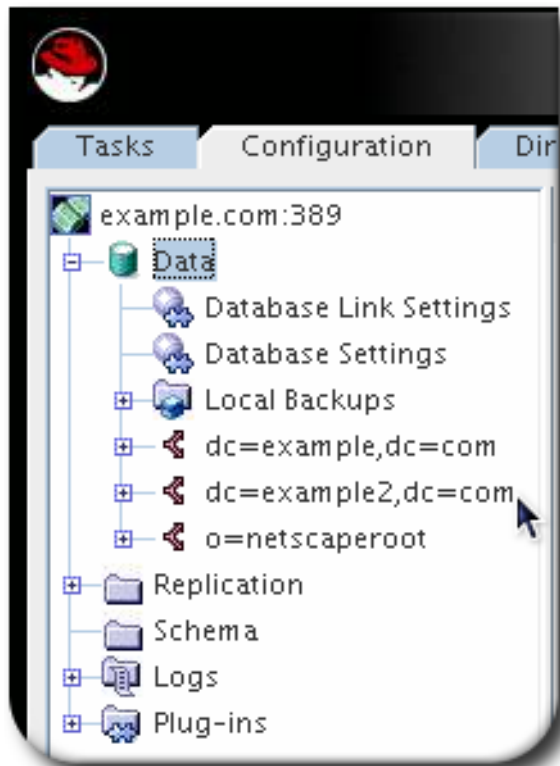
The suffix must be named with **dc** naming conventions, such as **dc=example,dc=com**.



4. Select the **Create associated database automatically** to create a database at the same time as the new root suffix, and enter a unique name for the new database in the **Database name** field, such as **example2**. The name can be a combination of alphanumeric characters, dashes (-), and underscores (_). No other characters are allowed.

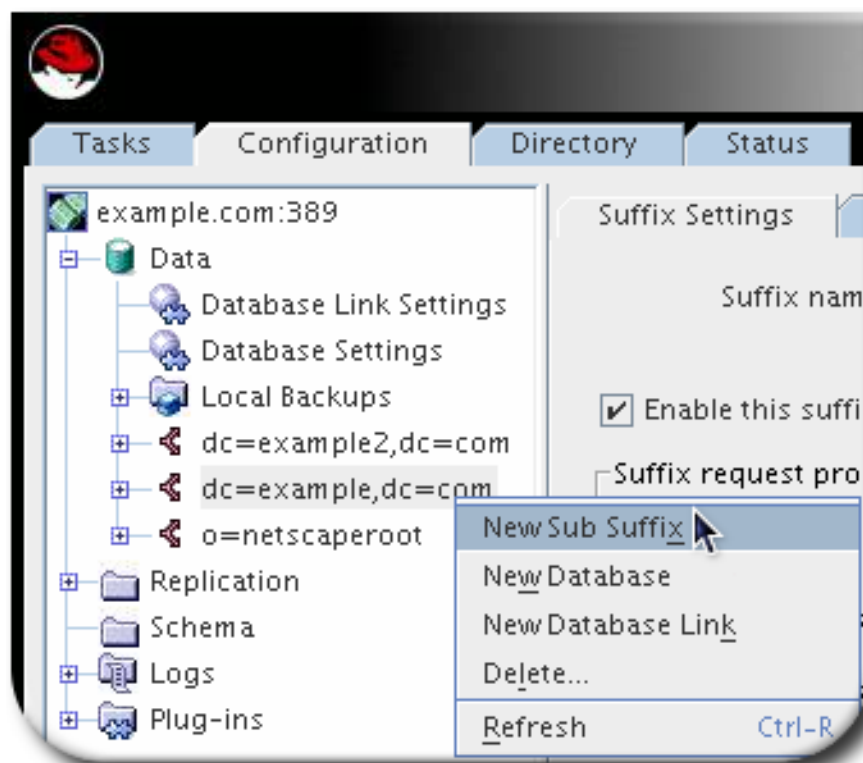
Deselect the check box to create a database for the new root suffix later. This option specifies a directory where the database will be created. The new root suffix will be disabled until a database is created.

The new root suffix is listed under the **Data** folder.



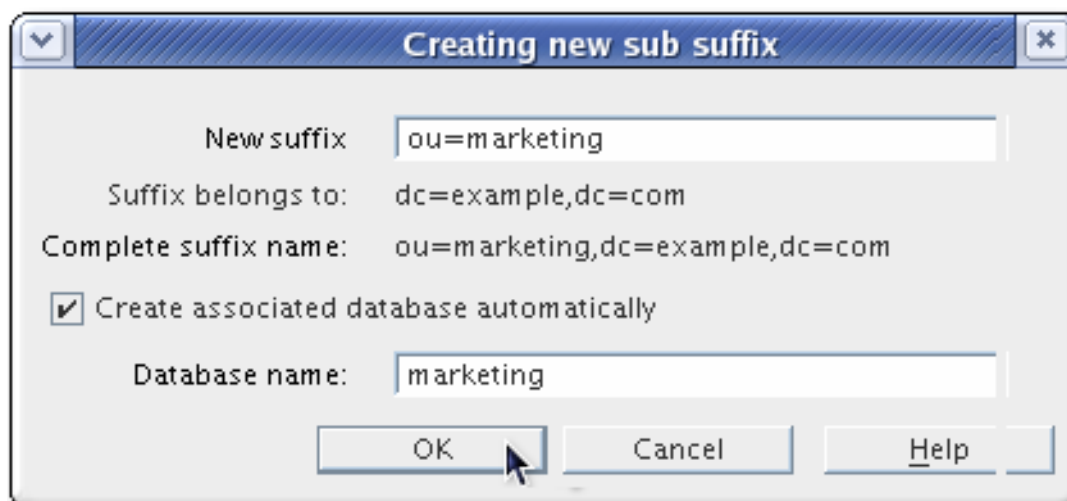
2.1.1.2. Creating a New Sub Suffix Using the Console

1. In the Directory Server Console, select the **Configuration** tab.
2. Under the **Data** in the left navigation pane, select the suffix under which to add a new sub suffix. Right-click the suffix, and select **New Sub Suffix** from the pop-up menu.



The **Create new sub suffix** dialog box is displayed.

3. Enter a unique suffix name in the **New suffix** field. The suffix must be named in line with **dc** naming conventions, such as **ou=groups**.

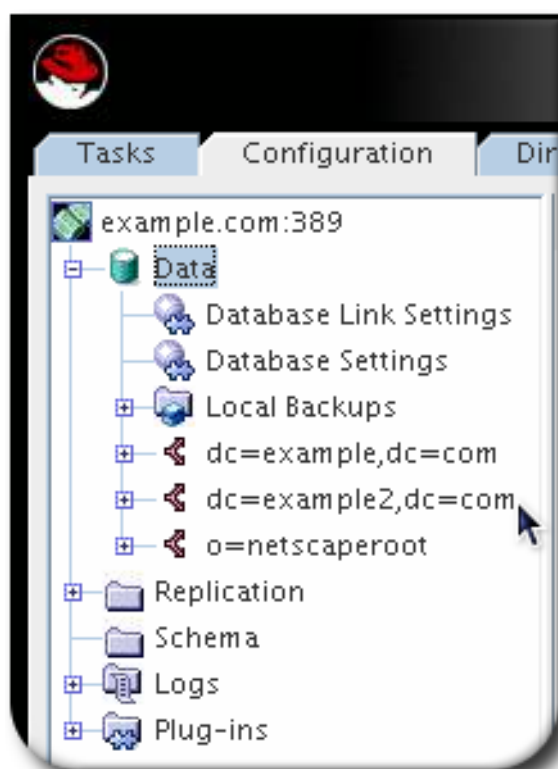


The root suffix is automatically added to the name. For example, if the sub suffix **ou=groups** is created under the **dc=example,dc=com** suffix, the Console automatically names it **ou=groups,dc=example,dc=com**.

4. Select the **Create associated database automatically** check box to create a database at the same time as the new sub suffix, and enter a unique name for the new database in the **Database name** field, such as **example2**. The name can be a combination of alphanumeric characters, dashes (-), and underscores (_). No other characters are allowed.

If the check box is not selected, then the database for the new sub suffix must be created later. The new sub suffix is disabled until a database is created.

The suffix appears automatically under its root suffix in the **Data** tree in the left navigation pane.



2.1.1.3. Creating Root and Sub Suffixes from the Command Line

Use the **ldapmodify** command-line utility to add new suffixes to the directory configuration file. The suffix configuration information is stored in the **cn=mapping tree,cn=config** entry.



NOTE

Avoid creating entries under the **cn=config** entry in the **dse.ldif** file. The **cn=config** entry in the simple, flat **dse.ldif** configuration file is not stored in the same highly scalable database as regular entries. As a result, if many entries, particularly entries that are likely to be updated frequently, are stored under **cn=config**, performance will suffer.

1. Add a new root suffix to the configuration file using the **ldapmodify** utility.

Example 2.1. Example Root Suffix Entry

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

```
dn: cn=dc=example\,dc=com,cn=mapping tree,cn=config
changetype: add
objectclass: top
objectclass: extensibleObject
objectclass: nsMappingTree
nsslapd-state: backend
nsslapd-backend: UserData
cn: dc=example,dc=com
```

2. Create a sub suffix for groups under this root suffix using **ldapmodify** to add the sub suffix entry:

```
dn: cn=ou=groups\,dc=example\,dc=com,cn=mapping tree,cn=config
changetype: add
objectclass: top
objectclass: extensibleObject
objectclass: nsMappingTree
nsslapd-state: backend
nsslapd-backend: GroupData
nsslapd-parent-suffix: dc=example,dc=com
cn: ou=groups,dc=example,dc=com
```



NOTE

To maintain suffixes using the Directory Server Console, respect the same spacing used to name the root and sub suffixes in the command line. For example, if a root suffix is named **ou=groups ,dc=example,dc=com**, with two spaces after **groups**, any sub suffixes created under this root will need to specify two spaces after **ou=groups**, as well.

The following table describes the attributes used to configure a suffix entry:

Table 2.1. Suffix Attributes

Attribute Name	Value
<i>dn</i>	Defines the DN for the suffix. The DN is contained in quotes. The value entered takes the form cn="dc=example,dc=com",cn=mapping tree,cn=config . This attribute is required.
<i>cn</i>	Defines the relative DN (RDN) of the entry. This attribute is required.
<i>objectclass</i>	Tells the server that the entry is root or sub suffix entry. It always takes the value nsMappingTree . This attribute is required.
<i>nsslapd-state</i>	<p>Determines how the suffix handles operations. This attribute takes the following values:</p> <ul style="list-style-type: none"> • backend: The back end (database) is used to process all operations. • disabled: The database is not available for processing operations. The server returns a <i>No such search object</i> error in response to requests made by client applications. • referral: A referral is returned for requests made to this suffix. • referral on update: The database is used for all operations except update requests, which receive a referral. <p>The default value is disabled.</p>
<i>nsslapd-referral</i>	Defines the LDAP URL of the referral to be returned by the suffix. This attribute can be multi-valued, with one referral per value. This attribute is required when the value of the <i>nsslapd-state</i> attribute is referral or referral on update .
<i>nsslapd-backend</i>	Gives the name of the database or database link used to process requests. This attribute can be multi-valued, with one database or database link per value. See Section 2.3, "Creating and Maintaining Database Links" for more information about database links. This attribute is required when the value of the <i>nsslapd-state</i> attribute is set to backend or referral on update .

Attribute Name	Value
<i>nsslapd-distribution-plugin</i>	Specifies the shared library to be used with the custom distribution function. This attribute is required only when more than one database is specified in the <i>nsslapd-backend</i> attribute. See Section 2.2, “Creating and Maintaining Databases” for more information about the custom distribution function.
<i>nsslapd-distribution-funct</i>	Specifies the name of the custom distribution function. This attribute is required only when more than one database is specified in the <i>nsslapd-backend</i> attribute. See Section 2.2, “Creating and Maintaining Databases” for more information about the custom distribution function.
<i>nsslapd-parent-suffix</i>	Provides the DN of the parent entry for a sub suffix. By default, this attribute is not present, which means that the suffix is regarded as a root suffix. For example, to create a sub suffix names o=sales,dc=example,dc=com under the root suffix dc=example,dc=com , add nsslapd-parent-suffix: dc=example,dc=com to the sub suffix.

2.1.2. Maintaining Suffixes

- [Section 2.1.2.1, “Viewing the Default Naming Context”](#)
- [Section 2.1.2.2, “Disabling a Suffix”](#)
- [Section 2.1.2.3, “Deleting a Suffix”](#)

2.1.2.1. Viewing the Default Naming Context

A naming context is analogous to the suffix; it is the root structure for naming directory entries. There can be multiple naming contexts, depending on the directory and data structure; for example, a standard Directory Server configuration has a user suffix such as **dc=example,dc=com**, a configuration suffix in **cn=config**, and an administrative configuration suffix in **o=netscaperoot**.

Many directory trees have multiple naming contexts to be used with different types of entries or with logical data divisions. Clients which access the Directory Server may not know what naming context they need to use. The Directory Server has a server configuration attribute which signals to clients what the default naming context is, if they have no other naming context configuration known to them.

The default naming context is set in the ***nsslapd-defaultnamingcontext*** attribute in **cn=config**. This value is propagated over to the root DSE and can be queried by clients anonymously by checking the ***defaultnamingcontext*** attribute in the root DSE.

For example:

```
[root@server ~]# ldapsearch -p 389 -h server.example.com -x -b "" -s base
| egrep namingcontext
```

```
namingContexts: dc=example,dc=com  
namingContexts: dc=example,dc=net  
namingContexts: dc=redhat,dc=com  
defaultnamingcontext: dc=example,dc=com
```



IMPORTANT

By default, the ***nsslapd-defaultnamingcontext*** attribute is included in the list of attributes which *can* be deleted, in the ***nsslapd-allowed-to-delete-attrs*** attribute. This allows the current default suffix to be deleted and then updates the server configuration accordingly.

If for some reason the ***nsslapd-defaultnamingcontext*** attribute is removed from the list of configuration attributes which can be deleted, then no changes to that attribute are preserved. If the default suffix is deleted, that change cannot be propagated to the server configuration. This means that the ***nsslapd-defaultnamingcontext*** attribute retains the old information instead of being blank (removed), which is the correct and current configuration.

To maintain configuration consistency, do not remove the ***nsslapd-defaultnamingcontext*** attribute from the ***nsslapd-allowed-to-delete-attrs*** list.

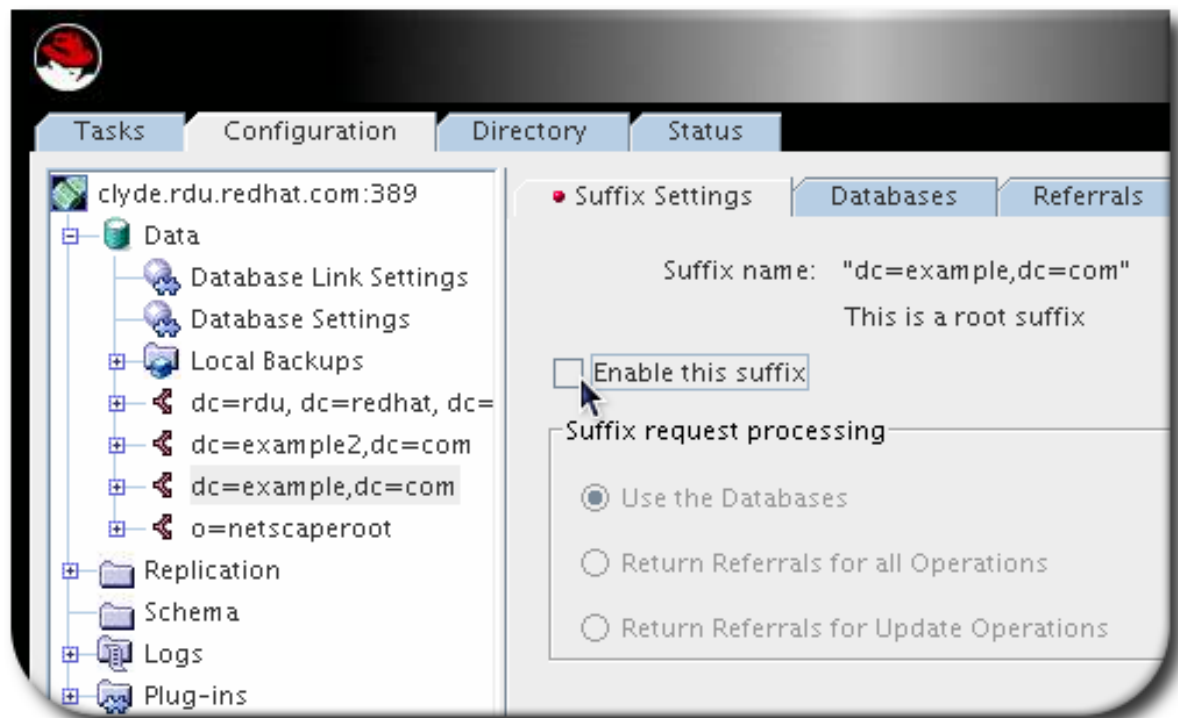
2.1.2.2. Disabling a Suffix

Sometimes, a database may need taken down for maintenance, but the data the database contains are not replicated. Rather than returning a referral, disable the suffix responsible for the database.

Once a suffix is disabled, the contents of the database related to the suffix are invisible to client applications when they perform LDAP operations such as search, add, and modify.

To disable a suffix:

1. In the Directory Server Console, select the **Configuration** tab.
2. Under **Data** in the left navigation pane, click the suffix to disable.
3. Click the **Suffix Setting** tab, and deselect the **Enable this suffix** check box.



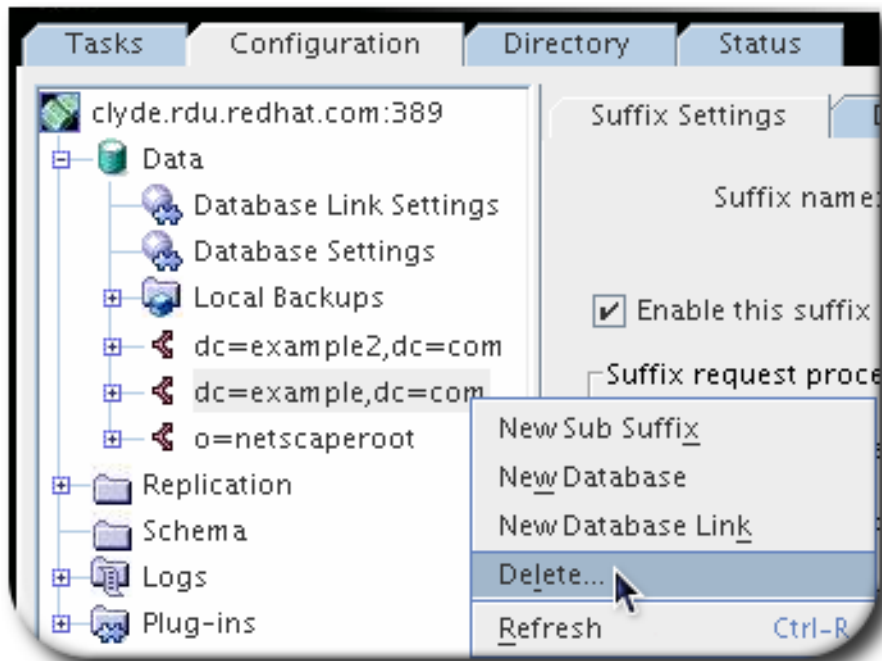
2.1.2.3. Deleting a Suffix



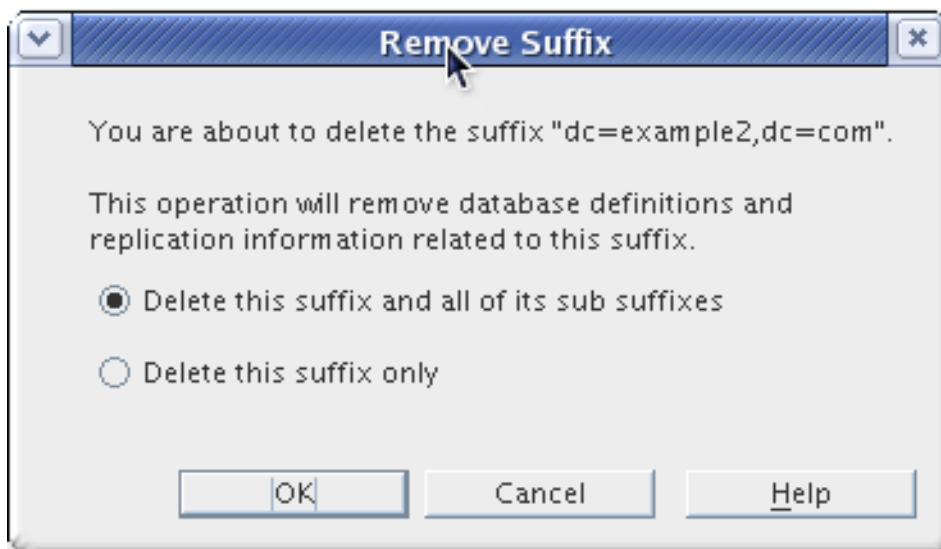
WARNING

Deleting a suffix also deletes all database entries and replication information associated with that suffix.

1. In the Directory Server Console, select the **Configuration** tab.
2. Under **Data** in the left navigation pane, select the suffix to delete.
3. Right-click the suffix, and select **Delete** from the menu.



4. Select either **Delete this suffix and all of its sub suffixes** or **Delete this suffix only**.



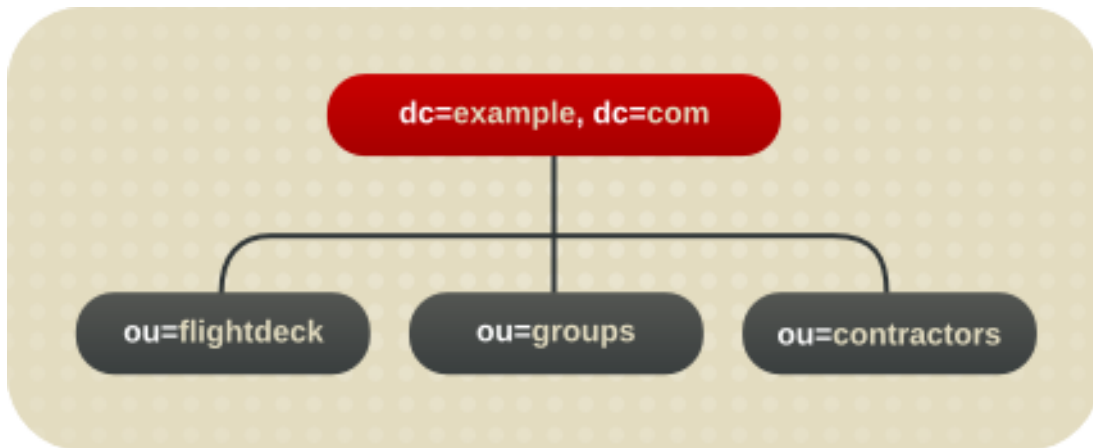
2.2. CREATING AND MAINTAINING DATABASES

After creating suffixes to organizing the directory data, create databases to contain that directory data. Databases are used to store directory data.

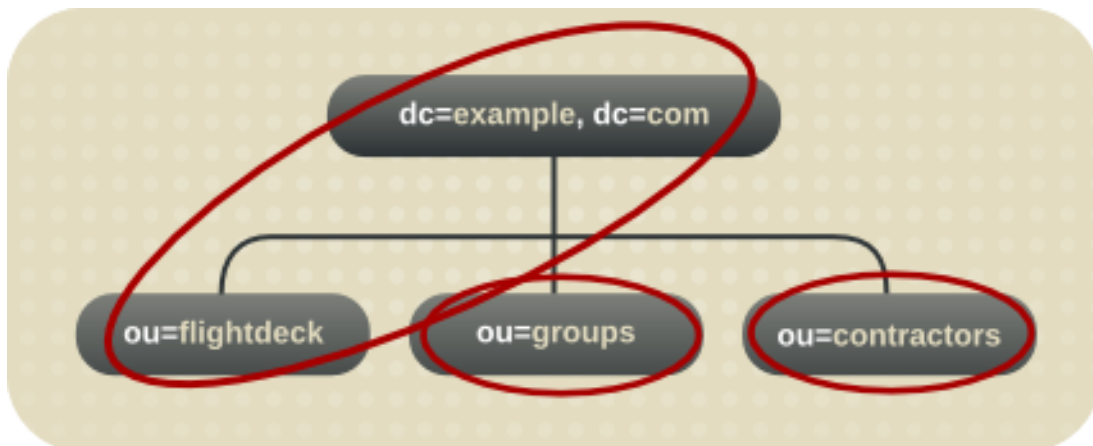
2.2.1. Creating Databases

The directory tree can be distributed over multiple Directory Server databases. There are two ways to distribute data across multiple databases:

- One database per suffix. The data for each suffix is contained in a separate database.



Three databases are added to store the data contained in separate suffixes.



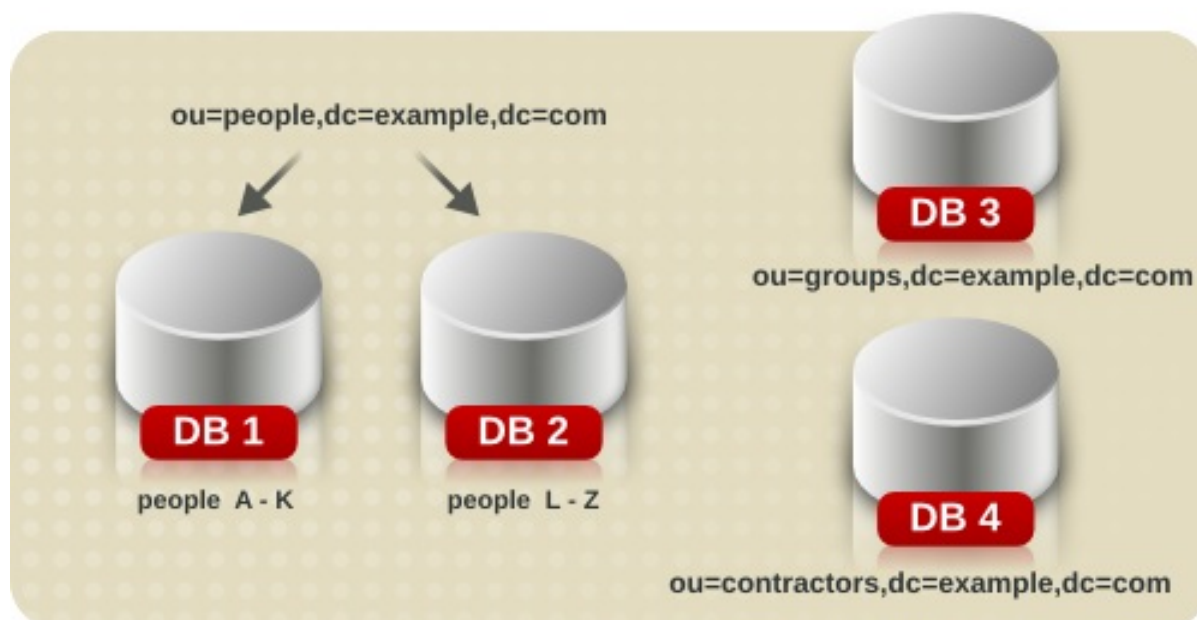
This division of the tree corresponds to three databases.



Database one contains the data for **ou=people** plus the data for **dc=example, dc=com**, so that clients can conduct searches based at **dc=example, dc=com**. Database two contains the data for **ou=groups**, and database three contains the data for **ou=contractors**.

- Multiple databases for one suffix.

Suppose the number of entries in the **ou=people** branch of the directory tree is so large that two databases are needed to store them. In this case, the data contained by **ou=people** could be distributed across two databases.

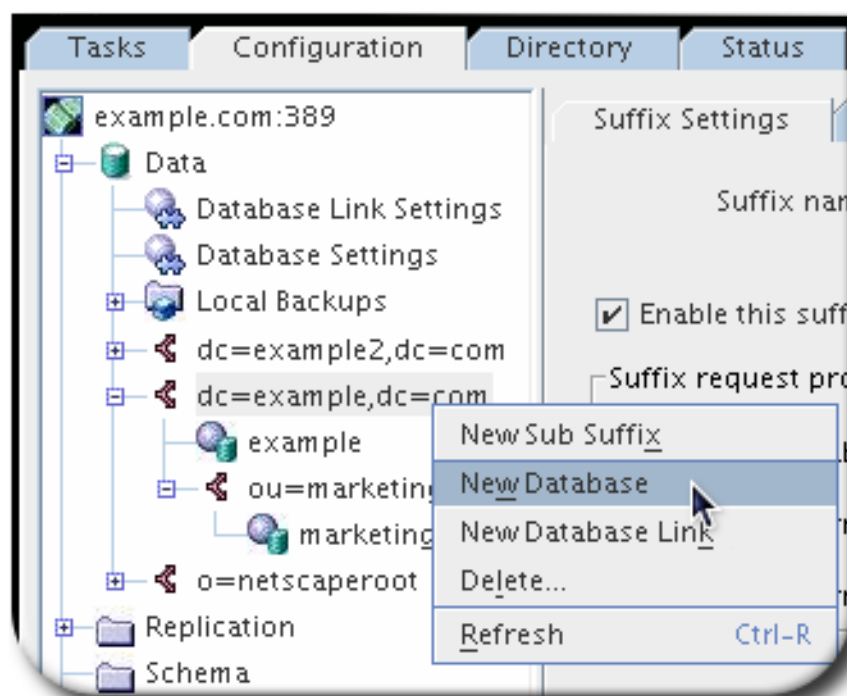


DB1 contains people with names from A-K, and **DB2** contains people with names from L-Z. **DB3** contains the `ou=groups` data, and **DB4** contains the `ou=contractors` data.

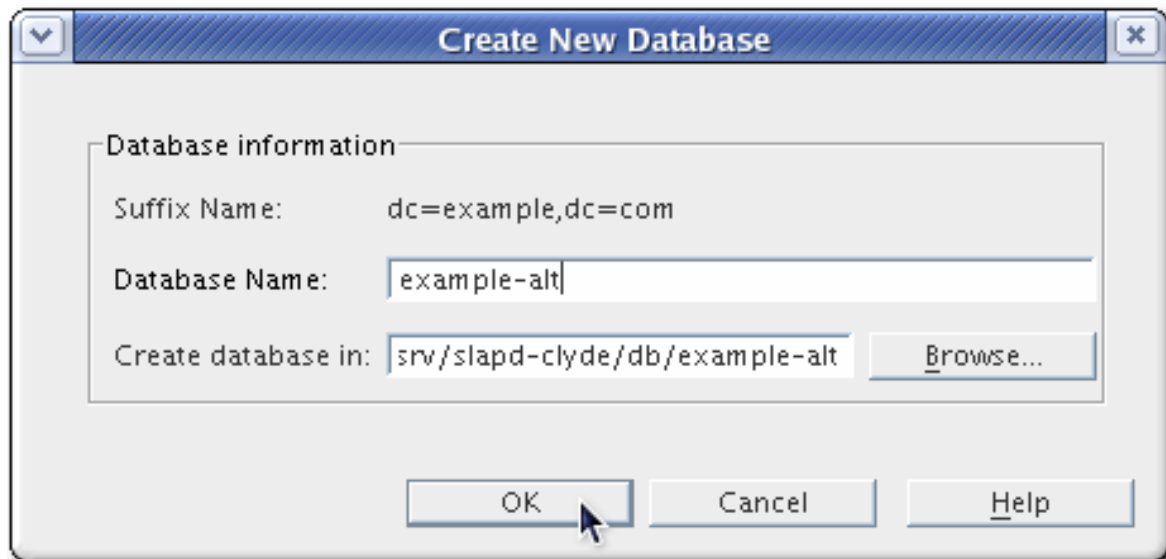
Custom distribution plug-in distributes data from a single suffix across multiple databases. Contact Red Hat Professional Services for information on how to create distribution logic for Directory Server.

2.2.1.1. Creating a New Database for an Existing Suffix Using the Console

1. In the Directory Server Console, select the **Configuration** tab.
2. In the left pane, expand **Data**, then click the suffix to which to add the new database.
3. Right-click the suffix, and select **New Database** from the pop-up menu.



4. Enter a unique name for the database, such as **example2**. The database name can be a combination of alphanumeric characters, dashes (-), and underscores (_).



The **Create database in** field is automatically filled with the default database directory (`/var/lib/dirsrv/slaped-instance_name/db`) and the name of the new database. It is also possible to enter or browse for a different directory location.

2.2.1.2. Creating a New Database for a Single Suffix from the Command Line

Use the **ldapmodify** command-line utility to add a new database to the directory configuration file. The database configuration information is stored in the **cn=ldb database,cn=plugins,cn=config** entry.

For example, add a new database to the server **example1**:

1. Run **ldapmodify** and create the entry for the new database.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=UserData,cn=ldb database,cn=plugins,cn=config
changetype: add
objectclass: extensibleObject
objectclass: nsBackendInstance
nsslapd-suffix: ou=people,dc=example,dc=com
```

The entry added corresponds to a database named **UserData** that contains the data for the root or sub suffix **ou=people,dc=example,dc=com**.

2. Create a root or sub suffix, as described in [Section 2.1.1.3, "Creating Root and Sub Suffixes from the Command Line"](#). The database name, given in the DN attribute, must correspond with the value in the **nsslapd-backend** attribute of the suffix entry.

2.2.1.3. Adding Multiple Databases for a Single Suffix

A single suffix can be distributed across multiple databases. However, to distribute the suffix, a custom distribution function has to be created to extend the directory. For more information on creating a custom distribution function, contact Red Hat Professional Services.

NOTE

Once entries have been distributed, they cannot be redistributed. The following restrictions apply:

- The distribution function cannot be changed once entry distribution has been deployed.
- The LDAP **modrdn** operation cannot be used to rename entries if that would cause them to be distributed into a different database.
- Distributed local databases cannot be replicated.
- The **ldapmodify** operation cannot be used to change entries if that would cause them to be distributed into a different database.

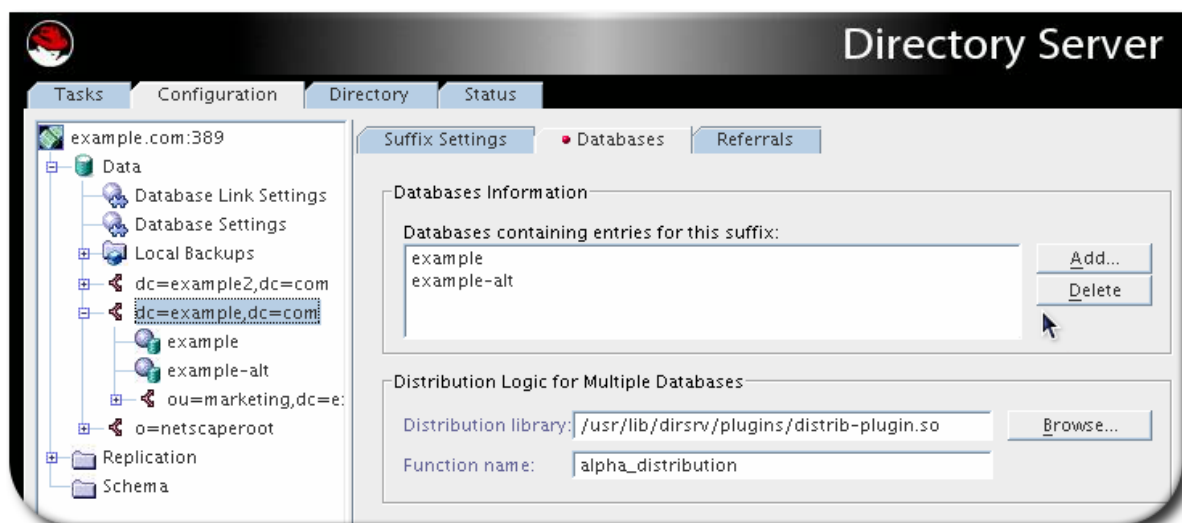
Violating these restrictions prevents Directory Server from correctly locating and returning entries.

After creating a custom distribution logic plug-in, add it to the directory.

The distribution logic is a function declared in a suffix. This function is called for every operation reaching this suffix, including subtree search operations that start above the suffix. A distribution function can be inserted into a suffix using both the Console and the command line.

2.2.1.3.1. Adding the Custom Distribution Function to a Suffix Using the Directory Server Console

1. In the Directory Server Console, select the **Configuration** tab.
2. Expand **Data** in the left navigation pane. Select the suffix to which to apply the distribution function.
3. Select the **Databases** tab in the right window.



4. The databases associated with the suffix are already listed in the **Databases** tab. Click **Add** to associate additional databases with the suffix.
5. Enter the path to the distribution library.
6. Enter the name of the distribution function in the **Function name** field.

2.2.1.3.2. Adding the Custom Distribution Function to a Suffix Using the Command Line

1. Run **ldapmodify**.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
```

2. Add the following attributes to the suffix entry itself, supplying the information about the custom distribution logic:

```
dn: suffix
changetype: modify
add: nsslapd-backend
nsslapd-backend: Database1
-
add: nsslapd-backend
nsslapd-backend: Database2
-
add: nsslapd-backend
nsslapd-backend: Database3
-
add: nsslapd-distribution-plugin
nsslapd-distribution-plugin: /full/name/of/a/shared/library
-
add: nsslapd-distribution-funct
nsslapd-distribution-funct: distribution-function-name
```

The **nsslapd-backend** attribute specifies all of the databases associated with this suffix. The **nsslapd-distribution-plugin** attribute specifies the name of the library that the plug-in uses. The **nsslapd-distribution-funct** attribute provides the name of the distribution function itself.

For more information about using the **ldapmodify** command-line utility, see [Section 3.2.4, “Adding and Modifying Entries Using ldapmodify”](#).

2.2.2. Maintaining Directory Databases

- [Section 2.2.2.1, “Placing a Database in Read-Only Mode”](#)
- [Section 2.2.2.2, “Deleting a Database”](#)
- [Section 2.2.2.3, “Configuring Transaction Logs for Frequent Database Updates”](#)

2.2.2.1. Placing a Database in Read-Only Mode

When a database is in read-only mode, you cannot create, modify, or delete any entries. One of the situations when read-only mode is useful is for manually initializing a consumer or before backing up or

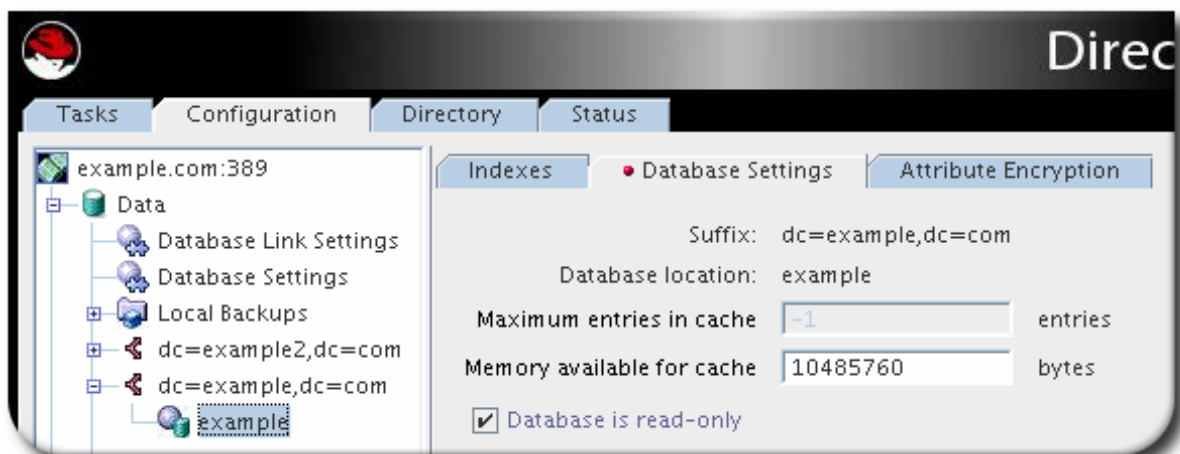
exporting data from the Directory Server. Read-only mode ensures a faithful image of the state of these databases at a given time.

The Directory Server Console and the command-line utilities do not automatically put the directory in read-only mode before export or backup operations because this would make your directory unavailable for updates. However, with multi-master replication, this might not be a problem.

- [Section 2.2.2.1.1, “Making a Database Read-Only Using the Console”](#)
- [Section 2.2.2.1.2, “Making a Database Read-Only from the Command Line”](#)
- [Section 2.2.2.1.3, “Placing the Entire Directory Server in Read-Only Mode”](#)

2.2.2.1.1. Making a Database Read-Only Using the Console

1. In the Directory Server Console, select the **Configuration** tab.
2. Expand **Data** in the left pane. Expand the suffix containing the database to put in read-only mode.
3. Select the database to put into read-only mode.
4. Select the **Database Settings** tab in the right pane.



5. Select the **database is read-only** check box.

The change takes effect immediately.

Before importing or restoring the database, ensure that the databases affected by the operation are *not* in read-only mode.

To disable read-only mode, open the database up in the Directory Server Console again and uncheck the **database is read-only** check box.

2.2.2.1.2. Making a Database Read-Only from the Command Line

To manually place a database into read-only mode:

1. Run **ldapmodify**.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
```

2. Change the read-only attribute to **on**

```
dn: cn=database_name,cn=ldbm database,cn=plugins,cn=config
changetype: modify
replace: nsslapd-readonly
nsslapd-readonly: on
```



NOTE

By default, the name of the database created at installation time is **userRoot**.

2.2.2.1.3. Placing the Entire Directory Server in Read-Only Mode

If the Directory Server maintains more than one database and all databases need to be placed in read-only mode, this can be done in a single operation.



WARNING

This operation also makes the Directory Server configuration read-only; therefore, you cannot update the server configuration, enable or disable plug-ins, or even restart the Directory Server while it is in read-only mode. Once read-only mode is enabled, it *cannot* be undone from the Console; you must modify the configuration files.

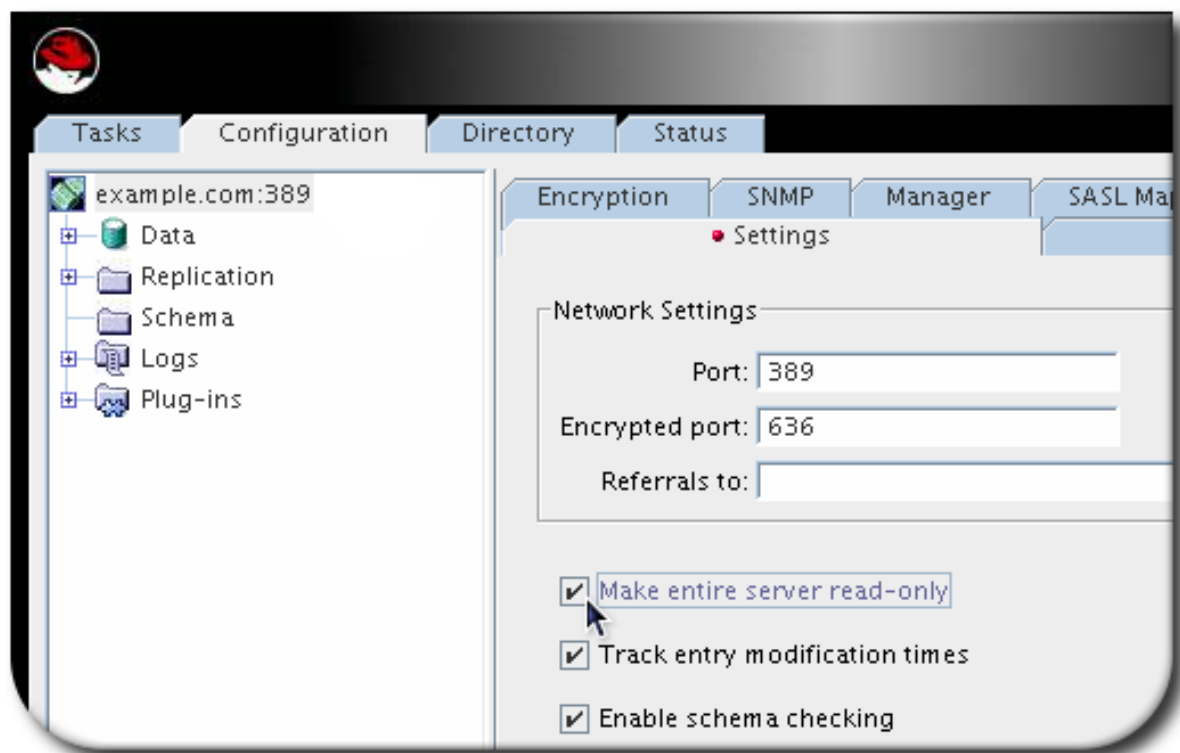


NOTE

If Directory Server contains replicas, *do not* use read-only mode because it will disable replication.

To put the Directory Server in read-only mode:

1. In the Directory Server Console, select the **Configuration** tab, and then select the top entry in the navigation tree in the left pane.
2. Select the **Settings** tab in the right pane.

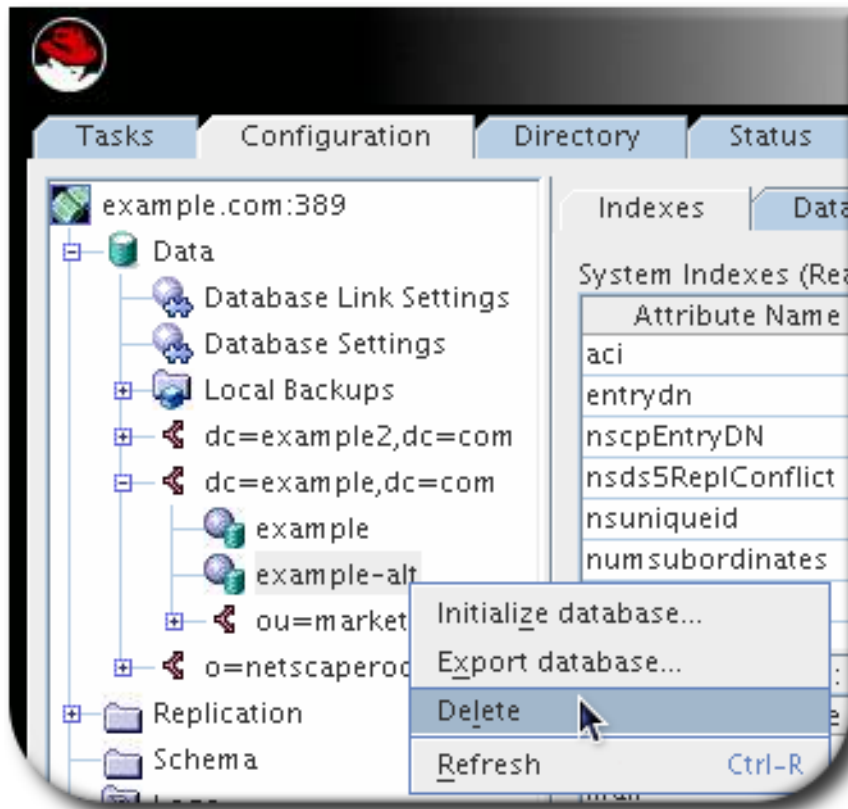


3. Select the **Make Entire Server Read-Only** check box.
4. Click **Save**, and then restart the server.

2.2.2.2. Deleting a Database

Deleting a database deletes the configuration information and entries for that database only, not the physical database itself.

1. In the Directory Server Console, select the **Configuration** tab.
2. Expand the **Data** folder, and then select the suffix.
3. Select the database to delete.
4. Right-click the database and select **Delete** from the pop-up menu.



5. Confirm that the database should be deleted in the **Delete Database** dialog box.

2.2.2.3. Configuring Transaction Logs for Frequent Database Updates

When the server is going to be asked to perform frequent database updates (LDAP adds, modifies, replication), the database transaction log files should be configured to be on a different disk than the primary database files.

Storing the transaction log files on a separate physical disk improves performance because the disk heads do not thrash moving between the log files and the data files.

1. Stop the Directory Server instance.

```
service dirsrv stop example
```

2. Create the new directory, if necessary, where the transaction logs will be located.

```
mkdir /home/exampledb-txnlogs
```

3. Set the appropriate file permissions on the directory so that the Directory Server user can access it; the default Directory Server user and group is **nobody:nobody**. However, Red Hat strongly recommends to use a different user and group name such as **dirsrv** during the installation.

```
chown nobody:nobody /home/exampledb-txnlogs
```

4. Open the **dse.ldif** file in the Directory Server instance's configuration directory.

```
[root@server ~]# vim /etc/dirsrv/slapd-instance_name/dse.ldif
```

5. Change the **nsslapd-db-logdirectory** directive for the new log file path:

```
nsslapd-db-logdirectory: /home/example/db-txnlogs
```

This attribute goes on the same entry that has the **nsslapd-dbcachesize** attribute.

6. Open the database directory.

```
[root@server ~]# cd /var/lib/dirsrv/slapd-instance_name/db
```

7. Remove all of the **__db.*** files.
8. Move the **log.*** files to the new location.
9. Start the Directory Server instance again.

```
[root@server ~]# service dirsrv start example
```

2.2.3. Configuring Attribute Encryption

The Directory Server offers a number of mechanisms to secure access to sensitive data, such as access control rules to prevent unauthorized users from reading certain entries or attributes within entries and SSL to protect data from eavesdropping and tampering on untrusted networks. However, if a copy of the server's database files should fall into the hands of an unauthorized person, they could potentially extract sensitive information from those files. Because information in a database is stored in plain text, some sensitive information, such as government identification numbers or passwords, may not be protected enough by standard access control measures.

For highly sensitive information, this potential for information loss could present a significant security risk. In order to remove that security risk, Directory Server allows portions of its database to be encrypted. Once encrypted, the data are safe even in the event that an attacker has a copy of the server's database files.

Database encryption allows attributes to be encrypted in the database. Both encryption and the encryption cipher are configurable per attribute per back end. When configured, every instance of a particular attribute, even index data, is encrypted for every entry stored in that database.

NOTE

There is one exception to encrypted data: any value which is used as the RDN for an entry is not encrypted within the entry DN. For example, if the **uid** attribute is encrypted, the value is encrypted in the entry but is displayed in the DN:

```
# entry-id: 16
dn: uid=jsmith1234,ou=People,dc=example,dc=com
nsUniqueId: ee91ea82-1dd111b2-9f36e9bc-39fb8550
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
givenName: John
sn: Smith
uid:: Sf04P9nJWGU1qiW9JJCGRg==
```

That would allow someone to discover the encrypted value.

Any attribute used within the entry DN cannot be effectively encrypted, since it will always be displayed in the DN. Be aware of what attributes are used to build the DN and design the attribute encryption model accordingly.

Indexed attributes may be encrypted, and attribute encryption is fully compatible with indexing. The contents of the index files that are normally derived from attribute values are also encrypted to prevent an attacker from recovering part or all of the encrypted data from an analysis of the indexes.

Since the server pre-encrypts all index keys before looking up an index for an encrypted attribute, there is some effect on server performance for searches that make use of an encrypted index, but the effect is not serious enough that it is no longer worthwhile to use an index.

2.2.3.1. Encryption Keys

In order to use attribute encryption, the server must be configured for SSL and have SSL enabled because attribute encryption uses the server's SSL encryption key and the same PIN input methods as SSL. The PIN must either be entered manually upon server startup or a PIN file must be used.

Randomly generated symmetric cipher keys are used to encrypt and decrypt attribute data. A separate key is used for each configured cipher. These keys are *wrapped* using the public key from the server's SSL certificate, and the resulting wrapped key is stored within the server's configuration files. The effective strength of the attribute encryption is never higher than the strength of the server's SSL key used for wrapping. Without access to the server's private key, it is not possible to recover the symmetric keys from the wrapped copies.

**WARNING**

There is no mechanism for recovering a lost key. Therefore, it is especially important to back up the server's certificate database safely. If the server's certificate were lost, it would not be possible to decrypt any encrypted data stored in its database.



WARNING

If the SSL certificate is expiring and needs to be renewed, export the encrypted back end instance before the renewal. Update the certificate, then re-import the exported LDIF file.

2.2.3.2. Encryption Ciphers

The encryption cipher is configurable on a per-attribute basis and must be selected by the administrator at the time encryption is enabled for an attribute. Configuration can be done through the Console or through the command line.

The following ciphers are supported:

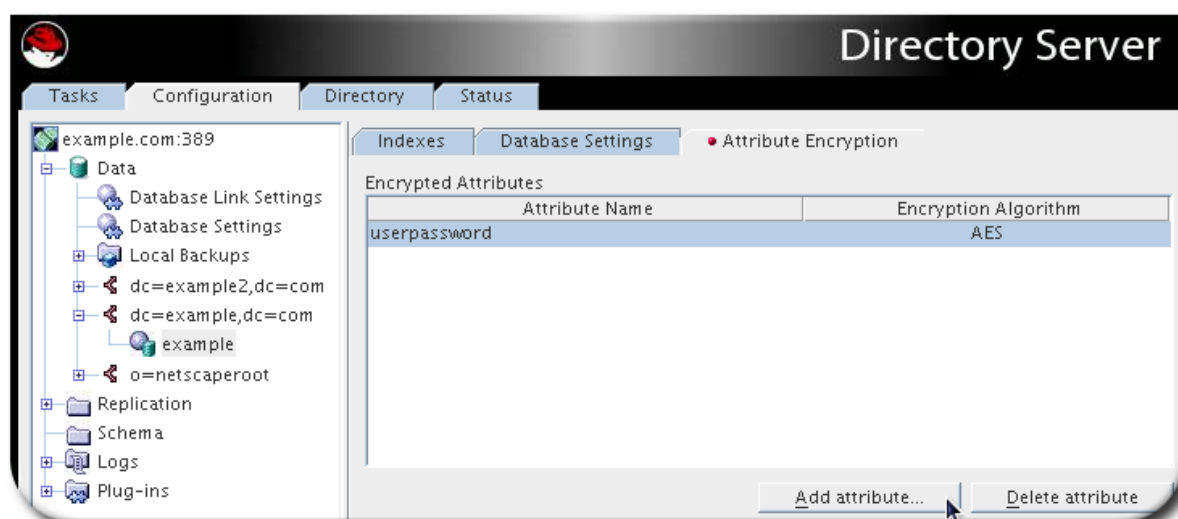
- Advanced Encryption Standard (AES)
- Triple Data Encryption Standard (3DES)

All ciphers are used in Cipher Block Chaining mode.

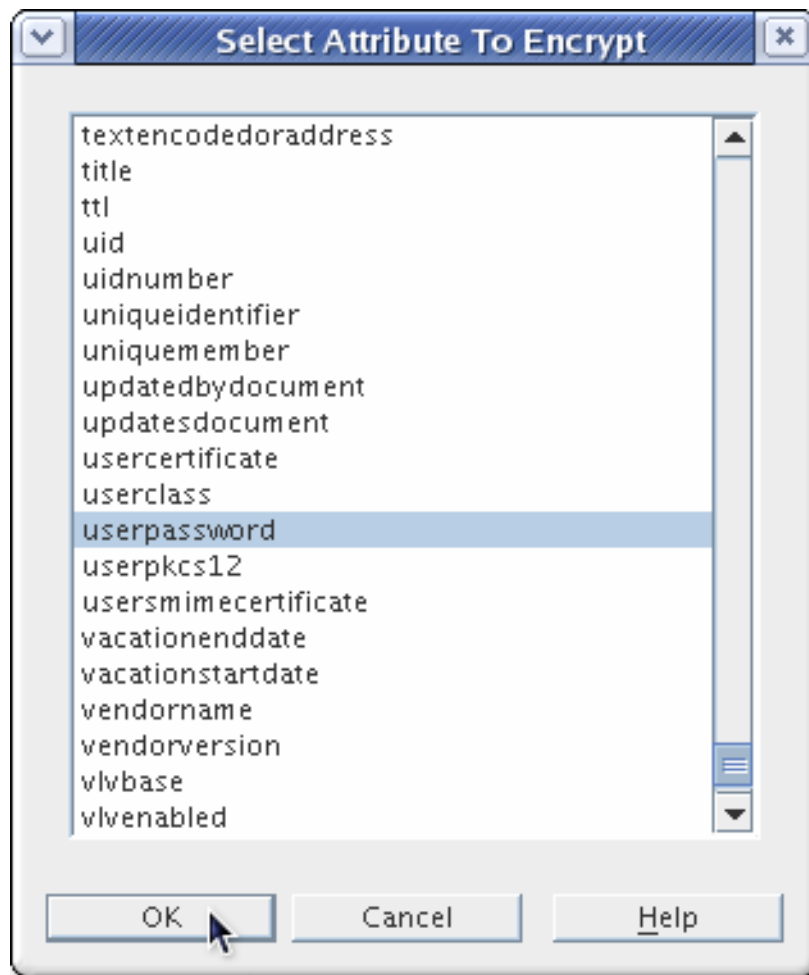
Once the encryption cipher is set, it should not be changed without exporting and re-importing the data.

2.2.3.3. Configuring Attribute Encryption from the Console

1. In the **Configuration** tab, select the **Data** node.
2. Expand the suffix, and select the database to edit.
3. Select the **Attribute Encryption** tab.

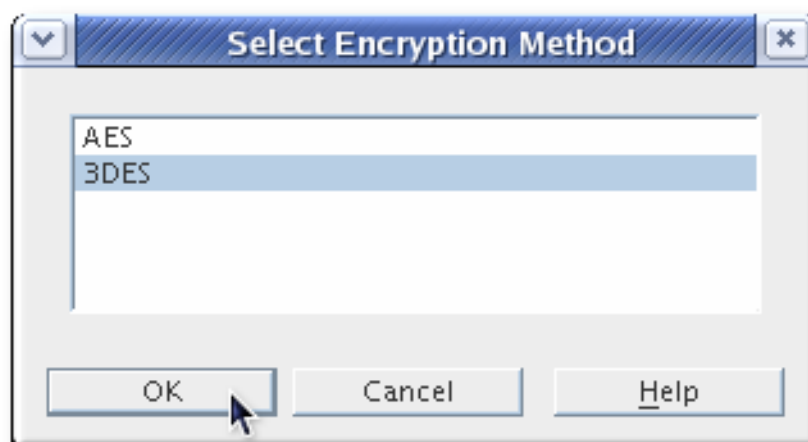


4. Click the **Add Attribute** button to open the list of attributes. Select the attribute to encrypt.

**NOTE**

For existing attribute values to be encrypted, the information must be exported from the database, then re-imported. See [Section 2.2.3.6, “Exporting and Importing an Encrypted Database”](#).

5. Select which encryption cipher to use.



**NOTE**

The encryption cipher to use is set separately for each attribute, so attribute encryption is applied to each attribute one at a time.

To remove encryption from attributes, select them from the list of encrypted attributes in the **Attribute Encryption** table, click the **Delete** button, and then click **Save** to apply the changes. Any deleted attributes have to be manually re-added after saving.

2.2.3.4. Configuring Attribute Encryption Using the Command Line

1. Run the **ldapmodify** command:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

2. Add an encryption entry for the attribute being encrypted. For example, this entry encrypts the **telephoneNumber** attribute with the AES cipher:

```
dn: cn=telephoneNumber,cn=encrypted attributes,cn=Database1,cn=ldbm
database,cn=plugins,cn=config
changetype: add
objectclass: top
objectclass: nsAttributeEncryption
cn: telephoneNumber
nsEncryptionAlgorithm: AES
```

3. For existing attributes in entries to be encrypted, the information must be exported, then re-imported. See [Section 2.2.3.6, "Exporting and Importing an Encrypted Database"](#).

For more information on attribute encryption configuration schema, see "Database Attributes under cn=attributeName,cn=encrypted attributes,cn=database_name,cn=ldbm database,cn=plugins,cn=config" in the *Directory Server Configuration and Command-Line Tool Reference*.

2.2.3.5. Enabling Attribute Encryption for Existing Attribute Values

To enable attribute encryption on an attribute with existing stored data, export the database to LDIF first, then make the configuration change, then re-import the data to the database. The server does not enforce consistency between encryption configuration and stored data; therefore, pay careful attention that all existing data are exported before enabling or disabling encryption.

2.2.3.6. Exporting and Importing an Encrypted Database

Exporting and importing encrypted databases is similar to exporting and importing regular databases. However, the encrypted information must be decrypted when it is exported to LDIF, then re-encrypted when it is imported to the database. Using the **-E** option when running the **db2ldif** and **ldif2db** scripts will decrypt the data on export and re-encrypt it on import.

1. Export the data using the **db2ldif** script, as follows:

```
db2ldif -n Database1 -E -a /path/to/output.ldif -s
"dc=example,dc=com" -s "o=userRoot"
```

See [Section 4.2.3, “Exporting to LDIF from the Command Line”](#) for more information.

2. Make any configuration changes.
3. Re-import the data using the **ldif2db** script, as follows:

```
ldif2db -n Database1 -E -i /path/to/output.ldif
```

See [Section 4.1.6, “Importing from the Command Line”](#) for more information.

NOTE

When enabling encryption for data that is already present in the database, several additional security concerns arise:

- It is possible for old, unencrypted data to persist in the server's database page pool backing file, even after a successful re-import with encryption. To remove this data, stop the server and delete the **db/guardian** file, then re-start the server. This will force recovery, a side-effect of which is deleting the backing file. However, it is possible that the data from the deleted file could still be recovered from the hard drive unless steps are taken to overwrite the disk blocks that it occupied.
- After enabling encryption and importing data, be sure to delete the LDIF file because it contains plain text values for the now-encrypted data. Ensure that the disk blocks that it occupied are overwritten.
- The unencrypted data previously stored in the server's database may persist on disk after a successful re-import with encryption. This is because the old database files are deleted as part of the import process. Ensure that the disk blocks that those files occupied are overwritten.
- Data stored in the server's replication log database is never encrypted; therefore, care should be taken to protect those files if replication is used.
- The server does not attempt to protect unencrypted data stored in memory. This data may be copied into a system page file by the operating system. For this reason, ensure that any page or swap files are adequately protected.

2.2.3.7. Updating Attribute Encryption Keys for New SSL/TLS Certificates

When SSL is first configured, there is no problem with attribute encryption. However, if the SSL certificate is changed, then attribute encryption fails, with messages like these:

```
Apr  4 13:00:35 smtp1 logger: [04/Apr/2017:13:00:34 -0700] -
attrcrypt_unwrap_key: failed to unwrap key for cipher AES
Apr  4 13:00:35 smtp1 logger: [04/Apr/2017:13:00:34 -0700] - Failed to
retrieve key for cipher AES in attrcrypt_cipher_init
Apr  4 13:00:35 smtp1 logger: [04/Apr/2017:13:00:34 -0700] - Failed to
initialize cipher AES in attrcrypt_init
Apr  4 13:00:35 smtp1 logger: [04/Apr/2017:13:00:34 -0700] -
attrcrypt_unwrap_key: failed to unwrap key for cipher AES
Apr  4 13:00:35 smtp1 logger: [04/Apr/2017:13:00:34 -0700] - Failed to
```

```
retrieve key for cipher AES in attrcrypt_cipher_init
Apr  4 13:00:35 smtp1 logger: [04/Apr/2017:13:00:34 -0700] - Failed to
initialize cipher AES in attrcrypt_init
```

This is because the previously-generated keys do not work with the new server certificate. To correct these errors, force the server to generate new keys for attribute encryption:

1. Stop the server.

```
service dirsrv stop
```

2. Open the **dse.ldif** file.

```
vim /etc/dirsrv/dse.ldif
```

3. There are special encryption key entries for the encryption ciphers used for attribute encryption under the database configuration. For example:

```
dn: cn=AES,cn=encrypted attribute keys,cn=userRoot,cn=ldbm
database,cn=plugins,cn=config
objectClass: top
objectClass: extensibleObject
cn: AES
nssymmetrickey::
mSLm/RlCLvPZrSdARHPowedF9zKx+kjVTww5ARE4w0lb12Y1YvrI3bNg=
```

Delete these entries.

4. Start the server again.

```
service dirsrv start
```

2.3. CREATING AND MAINTAINING DATABASE LINKS

Chaining means that a server contacts other servers on behalf of a client application and then returns the combined results. Chaining is implemented through a *database link*, which points to data stored remotely. When a client application requests data from a database link, the database link retrieves the data from the remote database and returns it to the client.

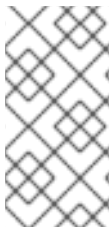
- [Section 2.3.1, “Creating a New Database Link”](#)
- [Section 2.3.2, “Configuring the Chaining Policy”](#)
- [Section 2.3.3, “Maintaining Database Links”](#)
- [Section 2.3.4, “Configuring Database Link Defaults”](#)
- [Section 2.3.5, “Deleting Database Links”](#)
- [Section 2.3.6, “Database Links and Access Control Evaluation”](#)

For more general information about chaining, see the chapter “Designing the Directory Topology,” in the *Directory Server Deployment Guide*. [Section 15.9, “Monitoring Database Link Activity”](#) covers how to monitor database link activity.

2.3.1. Creating a New Database Link

The basic database link configuration requires four piece of information:

- *Suffix information.* A suffix is created in the directory tree that is managed by the database link, not a regular database. This suffix corresponds to the suffix on the remote server that contains the data.
- *Bind credentials.* When the database link binds to a remote server, it impersonates a user, and this specifies the DN and the credentials for each database link to use to bind with remote servers.
- *LDAP URL.* This supplies the LDAP URL of the remote server to which the database link connects. The URL consists of the protocol (ldap or ldaps), the host name or IP address (IPv4 or IPv6) for the server, and the port.
- *List of failover servers.* This supplies a list of alternative servers for the database link to contact in the event of a failure. This configuration item is optional.



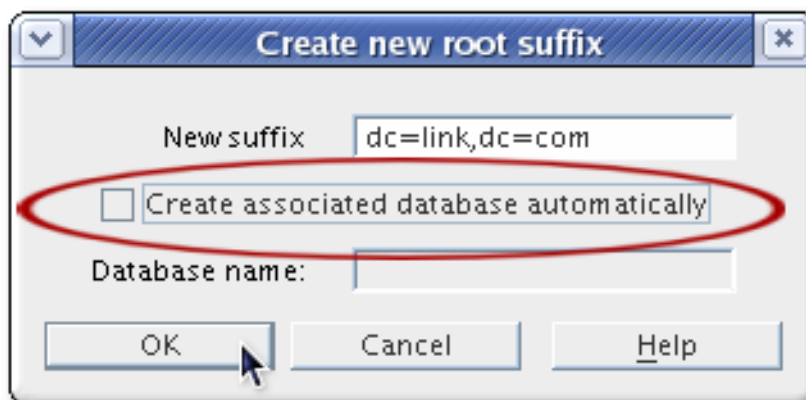
NOTE

If secure binds are required for simple password authentication ([Section 14.8.1, “Requiring Secure Binds”](#)), then any chaining operations will fail unless they occur over a secure connection. Using a secure connection (SSL/TLS and Start TLS connections or SASL authentication) is recommended, anyway.

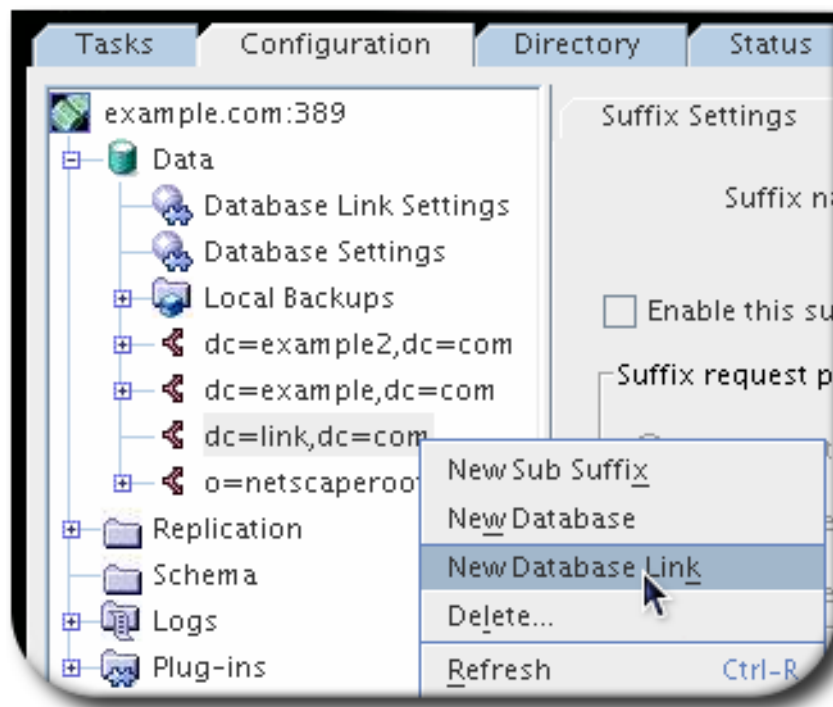
2.3.1.1. Creating a New Database Link Using the Console

1. In the Directory Server Console, select the **Configuration** tab.
2. Create a new suffix as described in [Section 2.1.1, “Creating Suffixes”](#).

Deselect the **Create associated database automatically** check box. It is simpler to configure a database link on a suffix without a database associated with it because having both a database and database link requires custom distribution functions to distribute directory data.



3. In the left pane, right-click the new suffix, and select **New Database Link** from the pop-up menu.



4. Fill in the database link name. The name can be a combination of alphanumeric characters, dashes (-), and underscores (_). No other characters, like spaces, are allowed.
5. Set the radio button for the appropriate method for authentication.

There are four authentication methods:

- *Simple* means that the server connects over the standard port with no encryption. The only required information is the bind DN and password for the user as whom the server connects to the remote server.
- *Server TLS/SSL Certificate* uses the local server's SSL certificate to authenticate to the remote server. A certificate must be installed on the local server for certificate-based authentication, and the remote server must have certificate mapping configured so that it can map the subject DN in the local server's certificate to the corresponding user entry.

Configuring SSL and certificate mapping is described in [Section 7.4, “Setting up TLS/SSL”](#).



NOTE

When the database link and remote server are configured to communicate using SSL, this does not mean that the client application making the operation request must also communicate using SSL. The client can bind using a normal port.

- *SASL/DIGEST-MD5* requires only the bind DN and password to authenticate.
- *SASL/GSSAPI* requires the local server to have a Kerberos keytab (as in [Section 7.12.2.2](#),

“[About the KDC Server and Keytabs](#)”), and the remote server to have a SASL mapping to map the local server's principal to the real user entry (as in [Section 7.11.3.1, “Configuring SASL Identity Mapping from the Console”](#)).

6. In the **Remote Server Information** section, select the connection type for the local server to use to connect to the remote server. There are three options:
 - *Use LDAP*. This sets a standard, unencrypted connection.
 - *Use TLS/SSL*. This uses a secure connection over the server's secure LDAPS port, such as **636**. This setting is required to use TLS/SSL.

When using SSL/TLS, make sure that the remote server's port number is set to its secure port.

- *Use Start TLS*. This uses Start TLS to establish a secure connection over the server's standard port.



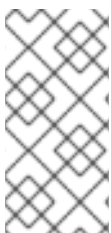
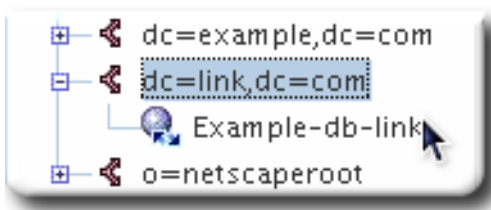
NOTE

If secure binds are required for simple password authentication ([Section 14.8.1, “Requiring Secure Binds”](#)), then any chaining operations will fail unless they occur over a secure connection. Using a secure connection (SSL/TLS and Start TLS connections or SASL authentication) is recommended, anyway.

7. In the **Remote Server Information** section, fill in the name (host name, IPv4 address, or IPv6 address) and port number for the remote server.

For any failover servers, fill in the host name and port number, and click the **Add** button. A failover server is a backup server, so that if the primary remote server fails, the database link contacts the first server in the failover servers list and cycles through the list until a server is accessed.

The new database link is listed under the suffix, in place of the database.



NOTE

The Console provides a checklist of information that needs to be present on the *remote* server for the database link to bind successfully. To view this checklist, click the new database link, and click the **Authentication** tab. The checklist is in the **Remote server checklist** box.

2.3.1.2. Creating a Database Link from the Command Line

1. Use the **ldapmodify** command-line utility to create a new database link. The new instance must be located in the **cn=chaining database,cn=plugins,cn=config** entry.

■

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

2. Specify the configuration information for the database link:

```
dn: cn=examplelink,cn=chaining database,cn=plugins,cn=config
changetype: add
objectclass: top
objectclass: extensibleObject
objectclass: nsBackendInstance
nsslapd-suffix: ou=people,dc=example,dc=com suffix being chained
nsfarmserverurl: ldap://people.example.com:389/ LDAP URL to remote
server
nsMultiplexorBindDN: cn=proxy admin,cn=config bind DN
nsMultiplexorCredentials: secret bind password
cn: examplelink
```



NOTE

If secure binds are required for simple password authentication ([Section 14.8.1, “Requiring Secure Binds”](#)), then any chaining operations will fail unless they occur over a secure connection. Using a secure connection (SSL/TLS and Start TLS connections or SASL authentication) is recommended, anyway.

Default configuration attributes are contained in the **cn=default instance config,cn=chaining database,cn=plugins,cn=config** entry. These configuration attributes apply to all database links at creation time. Changes to the default configuration only affect new database links. The default configuration attributes on existing database links cannot be changed.

Each database link contains its own specific configuration information, which is stored with the database link entry itself, **cn=database_link,cn=chaining database,cn=plugins,cn=config**. For more information about configuration attributes, see the *Directory Server Configuration and Command-Line Tool Reference*.

- [Section 2.3.1.2.1, “Providing Suffix Information”](#)
- [Section 2.3.1.2.2, “Providing Bind Credentials”](#)
- [Section 2.3.1.2.3, “Providing an LDAP URL”](#)
- [Section 2.3.1.2.4, “Providing a List of Failover Servers”](#)
- [Section 2.3.1.2.5, “Using Different Bind Mechanisms”](#)
- [Section 2.3.1.2.6, “Summary of Database Link Configuration Attributes”](#)
- [Section 2.3.1.2.7, “Database Link Configuration Example”](#)

2.3.1.2.1. Providing Suffix Information

Use the **nsslapd-suffix** attribute to define the suffix managed by the database link. For example, for the database link to point to the people information for a remote site of the company, enter the following suffix information:

`nsslapd-suffix: l=Zanzibar,ou=people,dc=example,dc=com`

The suffix information is stored in the **cn=database_link**, **cn=chaining database**, **cn=plugins**, **cn=config** entry.



NOTE

After creating the database link, any alterations to the **nsslapd-nsslapd-suffix** attribute are applied only after the server containing the database link is restarted.

2.3.1.2.2. Providing Bind Credentials

For a request from a client application to be chained to a remote server, special bind credentials can be supplied for the client application. This gives the remote server the proxied authorization rights needed to chain operations. Without bind credentials, the database link binds to the remote server as **anonymous**.

Providing bind credentials involves the following steps:

1. On the remote server:

- o Create an administrative user for the database link.

For information on adding entries, see [Chapter 3, Creating Directory Entries](#).

- o Provide proxy access rights for the administrative user created in step 1 on the subtree chained to by the database link.

For more information on configuring ACIs, see [Chapter 13, Managing Access Control](#)

2. On the server containing the database link, use **ldapmodify** to provide a user DN for the database link in the **nsMultiplexorBindDN** attribute of the **cn=database_link,cn=chaining database,cn=plugins,cn=config** entry.

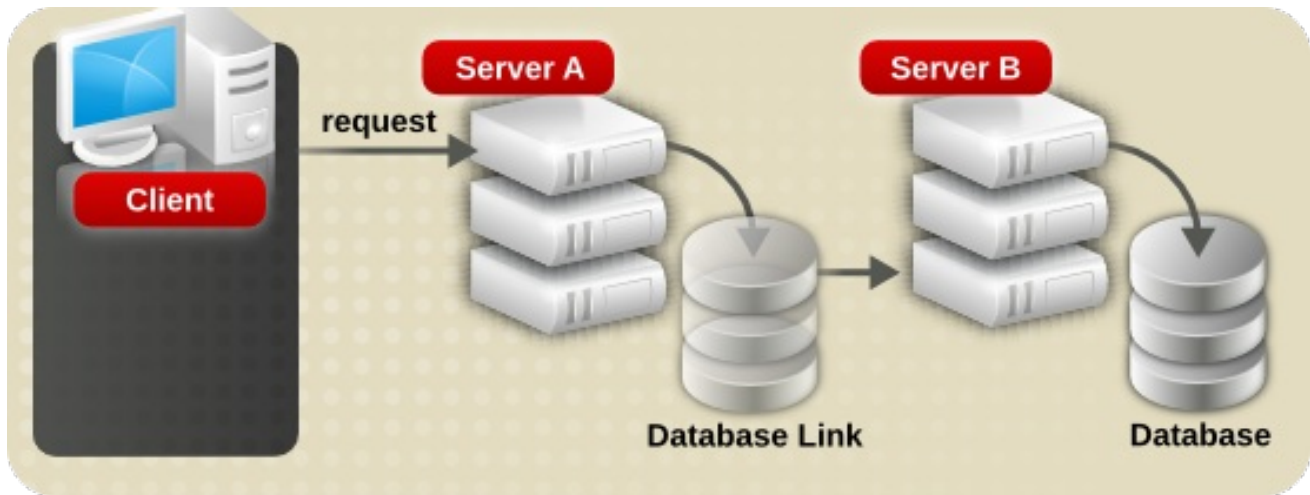


WARNING

The **nsMultiplexorBindDN** cannot be that of the Directory Manager.

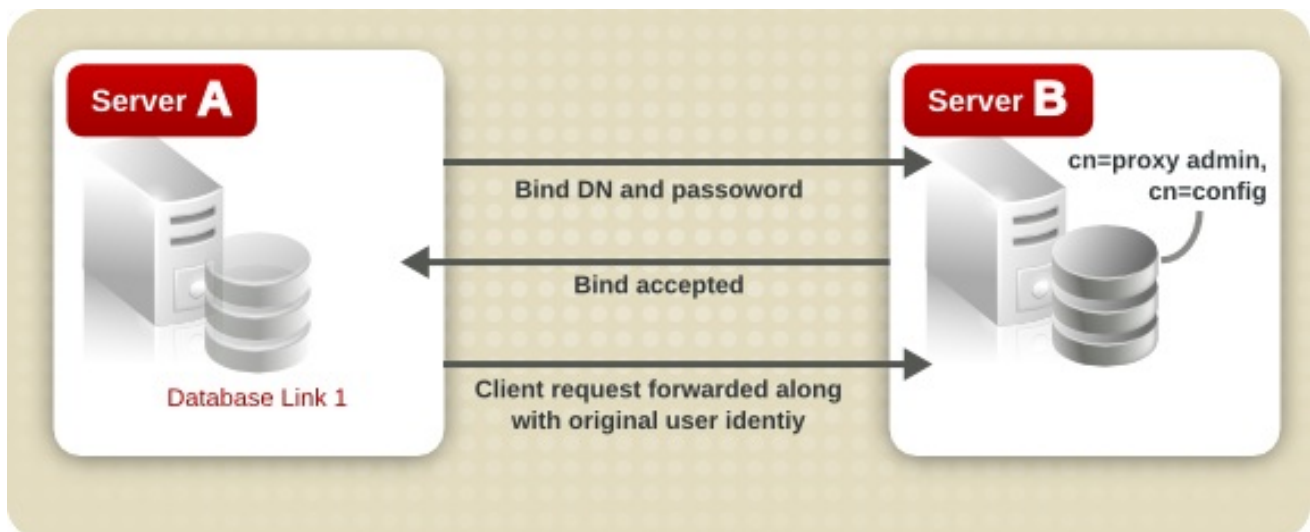
Use **ldapmodify** to provide a user password for the database link in the **nsMultiplexorCredentials** attribute of the **cn=database_link,cn=chaining database,cn=plugins,cn=config** entry.

For example, a client application sends a request to Server A. Server A contains a database link that chains the request to a database on Server B.



The database link on Server A binds to Server B using a special user as defined in the ***nsMultiplexorBindDN*** attribute and a user password as defined in the ***nsMultiplexorCredentials*** attribute. In this example, Server A uses the following bind credentials:

```
nsMultiplexorBindDN: cn=proxy admin,cn=config
nsMultiplexorCredentials: secret
```



Server B must contain a user entry corresponding to the ***nsMultiplexorBindDN***, and set the proxy authentication rights for this user. To set the proxy authorization correctly, set the proxy ACI as any other ACI.



WARNING

Carefully examine access controls when enabling chaining to avoid giving access to restricted areas of the directory. For example, if a default proxy ACI is created on a branch, the users that connect using the database link will be able to see all entries below the branch. There may be cases when not all of the subtrees should be viewed by a user. To avoid a security hole, create an additional ACI to restrict access to the subtree.

For more information on ACIs, see [Chapter 13, Managing Access Control](#).



NOTE

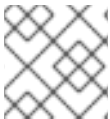
When a database link is used by a client application to create or modify entries, the attributes **creatorsName** and **modifiersName** do not reflect the real creator or modifier of the entries. These attributes contain the name of the administrative user granted proxied authorization rights on the remote data server.

2.3.1.2.3. Providing an LDAP URL

On the server containing the database link, identify the remote server that the database link connects with using an *LDAP URL*. Unlike the standard LDAP URL format, the URL of the remote server does not specify a suffix. It takes the form **ldap://server:port**, where the *server* can be a host name, IPv4 address, or IPv6 address.

The URL of the remote server using the **nsFarmServerURL** attribute is set in the **cn=database_link, cn=chaining database, cn=plugins, cn=config** entry of the configuration file.

```
nsFarmServerURL: ldap://example.com:389/
```



NOTE

Do not forget to use the trailing slash (/) at the end of the URL.

For the database link to connect to the remote server using LDAP over SSL, the LDAP URL of the remote server uses the protocol LDAPS instead of LDAP in the URL and points to the secure port of the server. For example:

```
nsFarmServerURL: ldaps://africa.example.com:636/
```



NOTE

SSL has to be enabled on the local Directory Server and the remote Directory Server to be chained over SSL. For more information on enabling SSL, see [Section 7.4, “Setting up TLS/SSL”](#).

When the database link and remote server are configured to communicate using SSL, this does not mean that the client application making the operation request must also communicate using SSL. The client can bind using a normal port.

2.3.1.2.4. Providing a List of Failover Servers

There can be additional LDAP URLs for servers included to use in the case of failure. Add alternate servers to the **nsFarmServerURL** attribute, separated by spaces.

```
nsFarmServerURL: ldap://example.com us.example.com:389
africa.example.com:1000/
```

In this sample LDAP URL, the database link first contacts the server **example.com** on the standard port to service an operation. If it does not respond, the database link then contacts the server **us.example.com** on port **389**. If this server fails, it then contacts **africa.example.com** on port

1000.

2.3.1.2.5. Using Different Bind Mechanisms

The local server can connect to the remote server using several different connection types and authentication mechanisms.

There are three ways that the local server can connect to the remote server:

- Over the standard LDAP port
- Over a dedicated TLS/SSL port
- Using Start TLS, which is a secure connection over a standard port



NOTE

If secure binds are required for simple password authentication ([Section 14.8.1, “Requiring Secure Binds”](#)), then any chaining operations will fail unless they occur over a secure connection. Using a secure connection (SSL/TLS and Start TLS connections or SASL authentication) is recommended, anyway.

Ultimately, there are two connection settings. The TLS/SSL option signifies that both of the servers are configured to run and accept connections over TLS/SSL, but there is no separate configuration attribute for enforcing TLS/SSL.

The connection type is identified in the ***nsUseStartTLS*** attribute. When this is **on**, then the server initiates a Start TLS connect over the standard port. If this is **off**, then the server either uses the LDAP port or the TLS/SSL port, depending on what is configured for the remote server in the ***nsFarmServerURL*** attribute.

For example, to use Start TLS:

```
nsUseStartTLS: on
```

For example, to use a standard connection or TLS/SSL connection:

```
nsUseStartTLS: off
```

There are four different methods which the local server can use to authenticate to the farm server.

- *empty*. If there is no bind mechanism set, then the server performs simple authentication and requires the ***nsMultiplexorBindDN*** and ***nsMultiplexorCredentials*** attributes to give the bind information.
- *EXTERNAL*. This uses an SSL certificate to authenticate the farm server to the remote server. Either the farm server URL must be set to the secure URL (**ldaps**) or the ***nsUseStartTLS*** attribute must be set to **on**.

Additionally, the remote server must be configured to map the farm server's certificate to its bind identity, as described in [Section 7.10.2, “Mapping DN's to Certificates”](#).

- *DIGEST-MD5*. This uses SASL authentication with DIGEST-MD5 encryption. As with simple authentication, this requires the *nsMultiplexorBindDN* and *nsMultiplexorCredentials* attributes to give the bind information.
- *GSSAPI*. This uses Kerberos-based authentication over SASL.

The farm server must be configured with a Kerberos keytab, and the remote server must have a defined SASL mapping for the farm server's bind identity. Setting up Kerberos keytabs and SASL mappings is described in [Section 7.11, “Setting up SASL Identity Mapping”](#).



NOTE

SASL connections can be established over standard connections or SSL/TLS connections.

For example:

```
nsBindMechanism: EXTERNAL
```



NOTE

If SASL is used, then the local server must also be configured to chain the SASL and password policy components. Add the components for the database link configuration, as described in [Section 2.3.2, “Configuring the Chaining Policy”](#). For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=config,cn=chaining database,cn=plugins,cn=config
changetype: modify
add: nsActiveChainingComponents
nsActiveChainingComponents: cn=password
policy,cn=components,cn=config
-
add: nsActiveChainingComponents
nsActiveChainingComponents: cn=sasl,cn=components,cn=config
^D
```

2.3.1.2.6. Summary of Database Link Configuration Attributes

The following table lists the attributes available for configuring a database link. Some of these attributes were discussed in the earlier sections. All instance attributes are defined in the **cn=database_link, cn=chaining database, cn=plugins, cn=config** entry.

Values defined for a specific database link take precedence over the global attribute value.

Table 2.2. Database Link Configuration Attributes

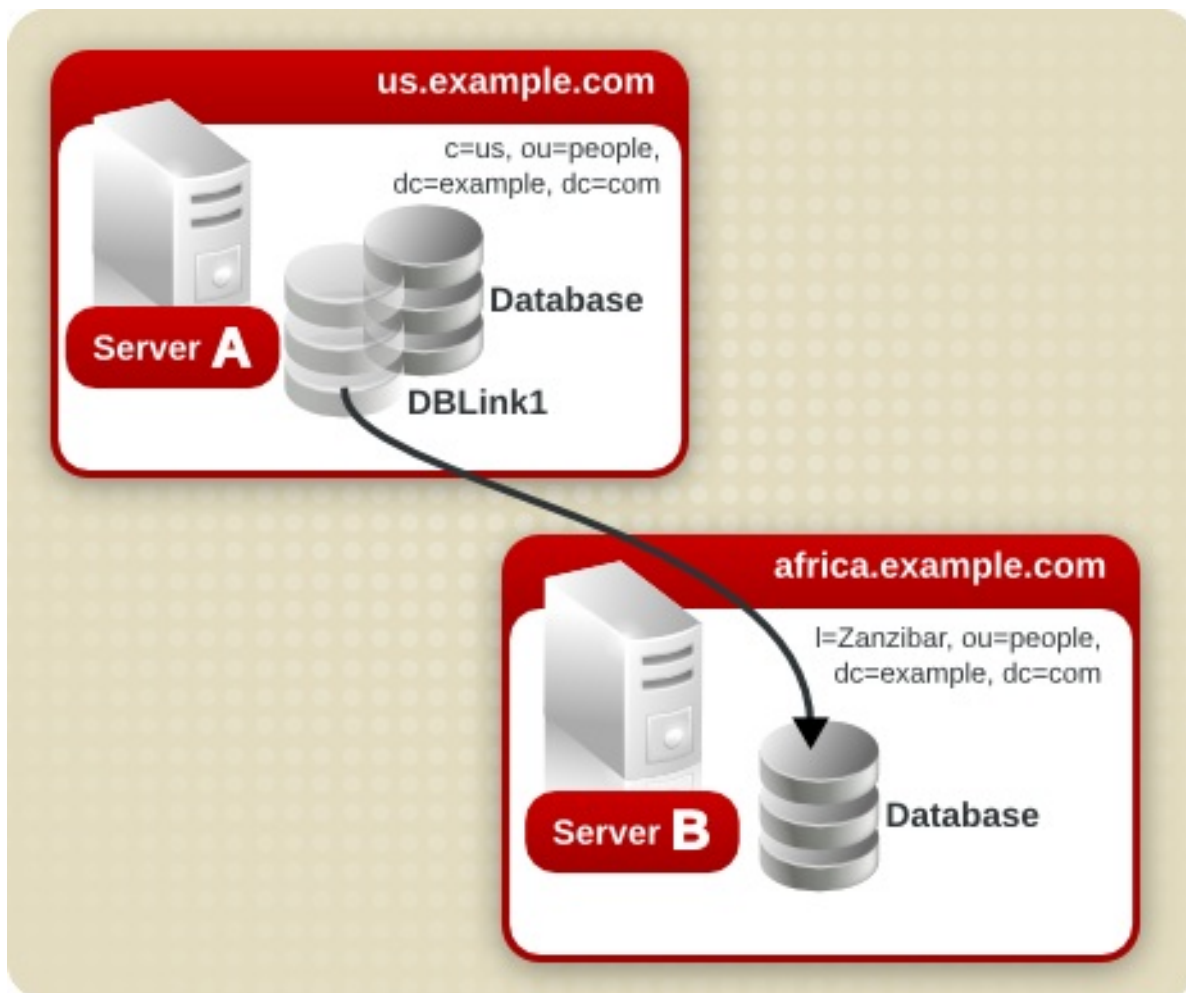
Attributes	Value
nsTransmittedControls [t]	Gives the OID of LDAP controls forwarded by the database link to the remote data server.

Attributes	Value
nsslapd-suffix	The suffix managed by the database link. Any changes to this attribute after the entry has been created take effect only after the server containing the database link is restarted.
nsslapd-timelimit	Default search time limit for the database link, given in seconds. The default value is 3600 seconds.
nsslapd-sizelimit	Default size limit for the database link, given in number of entries. The default value is 2000 entries.
nsFarmServerURL	Gives the LDAP URL of the remote server (or farm server) that contains the data. This attribute can contain optional servers for failover, separated by spaces. If using cascading chaining, this URL can point to another database link.
nsUseStartTLS	Sets whether to use Start TLS to establish a secure connection over a standard port. The default is off , which is used for both simple (standard) connections and TLS/SSL connections.
nsBindMethod	<p>Sets the authentication method to use to authenticate (bind) to the remote server. The default, if this attribute is not given, is to authenticate using a simple bind, requiring the <i>nsMultiplexorBindDN</i> and <i>nsMultiplexorCredentials</i> attributes for the bind information.</p> <div> <p>null; this is a simple bind.</p> <p>EXTERNAL (certificate-based); certificate mapping must be enabled for this.</p> <p>DIGEST-MD5 (SASL); this, like a simple bind, requires the bind DN and password.</p> <p>GSSAPI (SASL); this requires the keytab to be configured on the local server and SASL identity mapping on the remote server.</p> </div>
nsMultiplexorBindDN	DN of the administrative entry used to communicate with the remote server. The term <i>multiplexor</i> in the name of the attribute means the server which contains the database link and communicates with the remote server. This bind DN cannot be the Directory Manager. If this attribute is not specified, the database link binds as anonymous .

Attributes	Value
nsMultiplexorCredentials	Password for the administrative user, given in plain text. If no password is provided, it means that users can bind as anonymous . The password is encrypted in the configuration file.
nsCheckLocalACI	Reserved for advanced use only. Controls whether ACIs are evaluated on the database link as well as the remote data server. Takes the values on or off . Changes to this attribute occur only after the server has been restarted. The default value is off .
nsProxiedAuthorization	Reserved for advanced use only. Disables proxied authorization. A value of off means proxied authorization is disabled. The default value is on .
nsActiveChainingComponents ^[†]	Lists the components using chaining. A component is any functional unit in the server. The value of this attribute in the database link instance overrides the value in the global configuration attribute. To disable chaining on a particular database instance, use the value none . The default policy is not to allow chaining. For more information, see Section 2.3.2.1, “Chaining Component Operations” .
nsReferralOnScopedSearch	Controls whether referrals are returned by scoped searches. This attribute is for optimizing the directory because returning referrals in response to scoped searches is more efficient. Takes the values on or off . The default value is off .
nsHopLimit	Maximum number of times a request can be forwarded from one database link to another. The default value is 10 .
^[†] Can be both a global and instance attribute. This global configuration attribute is located in the cn=config,cn=chaining database,cn=plugins,cn=config entry. The global attributes are dynamic, meaning any changes made to them automatically take effect on all instances of the database link within the directory.	

2.3.1.2.7. Database Link Configuration Example

Suppose a server within the **us.example.com** domain contains the subtree **l=Walla Walla,ou=people,dc=example,dc=com** on a database and that operation requests for the **l=Zanzibar,ou=people,dc=example,dc=com** subtree should be chained to a different server in the **africa.example.com** domain.



1. Run **ldapmodify** to add a database link to Server A:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

2. Specify the configuration information for the database link:

```
dn: cn=DBLink1,cn=chaining database,cn=plugins,cn=config
changetype: add
objectclass: top
objectclass: extensibleObject
objectclass: nsBackendInstance
nsslapd-suffix: c=africa,ou=people,dc=example,dc=com
nsfarmserverurl: ldap://africa.example.com:389/
nsMultiplexorBindDN: cn=proxy admin,cn=config
nsMultiplexorCredentials: secret
cn: DBLink1

dn: cn=c=africa\,ou=people\,dc=example\,dc=com,cn=mapping
tree,cn=config
objectclass: top
objectclass: extensibleObject
objectclass: nsMappingTree
nsslapd-state: backend
nsslapd-backend: DBLink1
nsslapd-parent-suffix: ou=people,dc=example,dc=com
cn: c=africa\,ou=people\,dc=example\,dc=com
```

■

In the first entry, the **nsslapd-suffix** attribute contains the suffix on Server B to which to chain from Server A. The **nsFarmServerURL** attribute contains the LDAP URL of Server B.

The second entry creates a new suffix, allowing the server to route requests made to the new database link. The **cn** attribute contains the same suffix specified in the **nsslapd-suffix** attribute of the database link. The **nsslapd-backend** attribute contains the name of the database link. The **nsslapd-parent-suffix** attribute specifies the parent of this new suffix, **ou=people,dc=example,dc=com**.

3. Create an administrative user on Server B, as follows:

```
dn: cn=proxy admin,cn=config
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: proxy admin
sn: proxy admin
userPassword: secret
description: Entry for use by database links
```



WARNING

Do not use the Directory Manager user as the proxy administrative user on the remote server. This creates a security hole.

4. Add the following proxy authorization ACL to the **l=Zanzibar,ou=people,dc=example,dc=com** entry on Server B:

```
aci: (targetattr = "*")(version 3.0; acl "Proxied authorization
for database links"; allow (proxy) userdn = "ldap:///cn=proxy
admin,cn=config";)
```

This ACL gives the proxy admin user read-only access to the data contained on the remote server within the **l=Zanzibar,ou=people,dc=example,dc=com** subtree only.



NOTE

When a user binds to a database link, the user's identity is sent to the remote server. Access controls are always evaluated on the remote server. For the user to modify or write data successfully to the remote server, set up the correct access controls on the remote server. For more information about how access controls are evaluated in the context of chained operations, see [Section 2.3.6, "Database Links and Access Control Evaluation"](#).

2.3.2. Configuring the Chaining Policy

These procedures describe configuring how Directory Server chains requests made by client applications to Directory Servers that contain database links. This chaining policy applies to all database links created on Directory Server.

2.3.2.1. Chaining Component Operations

A component is any functional unit in the server that uses internal operations. For example, plug-ins are considered to be components, as are functions in the front-end. However, a plug-in may actually be comprised of multiple components (for example, the ACI plug-in).

Some components send internal LDAP requests to the server, expecting to access local data only. For such components, control the chaining policy so that the components can complete their operations successfully. One example is the certificate verification function. Chaining the LDAP request made by the function to check certificates implies that the remote server is trusted. If the remote server is not trusted, then there is a security problem.

By default, all internal operations are not chained and no components are allowed to chain, although this can be overridden.

Additionally, an ACI must be created on the remote server to allow the specified plug-in to perform its operations on the remote server. The ACI must exist in the *suffix* assigned to the database link.

The following table lists component names, the potential side-effects of allowing them to chain internal operations, and the permissions they need in the ACI on the remote server:

Table 2.3. Components Allowed to Chain

Component Name	Description	Permissions
ACI plug-in	This plug-in implements access control. Operations used to retrieve and update ACI attributes are not chained because it is not safe to mix local and remote ACI attributes. However, requests used to retrieve user entries may be chained by setting the chaining components attribute, <i>nsActiveChainingComponents: cn=ACI Plugin,cn=plugins,cn=config</i> .	Read, search, and compare
Resource limit component	This component sets server limits depending on the user bind DN. Resource limits can be applied on remote users if the resource limitation component is allowed to chain. To chain resource limit component operations, add the chaining component attribute, <i>nsActiveChainingComponents: cn=resource limits,cn=components,cn=config</i> .	Read, search, and compare

Component Name	Description	Permissions
Certificate-based authentication checking component	This component is used when the external bind method is used. It retrieves the user certificate from the database on the remote server. Allowing this component to chain means certificate-based authentication can work with a database link. To chain this component's operations, add the chaining component attribute, <i>nsActiveChainingComponents: cn=certificate-based authentication,cn=components,cn=config</i> .	Read, search, and compare
Password policy component	This component is used to allow SASL binds to the remote server. Some forms of SASL authentication require authenticating with a user name and password. Enabling the password policy allows the server to verify and implement the specific authentication method requested and to apply the appropriate password policies. To chain this component's operations, add the chaining component attribute, <i>nsActiveChainingComponents: cn=password policy,cn=components,cn=config</i> .	Read, search, and compare
SASL component	This component is used to allow SASL binds to the remote server. To chain this component's operations, add the chaining component attribute, <i>nsActiveChainingComponents: cn=password policy,cn=components,cn=config</i> .	Read, search, and compare

Component Name	Description	Permissions
Referential Integrity plug-in	<p>This plug-in ensures that updates made to attributes containing DNs are propagated to all entries that contain pointers to the attribute. For example, when an entry that is a member of a group is deleted, the entry is automatically removed from the group. Using this plug-in with chaining helps simplify the management of static groups when the group members are remote to the static group definition. To chain this component's operations, add the chaining component attribute, <i>nsActiveChainingComponents: cn=referential integrity postoperation,cn=plugins,cn=config</i>.</p>	Read, write, search, and compare
Attribute Uniqueness plug-in	<p>This plug-in checks that all the values for a specified attribute are unique (no duplicates). If this plug-in is chained, it confirms that attribute values are unique even on attributes changed through a database link. To chain this component's operations, add the chaining component attribute, <i>nsActiveChainingComponents: cn=attribute uniqueness,cn=plugins,cn=config</i></p>	Read, search, and compare
Roles component	<p>This component chains the roles and roles assignments for the entries in a database. Chaining this component maintains the roles even on chained databases. To chain this component's operations, add the chaining component attribute, <i>nsActiveChainingComponents: cn=roles,cn=components,cn=config</i>.</p>	Read, write, search, and compare

NOTE

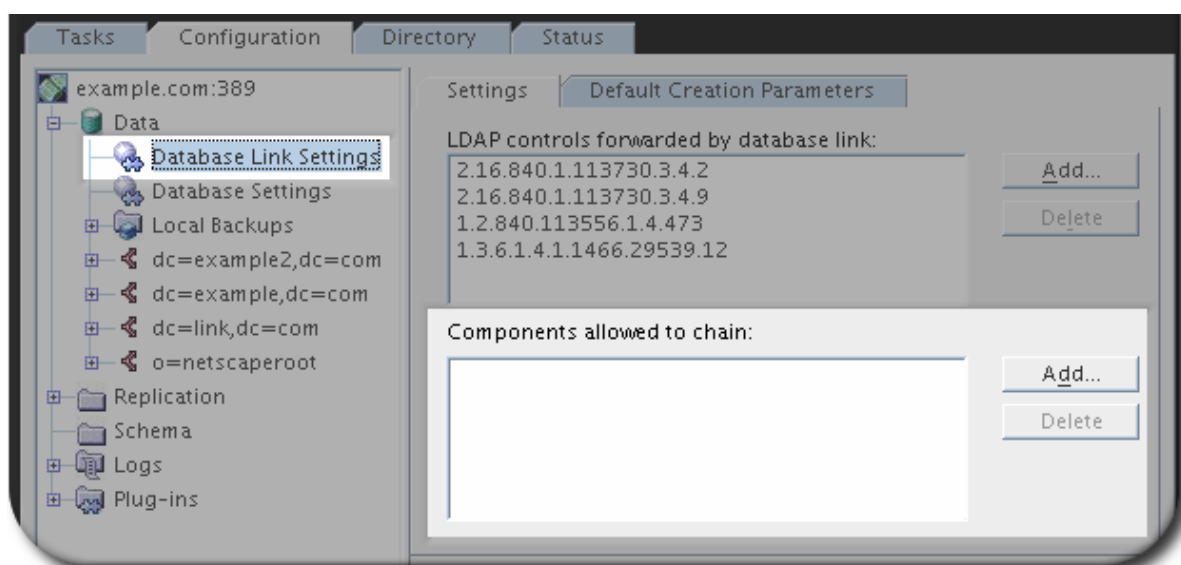
The following components cannot be chained:

- Roles plug-in
- Password policy component
- Replication plug-ins
- Referential Integrity plug-in

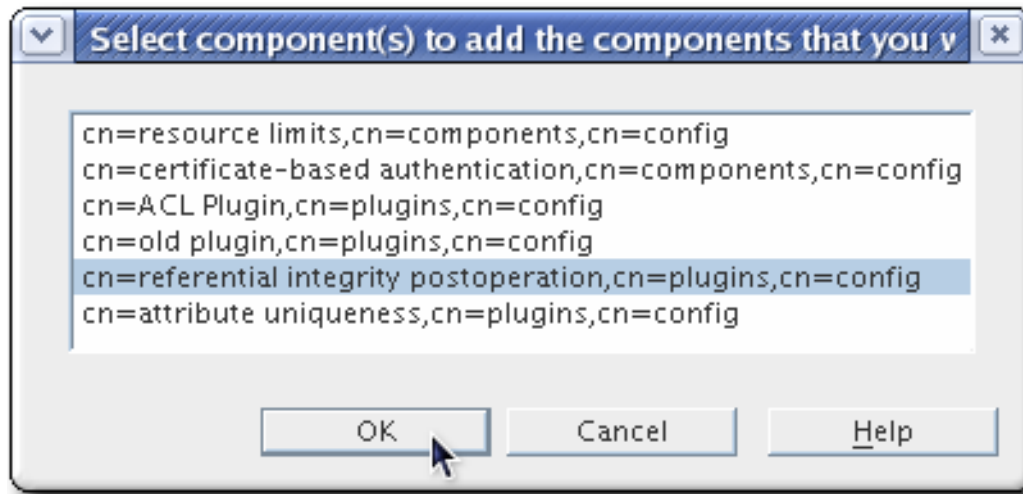
When enabling the Referential Integrity plug-in on servers issuing chaining requests, be sure to analyze performance, resource, and time needs as well as integrity needs. Integrity checks can be time-consuming and draining on memory and CPU. For further information on the limitations surrounding ACIs and chaining, see [Section 13.1.4, “ACI Limitations”](#).

2.3.2.1.1. Chaining Component Operations Using the Console

1. In the Directory Server Console, select the **Configuration** tab.
2. Expand **Data** in the left pane, and click **Database Link Settings**.
3. Select the **Settings** tab in the right window.



4. Click the **Add** button in the **Components allowed to chain** section.
5. Select the component to chain from the list, and click **OK**.



6. Restart the server in order for the change to take effect.

After allowing the component to chain, create an ACL in the suffix on the remote server to which the operation will be chained. For example, this creates an ACL for the Referential Integrity plug-in:

```
aci: (targetattr "*")
(target="ldap:///ou=customers,l=us,dc=example,dc=com")
(version 3.0; acl "RefInt Access for chaining"; allow
(read,write,search,compare) userdn = "ldap:///cn=referential
integrity
postoperation,cn=plugins,cn=config";)
```

2.3.2.1.2. Chaining Component Operations from the Command Line

1. Specify components to include in chaining using the ***nsActiveChainingComponents*** attribute in the ***cn=config,cn=chaining database,cn=plugins,cn=config*** entry of the configuration file.

For example, to allow the referential integrity component to chain operations, add the following to the database link configuration file:

```
nsActiveChainingComponents: cn=referential integrity
postoperation,cn=components,cn=config
```

See [Table 2.3, “Components Allowed to Chain”](#) for a list of the components which can be chained.

2. Restart the server for the change to take effect.

```
service dirsrv restart instance
```

3. Create an ACL in the suffix on the remote server to which the operation will be chained. For example, this creates an ACL for the Referential Integrity plug-in:

```
aci: (targetattr "*")
(target="ldap:///ou=customers,l=us,dc=example,dc=com")
(version 3.0; acl "RefInt Access for chaining"; allow
```

```
(read,write,search,compare) userdn = "ldap:///cn=referential
integrity postoperation,cn=plugins,cn=config";)
```

2.3.2.2. Chaining LDAP Controls

It is possible to *not* chain operation requests made by LDAP controls. By default, requests made by the following controls are forwarded to the remote server by the database link:

- *Virtual List View (VLV)*. This control provides lists of parts of entries rather than returning all entry information.
- *Server-side sorting*. This control sorts entries according to their attribute values, usually using a specific matching rule.
- *Dereferencing*. This control tracks back over references in entry attributes in a search and pulls specified attribute information from the referenced entry and returns it with the rest of the search results.
- *Managed DSA*. This controls returns smart referrals as entries, rather than following the referral, so the smart referral itself can be changed or deleted.
- *Loop detection*. This control keeps track of the number of times the server chains with another server. When the count reaches the configured number, a loop is detected, and the client application is notified. For more information about using this control, see [Section 2.4.4, “Detecting Loops”](#).

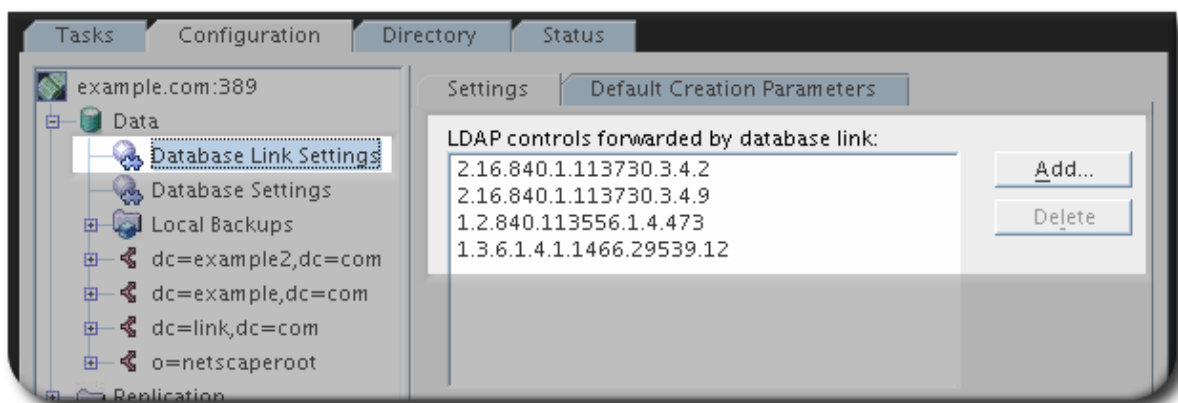


NOTE

Server-side sorting and VLV controls are supported only when a client application request is made to a single database. Database links cannot support these controls when a client application makes a request to multiple databases.

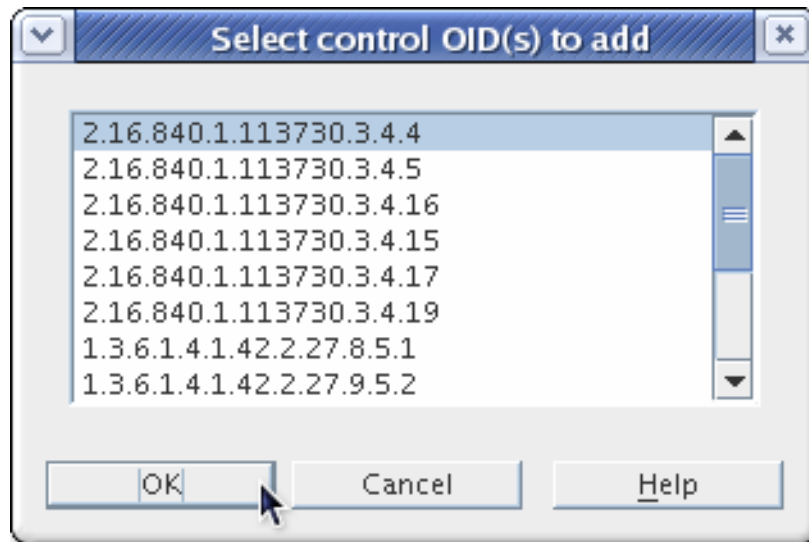
2.3.2.2.1. Chaining LDAP Controls Using the Console

1. In the Directory Server Console, select the **Configuration** tab.
2. Expand the **Data** folder in the left pane, and click **Database Link Settings**.
3. Select the **Settings** tab in the right window.



4. Click the **Add** button in the **LDAP Controls forwarded by the database link** section to add an LDAP control to the list.

5. Select the OID of a control to add to the list, and click **OK**.



2.3.2.2.2. Chaining LDAP Controls from the Command Line

To chain controls, alter the controls that the database link forwards by changing the ***nsTransmittedControls*** attribute of the ***cn=config, cn=chaining database, cn=plugins, cn=config*** entry. For example, to forward the virtual list view control, add the following to the database link entry in the configuration file:

```
nsTransmittedControls: 2.16.840.1.113730.3.4.9
```

In addition, if clients of the Directory Server create their own controls and their operations should be chained to remote servers, add the OID of the custom control to the ***nsTransmittedControls*** attribute.

The LDAP controls which can be chained and their OIDs are listed in the following table:

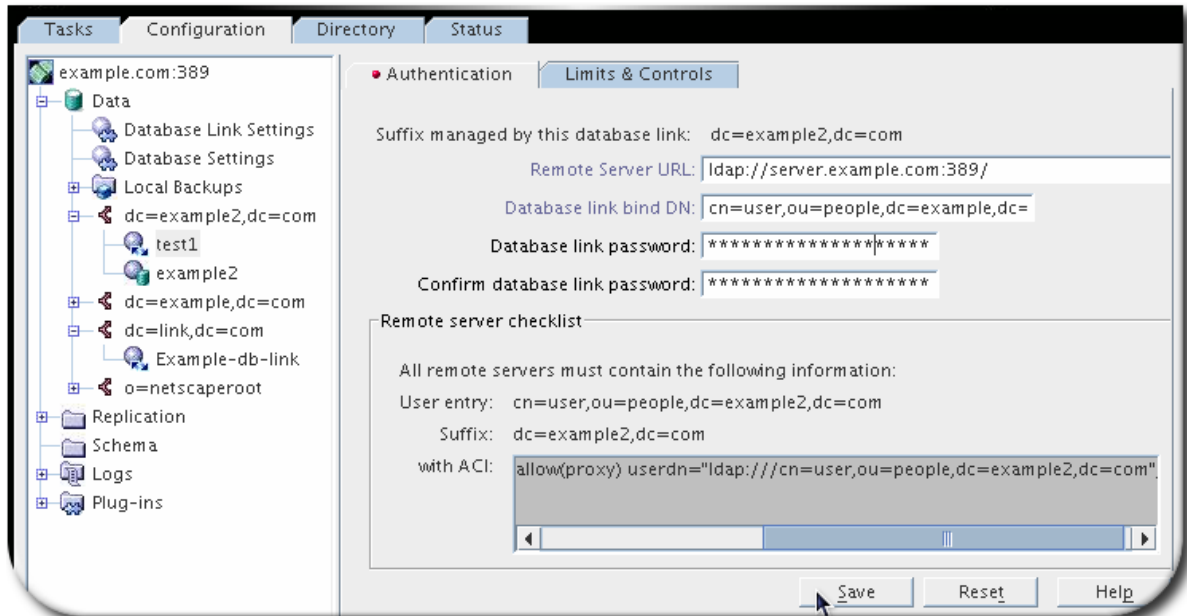
Table 2.4. LDAP Controls and Their OIDs

Control Name	OID
Virtual list view (VLV)	2.16.840.1.113730.3.4.9
Server-side sorting	1.2.840.113556.1.4.473
Managed DSA	2.16.840.1.113730.3.4.2
Loop detection	1.3.6.1.4.1.1466.29539.12
Dereferencing searches	1.3.6.1.4.1.4203.666.5.16

2.3.3. Maintaining Database Links

All of the information for the database link for the connection to the remote server.

1. In the Directory Server Console, select the **Configuration** tab.
2. In the left pane, expand the **Data** folder, and select the database link under the suffix.
3. In the right navigation pane, click the **Authentication** tab.

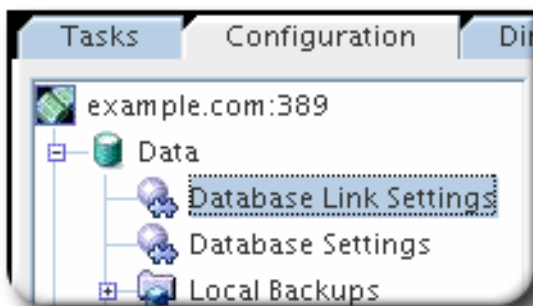


4. Change the connection information.
 - The LDAP URL for the remote server.[]
 - The bind DN and password used by the database link to bind to the remote server.

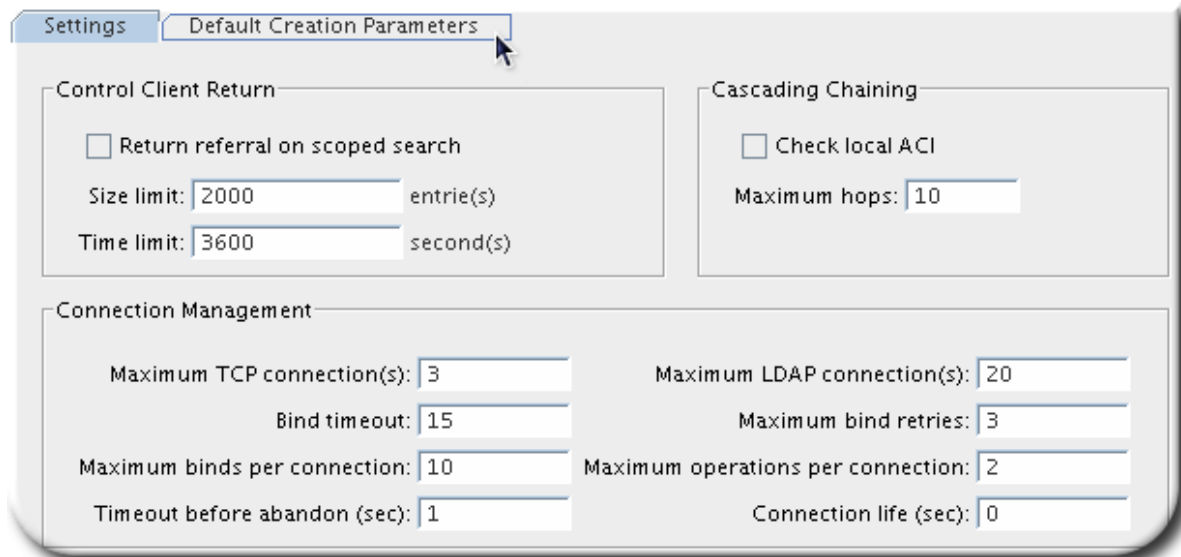
2.3.4. Configuring Database Link Defaults

Configuring the default settings for database links defines the settings used for cascading chaining (the number of hops allowed for a client request), the connection rules for the remote server, and how the server responds to client requests.

1. Select the **Configuration** tab.
2. Expand the **Data** folder in the left pane, and click **Database Link Settings**. Open the **Default Creation Parameters** tab.



3. Fill in the new configuration parameters.



Settings | **Default Creation Parameters**

Control Client Return

☐ Return referral on scoped search

Size limit: 2000 entrie(s)

Time limit: 3600 second(s)

Cascading Chaining

☐ Check local ACL

Maximum hops: 10

Connection Management

Maximum TCP connection(s): 3

Maximum LDAP connection(s): 20

Bind timeout: 15

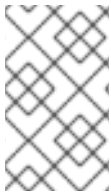
Maximum bind retries: 3

Maximum binds per connection: 10

Maximum operations per connection: 2

Timeout before abandon (sec): 1

Connection life (sec): 0



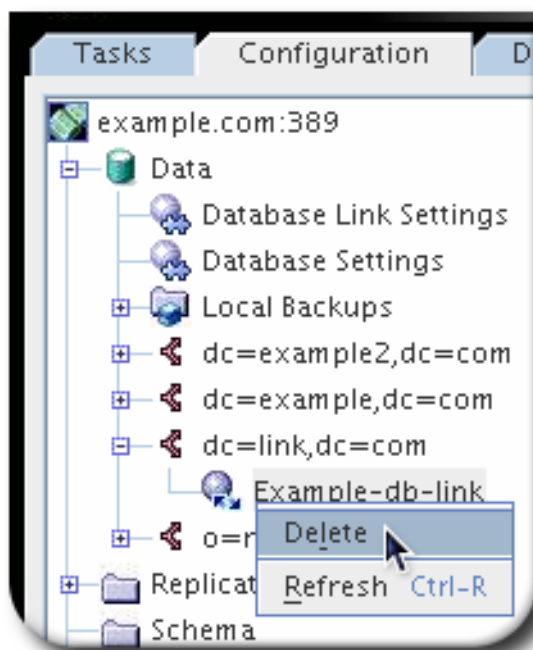
NOTE

Changes made to the default settings of a database link are not applied retroactively. Only the database links created after changes are made to the default settings will reflect the changes.

2.3.5. Deleting Database Links

To delete a database link, right-click the database link, and select **Delete** from the pop-up menu. Confirm the delete when prompted.

1. In the Directory Server Console, select the **Configuration** tab.
2. Under **Data** in the left navigation pane, open the suffix and select the database link to delete.
3. Right-click the database link, and select **Delete** from the menu.



2.3.6. Database Links and Access Control Evaluation

When a user binds to a server containing a database link, the database link sends the user's identity to the remote server. Access controls are always evaluated on the remote server. Every LDAP operation evaluated on the remote server uses the original identity of the client application passed using the proxied authorization control. Operations succeed on the remote server only if the user has the correct access controls on the subtree contained on the remote server. This requires adding the usual access controls to the remote server with a few restrictions:

- Not all types of access control can be used.

For example, role-based or filter-based ACIs need access to the user entry. Because the data are accessed through database links, only the data in the proxy control can be verified. Consider designing the directory in a way that ensures the user entry is located in the same database as the user's data.

- All access controls based on the IP address or DNS domain of the client may not work since the original domain of the client is lost during chaining. The remote server views the client application as being at the same IP address and in the same DNS domain as the database link.



NOTE

Directory Server supports both IPv4 and IPv6 IP addresses.

The following restrictions apply to the ACIs used with database links:

- ACIs must be located with any groups they use. If the groups are dynamic, all users in the group must be located with the ACI and the group. If the group is static, it links to remote users.
- ACIs must be located with any role definitions they use and with any users intended to have those roles.
- ACIs that link to values of a user's entry (for example, **userattr** subject rules) will work if the user is remote.

Though access controls are always evaluated on the remote server, they can also be evaluated on both the server containing the database link and the remote server. This poses several limitations:

- During access control evaluation, contents of user entries are not necessarily available (for example, if the access control is evaluated on the server containing the database link and the entry is located on a remote server).

For performance reasons, clients cannot do remote inquiries and evaluate access controls.

- The database link does not necessarily have access to the entries being modified by the client application.

When performing a modify operation, the database link does not have access to the full entry stored on the remote server. If performing a delete operation, the database link is only aware of the entry's DN. If an access control specifies a particular attribute, then a delete operation will fail when being conducted through a database link.



NOTE

By default, access controls set on the server containing the database link are not evaluated. To override this default, use the ***nsCheckLocalACI*** attribute in the ***cn=database_link, cn=chaining database, cn=plugins, cn=config*** entry. However, evaluating access controls on the server containing the database link is not recommended except with cascading chaining.

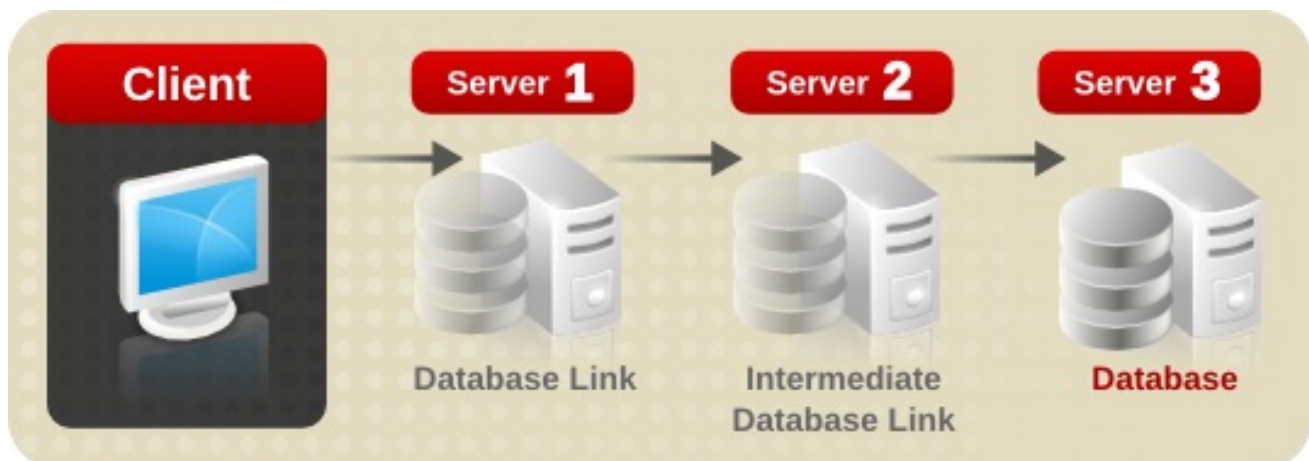
2.4. CONFIGURING CASCADING CHAINING

The database link can be configured to point to another database link, creating a cascading chaining operation. A cascading chain occurs any time more than one hop is required to access all of the data in a directory tree.

- [Section 2.4.1, “Overview of Cascading Chaining”](#)
- [Section 2.4.2, “Configuring Cascading Chaining Using the Console”](#)
- [Section 2.4.3, “Configuring Cascading Chaining from the Command Line”](#)
- [Section 2.4.4, “Detecting Loops”](#)
- [Section 2.4.5, “Summary of Cascading Chaining Configuration Attributes”](#)
- [Section 2.4.6, “Cascading Chaining Configuration Example”](#)

2.4.1. Overview of Cascading Chaining

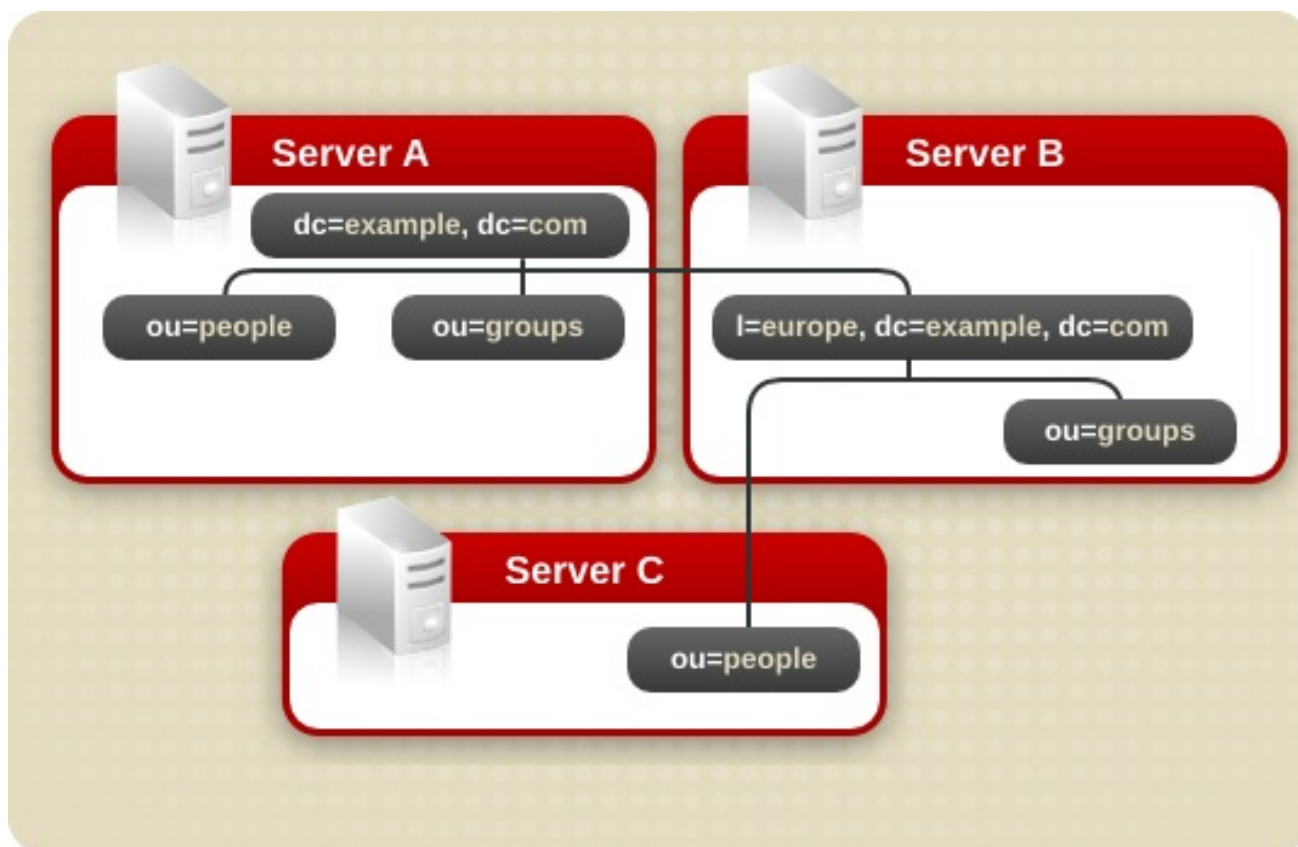
Cascading chaining occurs when more than one hop is required for the directory to process a client application's request.



The client application sends a modify request to Server 1. Server one contains a database link that forwards the operation to Server 2, which contains another database link. The database link on Server 2 forwards the operations to server three, which contains the data the clients wants to modify in a database. Two hops are required to access the piece of data the client want to modify.

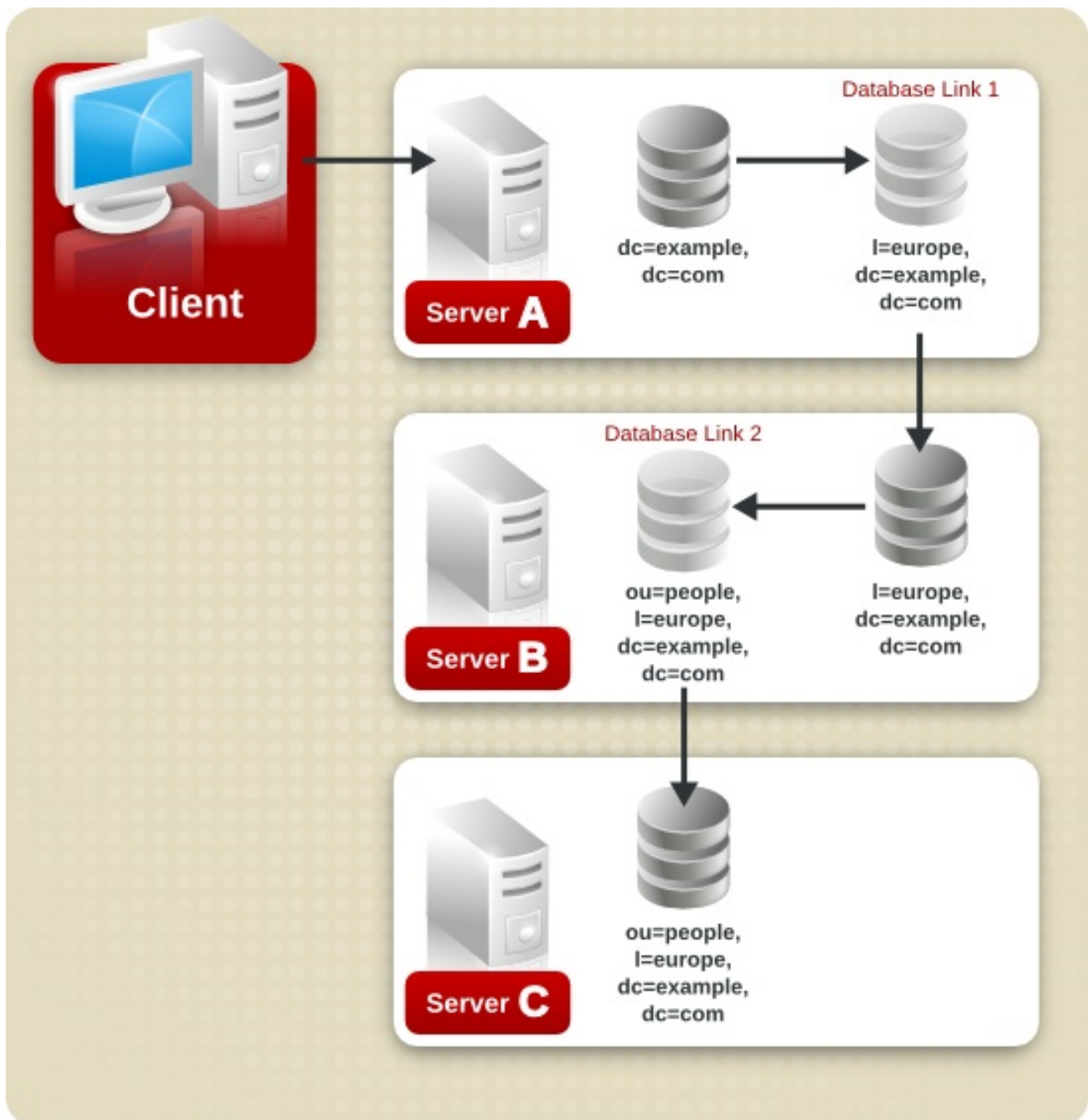
During a normal operation request, a client binds to the server, and then any ACIs applying to that client are evaluated. With cascading chaining, the client bind request is evaluated on Server 1, but the ACIs applying to the client are evaluated only after the request has been chained to the destination server, in the above example Server 2.

For example, on Server A, a directory tree is split:



The root suffix **dc=example, dc=com** and the **ou=people** and **ou=groups** sub suffixes are stored on Server A. The **l=europe, dc=example, dc=com** and **ou=groups** suffixes are stored in on Server B, and the **ou=people** branch of the **l=europe, dc=example, dc=com** suffix is stored on Server C.

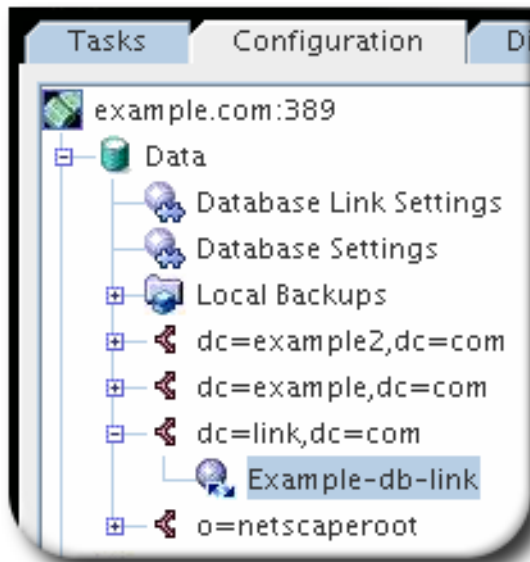
With cascading configured on servers A, B, and C, a client request targeted at the **ou=people, l=europe, dc=example, dc=com** entry would be routed by the directory as follows:



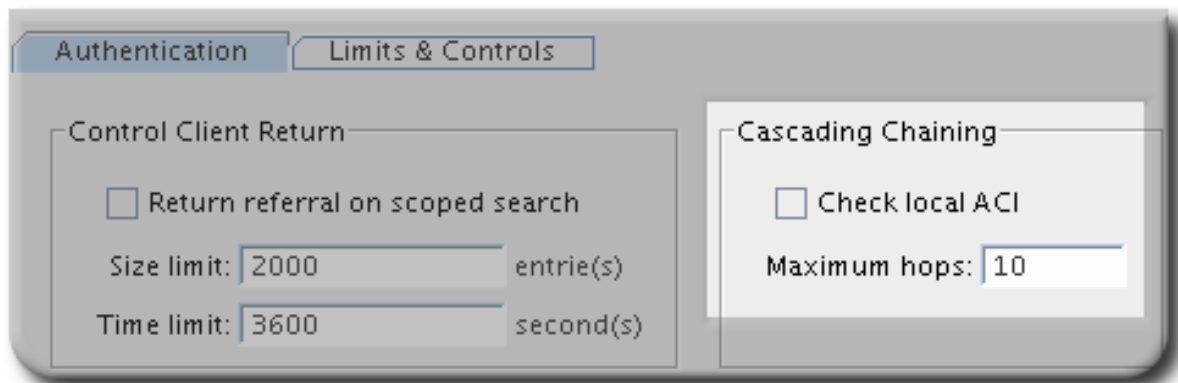
First, the client binds to Server A and chains to Server B using Database Link 1. Then Server B chains to the target database on Server C using Database Link 2 to access the data in the **ou=people, l=europa, dc=example, dc=com** branch. Because at least two hops are required for the directory to service the client request, this is considered a cascading chain.

2.4.2. Configuring Cascading Chaining Using the Console

1. Select the **Configuration** tab. Expand the **Data** folder in the left pane, and select the suffix, then the database link.



2. Click the **Limits and Controls** tab in the right navigation pane.
3. Select the **Check local ACI** check box to enable the evaluation of local ACIs on the intermediate database links involved in the cascading chain. Selecting this check box may require adding the appropriate local ACIs to the database link.



4. Enter the maximum number of times a database link can point to another database link in the **Maximum hops** field.

By default, the maximum is ten hops. After ten hops, a loop is detected by the server, and an error is returned to the client application.

2.4.3. Configuring Cascading Chaining from the Command Line

To configure a cascade of database links through the command line:

1. Point one database link to the URL of the server containing the intermediate database link.

To create a cascading chain, the **nsFarmServerURL** attribute of one database link must contain the URL of the server containing another database link. Suppose the database link on the server called **example1.com** points to a database link on the server called **africa.example.com**. For example, the **cn=database_link, cn=chaining database, cn=plugins, cn=config** entry of the database link on Server 1 would contain the following:

```
nsFarmServerURL: ldap://africa.example.com:389/
```

2. Configure the intermediate database link or links (in the example, Server 2) to transmit the Proxy Authorization Control.

By default, a database link does not transmit the Proxy Authorization Control. However, when one database link contacts another, this control is used to transmit information needed by the final destination server. The intermediate database link needs to transmit this control. To configure the database link to transmit the proxy authorization control, add the following to the **cn=config,cn=chaining database,cn=plugins,cn=config** entry of the intermediate database link:

```
nsTransmittedControls: 2.16.840.1.113730.3.4.12
```

The OID value represents the Proxy Authorization Control. For more information about chaining LDAP controls, see [Section 2.3.2.2, “Chaining LDAP Controls”](#).

3. Create a proxy administrative user ACI on all intermediate database links.

The ACI must exist on the server that contains the intermediate database link that checks the rights of the first database link before translating the request to another server. For example, if Server 2 does not check the credentials of Server 1, then anyone could bind as **anonymous** and pass a proxy authorization control allowing them more administrative privileges than appropriate. The proxy ACI prevents this security breach.

1. Create a database, if one does not already exist, on the server containing the intermediate database link. This database will contain the admin user entry and the ACI. For information about creating a database, see [Section 2.2.1, “Creating Databases”](#).
2. Create an entry that corresponds to the administrative user in the database.
3. Create an ACI for the administrative user that targets the appropriate suffix. This ensures the administrator has access only to the suffix of the database link. For example:

```
aci: (targetattr = "*")(version 3.0; acl "Proxied authorization
for database links";
    allow (proxy) userdn = "ldap:///cn=proxy admin,cn=config");
```

This ACI is like the ACI created on the remote server when configuring simple chaining.



WARNING

Carefully examine access controls when enabling chaining to avoid giving access to restricted areas of the directory. For example, if a default proxy ACI is created on a branch, the users that connect through the database link will be able to see all entries below the branch. There may be cases when not all of the subtrees should be viewed by a user. To avoid a security hole, create an additional ACI to restrict access to the subtree.

4. Enable local ACI evaluation on all intermediate database links.

To confirm that the proxy administrative ACI is used, enable evaluation of local ACIs on all

intermediate database links involved in chaining. Add the following attribute to the **cn=database_link, cn=chaining database, cn=plugins, cn=config** entry of each intermediate database link:

```
nsCheckLocalACI: on
```

Setting this attribute to **on** in the **cn=default instance config, cn=chaining database, cn=plugins, cn=config** entry means that all new database link instances will have the **nsCheckLocalACI** attribute set to **on** in their **cn=database_link, cn=chaining database, cn=plugins, cn=config** entry.

5. Create client ACIs on all intermediate database links and the final destination database.

Because local ACI evaluation is enabled, the appropriate client application ACIs must be created on all intermediate database links, as well as the final destination database. To do this on the intermediate database links, first create a database that contains a suffix that represents a root suffix of the final destination suffix.

For example, if a client request made to the **c=africa, ou=people, dc=example, dc=com** suffix is chained to a remote server, all intermediate database links need to contain a database associated with the **dc=example, dc=com** suffix.

Add any client ACIs to this superior suffix entry. For example:

```
aci: (targetattr = "*")(version 3.0; acl "Client authentication for
database link users";
    allow (all) userdn = "ldap:///uid=* ,cn=config";)
```

This ACI allows client applications that have a **uid** in the **cn=config** entry of Server 1 to perform any type of operation on the data below the **ou=people, dc=example, dc=com** suffix on server three.

2.4.4. Detecting Loops

An LDAP control included with Directory Server prevents loops. When first attempting to chain, the server sets this control to be the maximum number of hops, or chaining connections, allowed. Each subsequent server decrements the count. If a server receives a count of **0**, it determines that a loop has been detected and notifies the client application.

The number of hops allowed is defined using the **nsHopLimit** attribute. If not specified, the default value is **10**.

To use the control, add the following OID to the **nsTransmittedControl** attribute in the **cn=config, cn=chaining database, cn=plugins, cn=config** entry:

```
nsTransmittedControl: 1.3.6.1.4.1.1466.29539.12
```

If the control is not present in the configuration file of each database link, loop detection will not be implemented.

2.4.5. Summary of Cascading Chaining Configuration Attributes

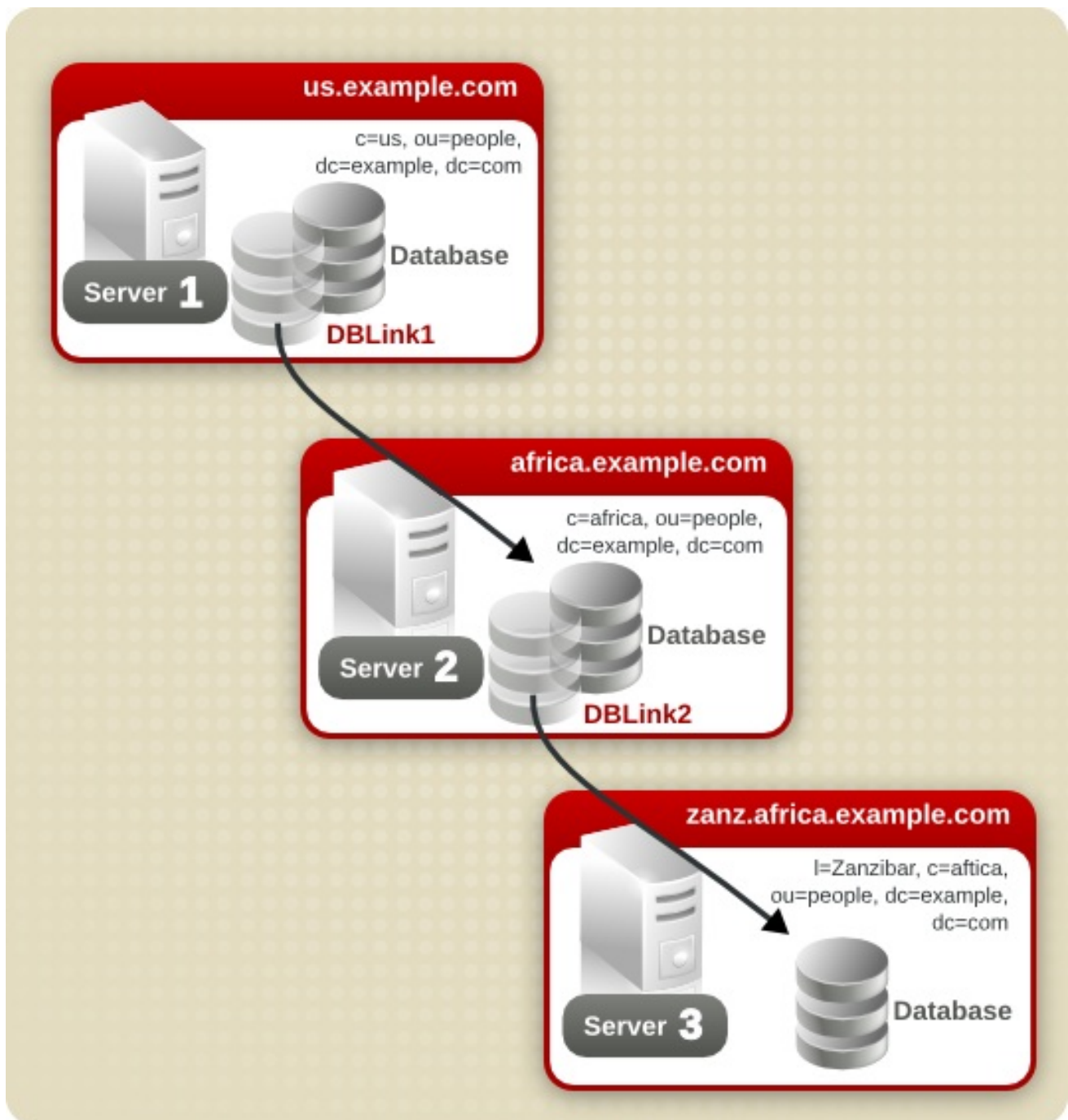
The following table describes the attributes used to configure intermediate database links in a cascading chain:

Table 2.5. Cascading Chaining Configuration Attributes

Attribute	Description
nsFarmServerURL	URL of the server containing the next database link in the cascading chain.
nsTransmittedControls	<p>Enter the following OIDs to the database links involved in the cascading chain:</p> <pre>nsTransmittedControls: 2.16.840.1.113730.3.4.12 nsTransmittedControls: 1.3.6.1.4.1.1466.29539.12</pre> <p>The first OID corresponds to the Proxy Authorization Control. The second OID corresponds to the Loop Detection Control.</p>
aci	<p>This attribute must contain the following ACI:</p> <pre>aci: (targetattr = "*")(version 3.0; acl "Proxied authorization for database links"; allow (proxy) userdn = "ldap:///cn=proxy admin,cn=config");)</pre>
nsCheckLocalACI	<p>To enable evaluation of local ACIs on all database links involved in chaining, turn local ACI evaluation on, as follows:</p> <pre>nsCheckLocalACI: on</pre>

2.4.6. Cascading Chaining Configuration Example

To create a cascading chain involving three servers as in the diagram below, the chaining components must be configured on all three servers.



- [Section 2.4.6.1, “Configuring Server One”](#)
- [Section 2.4.6.2, “Configuring Server Two”](#)
- [Section 2.4.6.3, “Configuring Server Three”](#)

2.4.6.1. Configuring Server One

1. Run **ldapmodify** and specify the configuration information for the database link, **DBLink1**, on Server 1:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

```
dn: cn=DBLink1,cn=chaining database,cn=plugins,cn=config
changetype: add
```

```

objectclass: top
objectclass: extensibleObject
objectclass: nsBackendInstance
nsslapd-suffix: c=africa,ou=people,dc=example,dc=com
nsfarmserverurl: ldap://africa.example.com:389/
nsMultiplexorBindDN: cn=server1 proxy admin,cn=config
nsMultiplexorCredentials: secret
cn: DBLink1
nsCheckLocalACI:off

dn: cn=c=africa\,ou=people\,dc=example\,dc=com,cn=mapping
tree,cn=config
changetype: add
objectclass: nsMappingTree
nsslapd-state: backend
nsslapd-backend: DBLink1
nsslapd-parent-suffix: ou=people,dc=example,dc=com
cn: c=africa\,ou=people\,dc=example\,dc=com

```

The first section creates the entry associated with **DBLink1**. The second section creates a new suffix, allowing the server to direct requests made to the database link to the correct server. The **nsCheckLocalACI** attribute does not need to be configured to check local ACIs, as this is only required on the database link, **DBLink2**, on Server 2.

2. To implement loop detection, to specify the OID of the loop detection control in the **nsTransmittedControl** attribute stored in **cn=config, cn=chaining database, cn=plugins, cn=config** entry on Server 1.

```

dn: cn=config,cn=chaining database,cn=plugins,cn=config
changetype: modify
add: nsTransmittedControl
nsTransmittedControl: 1.3.6.1.4.1.1466.29539.12

```

As the **nsTransmittedControl** attribute is usually configured by default with the loop detection control OID **1.3.6.1.4.1.1466.29539.12** value, it is wise to check beforehand whether it already exists. If it does exist, this step is not necessary.

2.4.6.2. Configuring Server Two

1. Create a proxy administrative user on Server 2. This administrative user will be used to allow Server 1 to bind and authenticate to Server 2. It is useful to choose a proxy administrative user name which is specific to Server 1, as it is the proxy administrative user which will allow server *one* to bind to Server 2. Create the proxy administrative user, as follows:

```

dn: cn=server1 proxy admin,cn=config
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: server1 proxy admin
sn: server1 proxy admin
userPassword: secret
description: Entry for use by database links

```

**WARNING**

Do not use the Directory Manager or Administrator ID user as the proxy administrative user on the remote server. This creates a security hole.

2. Configure the database link, **DBLink2**, on Server 2:

```
dn: cn=DBLink2,cn=chaining database,cn=plugins,cn=config
objectclass: top
objectclass: extensibleObject
objectclass: nsBackendInstance
nsslapd-suffix: l=Zanzibar,c=africa,ou=people,dc=example,dc=com
nsfarmserverurl: ldap://zanz.africa.example.com:389/
nsMultiplexorBindDN: cn=server2 proxy admin,cn=config
nsMultiplexorCredentials: secret
cn: DBLink2
nsCheckLocalACI:on

dn:
cn=l=Zanzibar\,c=africa\,ou=people\,dc=example\,dc=com,cn=mapping
tree,cn=config
objectclass: top
objectclass: extensibleObject
objectclass: nsMappingTree
nsslapd-state: backend
nsslapd-backend: DBLink2
nsslapd-parent-suffix: c=africa,ou=people,dc=example,dc=com
cn: l=Zanzibar\,c=africa\,ou=people\,dc=example\,dc=com
```

Since database link DBLink2 is the intermediate database link in the cascading chaining configuration, set the ***nsCheckLocalACI*** attribute to **on** to allow the server to check whether it should allow the client and proxy administrative user access to the database link.

3. The database link on Server 2 must be configured to transmit the proxy authorization control and the loop detection control. To implement the proxy authorization control and the loop detection control, specify both corresponding OIDs. Add the following information to the **cn=config,cn=chaining database,cn=plugins,cn=config** entry on Server 2:

```
dn: cn=config,cn=chaining database,cn=plugins,cn=config
changetype: modify
add: nsTransmittedControl
nsTransmittedControl: 2.16.840.1.113730.3.4.12
nsTransmittedControl: 1.3.6.1.4.1.1466.29539.12
```

nsTransmittedControl: 2.16.840.1.113730.3.4.12 is the OID for the proxy authorization control. **nsTransmittedControl: 1.3.6.1.4.1.1466.29539.12** is the or the loop detection control.

Check beforehand whether the loop detection control is already configured, and adapt the above command accordingly.

4. Configure the ACIs. On Server 2, ensure that a suffix exists above the **l=Zanzibar,c=africa,ou=people,dc=example,dc=com** suffix, so that the following actions are possible:
 - o Add the database link suffix
 - o Add a local proxy authorization ACI to allow Server 1 to connect using the proxy authorization administrative user created on Server 2
 - o Add a local client ACI so the client operation succeeds on Server 2, and it can be forwarded to server three. This local ACI is needed because local ACI checking is turned on for the **DBLink2** database link.

Both ACIs will be placed on the database that contains the **c=africa,ou=people,dc=example,dc=com** suffix.



NOTE

To create these ACIs, the database corresponding to the **c=africa,ou=people,dc=example,dc=com** suffix must already exist to hold the entry. This database needs to be associated with a suffix above the suffix specified in the *nsslapd-suffix* attribute of each database link. That is, the suffix on the final destination server should be a sub suffix of the suffix specified on the intermediate server.

1. Add the local proxy authorization ACI to the **c=africa,ou=people,dc=example,dc=com** entry:

```
aci:(targetattr="*")
(target="l=Zanzibar,c=africa,ou=people,dc=example,dc=com")
(version 3.0; acl "Proxied authorization for database
links"; allow (proxy)
userdn = "ldap:///cn=server1 proxy admin,cn=config");)
```

2. Then add the local client ACI that will allow the client operation to succeed on Server 2, given that ACI checking is turned on. This ACI is the same as the ACI created on the destination server to provide access to the **l=Zanzibar,c=africa,ou=people,dc=example,dc=com** branch. All users within **c=us,ou=people,dc=example,dc=com** may need to have update access to the entries in **l=Zanzibar,c=africa,ou=people,dc=example,dc=com** on server three. Create the following ACI on Server 2 on the **c=africa,ou=people,dc=example,dc=com** suffix to allow this:

```
aci:(targetattr="*")
(target="l=Zanzibar,c=africa,ou=people,dc=example,dc=com")
(version 3.0; acl "Client authorization for database links";
allow (all)
userdn = "ldap:///uid=*,c=us,ou=people,dc=example,dc=com");)
```

This ACI allows clients that have a UID in **c=us,ou=people,dc=example,dc=com** on Server 1 to perform any type of operation on the **l=Zanzibar,c=africa,ou=people,dc=example,dc=com** suffix tree on server three. If there are users on Server 2 under a different suffix that will require additional rights on server three, it may be necessary to add additional client ACIs on Server 2.

2.4.6.3. Configuring Server Three

1. Create an administrative user on server three for Server 2 to use for proxy authorization:

```
dn: cn=server2 proxy admin,cn=config
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: server2 proxy admin
sn: server2 proxy admin
userPassword: secret
description: Entry for use by database links
```

2. Then add the same local proxy authorization ACL to server three as on Server 2. Add the following proxy authorization ACL to the **l=Zanzibar,ou=people,dc=example,dc=com** entry:

```
aci: (targetattr = "*")(version 3.0; acl "Proxied authorization
for database links"; allow (proxy) userdn = "ldap:///cn=server2
proxy admin,cn=config";)
```

This ACL gives the Server 2 proxy admin read-only access to the data contained on the remote server, server three, within the **l=Zanzibar,ou=people,dc=example,dc=com** subtree only.

3. Create a local client ACL on the **l=Zanzibar,ou=people,dc=example,dc=com** subtree that corresponds to the original client application. Use the same ACL as the one created for the client on Server 2:

```
aci: (targetattr = "*")
(target="l=Zanzibar,c=africa,ou=people,dc=example,dc=com")
(version 3.0; acl "Client authentication for database link
users"; allow (all)
userdn = "ldap:///uid=*,c=us,ou=people,dc=example,dc=com";)
```

The cascading chaining configuration is now set up. This cascading configuration allows a user to bind to Server 1 and modify information in the **l=Zanzibar,c=africa,ou=people,dc=example,dc=com** branch on server three. Depending on your security needs, it may be necessary to provide more detailed access control.

2.5. USING REFERRALS

Referrals tell client applications which server to contact for a specific piece of information. This redirection occurs when a client application requests a directory entry that does not exist on the local server or when a database has been taken off-line for maintenance. This section contains the following information about referrals:

- [Section 2.5.1, “Starting the Server in Referral Mode”](#)
- [Section 2.5.2, “Setting Default Referrals”](#)
- [Section 2.5.3, “Creating Smart Referrals”](#)
- [Section 2.5.4, “Creating Suffix Referrals”](#)

For conceptual information on how to use referrals in the directory, see the *Directory Server Deployment Guide*.

2.5.1. Starting the Server in Referral Mode

Referrals are used to redirect client applications to another server while the current server is unavailable or when the client requests information that is not held on the current server. For example, starting Directory Server in referral mode while there are configuration changes being made to the Directory Server will refer all clients to another supplier while that server is unavailable. Starting the Directory Server in referral mode is done with the **refer** command.

Run **nsslapd** with the **refer** option.

```
/usr/sbin/ns-slapd refer -D /etc/dirsrv/slapd-instance_name [-p port] -r referral_url
```

- **/etc/dirsrv/slapd-*instance_name*** is the directory where the Directory Server configuration files are. This is the default location on Red Hat Enterprise Linux 6 (64-bit).
- *port* is the optional port number of the Directory Server to start in referral mode.
- *referral_url* is the referral returned to clients. The format of an LDAP URL is covered in [Appendix C, LDAP URLs](#).

2.5.2. Setting Default Referrals

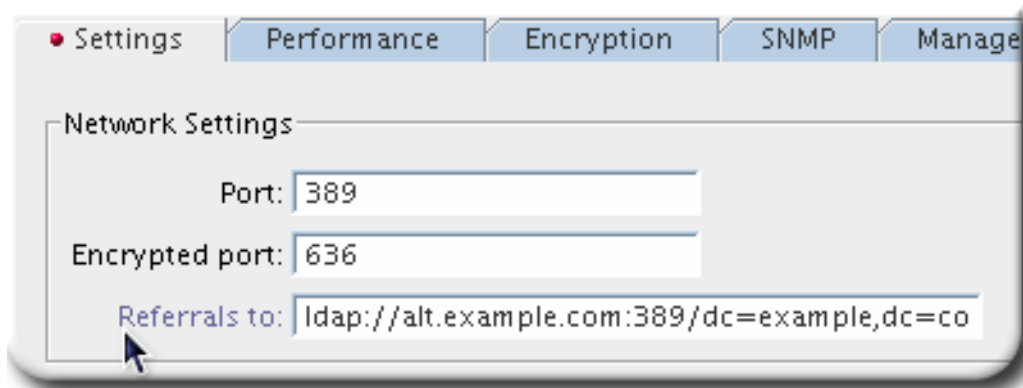
Default referrals are returned to client applications that submit operations on a DN not contained within any of the suffixes maintained by the directory. The following procedures describes setting a default referral for the directory using the console and the command-line utilities.

2.5.2.1. Setting a Default Referral Using the Console

1. In the Directory Server Console, select the **Configuration** tab.
2. Select the top entry in the navigation tree in the left pane.



3. Select the **Settings** tab in the right pane.
4. Enter an LDAP URL for the referral.



Enter multiple referral URLs separated by spaces and in quotes:

```
"ldap://dir1.example.com:389/dc=example,dc=com"
"ldap://dir2.example.com/"
```

For more information about LDAP URLs, see [Appendix C, LDAP URLs](#).

2.5.2.2. Setting a Default Referral from the Command Line

ldapmodify can add a default referral to the **cn=config** entry in the directory's configuration file. For example, to add a new default referral from one Directory Server, **dir1.example.com**, to a server named **dir2.example.com**, add a new line to the **cn=config** entry.

1. Run the **ldapmodify** utility and add the default referral to the **dir2.example.com** server:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=config
changetype: modify
replace: nsslapd-referral
nsslapd-referral: ldap://dir2.example.com/
```

After adding the default referral to the **cn=config** entry of the directory, the directory will return the default referral in response to requests made by client applications. The Directory Server does not need to be restarted.

2.5.3. Creating Smart Referrals

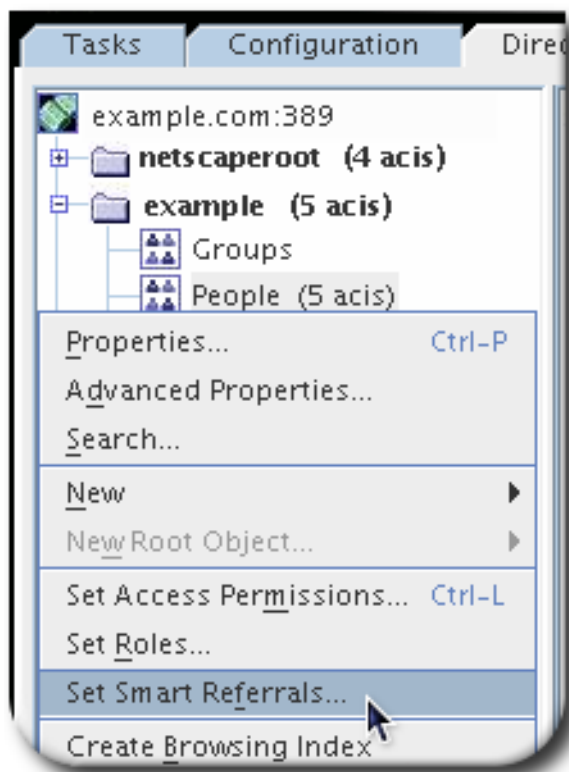
Smart referrals map a directory entry or directory tree to a specific LDAP URL. Using smart referrals, client applications can be referred to a specific server or a specific entry on a specific server.

For example, a client application requests the directory entry **uid=jdoe,ou=people,dc=example,dc=com**. A smart referral is returned to the client that points to the entry **cn=john doe,o=people,l=europe,dc=example,dc=com** on the server **directory.europe.example.com**.

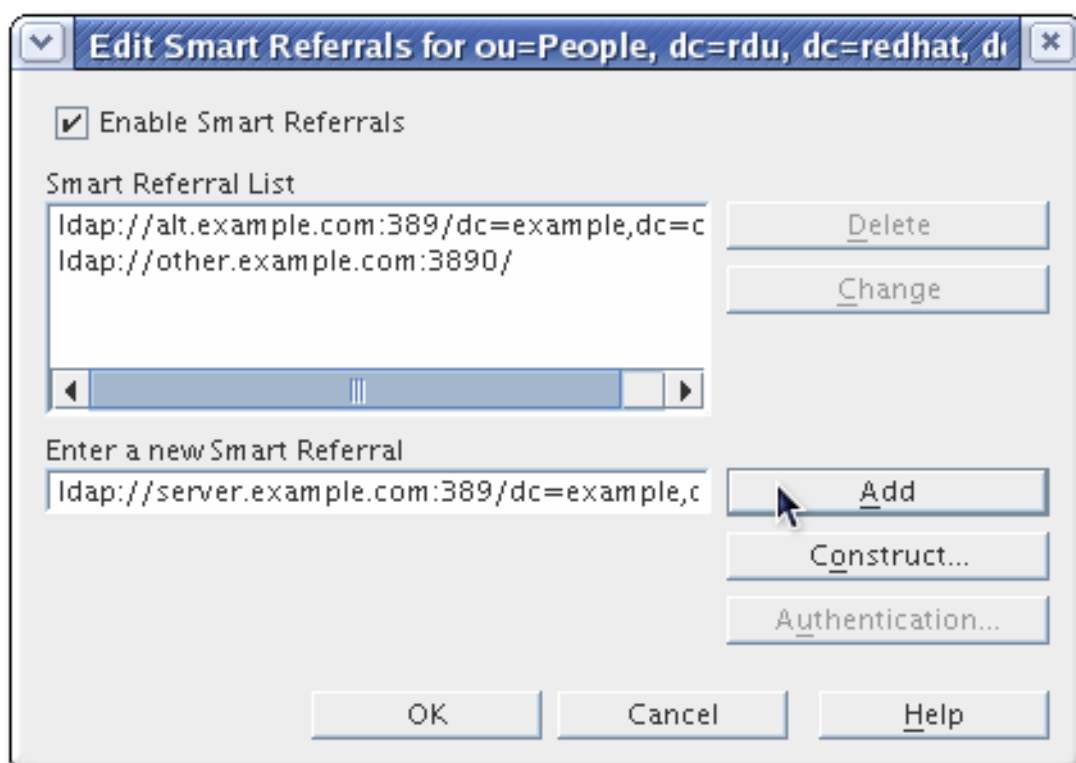
The way the directory uses smart referrals conforms to the standard specified in RFC 2251 section 4.1.11. The RFC can be downloaded at <http://www.ietf.org/rfc/rfc2251.txt>.

2.5.3.1. Creating Smart Referrals Using the Directory Server Console

1. In the Directory Server Console, select the **Directory** tab.
2. Browse through the tree in the left navigation pane, and select the entry for which to add the referral.
3. Right-click the entry, and select **Set Smart Referrals**.



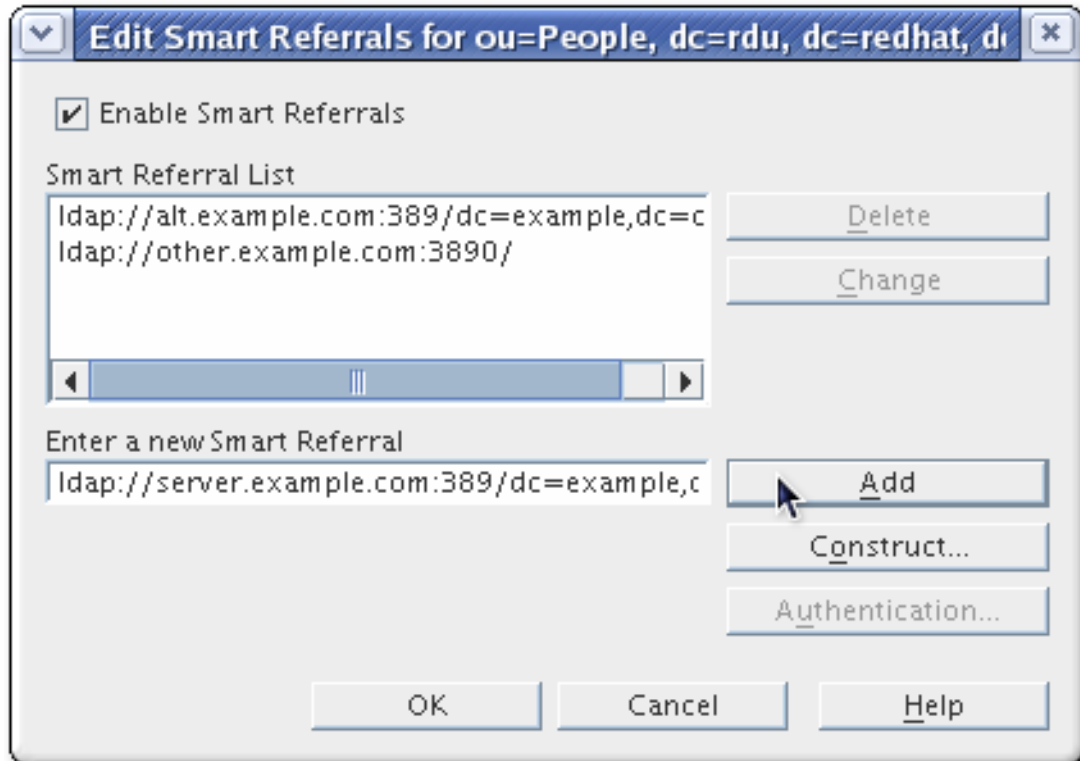
4. Select the **Enable Smart Referral** check box. (Unchecking the option removes all smart referrals from the entry and deletes the **referral** object class from the entry.)



5. In the **Enter a new Smart Referral** field, enter a referral in the LDAP URL format, and then click **Add**. The LDAP URL must be in the following format:

```
ldap://server:port/[optional_dn]
```

server can be the host name, IPv4 address, or IPv6 address for the server. *optional_dn* is the explicit DN for the server to return to the requesting client application.

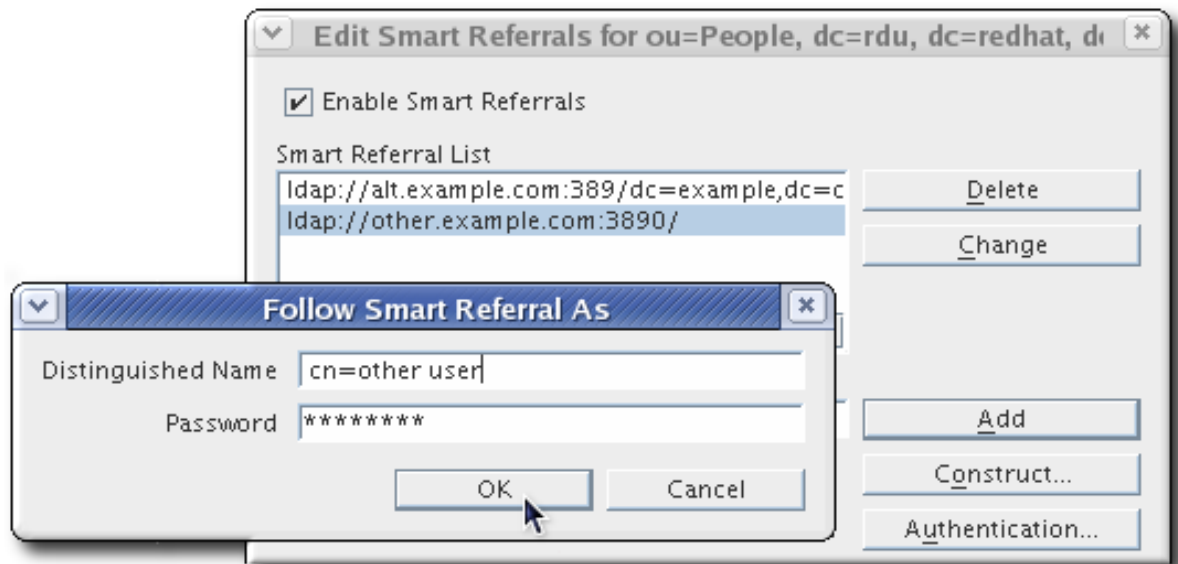


Construct opens a wizard to direct the process of adding a referral.

The **Smart Referral List** lists the referrals currently in place for the selected entry. The entire list of referrals is returned to client applications in response to a request with the **Return Referrals for All Operations** or **Return Referrals for Update Operations** options in the **Suffix Settings** tab, which is available under the **Configuration** tab.

To modify the list, click **Edit** to edit the selected referral or **Delete** to delete the selected referral.

6. To set the referral to use different authentication credentials, click **Authentication**, and specify the appropriate DN and password. This authentication remains valid only until the Console is closed; then it is reset to the same authentication used to log into the Console.



2.5.3.2. Creating Smart Referrals from the Command Line

Use the **ldapmodify** command-line utility to create smart referrals from the command line.

To create a smart referral, create the relevant directory entry, and add the **referral** object class. This object class allows a single attribute, **ref**. The **ref** attribute must contain an LDAP URL.

For example, add the following to return a smart referral for an existing entry, **uid=jdoe**:

```
dn: uid=jdoe,ou=people,dc=example,dc=com
objectclass: referral
ref:
ldap://directory.europe.example.com/cn=john%20doe,ou=people,l=europe,dc=example,dc=com
```



NOTE

Any information after a space in an LDAP URL is ignored by the server. For this reason, use **%20** instead of spaces in any LDAP URL used as a referral.

To add the entry **uid=jdoe,ou=people,dc=example,dc=com** with a referral to **directory.europe.example.com**, include the following in the LDIF file before importing:

```
dn: uid=jdoe,ou=people,dc=example,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
objectclass: referral
cn: john doe
sn: doe
uid: jdoe
ref:
ldap://directory.europe.example.com/cn=john%20doe,ou=people,l=europe,dc=example,dc=com
```

Use the **-M** option with **ldapmodify** when there is already a referral in the DN path. For more information on smart referrals, see the *Directory Server Deployment Guide*.

2.5.4. Creating Suffix Referrals

The following procedure describes creating a referral in a *suffix*. This means that the suffix processes operations using a referral rather than a database or database link.



WARNING

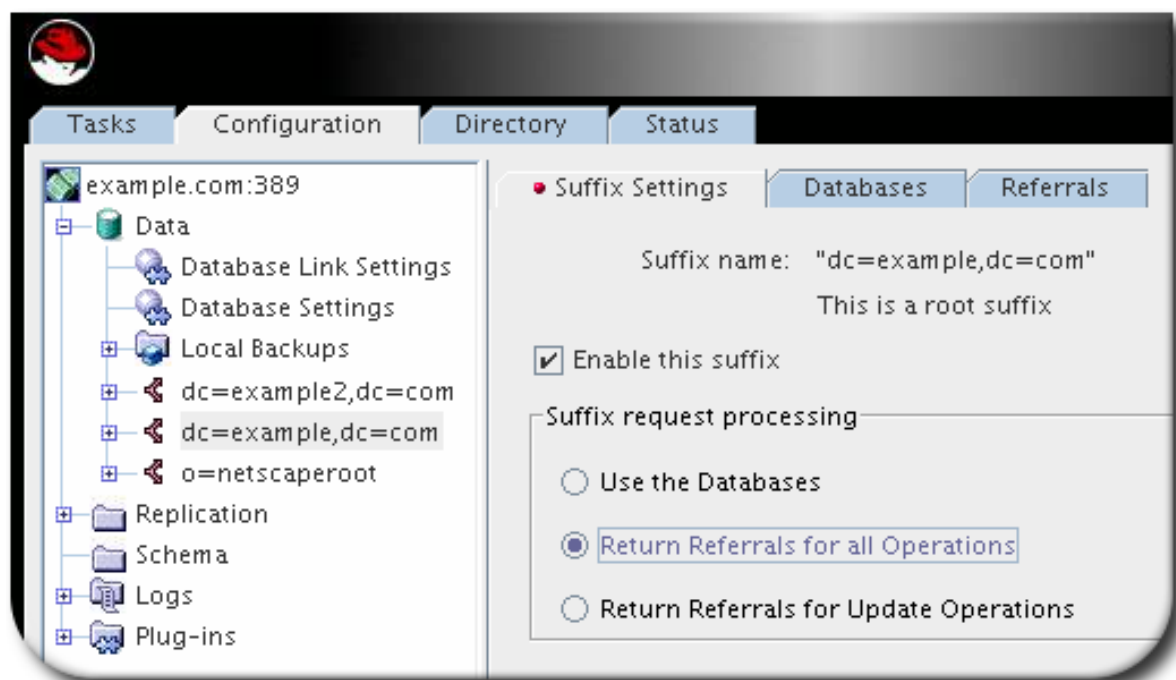
When a suffix is configured to return referrals, the ACIs contained by the database associated with the suffix are ignored.

2.5.4.1. Creating Suffix Referrals Using the Console

Referrals can be used to point a client application temporarily to a different server. For example, adding a referral to a suffix so that the suffix points to a different server allows the database associated with the suffix is taken off-line for maintenance without affecting the users of the Directory Server database.

To set referrals in a suffix:

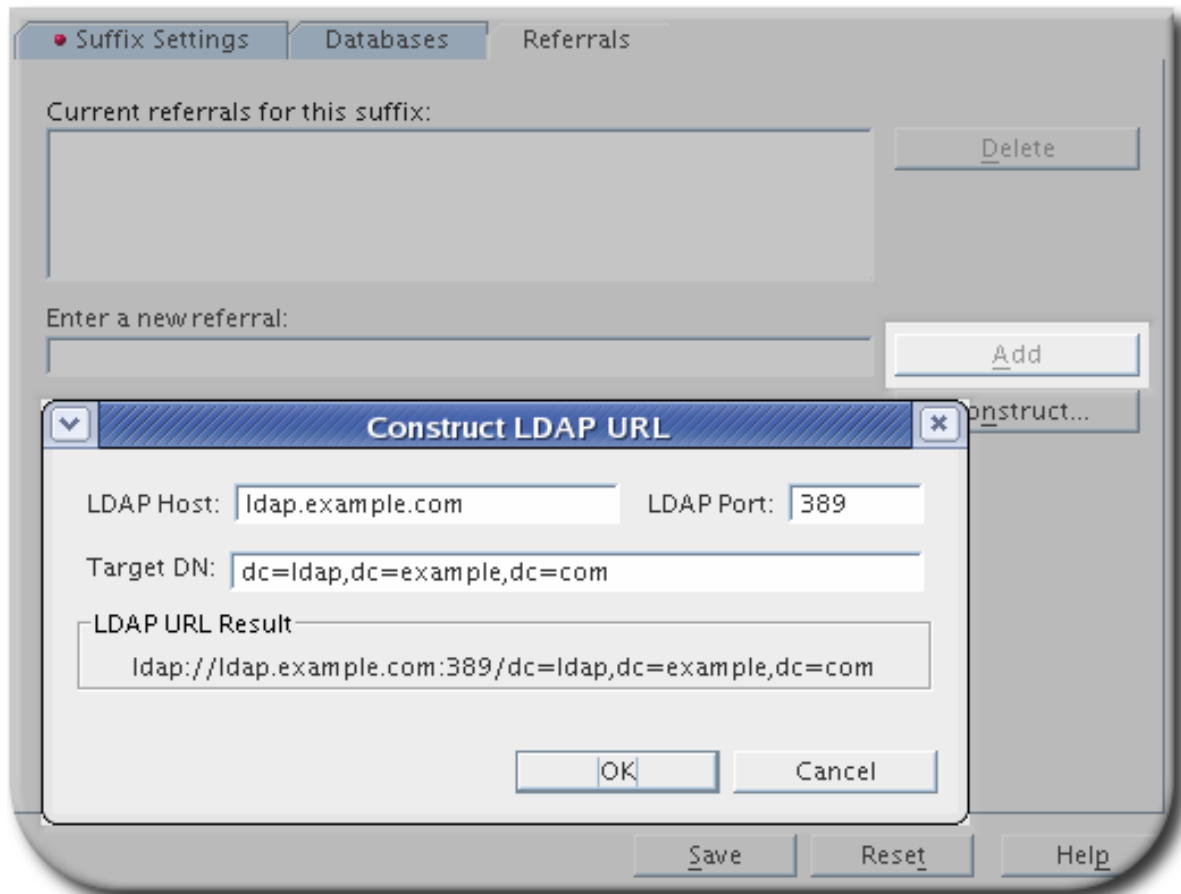
1. In the Directory Server Console, select the **Configuration** tab.
2. Under **Data** in the left pane, select the suffix for which to add a referral.
3. Click the **Suffix Settings** tab, and select the **Return Referrals for ... Operations** radio button.



Selecting **Return Referrals for Update Operations** means that the directory redirects only update and write requests to a read-only database. For example, there may be a local copy

of directory data, and that data should be available for searches but not for updates, so it is replicated across several servers. Enabling referrals for that Directory Server only for update requests means that when a client asks to update an entry, the client is referred to the server that owns the data, where the modification request can proceed.

4. Click the **Referrals** tab. Enter an LDAP URL in the^[1] in the **Enter a new referral** field, or click **Construct** to create an LDAP URL.



5. Click **Add** to add the referral to the list.

You can enter multiple referrals. The directory returns the entire list of referrals in response to requests from client applications.

2.5.4.2. Creating Suffix Referrals from the Command Line

Add a suffix referral to the root or sub suffix entry in the directory configuration file under the **cn=mapping tree,cn=config** branch.

Run **ldapmodify** and add a suffix referral to the **ou=people,dc=example,dc=com** root suffix:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=ou=people,dc=example,dc=com,cn=mapping tree,cn=config
changetype: add
objectclass: extensibleObject
objectclass: nsMappingTree
nsslapd-state: referral
nsslapd-referral: ldap://zanzibar.com/
```

The ***nsslapd-state*** attribute is set to **referral**, meaning that a referral is returned for requests made to this suffix. The ***nsslapd-referral*** attribute contains the LDAP URL of the referral returned by the suffix, in this case a referral to the **zanzibar.com** server.

The ***nsslapd-state*** attribute can also be set to **referral on update**. This means that the database is used for all operations except update requests. When a client application makes an update request to a suffix set to **referral on update**, the client receives a referral.

For more information about the suffix configuration attributes, see [Table 2.1, “Suffix Attributes”](#).

□ Unlike the standard LDAP URL format, the URL of the remote server does not specify a suffix. It has the form **ldap://server:port/**, where *server* can be the host name, IPv4 address, or IPv6 address.

[1] [Appendix C, LDAP URLs](#) has more information about the structure of LDAP URLs.

CHAPTER 3. CREATING DIRECTORY ENTRIES

This chapter discusses how to use the Directory Server Console and the `ldapmodify` and `ldapdelete` command-line utilities to modify the contents of your directory.

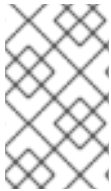
Entries stored in Active Directory can be added to the Directory Server through Windows Sync; see [Chapter 12, *Synchronizing Red Hat Directory Server with Microsoft Active Directory*](#) for more information on adding or modifying synchronized entries through Windows User Sync.

3.1. MANAGING ENTRIES FROM THE DIRECTORY CONSOLE

You can use the **Directory** tab and the **Property Editor** on the Directory Server Console to add, modify, or delete entries individually.

To add several entries simultaneously, use the command-line utilities described in [Section 3.2, “Managing Entries from the Command Line”](#).

- [Section 3.1.1, “Creating a Root Entry”](#)
- [Section 3.1.2, “Creating Directory Entries”](#)
- [Section 3.1.3, “Modifying Directory Entries”](#)
- [Section 3.1.4, “Deleting Directory Entries”](#)



NOTE

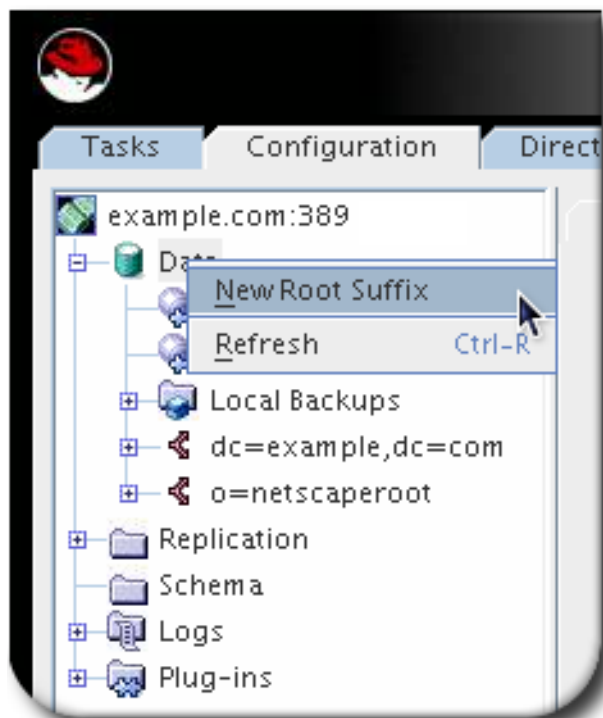
You cannot modify your directory unless the appropriate access control rules have been set. For information on creating access control rules for your directory, see [Chapter 13, *Managing Access Control*](#).

3.1.1. Creating a Root Entry

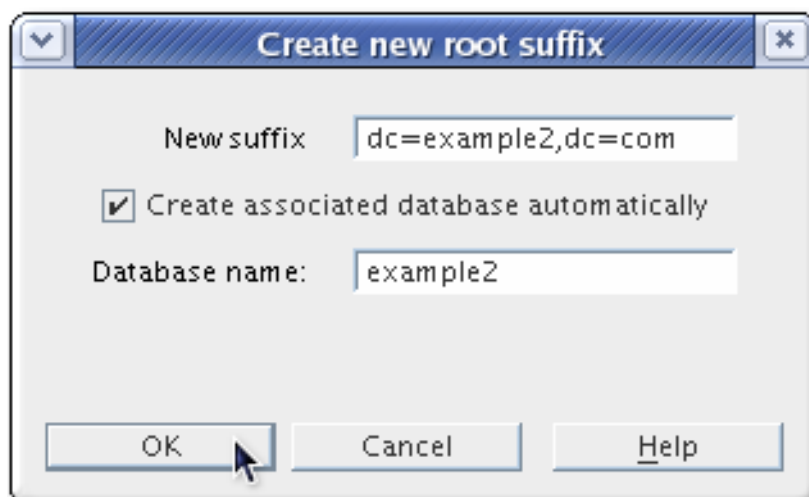
Each time a new database is created, it is associated with the suffix that will be stored in the database. The directory entry representing that suffix is not automatically created.

To create a root entry for a database:

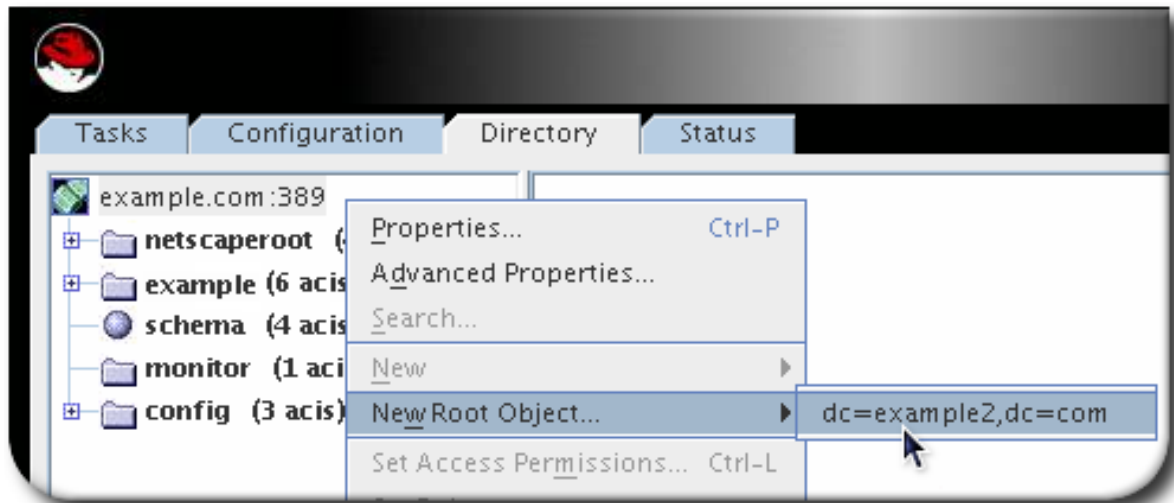
1. In the Directory Server Console, select the **Configuration** tab.
2. Right-click on the **Data** entry in the left menu, and select **New Root Suffix** from the menu.



3. Fill in the new suffix and database information.

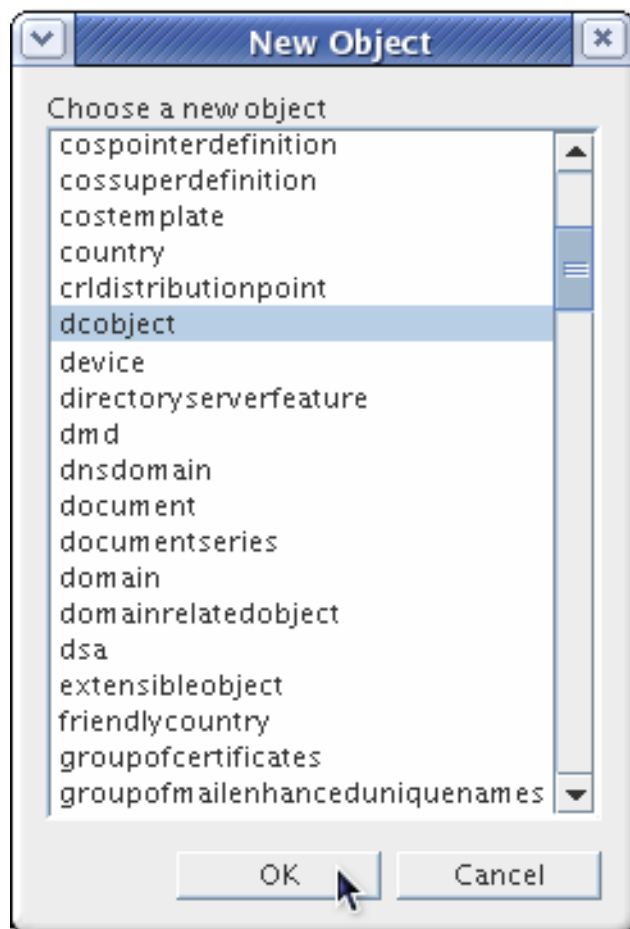


4. In the **Directory** tab, right-click the top object representing the Directory Server, and choose **New Root Object**.



The secondary menu under **New Root Object** displays the new suffixes without a corresponding directory entry. Choose the suffix corresponding to the entry to create.

5. In the **New Object** window, select the object class corresponding to the new entry.



The object class must contain the attribute used to name the suffix. For example, if the entry corresponds to the suffix **ou=people,dc=example,dc=com**, then choose the **organizationalUnit** object class or another object class that allows the **ou** attribute.

6. Click **OK** in the New Object window.

The **Property Editor** for the new entry opens. You can either accept the current values by clicking **OK** or modify the entry, as explained in [Section 3.1.3, “Modifying Directory Entries”](#).

3.1.2. Creating Directory Entries

Directory Server Console offers predefined templates, with preset forms, for new directory entries.

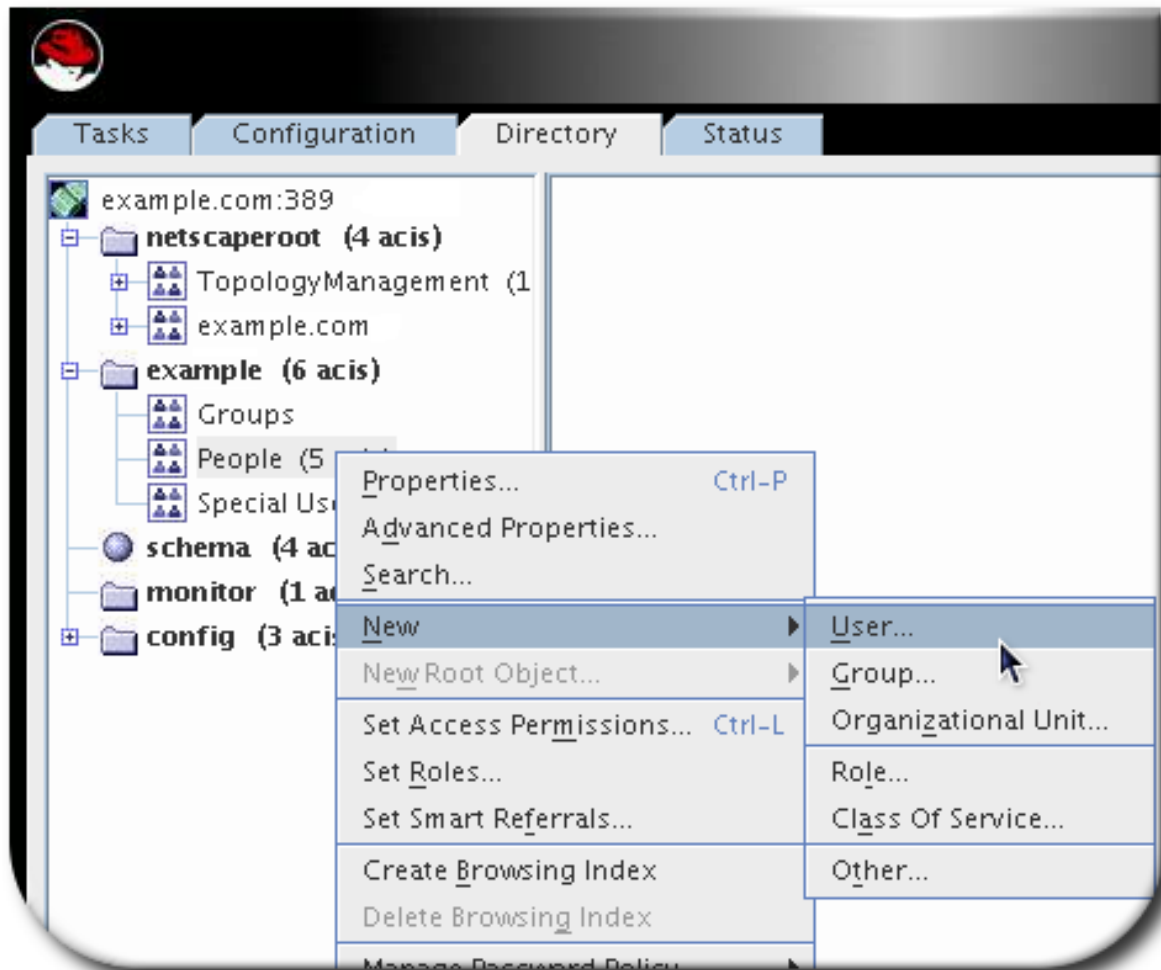
[Table 3.1, “Entry Templates and Corresponding Object Classes”](#) shows what type of object class is used for each template.

Table 3.1. Entry Templates and Corresponding Object Classes

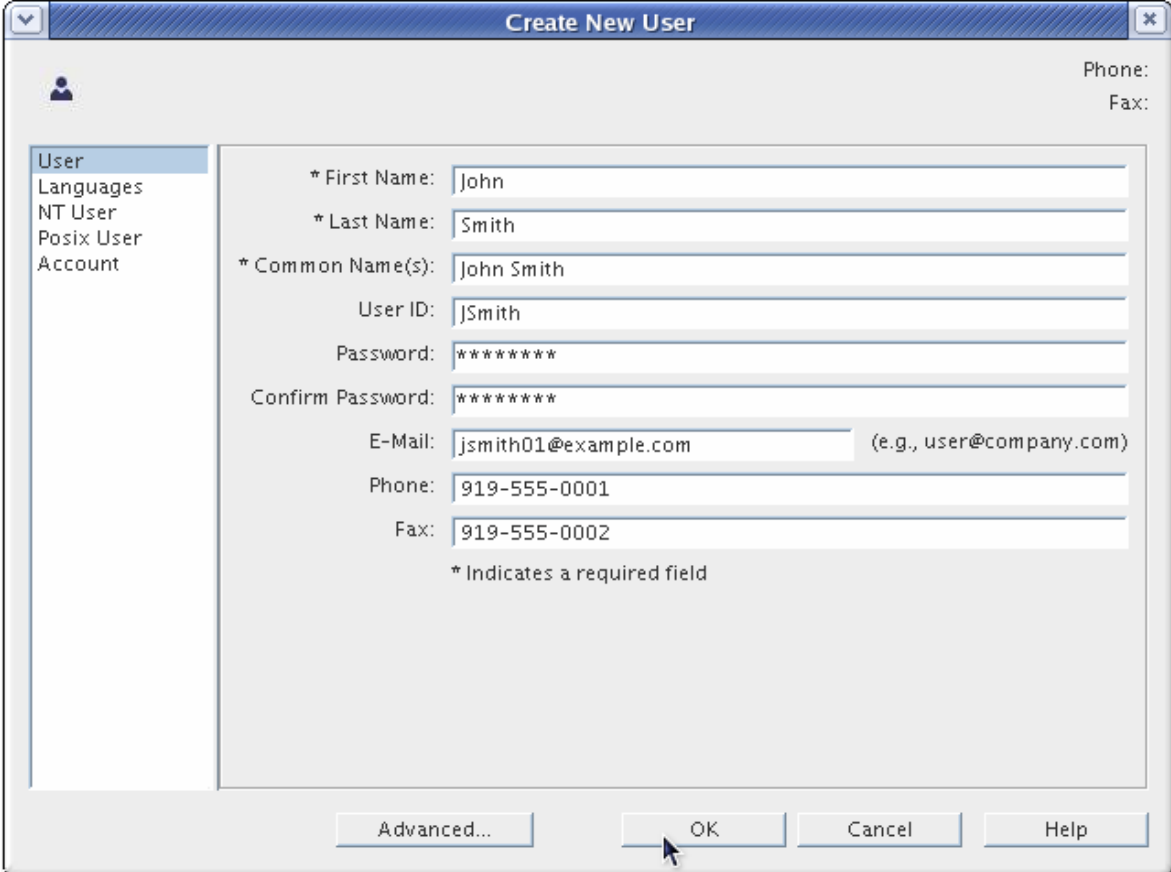
Template	Object Class
User	inetOrgPerson
Group	groupOfUniqueNames
Organizational Unit	organizationalUnit
Role	nsRoleDefinition
Class of Service	cosSuperDefinition

Another type, **Other** allows any kind of entry to be created by allowing users to select the specific object classes and attributes to apply.

1. In the Directory Server Console, select the **Directory** tab.
2. In the left pane, right-click the main entry to add the new entry, and select the type of entry: **User**, **Group**, **Organizational Unit**, **Role**, **Class of Service**, or **Other**.

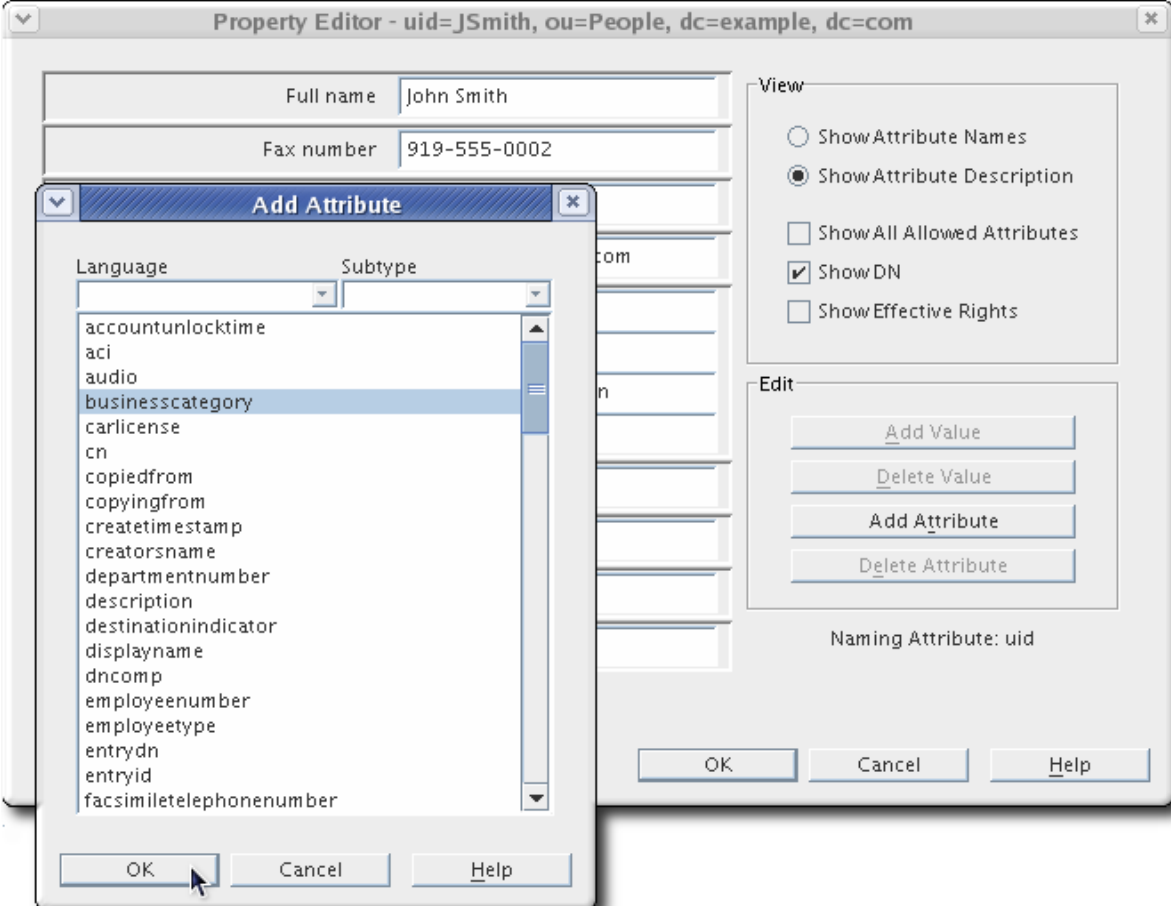


3. If the new entry type was **Other**, then a list of object classes opens. Select an object class from the list to define the new entry.
4. Supply a value for all the listed attributes. Required attributes are marked with an asterisk (*).



The "Create New User" dialog box is shown. It has a sidebar on the left with a tree view containing "User", "Languages", "NT User", "Posix User", and "Account". The "User" item is selected. The main area contains several text input fields: "* First Name:" (John), "* Last Name:" (Smith), "* Common Name(s):" (John Smith), "User ID:" (JSmith), "Password:" (masked with asterisks), "Confirm Password:" (masked with asterisks), "E-Mail:" (jsmith01@example.com), "Phone:" (919-555-0001), and "Fax:" (919-555-0002). A note at the bottom states "* Indicates a required field". At the bottom of the dialog are four buttons: "Advanced...", "OK", "Cancel", and "Help". A mouse cursor is pointing at the "OK" button.

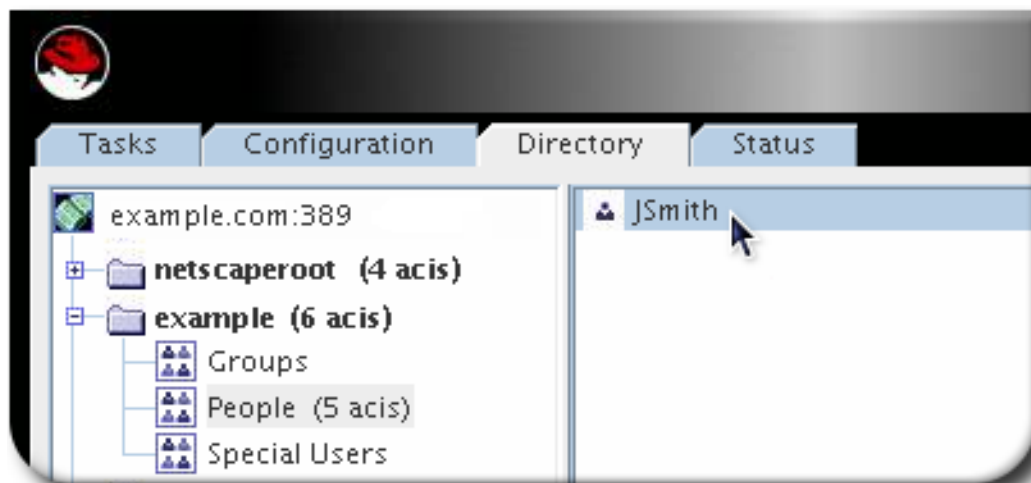
5. To display the full list of attributes available for the object class (entry type), click the **Advanced** button.



The "Property Editor" dialog box is shown, titled "Property Editor - uid=JSmith, ou=People, dc=example, dc=com". It displays fields for "Full name" (John Smith) and "Fax number" (919-555-0002). On the right, there is a "View" section with radio buttons for "Show Attribute Names" and "Show Attribute Description" (selected), and checkboxes for "Show All Allowed Attributes", "Show DN" (checked), and "Show Effective Rights". Below this is an "Edit" section with buttons for "Add Value", "Delete Value", "Add Attribute", and "Delete Attribute". At the bottom, it says "Naming Attribute: uid". Overlaid on top of the "Property Editor" is the "Add Attribute" dialog box. It has a "Language" dropdown and a "Subtype" dropdown. Below these is a list of attributes: accountunlocktime, aci, audio, businesscategory (highlighted), carlicense, cn, copiedfrom, copyingfrom, createtimestamp, creatorsname, departmentnumber, description, destinationindicator, displayName, dncomp, employeenumber, employeetype, entrydn, entryid, and facsimiletelephonenumber. At the bottom of the "Add Attribute" dialog are "OK", "Cancel", and "Help" buttons. A mouse cursor is pointing at the "OK" button.

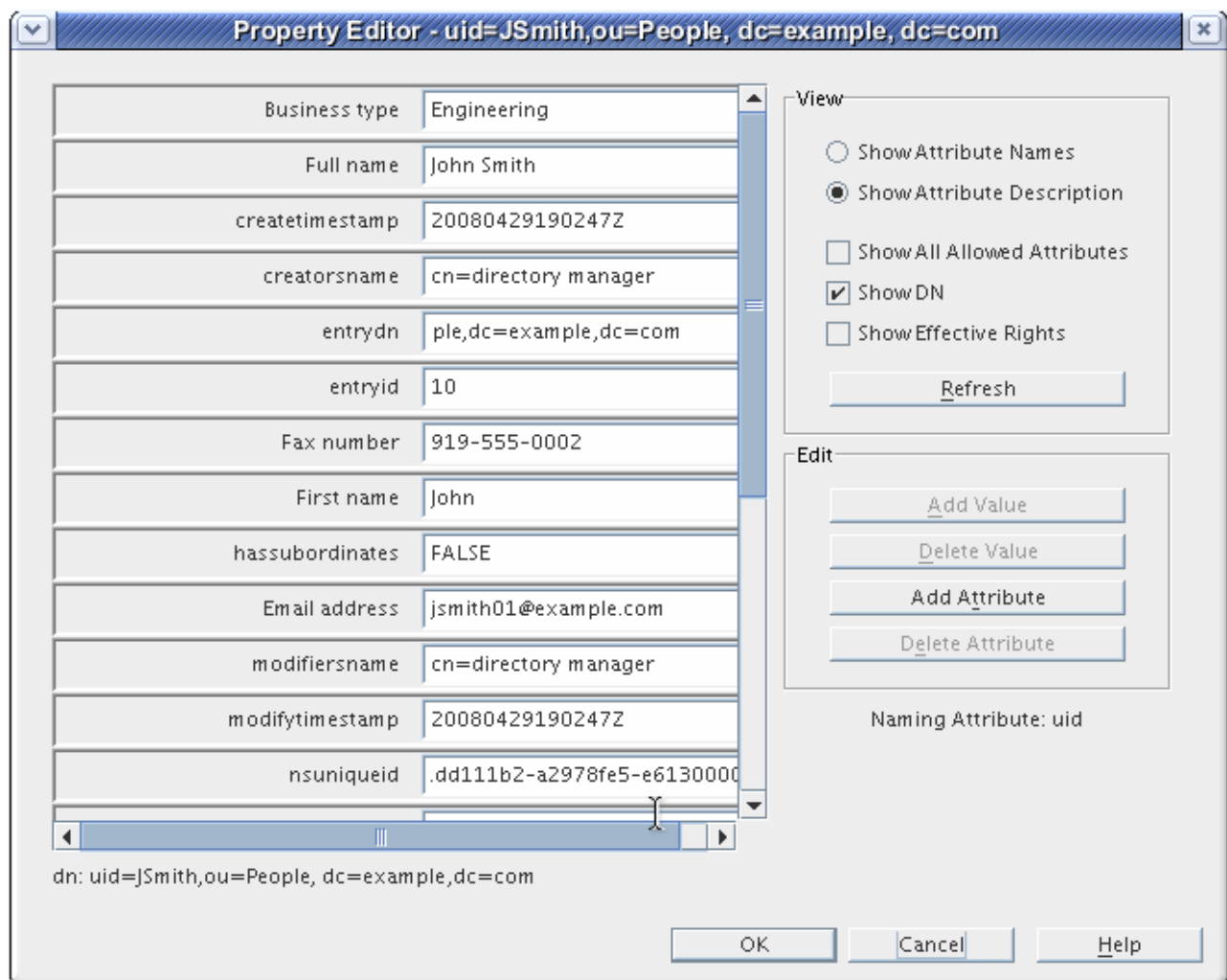
In the **Property Editor**, select any additional attributes, and fill in the attribute values.

- Click **OK** to save the entry. The new entry is listed in the right pane.



3.1.3. Modifying Directory Entries

Modifying directory entries in Directory Server Console uses a dialog window called the **Property Editor**. The **Property Editor** contains the list of object classes and attributes belonging to an entry and can be used to edit the object classes and attributes belonging to that entry by adding and removing object classes, attributes and attribute values, and attribute subtypes.



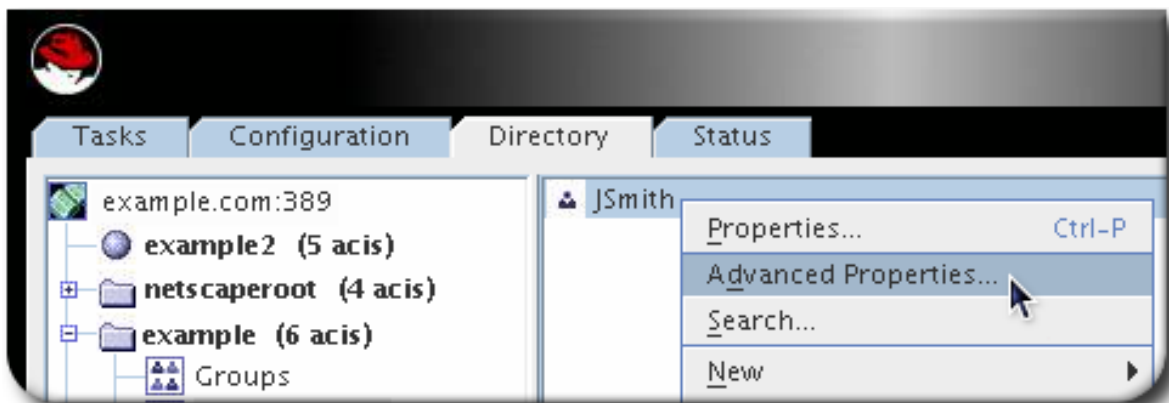
The **Property Editor** can be opened in several ways:

- From the **Directory** tab, by right-clicking an entry, and selecting **Advanced Properties** from the pop-up menu.
- From the **Directory** tab, by double-clicking an entry and clicking the **Advanced** button
- From the **Create...** new entry forms, by clicking the **Advanced** button
- From the **New Object** window, by clicking **OK**

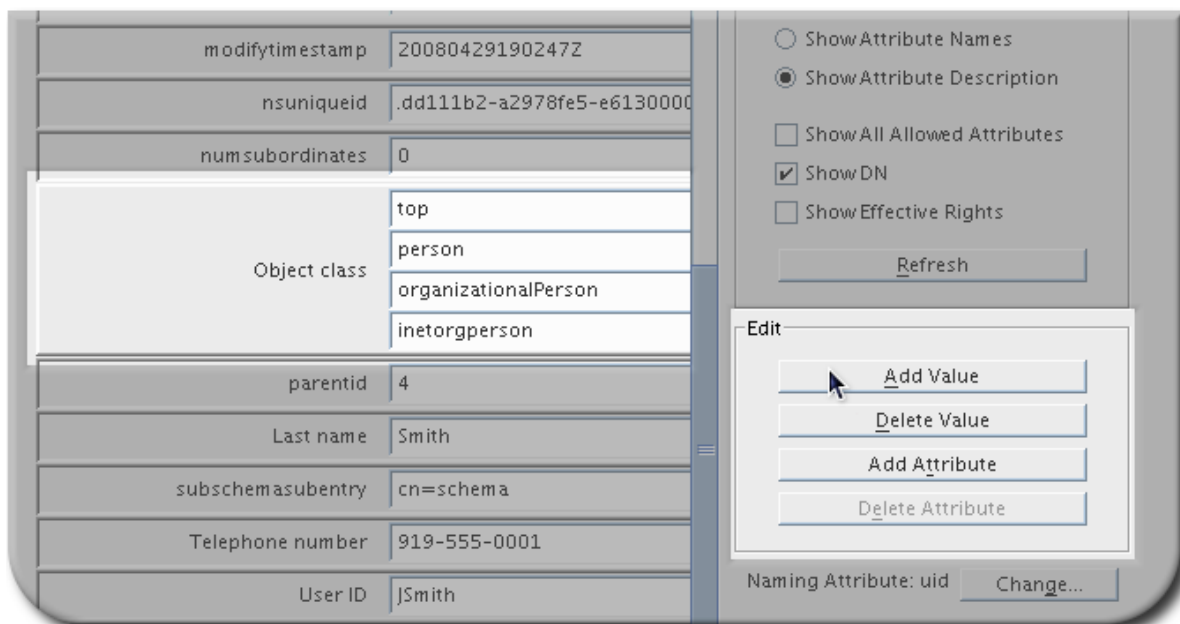
3.1.3.1. Adding or Removing an Object Class to an Entry

To add an object class to an entry:

1. In the **Directory** tab of the Directory Server Console, right-click the entry to modify, and select **Advanced** from the pop-up menu.

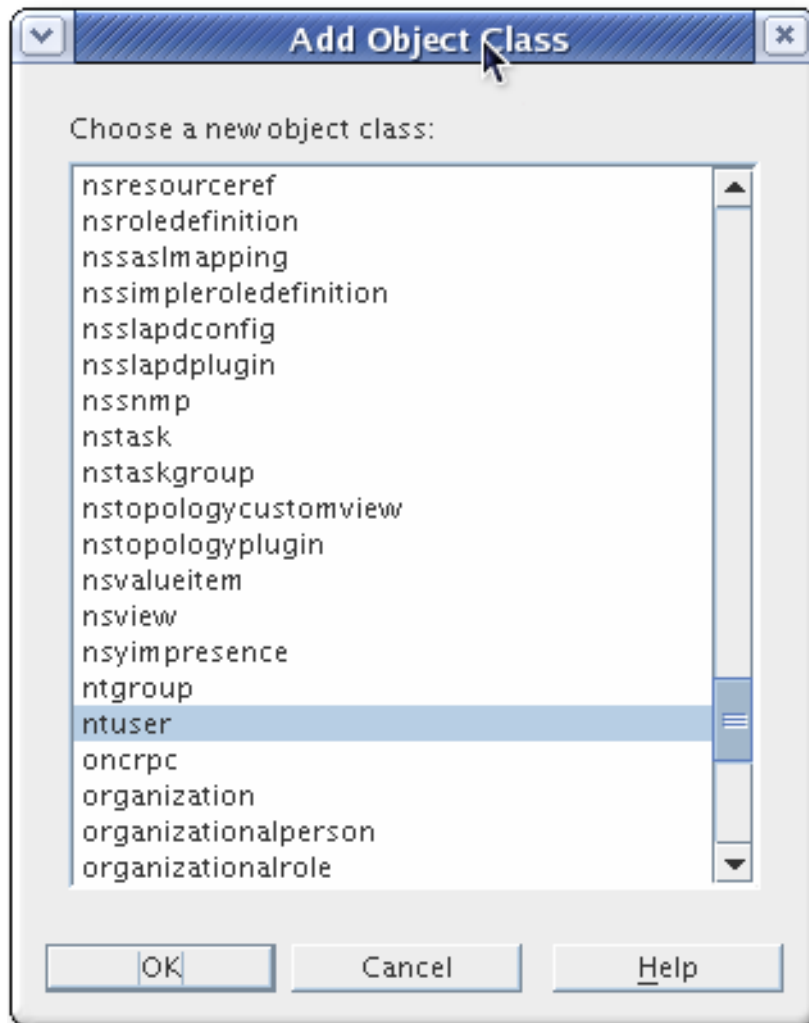


2. Select the object class field, and click **Add Value**.



The **Add Object Class** window opens. It shows a list of object classes that can be added to the entry.

3. Select the object class to add, and click **OK**.



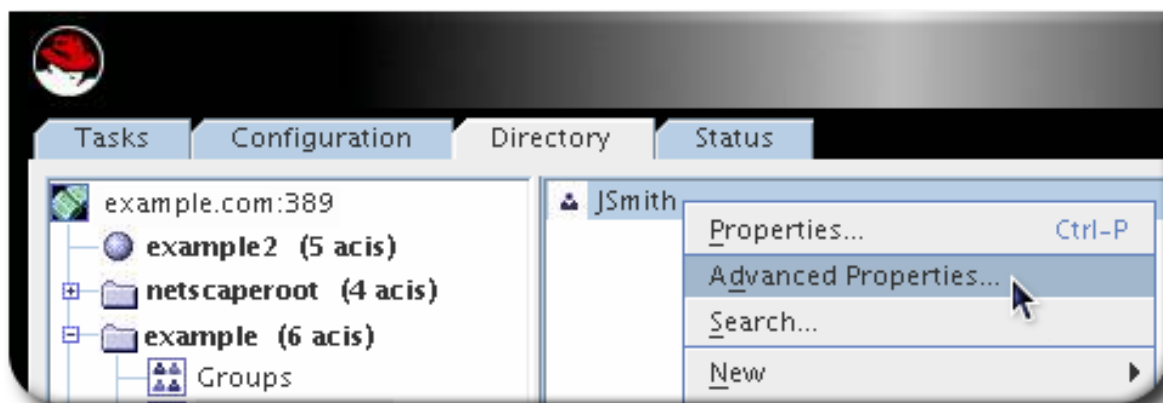
To remove an object class from an entry, click the text box for the object class to remove, and then click **Delete Value**.

3.1.3.2. Adding an Attribute to an Entry

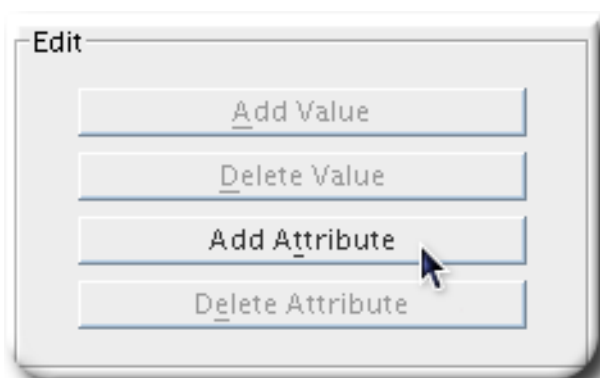
Before you can add an attribute to an entry, the entry must contain an object class that either requires or allows the attribute. See [Section 3.1.3.1, “Adding or Removing an Object Class to an Entry”](#) and [Chapter 8, *Managing the Directory Schema*](#) for more information.

To add an attribute to an entry:

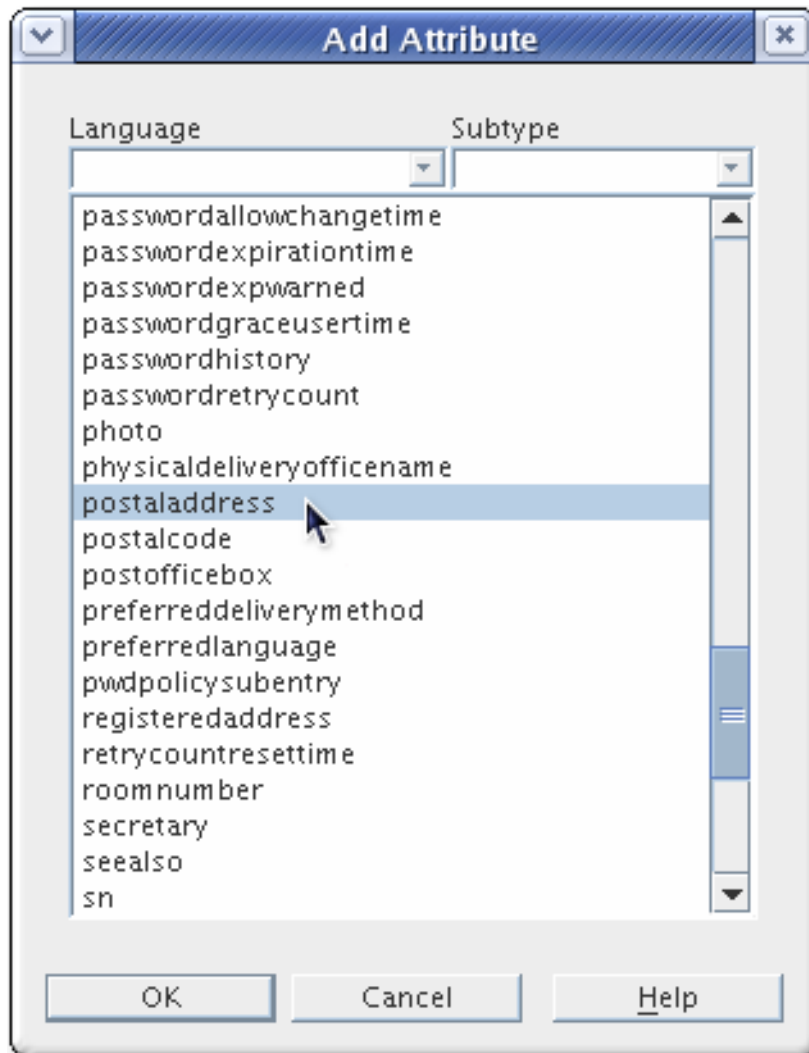
1. In the **Directory** tab of the Directory Server Console, right-click the entry to modify, and select **Advanced** from the pop-up menu.



2. Click **Add Attribute**.



3. Select the attribute to add from the list, and click **OK**.



NOTE

If the attribute you want to add is not listed, add the object class containing the attribute first, then add the attribute. See [Section 3.1.3.1, “Adding or Removing an Object Class to an Entry”](#) for instructions on adding an object class. If you do not know which object class contains the attribute you need, look up the attribute in the [Red Hat Directory Server 9 Configuration, Command, and File Reference](#), which lists the object classes which use that attribute.

4. Type in the value for the new attribute in the field to the right of the attribute name.

Last name	Smith
Mailing address	
modifiersname	cn=directory manager

To remove the attribute and all its values from the entry, select **Delete Attribute** from the **Edit** menu.

3.1.3.3. Adding Very Large Attributes

The configuration attribute ***nsslapd-maxbersize*** sets the maximum size limit for LDAP requests. The default configuration of Directory Server sets this attribute at 2 megabytes. LDAP add or modify operations will fail when attempting to add very large attributes that result in a request that is larger than 2 megabytes.

To add very large attributes, first change the setting for the ***nsslapd-maxbersize*** configuration attribute to a value larger than the largest LDAP request you will make.

When determining the value to set, consider *all* elements of the LDAP add and modify operations used to add the attributes, not just the single attribute. There are a number of different factors to consider, including the following:

- The size of each attribute name in the request
- The size of the values of each of the attributes in the request
- The size of the DN in the request
- Some overhead, usually 10 kilobytes

One common issue that requires increasing the ***nsslapd-maxbersize*** setting is using attributes which hold CRL values, such as ***certificateRevocationList***, ***authorityRevocationList***, and ***deltaRevocationList***.

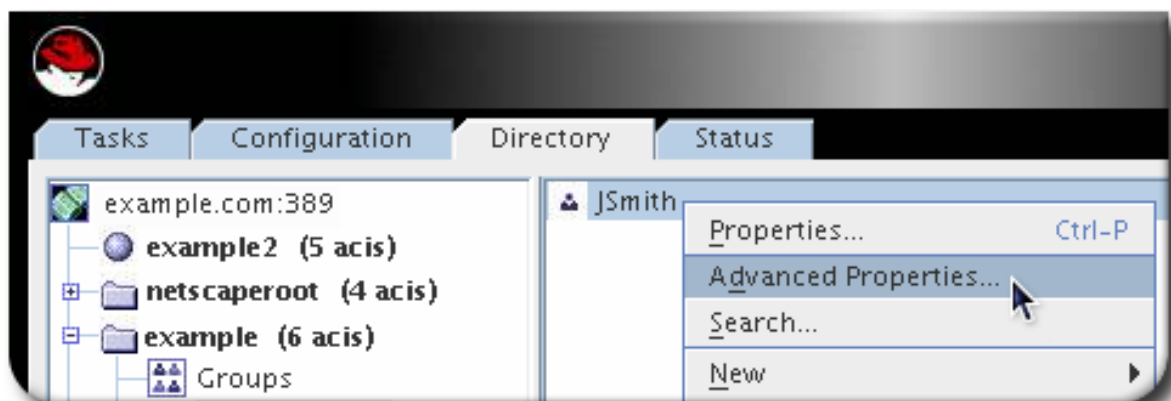
Note that this limit is additionally applied to replication processes.

For further information about the ***nsslapd-maxbersize*** attribute and a workaround in replication scenarios, see the corresponding section in the [Red Hat Directory Server Configuration, Command, and File Reference](#).

3.1.3.4. Adding Attribute Values

Multi-valued attributes allow multiple value for one attribute to be added to an entry.

1. In the **Directory** tab of the Directory Server Console, right-click the entry to modify, and select **Advanced** from the pop-up menu.



2. Select the attribute to which to add a value, and then click **Add Value**.

parentid	4
Mailing address	108 W. Callender St.
Last name	Smith
subschemasubentry	cn=schema

Edit

Add Value
Delete Value
Add Attribute
Delete Attribute

3. Type in the new attribute value.

Mailing address	108 W. Callender St.
-----------------	----------------------

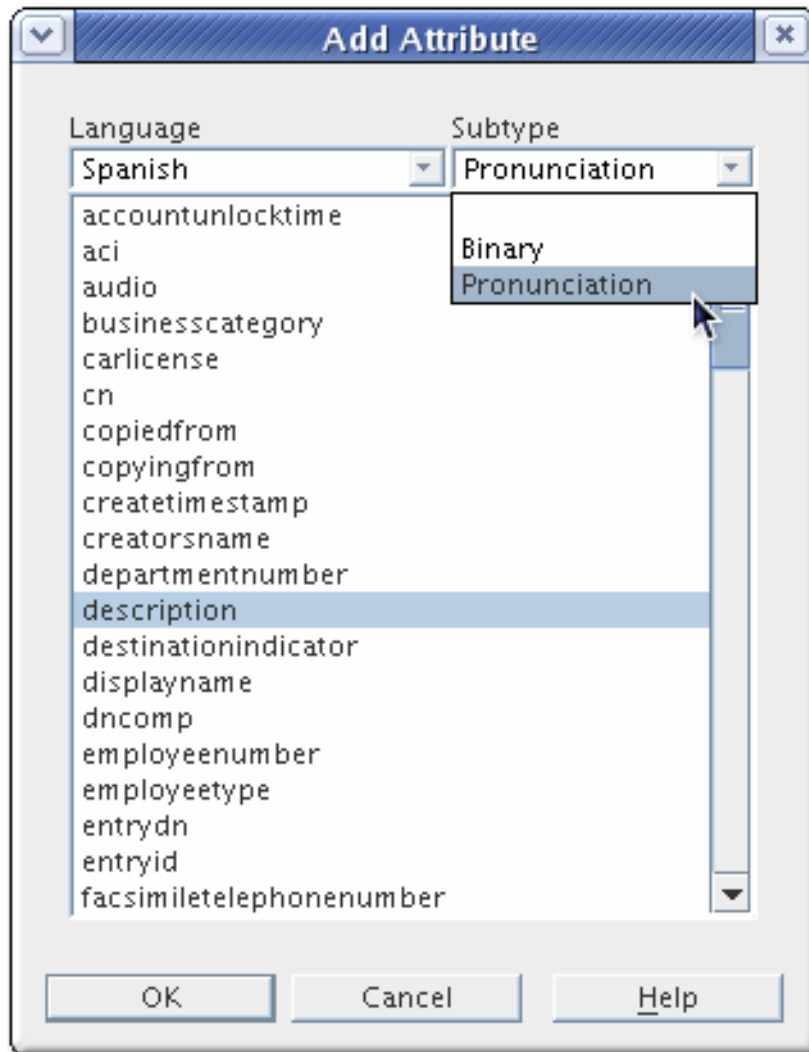
To remove an attribute value from an entry, click the text box of the attribute value to remove, and click **Delete Value**.

3.1.3.5. Adding an Attribute Subtype

A subtype allows the same entry value to be represented in different ways, such as providing a foreign-character set version. There are three different kinds of subtypes to attributes which can be added to an entry: language, binary, and pronunciation.

To add a subtype to an entry:

1. In the **Directory** tab of the Directory Server Console, right-click the entry to modify, and select **Properties** from the pop-up menu.
2. Click **Add Attribute**, and select the attribute to add from the list.
3. Add a language subtype by selecting a value from the **Language** drop-down list. Add either a binary or pronunciation subtype by selecting a value from the **Subtype** drop-down list.



Language Subtype

Sometimes a user's name can be more accurately represented in characters of a language other than the default language. For example, a user, Noriko, has a name in Japanese and prefers that her name be represented by Japanese characters when possible. You can select Japanese as a language subtype for the **givenname** attribute so that other users can search for her name in Japanese as well as English. For example:

```
givenname;lang-ja
```

To specify a language subtype for an attribute, add the subtype to the attribute name as follows:

```
attribute;lang-subtype:attribute value
```

attribute is the attribute being added to the entry and *subtype* is the two character abbreviation for the language. The supported language subtypes are listed in [Table D.2, "Supported Language Subtypes"](#).

Only one language subtype can be added per attribute *instance* in an entry. To assign multiple language subtypes, add another attribute instance to the entry, and then assign the new language subtype. For example, the following is illegal:

```
cn;lang-ja;lang-en-GB:value
```

Instead, use:


```
cn;lang-ja:ja-value  
cn;lang-en-GB:value
```

Binary Subtype

Assigning the binary subtype to an attribute indicates that the attribute value is binary, such as user certificates (**usercertificate;binary**).

Although you can store binary data within an attribute that does not contain the **binary** subtype (for example, **jpegphoto**), the **binary** subtype indicates to clients that multiple variants of the attribute type may exist.

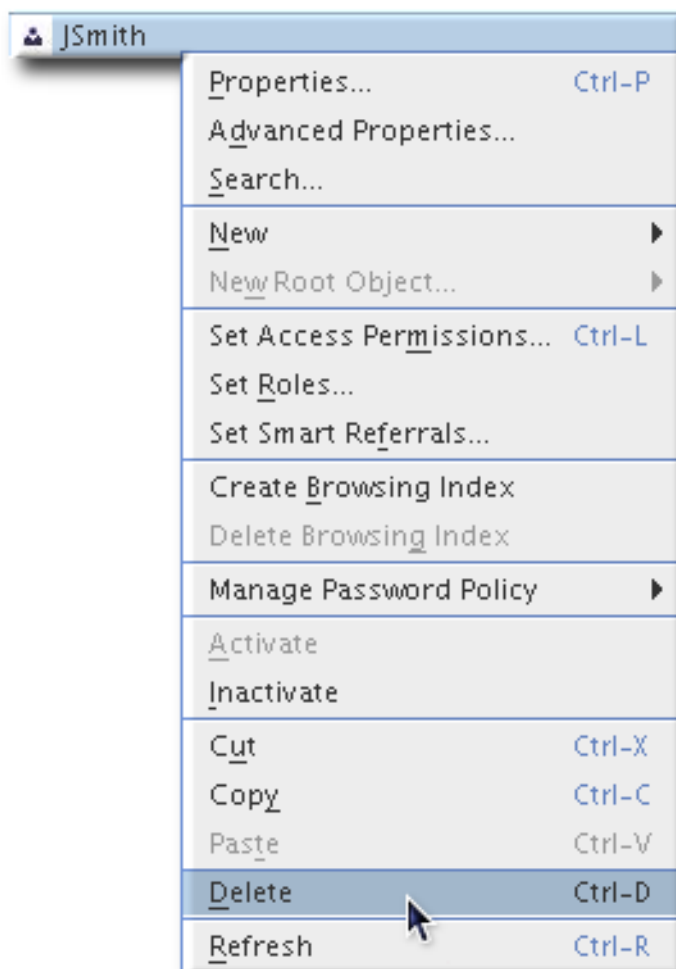
Pronunciation Subtype

Assigning the pronunciation subtype to an attribute indicates that the attribute value is a phonetic representation. The subtype is added to the attribute name as *attribute;phonetic*. This subtype is commonly used in combination with a language subtype for languages that have more than one alphabet, where one is a phonetic representation.

This subtype is useful with attributes that are expected to contain user names, such as **cn** or **givenname**. For example, **givenname;lang-ja;phonetic** indicates that the attribute value is the phonetic version of the user's Japanese name.

3.1.4. Deleting Directory Entries

1. In the Directory Server Console, select the **Directory** tab.
2. Right-click the entry to delete, and select **Delete** from the right-click menu.



WARNING

The server deletes the entry or entries immediately. There is no way to undo the delete operation.

3.2. MANAGING ENTRIES FROM THE COMMAND LINE

The command-line utilities allow you to manipulate the contents of your directory. They can be useful to write scripts to perform bulk management of the directory or to test the Directory Server. For example, you might want to ensure that it returns the expected information after you have made changes to access control information.

With command-line utilities, information can be provided directly from the command line or through an LDIF input file.

- [Section 3.2.1, “Providing Input from the Command Line”](#)
- [Section 3.2.2, “Creating a Root Entry from the Command Line”](#)
- [Section 3.2.3, “Adding Entries Using LDIF”](#)
- [Section 3.2.4, “Adding and Modifying Entries Using Idapmodify”](#)

- [Section 3.2.5, “Deleting Entries Using ldapdelete”](#)
- [Section 3.2.6, “Using Special Characters”](#)



NOTE

You cannot modify your directory unless the appropriate access control rules have been set. For information on creating access control rules for the directory, see [Chapter 13, Managing Access Control](#).

3.2.1. Providing Input from the Command Line

When you provide input to the **ldapmodify** and **ldapdelete** utilities directly from the command line, you must use LDIF statements. For detailed information on LDIF statements, see [Section 3.3, “Using LDIF Update Statements to Create or Modify Entries”](#).

The **ldapmodify** and **ldapdelete** utilities read the statements that you enter in exactly the same way as if they were read from a file. When all of the input has been entered, enter the character that the shell recognizes as the end of file (EOF) escape sequence. The utility then begins operations based on the supplied inputs.

While the EOF escape sequence depends on the type of machine, the EOF escape sequence almost always control-D (^D).

For example, to input some LDIF update statements to **ldapmodify**:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=Barry Nixon,ou=people,dc=example,dc=com
changetype: modify
delete: telephonenumber
-
add: manager
manager: cn=Harry Cruise,ou=people,dc=example,dc=com
^D
```

When adding an entry from the command line or from LDIF, make sure that an entry representing a subtree is created before new entries are created under that branch. For example, to place an entry in a **People** subtree, create an entry representing that subtree before creating entries within the subtree. For example:

```
dn: dc=example,dc=com
dn: ou=People,dc=example,dc=com
...People subtree entries. ...
dn: ou=Group,dc=example,dc=com
...Group subtree entries. ...
```

3.2.2. Creating a Root Entry from the Command Line

The **ldapmodify** command-line utility can be used to create a new root entry in a database. For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: Suffix_Name
```

```
changetype: add  
objectclass: newobjectclass
```

The DN corresponds to the DN of the root or sub-suffix contained by the database. The *newobjectclass* value depends upon the type of object class you are adding to the database. You may need to specify additional required attributes depending on the type of root object being added.



NOTE

You can use **ldapmodify** to add root objects only if you have one database per suffix. If you create a suffix that is stored in several databases, you must use the **ldif2db** utility with the **-noption** parameter to specify the database that will hold the new entries. For information, see [Section 4.1.6, “Importing from the Command Line”](#).

3.2.3. Adding Entries Using LDIF

You can use an LDIF file to add multiple entries or to import an entire database. To add entries using an LDIF file and the Directory Server Console:

1. Define the entries in an LDIF file.

LDIF files are described in [Appendix B, LDAP Data Interchange Format](#).

2. Import the LDIF file from the Directory Server Console.

See [Section 4.1.4, “Importing a Database from the Console”](#) for information about LDIF file formats. When you import the LDIF file, select **Append to database** in the **Import** dialog box so that the server will only import entries that do not currently exist in the directory.

You can also add entries described in an LDIF file from the command line using the **ldapmodify** command with the **-f** option.

3.2.4. Adding and Modifying Entries Using ldapmodify

The **ldapmodify** command can add and modify entries in an existing Directory Server database. The **ldapmodify** command opens a connection to the specified server using the supplied distinguished name and password and modifies the entries based on LDIF update statements contained in a specified file. Because **ldapmodify** uses LDIF update statements, **ldapmodify** can do everything that **ldapdelete** can do.

Consider the following when using **ldapmodify**:

- If the server detects an attribute or object class in the entry that is not known to the server, then the modify operation will fail when it reaches the erroneous entry. All entries that were processed before the error was encountered will be successfully added or modified. If you run **ldapmodify** with the **-c** option (do not stop on errors), all correct entries processed after the erroneous entry will be successfully added or modified.
- If a required attribute is not present, the modify operation fails. This happens even if the offending object class or attribute is not being modified.

**NOTE**

To create the root entry a database suffix (such as **dc=example,dc=com**) using **ldapmodify**, you must bind to the directory as the Directory Manager.

3.2.4.1. Adding Entries Using ldapmodify

Typically, to add the entries using **ldapmodify**, pass the **-a** option to indicate an add operation and the LDIF file to use which contains the new entry information (and, optionally, the bind credentials and any connection information). For example:

```
ldapmodify -a -D "cn=directory manager" -w -p 389 -h server.example.com -x
-f new.ldif
```

The entries to be created are specified in the file **new.ldif**. (In this example, the LDIF statements in the **new.ldif** file do not specify a change type. They follow the format defined in [Section B.1, “About the LDIF File Format”](#).)

If the new entry is not passed in a given LDIF file, then the **ldapmodify** utility waits for the DN of the new entry and then each object class and attribute for the entry, each on a new line in LDIF format. When the last attribute is entered, hit enter twice to submit the new entry.

[Table 3.2, “ldapmodify Parameters Used for Adding Entries”](#) describes the **ldapmodify** parameters used in the example.

Table 3.2. ldapmodify Parameters Used for Adding Entries

Parameter Name	Description
-a	Specifies that the modify operation will add new entries to the directory.
-D	Specifies the distinguished name with which to authenticate to the server. The value must be a DN recognized by the Directory Server, and it must also have the authority to modify the entries.
-w	Specifies the password associated with the distinguished name specified in the -D parameter.
-h	Specifies the name of the host on which the server is running.
-p	Specifies the port number that the server uses.
-f	Optional parameter that specifies the file containing the LDIF update statements used to define the modifications. If you do not supply this parameter, the update statements are read from stdin . For information on supplying LDIF update statements from the command line, see Section 3.2.1, “Providing Input from the Command Line” .

3.2.4.2. Modifying Entries Using `ldapmodify`

Typically, to edit entries using `ldapmodify`, specify the DN and password to bind to the Directory Server, the port and host of the Directory Server, and the LDIF file to use, as when adding entries with `ldapmodify`. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x -f
modify_statements
```

The entries to modify are specified in the file `modify_statements`. Before the entries can be modified, you must first create the `modify_statements` file with the appropriate LDIF update statements; LDIF update statements are described in [Section 3.3, “Using LDIF Update Statements to Create or Modify Entries”](#).

[Table 3.3, “ldapmodify Parameters Used for Modifying Entries”](#) describes the `ldapmodify` parameters used in the example.

Table 3.3. ldapmodify Parameters Used for Modifying Entries

Parameter Name	Description
-D	Specifies the distinguished name with which to authenticate to the server. The value must be a DN recognized by the Directory Server, and it must also have the authority to modify the entries.
-w	Specifies the password associated with the distinguished name specified in the -D parameter.
-h	Specifies the name of the host on which the server is running.
-p	Specifies the port number that the server uses.
-f	Optional parameter that specifies the file containing the LDIF update statements used to define the modifications. If you do not supply this parameter, the update statements are read from stdin . For information on supplying LDIF update statements from the command line, see Section 3.2.1, “Providing Input from the Command Line” .
-x	Disables SASL to allow a simple bind to the server.

3.2.5. Deleting Entries Using `ldapdelete`

The `ldapdelete` command-line utility opens a connection to the specified server using the provided distinguished name and password and deletes the specified entry or entries.



NOTE

You can only delete entries at the end of a branch. You cannot delete entries that are branch points in the directory tree.

For example, of the following three entries, only the last two entries can be deleted.

```
ou=People,dc=example,dc=com
cn=Paula Simon,ou=People,dc=example,dc=com
cn=Jerry O'Connor,ou=People,dc=example,dc=com
```

The entry that identifies the **People** subtree can be deleted only if there are not any entries below it. To delete **ou=People,dc=example,dc=com**, you must first delete Paula Simon and Jerry O'Connor's entries and all other entries in that subtree.

Like **ldapmodify**, running **ldapdelete** requires the DN and password to bind to the Directory Server, the port and host of the Directory Server, and the DNs of the entries to delete. For example:

```
ldapdelete -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x "cn=Robert Jenkins,ou=People,dc=example,dc=com"
"cn=Lisa Jangles,ou=People,dc=example,dc=com"
```

The DNs of the entries to delete (**cn=Robert Jenkins,ou=People,dc=example,dc=com** and **cn=Lisa Jangles,ou=People,dc=example,dc=com**) are appended to the end of the delete command.

[Table 3.4, “ldapdelete Parameters Used for Deleting Entries”](#) describes the **ldapdelete** parameters used in the example:

Table 3.4. ldapdelete Parameters Used for Deleting Entries

Parameter Name	Description
-D	Specifies the distinguished name with which to authenticate to the server. The value must be a DN recognized by the Directory Server, and it must also have the authority to modify the entries.
-w	Specifies the password associated with the distinguished name specified in the -D parameter.
-h	Specifies the name of the host on which the server is running.
-p	Specifies the port number that the server uses.
-x	Disables SASL to allow a simple bind to the server.

3.2.6. Using Special Characters

When using the Directory Server command-line client tools, you may need to specify values that contain characters that have special meaning to the command-line interpreter, such as space (), asterisk (*), or backslash (\). When this situation occurs, enclose the value in quotation marks (""). For example:

```
-D "cn=Barbara Jensen,ou=Product Development,dc=example,dc=com"
```

Depending on the command-line utility, use either single or double quotation marks; see your operating system documentation for more information.

Additionally, if a DN contains commas, you must escape the commas with a backslash (\). For example:

```
-D "cn=Patricia Fuentes,ou=people,o=example.com Bolivia\,S.A."
```

To delete user Patricia Fuentes from the **example.com Bolivia, S.A.** tree, use the following command:

```
ldapdelete -D "cn=directory manager" -w secret -p 389 -h  
server.example.com -x "cn=Patricia Fuentes,ou=People,o=example.com  
Bolivia\,S.A."
```

3.3. USING LDIF UPDATE STATEMENTS TO CREATE OR MODIFY ENTRIES

LDIF update statements define how **ldapmodify** changes the directory entry. In general, LDIF update statements contain the following information:

- The DN of the entry to be modified.
- A changetype that defines how a specific entry is to be modified (**add**, **delete**, **modify**, **modrdn**).
- A series of attributes and their changed values.

A change type is required unless **ldapmodify** is run with the **-a** parameter. If you specify the **-a** parameter, then an add operation (**changetype: add**) is assumed. However, any other change type overrides the **-a** parameter.

If you specify a modify operation (**changetype: modify**), a change operation is required that indicates how the entry should be changed.

If you specify **changetype: modrdn**, change operations are required that specify how the relative distinguished name (RDN) is to be modified. A distinguished name's RDN is the left-most value in the DN. For example, the distinguished name **uid=ssarette,dc=example,dc=com** has an RDN of **uid=ssarette**.

The general format of LDIF update statements is as follows:

```
dn: distinguished_name  
changetype: changetype_identifier  
change_operation_identifier: list_of_attributes  
change_operation_identifier: list_of_attributes
```


A dash (-) must be used to denote the end of a change operation if subsequent change operations are specified. For example, the following statement adds the telephone number and manager attributes to the entry:

```
dn: cn=Lisa Jangles,ou=People,dc=example,dc=com
changetype: modify
add: telephonenumber
telephonenumber: (408) 555-2468
-
add: manager
manager: cn=Harry Cruise,ou=People,dc=example,dc=com
```

In addition, the line continuation operator is a single space. Therefore, the following two statements are identical:

```
dn: cn=Lisa Jangles,ou=People,dc=example,dc=com

dn: cn=Lisa Jangles,
   ou=People,
   dc=example,dc=com
```

The following sections describe the change types in detail.

3.3.1. Adding an Entry Using LDIF

changetype: add adds an entry to the directory. When you add an entry, make sure to create an entry representing a branch point before you try to create new entries under that branch. That is, to place an entry in a **People** and a **Groups** subtree, then create the branch point for those subtrees before creating entries within the subtrees. For example:

```
dn: dc=example,dc=com
changetype: add
objectclass: top
objectclass: organization
o: example.com

dn: ou=People,dc=example,dc=com
changetype: add
objectclass: top
objectclass: organizationalUnit
ou: People
ou: Marketing

dn: cn=Pete Minsky,ou=People,dc=example,dc=com
changetype: add
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: Pete Minsky
givenName: Pete
sn: Minsky
ou: People
ou: Marketing
uid: pminsky
```

```
dn: cn=Sue Jacobs,ou=People,dc=example,dc=com
changetype: add
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: Sue Jacobs
givenName: Sue
sn: Jacobs
ou: People
ou: Marketing
uid: sjacobs
```

```
dn: ou=Groups,dc=example,dc=com
changetype: add
objectclass: top
objectclass: organizationalUnit
ou: Groups
```

```
dn: cn=Administrators,ou=Groups,dc=example,dc=com
changetype: add
objectclass: top
objectclass: groupOfNames
member: cn=Sue Jacobs,ou=People,dc=example,dc=com
member: cn=Pete Minsky,ou=People,dc=example,dc=com
cn: Administrators
```

```
dn: ou=example.com Bolivia\, S.A.,dc=example,dc=com
changetype: add
objectclass: top
objectclass: organizationalUnit
ou: example.com Bolivia\, S.A.
```

```
dn: cn=Carla Flores,ou=example.com Bolivia\,S.A.,dc=example,dc=com
changetype: add
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: Carla Flores
givenName: Carla
sn: Flores
ou: example.com Bolivia\, S.A.
uid: cflores
```

3.3.2. Renaming an Entry Using LDIF

changetype: modrdn changes an entry's distinguished name. Most commonly, it changes the *relative DN*, or RDN, the left-most element in the distinguished name. The RDN for **cn=Barry Nixon,ou=People,dc=example,dc=com** is **cn=Barry Nixon**, and the RDN for **ou=People,dc=example,dc=com** is **ou=People**.

The following command renames Sue Jacobs to Susan Jacobs:

```
dn: cn=Sue Jacobs,ou=Marketing,dc=example,dc=com
changetype: modrdn
newrdn: cn=Susan Jacobs
deleteoldrdn: 0
```

Because **deleteoldrdn** is **0**, this example retains the existing RDN as a value in the new entry. The resulting entry would therefore have a common name (**cn**) attribute set to both Sue Jacobs and Susan Jacobs, in addition to all the other attributes included in the original entry. However, using the following command causes the server to delete **cn=Sue Jacobs**, so that only **cn=Susan Jacobs** remains in the entry:

```
dn: cn=Sue Jacobs,ou=Marketing,dc=example,dc=com
changetype: modrdn
newrdn: cn=Susan Jacobs
deleteoldrdn: 1
```

While it is most common for the **modrdn** operation to be used in a direct name change operation, it can be used to change other parts of the DN, which is both a rename operation *and* a move operation. **modrdn** can update the name of all leaf entries if the name of a subtree entry changes, or a single leaf can be moved to a new parent by changing its DN. This is covered in more detail in [Section 3.4](#), “Renaming and Moving Entries”.

3.3.3. Modifying an Entry Using LDIF

changetype: modify can add, replace, or remove attributes or attribute values in an entry. When you specify **changetype: modify**, you must also provide a change operation to indicate how the entry is to be modified. Change operations can be as follows:

- **add:** *attribute*

Adds the specified attribute or attribute value. If the attribute type does not currently exist for the entry, then the attribute and its corresponding value are created. If the attribute type already exists for the entry, then the specified attribute value is added to the existing value. If the particular attribute value already exists for the entry, then the operation fails, and the server returns an error.

- **replace:** *attribute*

The specified values are used to entirely replace the attribute's values. If the attribute does not already exist, it is created. If no replacement value is specified for the attribute, the attribute is deleted.

- **delete:** *attribute*

The specified attribute is deleted. If more than one value of an attribute exists for the entry, then all values of the attribute are deleted in the entry. To delete just one of many attribute values, specify the attribute and associated value on the line following the delete change operation.

This section contains the following topics:

- [Section 3.3.3.1, “Adding Attributes to Existing Entries Using LDIF”](#)
- [Section 3.3.3.2, “Changing an Attribute Value Using LDIF”](#)
- [Section 3.3.3.3, “Deleting All Values of an Attribute Using LDIF”](#)

- [Section 3.3.3.4, “Deleting a Specific Attribute Value Using LDIF”](#)

3.3.3.1. Adding Attributes to Existing Entries Using LDIF

Using **changetype: modify** with the add operation can add an attribute and an attribute value to an entry. For example, the following LDIF update statement adds a telephone number to the entry:

```
dn: cn=Barney Fife,ou=People,dc=example,dc=com
changetype: modify
add: telephonenumber
telephonenumber: 555-1212
```

The following example adds two telephone numbers to the entry:

```
dn: cn=Barney Fife,ou=People,dc=example,dc=com
changetype: modify
add: telephonenumber
telephonenumber: 555-1212
telephonenumber: 555-6789
```

The following example adds two **telephonenumber** attributes and a **manager** attribute to the entry:

```
dn: cn=Barney Fife,ou=People,dc=example,dc=com
changetype: modify
add: telephonenumber
telephonenumber: 555-1212
telephonenumber: 555-6789
-
add: manager
manager: cn=Sally Nixon,ou=People,dc=example,dc=com
```

The following example adds a **jpeg** photograph to the directory. In order to add this attribute to the directory, use the **-b** parameter, which indicates that **ldapmodify** should read the referenced file for binary values if the attribute value begins with a slash:

```
dn: cn=Barney Fife,ou=People,dc=example,dc=com
changetype: modify
add: jpegphoto
jpegphoto: /path/to/photo
```

You can also add a **jpeg** photograph to the directory using the following standard LDIF notation:

```
jpegphoto: < file:/path/to/photo
```

Using the standard notation means that the **-b** parameter does not need to be used with **ldapmodify**. However, you must add **version:1** to the beginning of the LDIF file or with LDIF update statements. For example:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
version: 1
dn: cn=Barney Fife,ou=People,dc=example,dc=com
```

```
changetype: modify
add: userCertificate
userCertificate;binary:< file: BarneyCert
```

**NOTE**

Standard LDIF notation can *only* be used with the **ldapmodify** command, not with other command-line utilities.

3.3.3.2. Changing an Attribute Value Using LDIF

changetype: modify with the replace operation changes all values of an attribute in an entry. For example, the following LDIF update statement changes Barney's manager from Sally Nixon to Wally Hensford:

```
dn: cn=Barney Fife,ou=People,dc=example,dc=com
changetype: modify
replace: manager
manager: cn=Wally Hensford,ou=People,dc=example,dc=com
```

If the entry has multiple instances of the attribute, then to change one of the attribute values, you must delete the attribute value first and then add the replacement value. For example, this entry has two telephone numbers:

```
cn=Barney Fife,ou=People,dc=example,dc=com
objectClass: inetOrgPerson
cn: Barney Fife
sn: Fife
telephonenumber: 555-1212
telephonenumber: 555-6789
```

To change the telephone number **555-1212** to **555-4321**, use the following LDIF update statement:

```
dn: cn=Barney Fife,ou=People,dc=example,dc=com
changetype: modify
delete: telephonenumber
telephonenumber: 555-1212
-
add: telephonenumber
telephonenumber: 555-4321
```

The entry is now as follows:

```
cn=Barney Fife,ou=People,dc=example,dc=com
objectClass: inetOrgPerson
cn: Barney Fife
sn: Fife
telephonenumber: 555-6789
telephonenumber: 555-4321
```

3.3.3.3. Deleting All Values of an Attribute Using LDIF

changetype: modify with the delete operation deletes an attribute from an entry. If the entry has more than one instance of the attribute, you must indicate which of the attributes to delete.

For example, the following LDIF update statement deletes all instances of the **telephonenumber** attribute from the entry, regardless of how many times it appears in the entry:

```
dn: cn=Barney Fife,ou=People,dc=example,dc=com
changetype: modify
delete: telephonenumber
```

To delete just a specific instance of the **telephonenumber** attribute, simply delete that specific attribute value, as described in the next section.

3.3.3.4. Deleting a Specific Attribute Value Using LDIF

Running **changetype: modify** with the delete operation can delete a single value for an attribute value from an entry, as well as deleting all instances of the attribute. For example, consider the following entry:

```
cn=Barney Fife,ou=People,dc=example,dc=com
objectClass: inetOrgPerson
cn: Barney Fife
sn: Fife
telephonenumber: 555-1212
telephonenumber: 555-6789
```

To delete the **555-1212** telephone number from this entry, use the following LDIF update statement:

```
dn: cn=Barney Fife,ou=People,dc=example,dc=com
changetype: modify
delete: telephonenumber
telephonenumber: 555-1212
```

Barney's entry then becomes:

```
cn=Barney Fife,ou=People,dc=example,dc=com
objectClass: inetOrgPerson
cn: Barney Fife
sn: Fife
telephonenumber: 555-6789
```

3.3.4. Deleting an Entry Using LDIF

changetype: delete is the change type which deletes an entire entry from the directory.



NOTE

You can only delete leaf entries. Therefore, when you delete an entry, make sure that no other entries exist under that entry in the directory tree. That is, you cannot delete an organizational unit entry unless you have first deleted all the entries that belong to the organizational unit.

For example, of the following three entries, only the last two entries can be deleted:

```
ou=People,dc=example,dc=com
cn=Paula Simon,ou=People,dc=example,dc=com
cn=Jerry O'Connor,ou=People,dc=example,dc=com
```

The entry that identifies the **People** subtree can be deleted only if no other entries exist below it.

The following LDIF update statements can be used to delete person entries:

```
dn: cn=Pete Minsky,ou=People,dc=example,dc=com
changetype: delete
dn: cn=Sue Jacobs,ou=People,dc=example,dc=com
changetype: delete
```



WARNING

Do not delete the suffix **o=NetscapeRoot**. The Admin Server uses this suffix to store information about installed Directory Servers. Deleting this suffix could force you to reinstall the Directory Server.

3.3.5. Modifying an Entry in an Internationalized Directory

If the attribute values in the directory are associated with languages other than English, the attribute values are associated with language tags. When using the **ldapmodify** command-line utility to modify an attribute that has an associated language tag, you must match the value and language tag exactly or the modify operation will fail.

For example, to modify an attribute value that has a language tag of **lang-fr**, include **lang-fr** in the modify operation, as follows:

```
dn: bjensen,dc=example,dc=com
changetype: modify
replace: homePostalAddress;lang-fr
homePostalAddress;lang-fr: 34 rue de Seine
```

3.4. RENAMING AND MOVING ENTRIES

Most rename operations are done on a leaf entry. However, some types of rename operations result in changes to the directory tree itself. These operations — subtree renames and new superior operations — are possible because of the way that entry IDs are stored in Directory Server indexes.

3.4.1. About Renaming Entries

Every DN is comprised of a long chain of elements that build on one another. Every entry has its own name, and then each of its children combine their name (the element on the far left) with the parent name. This pattern orients entries within the directory tree.

Example 3.1. Building Entry DNs

```

dc=example,dc=com => root suffix

ou=People,dc=example,dc=com => org unit

st=California,ou=People,dc=example,dc=com => state/province
                                l=Mountain
View,st=California,ou=People,dc=example,dc=com => city
                                ou=Engineering,l=Mountain
View,st=California,ou=People,dc=example,dc=com => org unit
uid=jsmith,ou=Engineering,l=Mountain
View,st=California,ou=People,dc=example,dc=com => leaf entry

```

Rename, or **modrdn**, operations result in changes to this tree.

3.4.1.1. Types of Rename Operations

When the naming attribute of an entry, the leftmost element of the DN, is changed, this is a *modrdn operation*. That's a special kind of modify operation because, in a sense, it moves the entry within the directory tree. For leaf entries (entries with no children), **modrdn** operations are lateral moves; the entry has the same parent, just a new name.

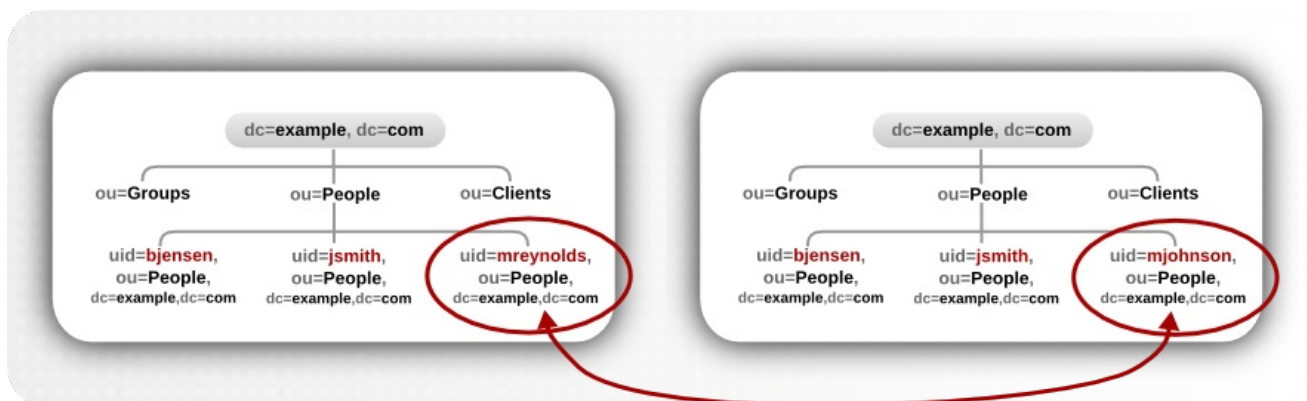


Figure 3.1. modrdn Operations for a Leaf Entry

For subtree entries, the **modrdn** operation not only renames the subtree entry itself, but also changes the DN components of all of the children entries *beneath* the subtree.

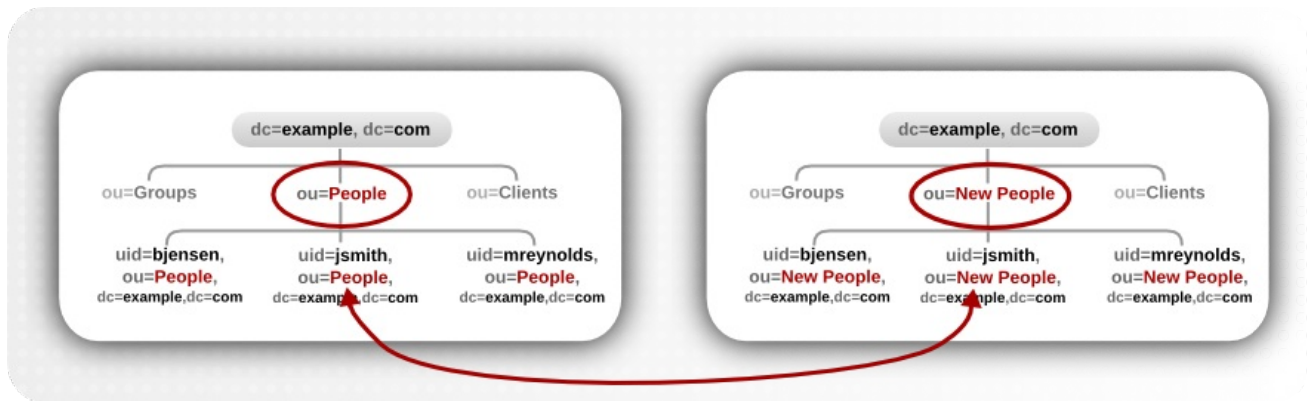


Figure 3.2. modrdn Operations for a Subtree Entry



IMPORTANT

Subtree **modrdn** operations also move and rename all of the child entries beneath the subtree entry. For large subtrees, this can be a time- and resource-intensive process. Plan the naming structure of your directory tree hierarchy so that it will not require frequent subtree rename operations.

A similar action to renaming a subtree is moving an entry from one subtree to another. This is an expanded type of **modrdn** operation, which simultaneously renames the entry (even if it is the same name) and sets a **newSuperior** attribute which moves the entry from one parent to another.

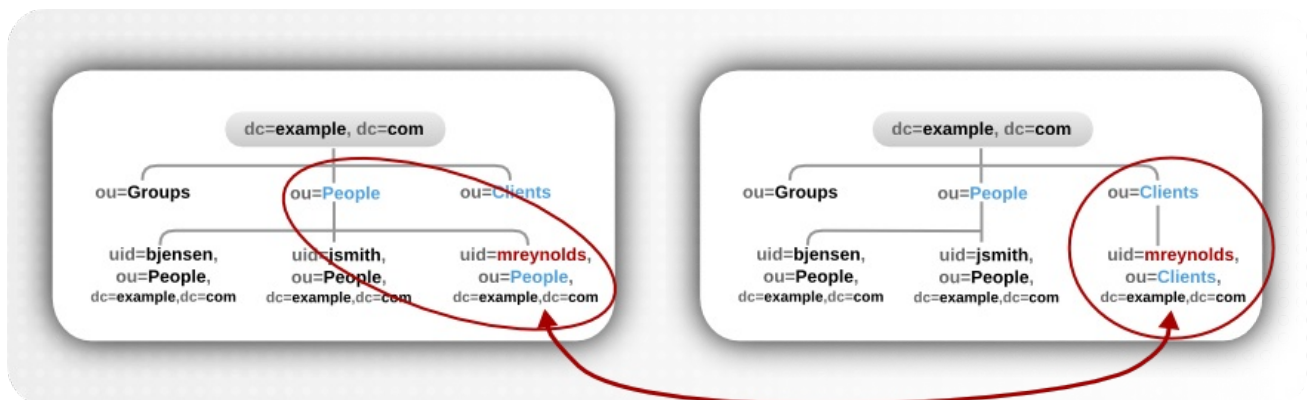


Figure 3.3. modrdn Operations to a New Parent Entry

3.4.1.2. The Relationships Between Entry Keys in entryrdn.db4

Both new superior and subtree rename operations are possible because of how entries are stored in the **entryrdn.db4** index. Each entry is identified by its own key (a *self-link*) and then a subkey which identifies its parent (the *parent link*) and any children. This has a format that lays out the directory tree hierarchy by treating parents and children as attribute to an entry, and every entry is describes by a unique ID and its RDN, rather than the full DN.

These relationships are visible in the **dbscan** output for the **entryrdn.db4** index.

```
numeric_id:RDN => self link
  ID: #; RDN: "rdn"; NRDN: normalized_rdn
P#:RDN => parent link
```

```

        ID: #; RDN: "rdn"; NRDN: normalized_rdn
C#:RDN => child link
        ID: #; RDN: "rdn"; NRDN: normalized_rdn

```

For example, the **ou=people** subtree has a parent of **dc=example,dc=com** and a child of **uid=jsmith**.

```

4:ou=people
  ID: 4; RDN: "ou=People"; NRDN: "ou=people"
P4:ou=people
  ID: 1; RDN: "dc=example,dc=com"; NRDN: "dc=example,dc=com"
C4:ou=people
  ID: 10; RDN: "uid=jsmith"; NRDN: "uid=jsmith"

```



NOTE

Version 8.2 and older of Directory Server used the **entrydn.db4** index, which stored entry keys according to their full DN. To upgrade to **entryrdn.db4**, run the **dn2rdn** tool.

The defined keys for parent and child entries makes it possible for the Directory Server to perform operations that move entries in the directory tree.

- Add operations look at the parent given in the modify statement and check for any existing leaf entries; if none exist, the new entry is added.
- For delete operations, the server first checks the self link (the full entry key), and then deletes both the self link and then the child link in the parent entry. For example, if you deleted **ou=people,dc=example,dc=com**, the server deletes the **ou=people** key (the self-link) and then deletes the child subkey from the **dc=example,dc=com** entry key.
- For new superior operations, the server changes the parent link in the entry key, removes the child link from the old parent, and adds a new child link to the new superior key.
- For subtree rename operations, the server changes the self link in the subtree entry key to update its RDN, then tracks any child entry keys (based on the child links) and updates the parent link RDN.

3.4.1.3. Considerations for Renaming Entries

There are some things to keep in mind when performing rename operations:

- You cannot rename the root suffix.
- Subtree rename operations have minimal effect on replication. Replication agreements are applied to an entire database, not a subtree within the database, so a subtree rename operation does not require re-configuring a replication agreement. All of the name changes after a subtree rename operation are replicated as normal.
- Renaming a subtree **may** require any synchronization agreements to be re-configured. Sync agreements are set at the suffix or subtree level, so renaming a subtree may break synchronization.
- Renaming a subtree **requires** that any subtree-level ACIs set for the subtree be re-configured manually, as well as any entry-level ACIs set for child entries of the subtree.

- You can rename a subtree with children, but you cannot delete a subtree with children.
- Trying to change the component of a subtree, like moving from **ou** to **dc**, may fail with a schema violation. For example, the **organizationalUnit** object class requires the **ou** attribute. If that attribute is removed as part of renaming the subtree, then the operation will fail.
- Any **memberOf** attributes managed by the MemberOf Plug-in will **not** be updated after a subtree-level rename operation. The MemberOf Plug-in does not check to see if *parent* entries change in order to initiate updates. If a subtree is renamed which contains either groups or group members, then launch a **cn=memberOf task** task or use the **fixup-memberof.pl** command to force the MemberOf Plug-in to make the changes.

See [Section 6.1.4.6, “Synchronizing memberOf Values”](#) to see how to clean up **memberOf** attribute references.

3.4.2. Renaming an Entry or Subtree

To rename an entry or subtree, use the **modrdn** changetype LDIF statement and use the **newrdn** attribute to indicate that the naming attribute is being changed and **deleteoldrdn** to set whether to retain the old RDN (0) or delete it (any non-zero integer). For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: ou=Engineering,ou=People,dc=example,dc=com
changetype: modrdn
newrdn: ou=Development
deleteoldrdn: 1
```

If **deleteoldrdn** is set to any non-zero integer, positive or negative, then the old RDN is removed.

LDIF statements for **modrdn** operations are described in [Section 3.3.2, “Renaming an Entry Using LDIF”](#).

3.4.3. Moving an Entry to a New Parent

Changing the parent entry requires setting two attributes:

- **newrdn** to set the RDN of the moved entry. This is required even if the RDN remains the same.
- **newSuperior** to give the DN of the new parent entry.

For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: uid=jsmith,l=Boston,ou=Engineering,ou=People,dc=example,dc=com
changetype: modrdn
newrdn: uid=jsmith
deleteoldrdn: 1
newSuperior: l=Mountain View,ou=Engineering,ou=People,dc=example,dc=com
```

3.4.4. Disabling Subtree Rename Operations

By default, renaming subtrees is allowed. However, this can be disabled by resetting the **nsslapd-subtree-rename-switch** parameter to **off**:



NOTE

When a new Directory Server 9.0 instance or an upgraded instance has subtree renames enabled, then the RDNs of the entries are mapped in the **entryrdn.db4** database. Instances with subtree rename disabled store their entry information in **entryrdn.db4**. When disabling subtree renames, then the server needs to switch from using one database to the other, which is done by exporting and re-importing the directory entries. This is described in [Section 3.4.1.2, “The Relationships Between Entry Keys in entryrdn.db4”](#).

1. Export the directory to LDIF.

```
/etc/dirsrv/slapd-instance_name/db2ldif -n database1 -a
/var/lib/dirsrv/slapd-instance_name/ldif/tree.ldif
```

More export options are covered in [Section 4.2, “Exporting Data”](#).

2. Turn off the **nsslapd-subtree-rename-switch** parameter.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=config,cn=ldb database,cn=plugins,cn=config
changetype: modify
replace: nsslapd-subtree-rename-switch
nsslapd-subtree-rename-switch: off
```

3. Import the LDIF file. This will populate the new **entryrdn.db4** file.

```
/etc/dirsrv/slapd-instance_name/ldif2db -D "cn=Directory Manager" -w
secret -i /var/lib/dirsrv/slapd-instance_name/ldif/tree.ldif -n
database1
```

More import options are covered in [Section 4.1, “Importing Data”](#).

3.4.5. Setting the DN Cache Size

A separate cache is used specifically to store DN information. As with the entry cache, the DN cache uses memory (RAM) used to store directory entries information in the internal representation. This is configured in the **nsslapd-dncachememsize**, and setting a large cache memory size improves the performance of rename operations.

The cache itself stores only the entry ID and the entry DN, so the table is relatively small when compared to the entry cache.

For the best move performance, make the entry cache large enough to contain all entries in the database.

This value can be modified from the command line by editing the **nsslapd-dncachememsize** attribute value for the LDBM plug-in instance for the database. For example:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
```

```
dn: cn=database_name,cn=ldbm database,cn=plugins,cn=config
changetype: modify
replace: nsslapd-dncachememsize
nsslapd-dncachememsize: 20971520
```

3.5. TRACKING MODIFICATIONS TO DIRECTORY ENTRIES

It can be useful to track when changes are made to entries. There are two aspects of entry modifications that the Directory Server tracks:

- Using change sequence numbers to track changes to the database. This is similar to change sequence numbers used in replication and synchronization. Every normal directory operation triggers a sequence number.
- Assigning creation and modification information. These attributes record the names of the user who created and most recently modified an entry, as well as the timestamps of when it was created and modified.



NOTE

The entry USN, modify time and name, and create time and name are all operational attributes and are not returned in a regular **ldapsearch**. For details on running a search for operational attributes, see [Section 10.5.7, “Searching for Operational Attributes”](#).

3.5.1. Tracking Modifications to the Database through Update Sequence Numbers

The USN Plug-in provides a way for LDAP clients to know that something — anything — in the database has changed.

3.5.1.1. An Overview of the Entry Sequence Numbers

When the USN Plug-in is enabled, update sequence numbers (USNs) are sequential numbers that are assigned to an entry whenever a write operation is performed against the entry. (Write operations include add, modify, modrdn, and delete operations. Internal database operations, like export operations, are not counted in the update sequence.) A USN counter keeps track of the most recently assigned USN.

3.5.1.1.1. Local and Global USNs

The USN is evaluated globally, for the entire database, not for the single entry. The USN is similar to the change sequence number for replication and synchronization, in that it simply ticks upward to track any changes in the database or directory. However, the entry USN is maintained separately from the CSNs, and USNs are not replicated.

The entry shows the change number for the last modification to that entry in the **entryUSN** operational attribute. (For details on running a search for operational attributes, see [Section 10.5.7, “Searching for Operational Attributes”](#).)

Example 3.2. Example Entry USN

```
dn: uid=jsmith,ou=People,dc=example,dc=com
mail: jsmith@example.com
uid: jsmith
givenName: John
objectClass: top
```

```
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
sn: Smith
cn: John Smith
userPassword: {SSHA}EfhKCI4iKl/ipZMSwLITQatz7v2lUnptxwZ/pw==
entryusn: 1122
```

The USN Plug-in has two modes, local mode and global mode:

- In local mode, each back end database has an instance of the USN Plug-in with a USN counter specific to that back end database. This is the default setting.
- In global mode, there is a global instance of the USN Plug-in with a global USN counter that applies to changes made to the entire directory.

When the USN Plug-in is set to local mode, results are limited to the local back end database. When the USN Plug-in is set to global mode, the returned results are for the entire directory.

The root DSE shows the most recent USN assigned to any entry in the database in the ***lastusn*** attribute. When the USN Plug-in is set to local mode, so each database has its own local USN counter, the ***lastUSN*** shows both the database which assigned the USN and the USN:

```
lastusn;database_name:USN
```

For example:

```
lastusn;example1: 2130
lastusn;example2: 2070
```

In global mode, when the database uses a shared USN counter, the ***lastUSN*** attribute shows the latest USN only:

```
lastusn: 4200
```

3.5.1.1.2. Importing USN Entries

When entries are imported, the USN Plug-in uses the ***nsslapd-entryusn-import-initval*** attribute to check if the entry has an assigned USN. If the value of ***nsslapd-entryusn-import-initval*** is numerical, the imported entry will use this numerical value as the entry's USN. If the value of ***nsslapd-entryusn-import-initval*** is not numerical, the USN Plug-in will use the value of the ***lastUSN*** attribute and increment it by one as the USN for the imported entry.

3.5.1.2. Configuring the USN Plug-in

The USN Plug-in must be enabled for USNs to be recorded on entries, as described in [Section 1.8.1, “Enabling Plug-ins in the Directory Server Console”](#). The plug-in can be enabled through the Directory Server Console or through the command line. For example:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=USN,cn=plugins,cn=config
changetype: modify
```

```
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
```

Then restart the server to apply the changes.

3.5.1.3. Enabling Global USN

The ***nsslapd-entryusn-global*** configuration parameter controls whether the USN Plug-in runs in local mode or global mode. The default is set to use a local USN counter, and the ***nsslapd-entryusn-global*** attribute value is off. To change whether the USN counter is in local or global mode, modify the ***nsslapd-entryusn-global*** attribute. For example:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=USN,cn=plugins,cn=config
changetype: modify
replace: nsslapd-entryusn-global
nsslapd-entryusn-global: on
```

The USN Plug-in must be enabled before the ***nsslapd-entryusn-global*** configuration parameter is turned on.

3.5.1.4. Cleaning up USN Tombstone Entries

The USN Plug-in moves entries to tombstone entries when the entry is deleted. If replication is enabled, then separate tombstone entries are kept by both the USN and Replication Plug-ins. Both tombstone entries are deleted by the replication process, but for server performance, it can be beneficial to delete the USN tombstones before converting a server to a replica or to free memory for the server.

The ***usn-tombstone-cleanup.pl*** command deletes USN tombstone entries for a specific database back end or specific suffix. Optionally, it can delete all of tombstone entries up to a certain USN. For example:

```
/usr/lib64/dirsrv/instance_name/usn-tombstone-cleanup.pl -D "cn=directory
manager" -w secret -s "ou=people,dc=example,dc=com" -m 1100
```

Either the back end must be specified using the **-n** option or the suffix, using the **-s** option. If both are given, then the suffix in the **-s** option is used.

The options for ***usn-tombstone-cleanup.pl*** command are listed in [Table 3.5, “usn-tombstone-cleanup.pl Options”](#). More details for this tool are in the *Configuration and Command-Line Tool Reference*.

Table 3.5. usn-tombstone-cleanup.pl Options

Option	Description
-D rootdn	Gives the user DN with root permissions, such as Directory Manager. The default is the DN of the Directory Manager, which is read from the <i>nsslapd-root</i> attribute under <i>cn=config</i> .

Option	Description
<code>-m maximum_USN</code>	Sets the upper bound for entries to delete. All tombstone entries with an entryUSN value up to the specified maximum (inclusive) are deleted, but not past that USN value. If no maximum USN value is set, then all back end tombstone entries are deleted.
<code>-n backendInstance</code>	Gives the name of the database containing the entries to clean (delete).
<code>-s suffix</code>	Gives the name of the suffix containing the entries to clean (delete).
<code>-w password</code>	The password associated with the user DN.

3.5.2. Tracking Entry Modifications through Operational Attributes

The Directory Server can maintain some special operational attributes for every directory entry, showing basic creation and modification information:

- *creatorsName*. The distinguished name of the person who initially created the entry.
- *createTimestamp*. The timestamp for when the entry was created in GMT (Greenwich Mean Time) format.
- *modifiersName*. The distinguished name of the person who last modified the entry.
- *modifyTimestamp*. The timestamp for when the entry was last modified in GMT format.

Operational attributes are not returned in default directory searches and must be explicitly requested, as described in [Section 10.5.7, “Searching for Operational Attributes”](#).

NOTE

When a database link is used by a client application to create or modify entries, the **creatorsName** and **modifiersName** attributes do not reflect the real creator or modifier of the entries. These attributes contain the name of the administrator who is granted proxy authorization rights on the remote server.

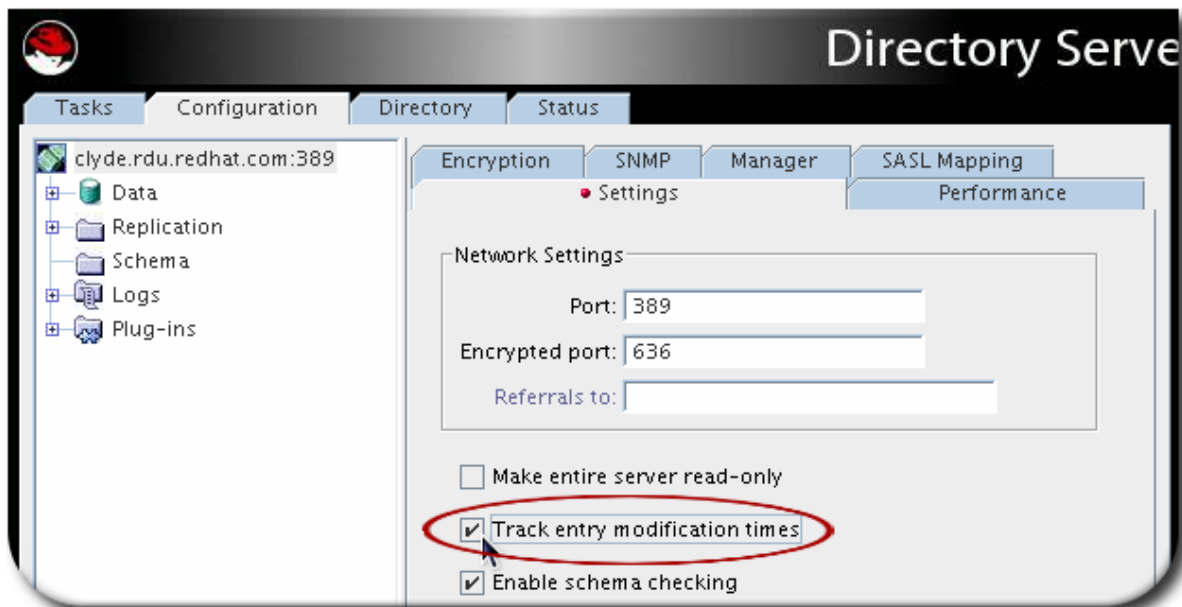
The access logs, however, will show both the proxy user (as **dn**) and the real user (as the **authzid** entity). For example:

```
[23/May/2011:18:13:56 +051800] conn=1175 op=0 BIND dn="cn=proxy
admin,ou=people,dc=example,dc=com" method=128 version=3
[23/May/2011:18:13:56 +051800] conn=1175 op=0 RESULT err=0
tag=97 nentries=0 etime=0 dn="cn=proxy
admin,ou=people,dc=example,dc=com"
[23/May/2011:18:13:56 +051800] conn=1175 op=1 SRCH
base="dc=example,dc=com" scope=2 filter="(objectClass=*)"
attrs=ALL authzid="uid=jsmith,ou=people,dc=example,dc=com"
```

For information on proxy authorization, see [Section 2.3.1.2.2, "Providing Bind Credentials"](#).

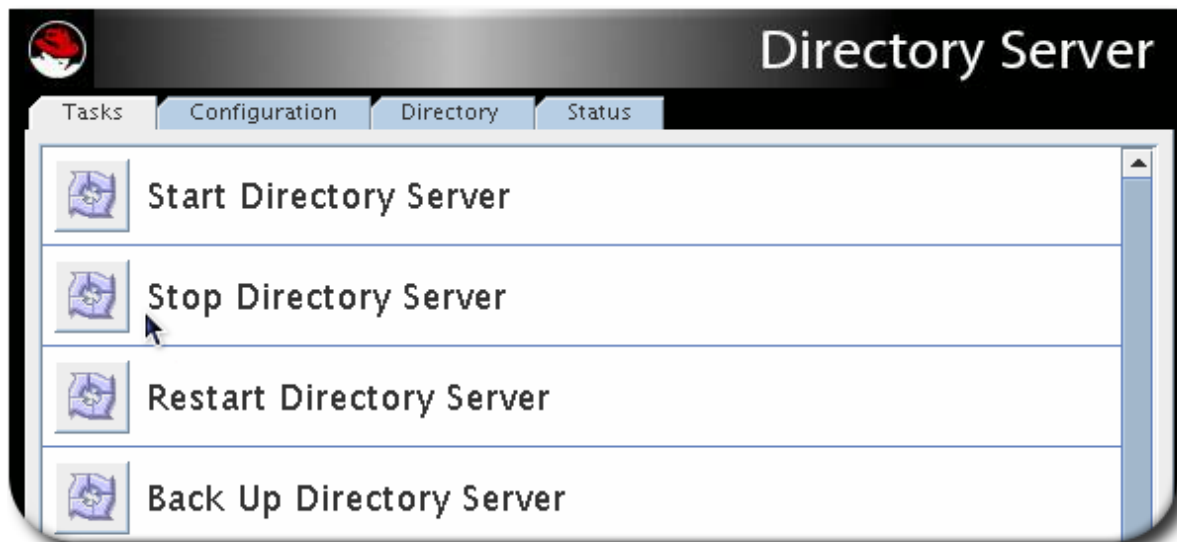
To enable the Directory Server to track when entries are created or modified:

1. In the Directory Server Console, select the **Configuration** tab, and then select the top entry in the navigation tree in the left pane.
2. Select the **Settings** tab in the right pane.
3. Select the **Track Entry Modification Times** check box.



The server adds the **creatorsName**, **createTimestamp**, **modifiersName**, and **modifyTimestamp** attributes to every newly created or modified entry.

4. Open the **Tasks** tab, and click **Restart Directory Server**.

**NOTE**

The Directory Server *must* be restarted for the changes to take effect.

3.5.3. Tracking the Bind DN for Plug-in Initiated Updates

One change to an entry can trigger other, automatic changes across the directory tree. When a user is deleted, for example, that user is automatically removed from any groups it belonged to by the Referential Integrity Plug-in.

The initial action is shown in the entry as being performed by whatever user account is bound to the server, but all related updates (by default) are shown as being performed by the plug-in, with no information about which user initiated that update. For example, using the MemberOf Plug-in to update user entries with group membership, the update to the group account is shown as being performed by the bound user, while the edit to the user entry is shown as being performed by the MemberOf Plug-in:

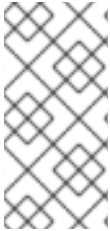
```
dn: cn=my_group,ou=groups,dc=example,dc=com
modifiersname: uid=jsmith,ou=people,dc=example,dc=com

dn: uid=bjensen,ou=people,dc=example,dc=com
modifiersname: cn=memberOf plugin,cn=plugins,cn=config
```

The ***nsslapd-plugin-binddn-tracking*** attribute allows the server to track which user originated an update operation, as well as the internal plug-in which actually performed it. The bound user is shown in the ***modifiersname*** and ***creatorsname*** operational attributes, while the plug-in which performed it is shown in the ***internalModifiersname*** and ***internalCreatorsname*** operational attributes. For example:

```
dn: uid=bjensen,ou=people,dc=example,dc=com
modifiersname: uid=jsmith,ou=people,dc=example,dc=com
internalModifiersname: cn=memberOf plugin,cn=plugins,cn=config
```

The ***nsslapd-plugin-binddn-tracking*** attribute tracks and maintains the relationship between the bound user and any updates performed for that connection.



NOTE

The ***internalModifiersname*** and ***internalCreatorsname*** attributes always show a plug-in as the identity. This plug-in could be an additional plug-in, such as the MemberOf Plug-in. If the change is made by the core Directory Server, then the plug-in is the database plug-in, **cn=ldbm database,cn=plugins,cn=config**.

The ***nsslapd-plugin-binddn-tracking*** attribute is disabled by default. To allow the server to track operations based on bind DN, enable that attribute using **ldapmodify**:

```
[jsmith@server ~]$ ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=config
changetype: modify
replace: nsslapd-plugin-binddn-tracking
nsslapd-plugin-binddn-tracking: on
```

3.5.4. Tracking Password Change Times

Password change operations are normally treated as any other modification to an entry, so the update time is recorded in the ***lastModified*** operational attribute. However, there can be times when the time of the last password change needs to be recorded separately, to make it easier to update passwords in Active Directory synchronization or to connect with other LDAP clients.

The ***passwordTrackUpdateTime*** attribute within the password policy tells the server to record a timestamp for the last time that the password was updated for an entry. The password change time itself is stored as an operational attribute on the user entry, ***pwdUpdateTime*** (which is separate from the ***modifyTimestamp*** or ***lastModified*** operational attributes).

The ***passwordTrackUpdateTime*** attribute can be set as part of the global password policy or on a subtree or user-level policy, depending on what clients need to access the password change time. Setting password policies is described in [Section 14.1, “Managing the Password Policy”](#).

3.6. MAINTAINING REFERENTIAL INTEGRITY

Referential Integrity is a database mechanism that ensures relationships between related entries are maintained. In the Directory Server, the Referential Integrity can be used to ensure that an update to one entry in the directory is correctly reflected in any other entries that reference to the updated entry.

For example, if a user's entry is removed from the directory and referential integrity is enabled, the server also removes the user from any groups of which the user is a member. If referential integrity is not enabled, the user remains a member of the group until manually removed by the administrator. This is an important feature if you are integrating the Directory Server with other products that rely on the directory for user and group management.

3.6.1. How Referential Integrity Works

When the Referential Integrity Plug-in is enabled, it performs integrity updates on specified attributes immediately after a delete or rename operation. By default, the Referential Integrity Plug-in is disabled.



NOTE

The Referential Integrity Plug-in should only be enabled on one supplier replica in a multi-master replication environment to avoid conflict resolution loops. When enabling the plug-in on servers issuing chaining requests, be sure to analyze performance resource and time needs, as well as your integrity needs. Integrity checks can be time-consuming and draining on memory and CPU.

When a user or group entry is deleted, updated, renamed, or moved within the directory, the operation is logged to the Referential Integrity log file. For the distinguished names (DN) in the log file, Directory Server searches and updates in intervals the attributes set in the plug-in configuration:

- For entries, marked in the log file as deleted, the corresponding attribute in the directory is deleted.
- For entries, marked in the log file as updated, the corresponding attribute in the directory is updated.
- For entries, marked in the log file as renamed or moved, the value of the corresponding attribute in the directory is renamed.

By default, when the Referential Integrity Plug-in is enabled, it performs integrity updates on the **member**, **uniquemember**, **owner**, and **seeAlso** attributes immediately after a delete or rename operation. However, the behavior of the Referential Integrity Plug-in can be configured to suit the needs of the directory in several different ways:

- Record referential integrity updates in the replication changelog.
- Modify the update interval.
- Select the attributes to which to apply referential integrity.
- Disable referential integrity.

All attributes used in referential integrity *must* be indexed for presence, equality, and subtring; not indexing those attributes results poor server performance for modify and delete operations.

```
nsIndexType: pres
nsIndexType: eq
nsIndexType: sub
```

See [Section 9.2, “Creating Standard Indexes”](#) for more information about checking and creating indexes.

3.6.2. Using Referential Integrity with Replication

There are certain limitations when using the Referential Integrity Plug-in in a replication environment:

- *Never* enable it on a dedicated consumer server (a server that contains only read-only replicas).
- *Never* enable it on a server that contains a combination of read-write and read-only replicas.
- It is possible to enable it on a supplier server that contains only read-write replicas.
- With multi-master replication, enable the plug-in on just one supplier.

If the replication environment satisfies the all of those condition, you can enable the Referential Integrity Plug-in.

1. Enable the Referential Integrity Plug-in as described in [Section 3.6.3, “Enabling and Disabling Referential Integrity”](#).
2. Configure the plug-in to record any integrity updates in the changelog.
3. Ensure that the Referential Integrity Plug-in is disabled on all consumer servers.

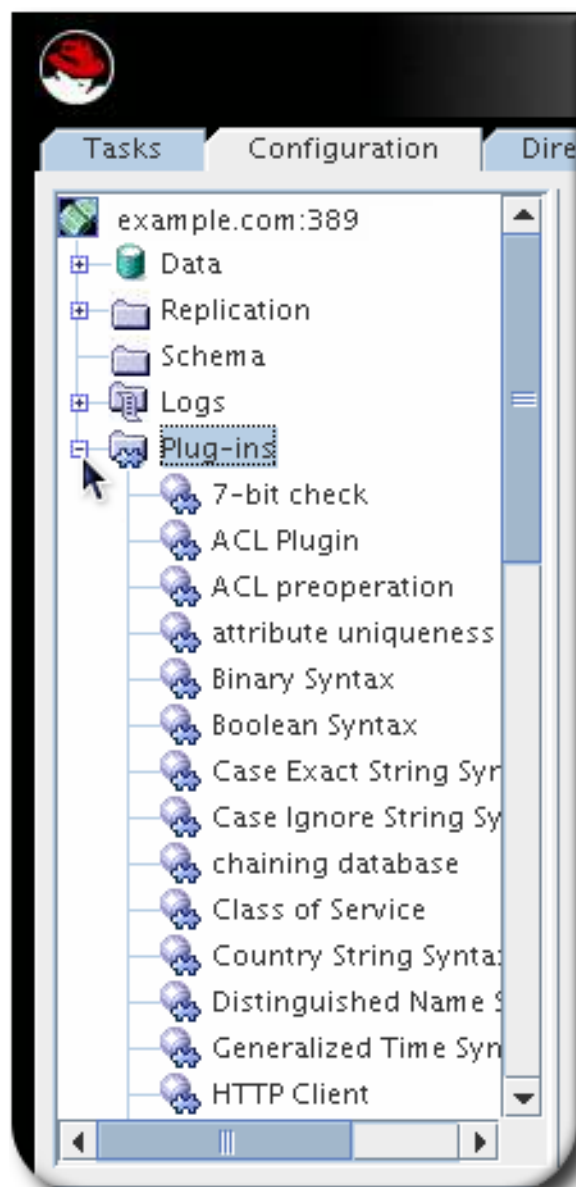
**NOTE**

Because the supplier server sends any changes made by the Referential Integrity Plug-in to consumer servers, it is unnecessary to run the Referential Integrity Plug-in on consumer servers.

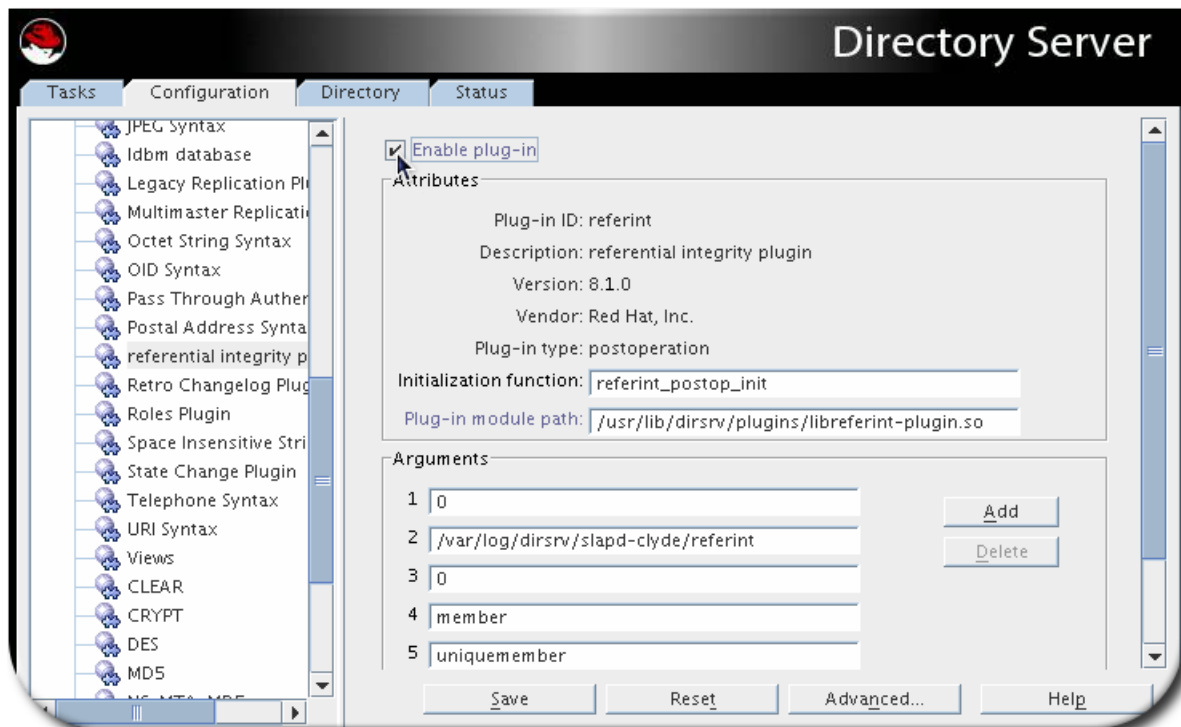
3.6.3. Enabling and Disabling Referential Integrity

3.6.3.1. Enabling and Disabling Referential Integrity in the Console

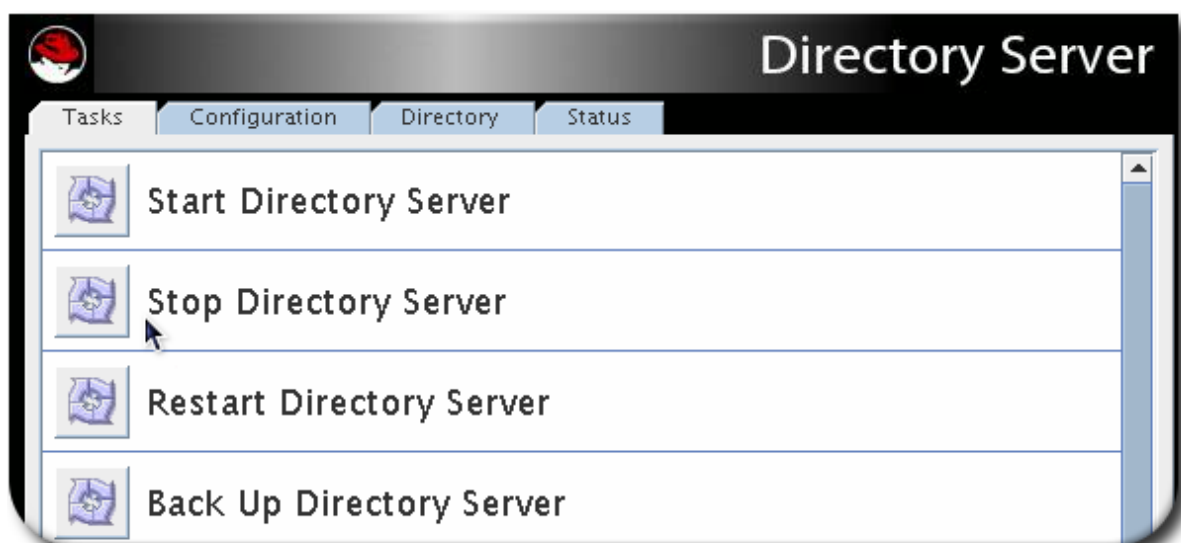
1. Select the **Configuration** tab, and expand the **Plugins** folder.
2. Select **Referential Integrity Postoperation Plug-in** from the list.



3. Check the **Enable plugin** check box to enable the plug-in; clear it to disable it.



4. Fill in the correct path to the plug-in by default; plug-ins are located in **/usr/lib64/dirsrv/plugins**.
5. Restart the Directory Server to apply the changes. In the **Tasks** tab, select **Restart the Directory Server**.



3.6.3.2. Enabling and Disabling Referential Integrity from the Command Line

To disable or enable the Referential Integrity Plug-in:

1. Use **ldapmodify** to edit the value of the **nsslapd-pluginEnabled** attribute. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-X

dn: cn=referential integrity postoperation,cn=plugins,cn=config
```

```
changetype: modify
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
```

2. Then restart the server.

```
service dirsrv restart
```

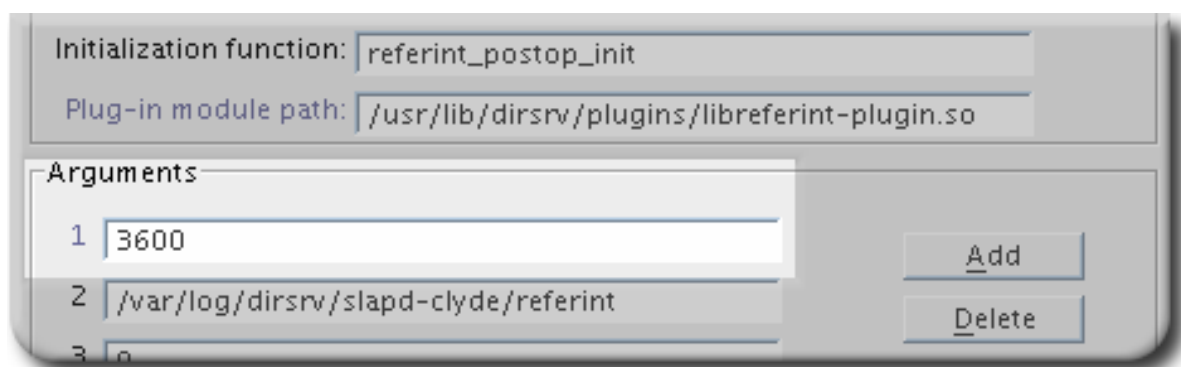
3.6.4. Modifying the Update Interval

By default, the server makes referential integrity updates immediately after a delete or a **modrdn** operation. To reduce the impact this operation has on your system, increase the amount of time between updates. Although there is no maximum update interval, the following intervals are commonly used:

- Update immediately
- 90 seconds
- 3600 seconds (updates occur every hour)
- 10,800 seconds (updates occur every 3 hours)
- 28,800 seconds (updates occur every 8 hours)
- 86,400 seconds (updates occur once a day)
- 604,800 seconds (updates occur once a week)

3.6.4.1. Modifying the Update Interval from the Console

1. Select the **Configuration** tab, and expand the **Plugins** folder. Select the **Referential Integrity Postoperation Plug-in**.
2. In the arguments list, replace the value in the first text box with the appropriate time interval.



3. Restart the Directory Server to apply the changes. In the **Tasks** tab, select **Restart the Directory Server**.

3.6.4.2. Modifying the Update Interval from the Command Line

1. Use **ldapmodify** to edit the value of the **nsslapd-pluginarg** attribute. For example:

The first argument listed sets the update interval for referential integrity checks. To change the interval, replace the ***nsslapd-pluginarg0*** attribute.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=referential integrity postoperation,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginarg0
nsslapd-pluginarg0: 600
```

2. Then restart the server.

```
service dirsrv restart
```

3.6.5. Modifying the Attribute List

3.6.5.1. Modifying the Attribute List from the Console

By default, the Referential Integrity Plug-in is set up to check for and update the ***member***, ***uniquemember***, ***owner***, and ***seeAlso*** attributes. You can add or delete attributes to be updated through the Directory Server Console, such as adding the ***nsroledn*** attribute if roles are being used.



NOTE

Keep in mind that any attribute specified in the Referential Integrity Plug-in parameter list *must* have equality indexing on all databases. Otherwise, the plug-in scans every entry of the databases for matching the deleted or modified DN, degrading performance severely. If you add an attribute, ensure that it is indexed in all the back ends.



NOTE

Improve the performance by removing any unused attributes from the list.

1. Select the **Configuration** tab, and expand the **Plugins** folder. Select the **Referential Integrity Postoperation Plug-in**.
2. In the **Arguments** section, use the **Add** and **Delete** buttons to modify the attributes in the list.

Arguments

1	3600
2	/var/log/dirsrv/slapd-clyde/referint
3	0
4	member
5	uniquemember
6	owner
7	seeAlso
8	uid

Add
Delete

- Restart the Directory Server to apply the changes. In the **Tasks** tab, select **Restart the Directory Server**.



NOTE

All attributes used in referential integrity *must* be indexed for presence and equality; not indexing those attributes results poor server performance for modify and delete operations. See [Section 9.2, “Creating Standard Indexes”](#) for more information about checking and creating indexes.

3.6.5.2. Modifying the Attribute List from the Command Line

By default, the Referential Integrity plug-in is set up to check for and update the *member*, *uniquemember*, *owner*, and *seeAlso* attributes.

To enable shared configuration entries, set the **nsslapd-pluginConfigArea** attribute:

```
nsslapd-pluginConfigArea:entry_DN
```

All the configuration attribute settings, for example adding or removing a shared entry, are dynamic and do not require a server restart to take effect.

The following example uses the **pluginarg*** attributes:

```
nsslapd-pluginarg0: 0
nsslapd-pluginarg1: /var/log/dirsrv/slapd-localhost/referint
nsslapd-pluginarg2: 0
nsslapd-pluginarg3: member
nsslapd-pluginarg4: uniquemember
nsslapd-pluginarg5: owner
nsslapd-pluginarg6: seeAlso
```

Referential Integrity plug-in parameter descriptions:

Legacy-style parameter	Description
nsslapd-pluginarg0	Sets the update delay: <ul style="list-style-type: none"> • -1: No check for referential integrity is performed. • 0: The check for referential integrity is performed immediately. • Positive integer value: Represents the interval in seconds for performing the referential integrity check.
nsslapd-pluginarg1	Sets the path to the log file.
nsslapd-pluginarg2	
nsslapd-pluginarg[3-10]	Sets the attributes on which the plug-in performs integrity updates.



NOTE

Keep in mind that any attribute specified in the Referential Integrity Plug-in parameter list *must* have equality indexing on all databases. Otherwise, the plug-in scans every entry of the databases for matching the deleted or modified DN, degrading performance severely. If you add an attribute, ensure that it is indexed in all the back ends.

CHAPTER 4. POPULATING DIRECTORY DATABASES

Databases contain the directory data managed by the Red Hat Directory Server.

4.1. IMPORTING DATA

Directory Server can populate a database with data in one of two ways: by importing data (either through the Directory Server Console or using the import tools) or by initializing a database for replication.

Table 4.1, “Import Method Comparison” describes the differences between an import and initializing databases.

Table 4.1. Import Method Comparison

Action	Import	Initialize Database
Overwrites database	No	Yes
LDAP operations	Add, modify, delete	Add only
Performance	More time-consuming	Fast
Partition specialty	Works on all partitions	Local partitions only
Response to server failure	Best effort (all changes made up to the point of the failure remain)	Atomic (all changes are lost after a failure)
LDIF file location	Local to Console	Local to Console or local to server
Imports configuration information (cn=config)	Yes	No

4.1.1. Importing Entries with Large Attributes

The **nsslapd-cachememsize** attribute defines the size allowed for the entry cache.

The import buffer is automatically set to 80% of the cache memory size setting. If the memory cache is 1GB, for example, then the import buffer is 800MB.

When importing a very large database or entries with large attributes (often with values like binary data like certificate chains, CRLs, or images), then set the **nsslapd-cachememsize** attribute high enough so that the import buffer has enough memory to process the entries.

4.1.2. Importing Large Numbers of Entries

When there are a large number of entries to be imported, the operating system itself may hit performance limits on what it allows the Directory Server to do. This is particularly true on x86 systems. This can cause import operations to fail because of resource constraints.

If necessary, set the system **ulimit** value to the maximum number of allowed processes for the system user.

For example:

```
[root@server ~]# ulimit -u 4096
```

Then run the import operation.

4.1.3. Setting EntryUSN Initial Values During Import

Entry update sequence numbers (USNs) are not preserved when entries are exported from one server and imported into another. As [Section 3.5.1, “Tracking Modifications to the Database through Update Sequence Numbers”](#) explains, entry USNs are assigned for operations that happen on a local server, so it does not make sense to import those USNs onto another server.

However, it is possible to configure an initial entry USN value for entries when importing a database or initializing a database (such as when a replica is initialized for replication). This is done by setting the ***nsslapd-entryusn-import-initval*** attribute, which sets a starting USN for all imported entries.

There are two possible values for ***nsslapd-entryusn-import-initval***:

- An integer, which is the explicit start number used for every imported entry.
- *next*, which means that every imported entry uses whatever the highest entry USN value was on the server before the import operation, incremented by one.

If ***nsslapd-entryusn-import-initval*** is not set, then all entry USNs begin at zero.

For example, if the highest value on the server is 1000 before the import or initialization operation, and the ***nsslapd-entryusn-import-initval*** value is *next*, then every imported entry is assigned a USN of 1001:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x "(cn=*)" entryusn

dn: dc=example,dc=com
entryusn: 1001
dn: ou=Accounting,dc=example,dc=com
entryusn: 1001
dn: ou=Product Development,dc=example,dc=com
entryusn: 1001
...
dn: uid=jsmith,ou=people,dc=example,dc=com
entryusn: 1001
...
```

To set an initial value for entry USNs, simply add the ***nsslapd-entryusn-import-initval*** attribute to the server into which data are being imported or to the master server which will perform the initialization.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -D
"cn=directory manager" -w secret -p 389 -h server.example.com -x

dn: cn=config
changetype: modify
add: nsslapd-entryusn-import-initval
nsslapd-entryusn-import-initval: next
```

**NOTE**

In multi-master replication, the ***nsslapd-entryusn-import-initval*** attribute is *not* replicated between servers. This means that the value must be set specifically on whichever supplier server is being used to initialize a replica.

For example, if Supplier1 has ***nsslapd-entryusn-import-initval*** set to *next* and is used to initialize a replica, then the entry USNs for imported entries have the highest value plus one. If Supplier2 does not have ***nsslapd-entryusn-import-initval*** set and is used to initialize a replica, then all entry USNs for imported entries begin at zero — even if Supplier1 and Supplier 2 have a multi-master replication agreement between them.

4.1.4. Importing a Database from the Console

When performing an import operation from the Directory Server Console, an **ldapmodify** operation is executed to append data, as well as to modify and delete entries. The operation is performed on all of the databases managed by the Directory Server and on remote databases to which the Directory Server has a configured database link.

Import operations can be run on a server instance that is local to the Directory Server Console or on a different host machine (a remote import operation).

You must be logged in as the Directory Manager in order to perform an import.

**NOTE**

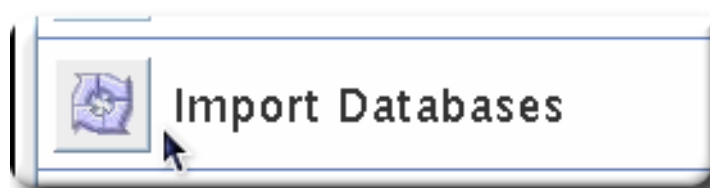
The LDIF files used for import operations must use UTF-8 character set encoding. Import operations do not convert data from local character set encoding to UTF-8 character set encoding.

**WARNING**

All imported LDIF files must also contain the root suffix.

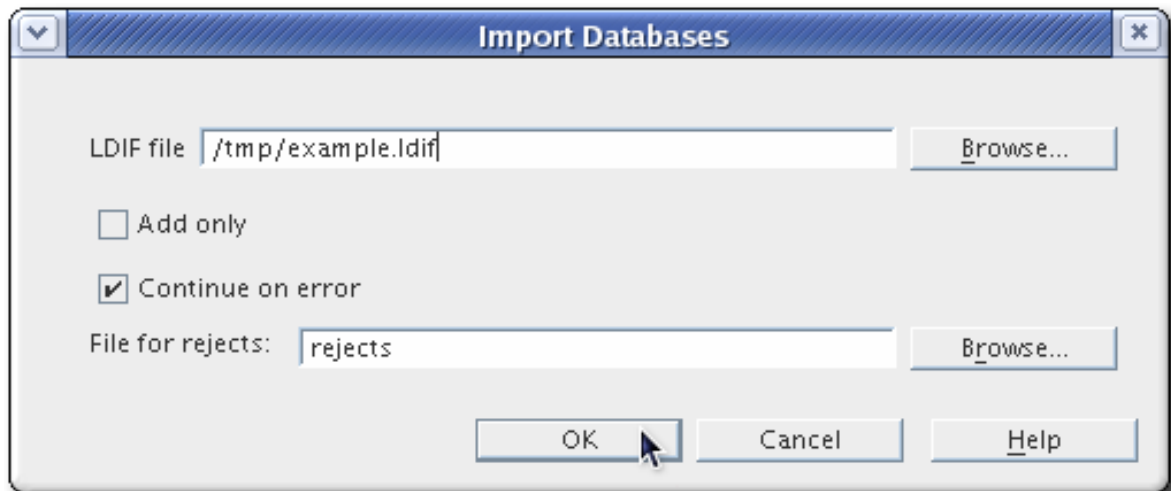
To import data from the Directory Server Console:

1. Select the **Tasks** tab. Scroll to the bottom of the screen, and select **Import Database**.



Alternatively, open the **Configuration** tab and select **Import** from the **Console** menu.

2. In the **Import Database** dialog box, enter the full path to the LDIF file to import in the **LDIF file** field, or click **Browse** to select the file to import.



If the Console is running on a machine remote to the directory, the field name appears as **LDIF file (on the machine running the Console)**. When browsing for a file, you are not browsing the current directory for the Directory Server host, but the filesystem of the machine running the Console.

When importing a database through a remote Console, *do not* use a relative path to the database. For remote imports, the operation fails with the error *Cannot write to file...* if a relative path is given for the file. Always use an absolute path for remote import operations.

3. In the **Options** box, select one or both of the following options:
 - *Add Only*. The LDIF file may contain modify and delete instructions in addition to the default add instructions. For the server to ignore operations other than add, select the **Add only** check box.
 - *Continue on Error*. Select the **Continue on error** check box for the server to continue with the import even if errors occur. For example, use this option to import an LDIF file that contains some entries that already exist in the database in addition to new ones. The server notes existing entries in the rejects file while adding all new entries.
4. In the **File for Rejects** field, enter the full path to the file in which the server is to record all entries it cannot import, or click **Browse** to select the file which will contain the rejects.

A reject is an entry which cannot be imported into the database; for example, the server cannot import an entry that already exists in the database or an entry that has no parent object. The Console will write the error message sent by the server to the rejects file.

Leaving this field blank means the server will not record rejected entries.

The server performs the import and also creates indexes.



NOTE

Trailing spaces are dropped during a remote Console import but are preserved during both local Console or **ldif2db** import operations.

4.1.5. Initializing a Database from the Console

The existing data in a database can be overwritten by initializing databases.

You must be logged in as the **Directory Manager** in order to initialize a database because an LDIF file that contains a root entry cannot be imported into a database except as the Directory Manager (root DN). Only the Directory Manager has access to the root entry, such as **dc=example,dc=com**.

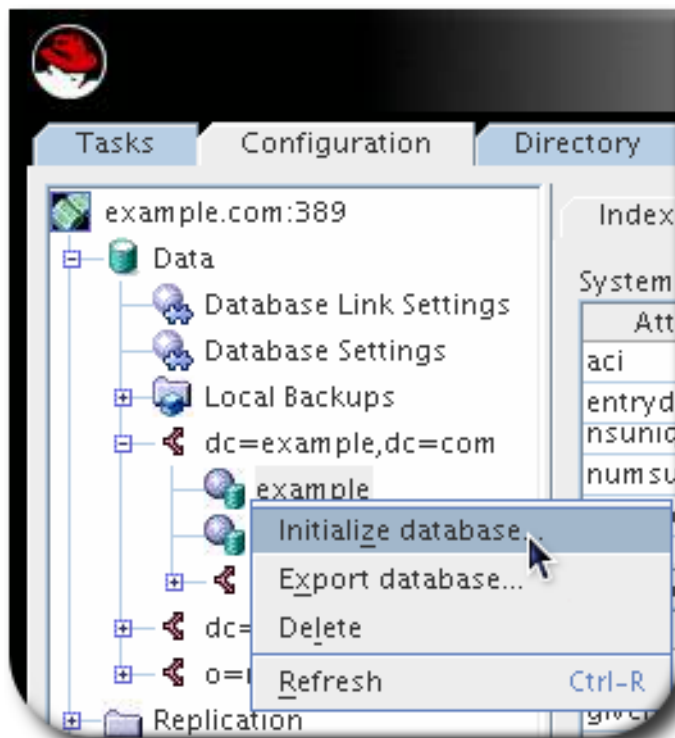


WARNING

When initializing databases from an LDIF file, be careful not to overwrite the **o=NetScapeRoot** suffix unless you are restoring data. Otherwise, initializing the database deletes information and may require re-installing the Directory Server.

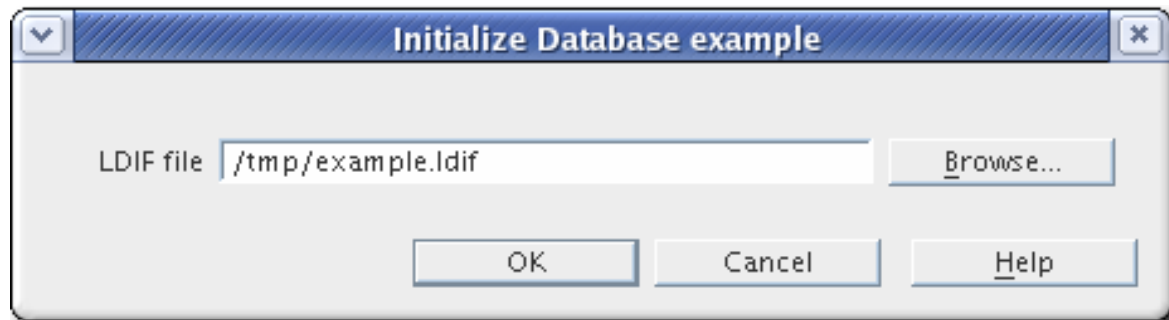
To initialize a database using the Directory Server Console:

1. Select the **Configuration** tab.
2. Expand the **Data** tree in the left navigation pane. Expand the suffix of the database to initialize, then click the database itself.
3. Right-click the database, and select **Initialize Database**.



Alternatively, select **Initialize Database** from the **Object** menu.

4. In the **LDIF file** field, enter the full path to the LDIF file to import, or click **Browse**.



5. If the Console is running from a machine local to the file being imported, click **OK** and proceed with the import immediately. If the Console is running from a machine remote to the server containing the LDIF file, select one of the following options, then click **OK**:

- *From local machine.* Indicates that the LDIF file is located on the local machine.
- *From server machine.* Indicates that the LDIF file is located on a remote server.

The default LDIF directory is `/var/lib/dirsrv/slaped-instance_name/ldif`.

4.1.6. Importing from the Command Line

There are four methods for importing data through the command line:

- *Using `ldif2db`.* This import method overwrites the contents of the database and requires the server to be stopped; see [Section 4.1.6.1, “Importing Using the `ldif2db` Command-Line Script](#)”.
- *Using `ldif2db.pl`.* This import method overwrites the contents of the database while the server is still running; see [Section 4.1.6.2, “Importing Using the `ldif2db.pl` Perl Script](#)”.
- *Using `ldif2ldap`.* This method appends the LDIF file through LDAP. This method is useful to append data to all of the databases; see [Section 4.1.6.3, “Importing Using the `ldif2ldap` Command-Line Script](#)”.
- *Creating a `cn=tasks` entry.* This method creates a temporary task entry which automatically launches an import operation. This is functionally like running `ldif2db`. See [Section 4.1.6.4, “Importing through the `cn=tasks` Entry](#)”.



NOTE

The LDIF files used for import operations must use UTF-8 character set encoding. Import operations do not convert data from local character set encoding to UTF-8 character set encoding.



WARNING

All imported LDIF files must also contain the root suffix.

**NOTE**

To import a database that has been encrypted, use the **-E** option with the script. See [Section 2.2.3.6, “Exporting and Importing an Encrypted Database”](#) for more information.

4.1.6.1. Importing Using the `ldif2db` Command-Line Script

The **ldif2db** script overwrites the data in the specified database. Also, the script requires that the Directory Server be stopped when the import begins.

By default, the script first saves and then merges any existing **o=NetScapeRoot** configuration information with the **o=NetScapeRoot** configuration information in the files being imported.

**WARNING**

This script overwrites the data in the database.

To import an LDIF:

1. Stop the server.

```
[root@server ~]# service dirsrv stop instance
```

2. Run the **ldif2db** command-line script.

```
[root@server ~]# /usr/lib64/dirsrv/slapd-instance_name/ldif2db -n
Database1 -i /var/lib/dirsrv/slapd-instance_name/ldif/demo.ldif -i
/var/lib/dirsrv/slapd-instance_name/ldif/demo2.ldif
```

On 32-bit installations, the **ldif2db** script is located in the **/usr/lib64/dirsrv/slapd-*instance_name*/** directory.

For more information about using this script, see the *Directory Server Configuration and Command-Line Tool Reference*.

**WARNING**

If the database specified in the **-n** option does not correspond with the suffix contained by the LDIF file, all of the data contained by the database is deleted, and the import fails. Make sure that the database name is not misspelled.

3. Start the server.

```
[root@server ~]# service dirsrv start instance
```

Table 4.2. Idif2db Parameters

Option	Description
-i	Specifies the full path name of the LDIF files to be imported. This option is required. To import more than one LDIF file at a time, use multiple -i arguments. When multiple files are imported, the server imports the LDIF files in the order which they are specified from the command line.
-n	Specifies the name of the database to which to import the data.

For more information about using this script, see the *Directory Server Configuration and Command-Line Tool Reference*.

4.1.6.2. Importing Using the Idif2db.pl Perl Script

As with the **ldif2db** script, the **ldif2db.pl** script overwrites the data in the specified database. This script requires the server to be running in order to perform the import.



WARNING

This script overwrites the data in the database.

Run the **ldif2db.pl** script.

```
[root@server ~]# ldif2db.pl -D "cn=Directory Manager" -w secret -i
/var/lib/dirsrv/slapd-instance_name/ldif/demo.ldif -n Database1
```

For more information about using this script, see the *Directory Server Configuration and Command-Line Tool Reference*.



NOTE

You do not need **root** privileges to run the script, but you must authenticate as the Directory Manager.

Table 4.3. Idif2db.pl Options

Option	Description
-D	Specifies the DN of the administrative user.
-w	Specifies the password of the administrative user.
-i	Specifies the LDIF files to be imported. This option is required. To import multiple LDIF files at a time, use multiple -i arguments. When multiple files are imported, the server imports the LDIF files in the order they are specified in the command line.
-n	Specifies the name of the database to which to import the data.

4.1.6.3. Importing Using the `ldif2ldap` Command-Line Script

The **ldif2ldap** script appends the LDIF file through LDAP. Using this script, data are imported to all directory databases at the same time. The server must be running in order to import using **ldif2ldap**.

To import LDIF using **ldif2ldap**:

```
[root@server ~]# ldif2ldap "cn=Directory Manager" secretpwd
/var/lib/dirsrv/slapped-instance_name/ldif/demo.ldif
```

The **ldif2ldap** script requires the DN of the administrative user, the password of the administrative user, and the absolute path and filename of the LDIF files to be imported.

For more information about using this script, see the *Directory Server Configuration and Command-Line Tool Reference*.

4.1.6.4. Importing through the `cn=tasks` Entry

The **cn=tasks**, **cn=config** entry in the Directory Server configuration is a container entry for temporary entries that the server uses to manage tasks. Several common directory tasks have container entries under **cn=tasks**, **cn=config**. Temporary task entries can be created under **cn=import**, **cn=tasks**, **cn=config** to initiate an import operation.

As with the **ldif2db** and **ldif2db.pl** scripts, an import operation in **cn=tasks** overwrites all of the information in the database.

This task entry requires three attributes:

- A unique name (**cn**)
- The filename of the LDIF file to import (**nsFilename**)
- The name of the database into which to import the file (**nsInstance**)

It is also possible to supply the DNs of suffixes to include or exclude from the import, analogous to the **-s** and **-x** options, respectively, for the **ldif2db** and **ldif2db.pl** scripts.

The entry is simply added using **ldapmodify**, as described in [Section 3.2.4, “Adding and Modifying Entries Using ldapmodify”](#). For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=example import,cn=import,cn=tasks,cn=config
changetype: add
objectclass: extensibleObject
cn: example import
nsFilename: /home/files/example.ldif
nsInstance: userRoot
nsIncludeSuffix: ou=People,dc=example,dc=com
nsExcludeSuffix: ou=Groups,dc=example,dc=com
```

As soon as the task is completed, the entry is removed from the directory configuration.

The *Directory Server Configuration and Command-Line Tool Reference* has more information on the available attributes for running Directory Server import tasks under the **cn=tasks** entries.

4.2. EXPORTING DATA

LDAP Data Interchange Format (LDIF) files are used to export database entries from the Directory Server databases. LDIF is a standard format described in RFC 2849, *The LDAP Data Interchange Format (LDIF) - Technical Specification*.

Exporting data can be useful for the following:

- Backing up the data in the database.
- Copying data to another Directory Server.
- Exporting data to another application.
- Repopulating databases after a change to the directory topology.

For example, if a directory contains one database, and its contents are split into two databases, then the two new databases receive their data by exporting the contents of the old databases and importing it into the two new databases, as illustrated in [Figure 4.1, “Splitting a Database Contents into Two Databases”](#).



NOTE

The export operations do not export the configuration information (**cn=config**), schema information (**cn=schema**), or monitoring information (**cn=monitor**).

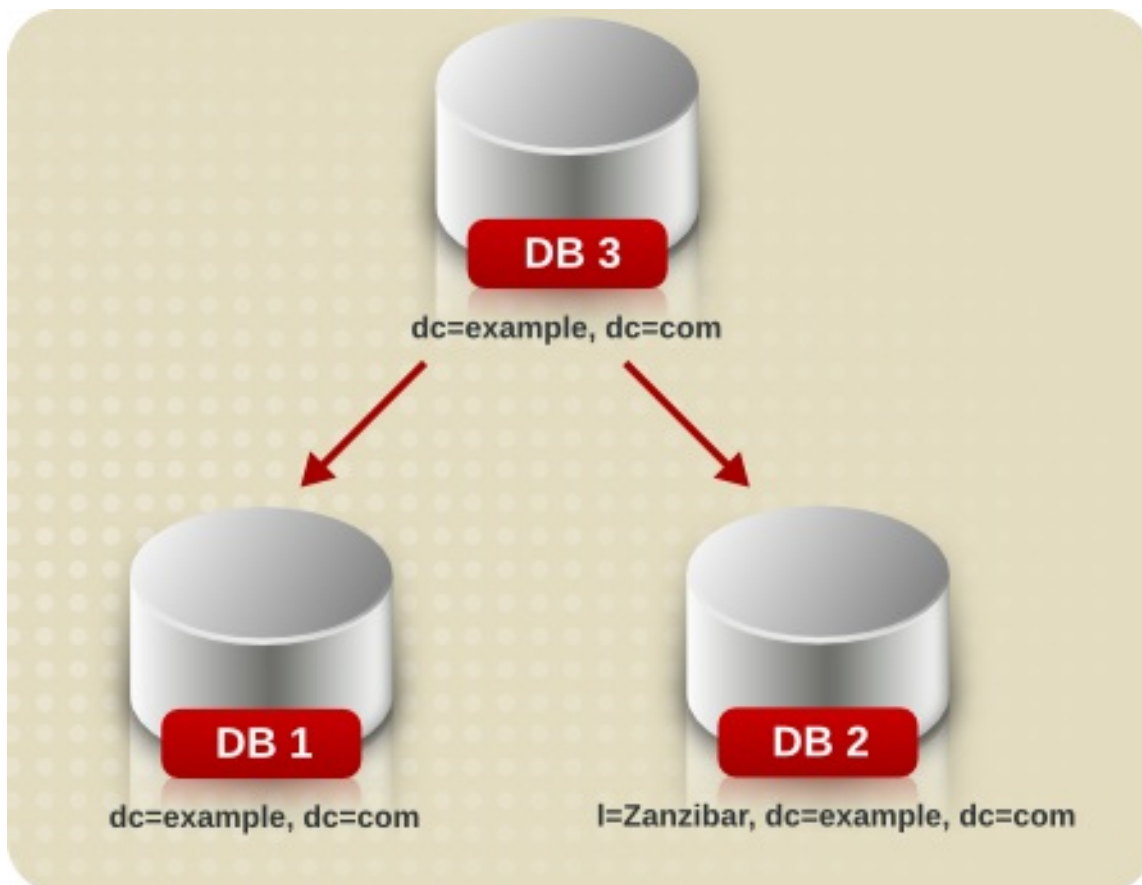


Figure 4.1. Splitting a Database Contents into Two Databases

The Directory Server Console or command-line utilities can be used to export data.

- [Section 4.2.1, “Exporting Directory Data to LDIF Using the Console”](#)
- [Section 4.2.2, “Exporting a Single Database to LDIF Using the Console”](#)
- [Section 4.2.3, “Exporting to LDIF from the Command Line”](#)



WARNING

Do not stop the server during an export operation.

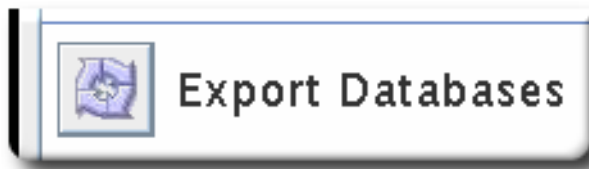
4.2.1. Exporting Directory Data to LDIF Using the Console

Some or all of directory data can be exported to LDIF, depending upon the location of the final exported file. When the LDIF file is on the server, only the data contained by the databases local to the server can be exported. If the LDIF file is remote to the server, all of the databases and database links can be exported.

Export operations can be run to get data from a server instance that is local to the Directory Server Console or from a different host machine (a remote export operation).

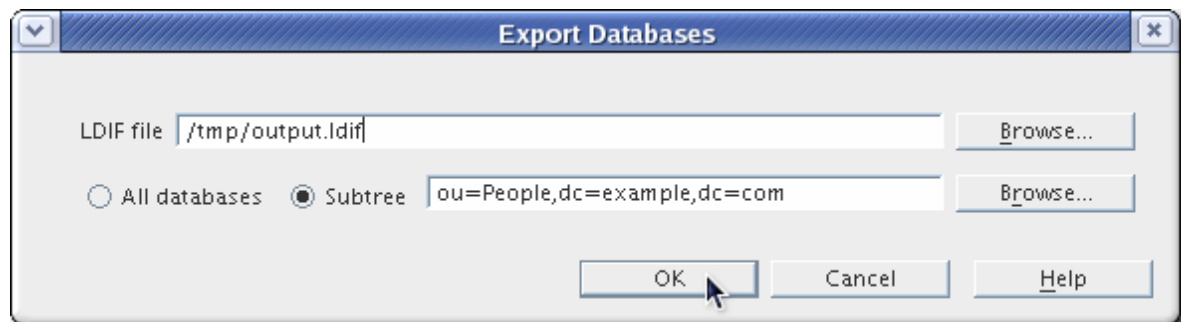
Export directory data to LDIF from the Directory Server Console while the server is running:

1. Select the **Tasks** tab. Scroll to the bottom of the screen, and click **Export Database(s)**.



Alternatively, select the **Configuration** tab and click the **Export from the Console** menu.

2. Enter the full path and filename of the LDIF file in the **LDIF File** field, or click **Browse** to locate the file.



Browse is not enabled if the Console is running on a remote server. When the **Browse** button is not enabled, the file is stored in the default directory, **/var/lib/dirsrv/slaped-instance_name/ldif**.

3. If the Console is running on a machine remote to the server, two radio buttons are displayed beneath the **LDIF File** field.
 - Select **To local machine** to export the data to an LDIF file on the machine from which the Console is running.
 - Select **To server machine** to export to an LDIF file located on the server's machine.
4. To export the whole directory, select the **Entire database** radio button.

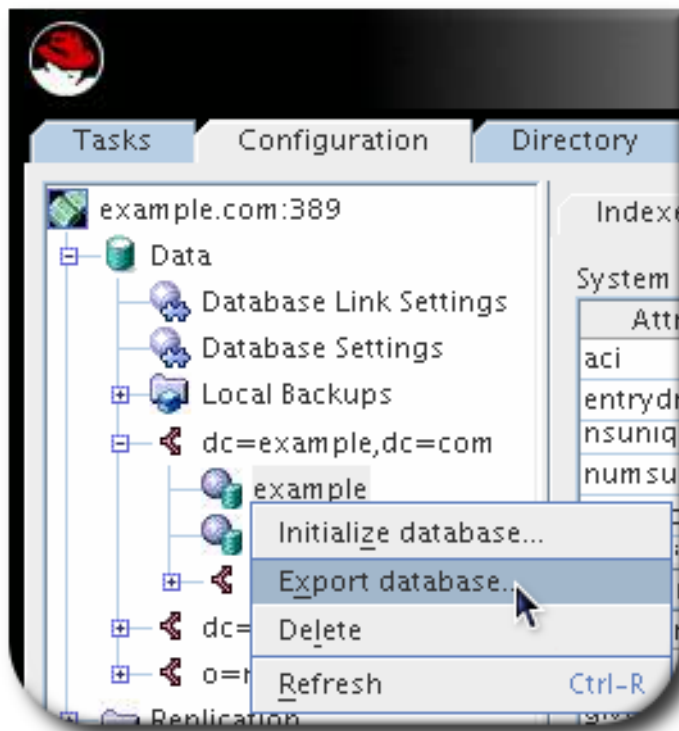
To export only a single subtree of the suffix contained by the database, select the **Subtree** radio button, and then enter the name of the suffix in the **Subtree** text box. This option exports a subtree that is contained by more than one database.

Alternatively, click **Browse** to select a suffix or subtree.

4.2.2. Exporting a Single Database to LDIF Using the Console

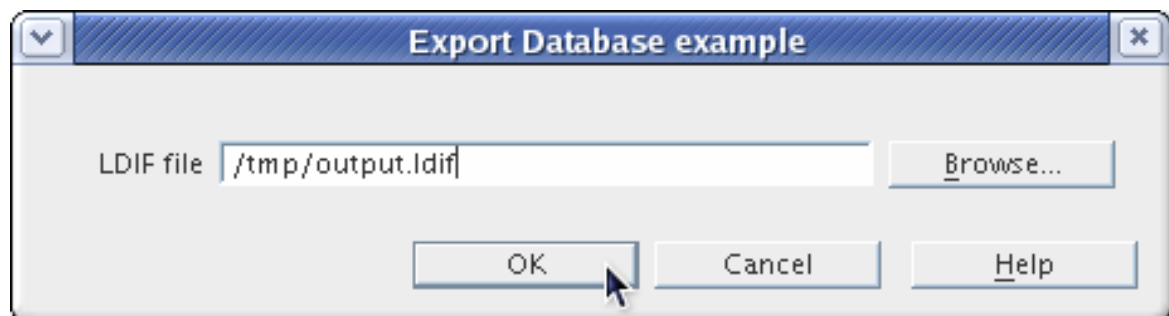
It is also possible to export a single database to LDIF. Do the following while the server is running:

1. Select the **Configuration** tab.
2. Expand the **Data** tree in the left navigation pane. Expand the suffix, and select the database under the suffix.
3. Right-click the database, and select **Export Database**.



Alternatively, select **Export Database** from the **Object** menu.

4. In the **LDIF file** field, enter the full path to the LDIF file, or click **Browse**.



When the **Browse** button is not enabled, the file is stored in the default directory, `/var/lib/dirsrv/slapd-instance_name/ldif`.

4.2.3. Exporting to LDIF from the Command Line

There are three methods for exporting data through the command line:

- *Using db2ldif.* This method runs the command-line utility; unlike the import script, **ldif2db**, this utility can be run while the server is running.
- *Using db2ldif.pl.* This Perl script behaves the same as the **db2ldif** command-line utility and takes the same options.
- *Creating a cn=tasks entry.* This method creates a temporary task entry which automatically launches an export operation. This is functionally like running **db2ldif**, with one exception: when running **db2ldif** or **db2ldif.pl** for a replica (with a **-r** option, the server must be stopped first. The **cn=tasks** entry can be added and export replica information while the server is still running. See [Section 4.2.3.2, “Exporting through the cn=tasks Entry”](#).

4.2.3.1. Exporting a Database Using db2ldif or db2ldif.pl

Databases can be exported to LDIF using the **db2ldif** command-line script or the **db2ldif.pl** Perl script. Both of these tools export all of the database contents or a part of their contents to LDIF when the server is running or stopped.

These script take the same options.



NOTE

The **-E** option is required to export a database that has been encrypted. See [Section 2.2.3.6, “Exporting and Importing an Encrypted Database”](#) for more information.



NOTE

If the database being exported is a replica, then the server must be stopped before the export script is run and the export script must have the **-r**.

To export to LDIF from the command linerun either the **db2ldif** command-line script or the **db2ldif.pl** Perl script. For example:

```
[root@server ~]# db2ldif -n database1 -a /export/output.ldif
```

This exports the database contents to **/export/output.ldif**. If the **-a** option is not specified, then the database information is exported to **/var/lib/dirsrv/slapd-*instance_name*/ldif/instance_name-database1-date.ldif**. For example:

```
[root@server ~]# db2ldif -n database1
```

It is also possible to specify which suffixes to export, using the **-s** option. For example:

```
[root@server ~]# db2ldif -s "dc=example,dc=com"
```

The LDIF file in this case would be **/var/lib/dirsrv/slapd-*instance_name*/ldif/instance_name-example-2017_04_30_112718.ldif**, using the name of the suffix rather than the database.

If the suffix specified is a root suffix, such as **dc=example,dc=com**, then it is not necessary to specify the database or to use the **-n** option.

For more information about using these scripts, see the *Directory Server Configuration and Command-Line Tool Reference*.

Table 4.4. db2ldif Options

Option	Description
-n	Specifies the name of the database from which the file is being exported.

Option	Description
-s	Specifies the suffix or suffixes to include in the export. If the suffix is a root suffix, such as dc=example , dc=com , then the -n option is not required. There can be multiple -s arguments.
-a	Defines the output file to which Directory Server exports the LDIF. This file must be an absolute path. If the -a option is not given, the output ldif is stored in the /var/lib/dirsrv/slaped-instance_name/ldif directory and is automatically named serverID-database-YYYY_MM_DD_hhmmxx.ldif with the -n option or serverID-firstsuffixvalue-YYYY_MM_DD_hhmmxx.ldif with the -s option.
-r	Specifies that the exported database is a consumer replica. In this case, the appropriate settings and entries are included with the LDIF to initialize the replica when the LDIF is imported.
-E	Decrypts an encrypted database so it can be exported.

4.2.3.2. Exporting through the cn=tasks Entry

The **cn=tasks**, **cn=config** entry in the Directory Server configuration is a container entry for temporary entries that the server uses to manage tasks. Several common directory tasks have container entries under **cn=tasks**, **cn=config**. Temporary task entries can be created under **cn=export**, **cn=tasks**, **cn=config** to initiate an export operation.

The export task entry requires three attributes:

- A unique name (**cn**)
- The filename of the LDIF file to which to export the database (**nsFilename**)
- The name of the database to export (**nsInstance**)

It is also possible to supply the DNs of suffixes to include or exclude from the export operation, analogous to the **-s** and **-x** options, respectively, for the **db2ldif** and **db2ldif.pl** scripts. Additionally, if the database is a replica, then the appropriate replica information can be included to initialize the new consumer when the LDIF is imported; this is set in the **nsExportReplica**, corresponding to the **-r** option.

The entry is simply added using **ldapmodify**, as described in [Section 3.2.4, “Adding and Modifying Entries Using ldapmodify”](#). For example:

```
ldapmodify -a -D "cn=directory manager" -w -p 389 -h server.example.com -x
dn: cn=example export,cn=export,cn=tasks,cn=config
```

```
changetype: add
objectclass: extensibleObject
cn: example export
nsInstance: userRoot
nsFilename: /home/files/example.ldif
nsExportReplica: true
nsIncludeSuffix: ou=People,dc=example,dc=com
nsExcludeSuffix: ou=Groups,dc=example,dc=com
```

As soon as the task is completed, the entry is removed from the directory configuration.

The *Directory Server Configuration and Command-Line Tool Reference* has more information on the available attributes for running Directory Server export tasks under the **cn=tasks** entries.

4.3. BACKING UP AND RESTORING DATA

Databases can be backed up and restored using the Directory Server Console or a command-line script.

- [Section 4.3.1, “Backing up All Databases”](#)
- [Section 4.3.2, “Backing up the dse.ldif Configuration File”](#)
- [Section 4.3.3, “Restoring All Databases”](#)
- [Section 4.3.4, “Restoring a Single Database”](#)
- [Section 4.3.5, “Restoring Databases That Include Replicated Entries”](#)
- [Section 4.3.6, “Restoring the dse.ldif Configuration File”](#)



WARNING

Do not stop the server during a backup or restore operation.

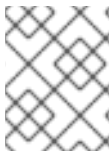


NOTE

The changelog database is backed up with the regular server database.

4.3.1. Backing up All Databases

The following procedures describe backing up all of the databases in the directory using the Directory Server Console and from the command line.



NOTE

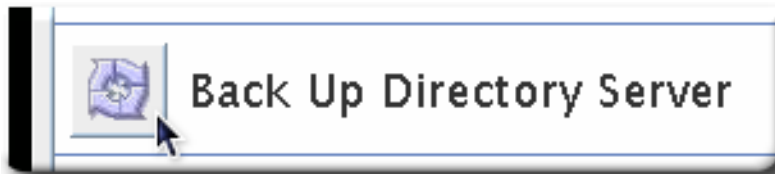
These backup methods cannot be used to back up the data contained by databases on a remote server that are chained using database links.

4.3.1.1. Backing up All Databases from the Console

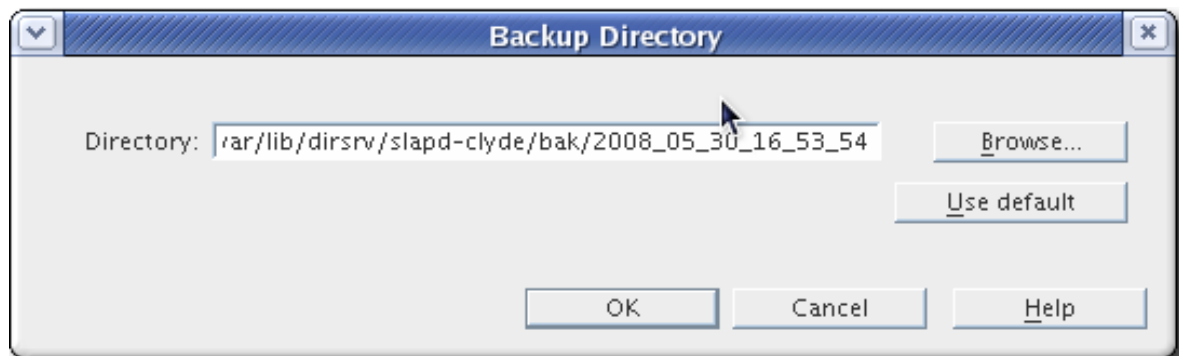
When backing up databases from the Directory Server Console, the server copies all of the database contents and associated index files to a backup location. A backup can be performed while the server is running.

To back up databases from the Directory Server Console:

1. Select the **Tasks** tab.
2. Click **Back Up Directory Server**.



3. Enter the full path of the directory to store the backup file in the **Directory** text box, or click **Use default**, and the server provides a name for the backup directory.



If the Console is running on the same machine as the directory, click **Browse** to select a local directory.

With the default location, the backup files are placed in **/var/lib/dirsrv/slapd-*instance_name*/bak**. By default, the backup directory name contains the name of the server instance and the time and date the backup was created (*instance_name-YYYY_MM_DD_hhmmss*).

4.3.1.2. Backing up All Databases from the Command Line

Databases can be backed up from the command line using either the **db2bak** command-line script or the **db2bak** Perl script. The command-line script works when the server is running or when the server is stopped; the Perl script can only be run when the server is running.

IMPORTANT

If the database being backed up is a master database, meaning it keeps a changelog, then it must be backed up using the **db2bak.p1** Perl script or using the Directory Server Console if the server is kept running. The changelog only writes its RUV entries to the database when the server is shut down; while the server is running, the changelog keeps its changes in memory. For the Perl script and the Console, these changelog RUVs are written to the database before the backup process runs. However, that step is not performed by the command-line script.

The **db2bak** should not be run on a running master server. Either use the Perl script or stop the server before performing the backup.

Configuration information *cannot* be backed up using this backup method. For information on backing up the configuration information, see [Section 4.3.2, “Backing up the dse.ldif Configuration File”](#).

To back up the directory from the command line using the **db2bak.p1** script, run the **db2bak.p1** Perl script, specifying the backup filename and directory.

```
[root@server ~]# /usr/lib[64]/dirsrv/slapd-example/db2bak.p1
/var/lib/dirsrv/slapd-instance_name/bak/instance_name-2017_04_30_16_27_56
```

The backup directory where the server saves the backed up databases can be specified with the script. If a directory is not specified, the backup file is stored in **/var/lib/dirsrv/slapd-*instance_name*/bak**. By default, the backup directory is named with the Directory Server instance name and the date of the backup (*serverID-YYYY_MM_DD_hhmmss*).

For more information about using these scripts, see the *Directory Server Configuration and Command-Line Tool Reference*.

4.3.1.3. Backing up the Database through the cn=tasks Entry

The **cn=tasks**, **cn=config** entry in the Directory Server configuration is a container entry for temporary entries that the server uses to manage tasks. Several common directory tasks have container entries under **cn=tasks**, **cn=config**. Temporary task entries can be created under **cn=backup**, **cn=tasks**, **cn=config** to initiate a backup operation.

The backup task entry requires three attributes:

- A unique name (**cn**).
- The directory to write the backup file to (**nsArchiveDir**). The backup file is named with the Directory Server instance name and the date of the backup (*serverID-YYYY_MM_DD_hhmmss*).
- The type of database (**nsDatabaseType**); the only option is **ldbm database**.

The entry is simply added using **ldapmodify**, as described in [Section 3.2.4, “Adding and Modifying Entries Using ldapmodify”](#). For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x

dn: cn=example backup,cn=backup,cn=tasks,cn=config
changetype: add
objectclass: extensibleObject
cn: example backup
```

```
nsArchiveDir: /export/backups/  
nsDatabaseType: ldbm database
```

As soon as the task is completed, the entry is removed from the directory configuration.

The *Directory Server Configuration and Command-Line Tool Reference* has more information on the available attributes for running tasks to back up databases under the **cn=tasks** entries.

4.3.2. Backing up the dse.ldif Configuration File

Directory Server automatically backs up the **dse.ldif** configuration file. When the Directory Server is started, the directory creates a backup of the **dse.ldif** file automatically in a file named **dse.ldif.startOK** in the **/etc/dirsrv/slapd-*instance_name*** directory.

When the **dse.ldif** file is modified, the file is first backed up to a file called **dse.ldif.bak** in the **/etc/dirsrv/slapd-*instance_name*** directory before the directory writes the modifications to the **dse.ldif** file.

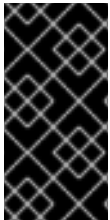
4.3.3. Restoring All Databases

The following procedures describe restoring all of the databases in the directory using the Directory Server Console and from the command line.



NOTE

Restoring a database from backup also restores the changelog.



IMPORTANT

While restoring databases, the server must be running. However, the databases will be unavailable for processing operations during the restore.

Therefore, **stop all replication processes before attempting to restore a database.**

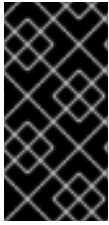
4.3.3.1. Restoring All Databases from the Console

If the databases become corrupted, restore data from a previously generated backup using the Directory Server Console. This process consists of stopping the server and then copying the databases and associated index files from the backup location to the database directory.



WARNING

Restoring databases overwrites any existing database files.



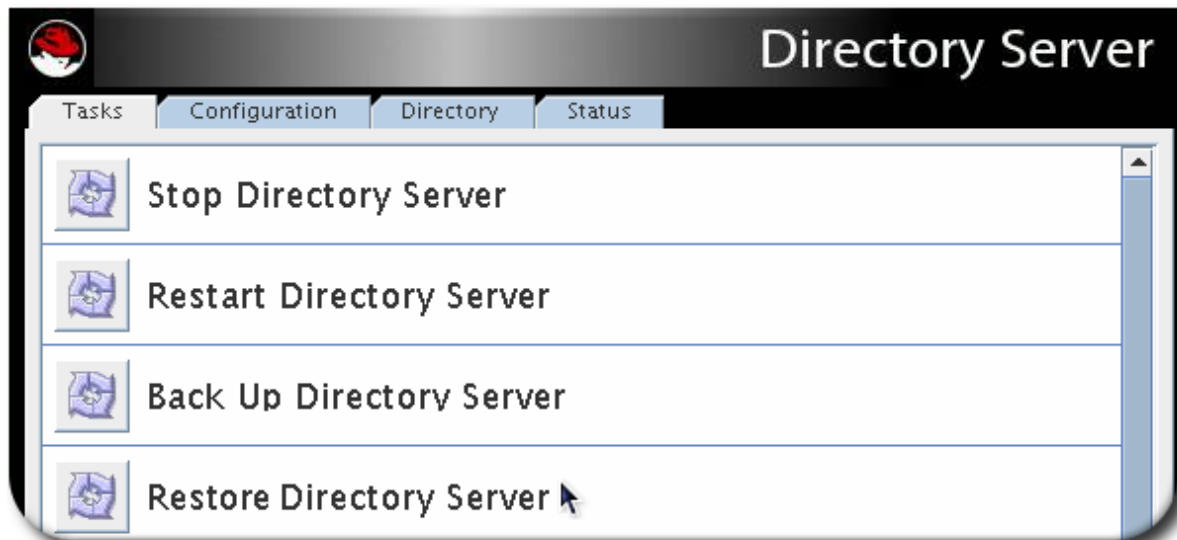
IMPORTANT

While restoring databases, the server must be running. However, the databases will be unavailable for processing operations during the restore.

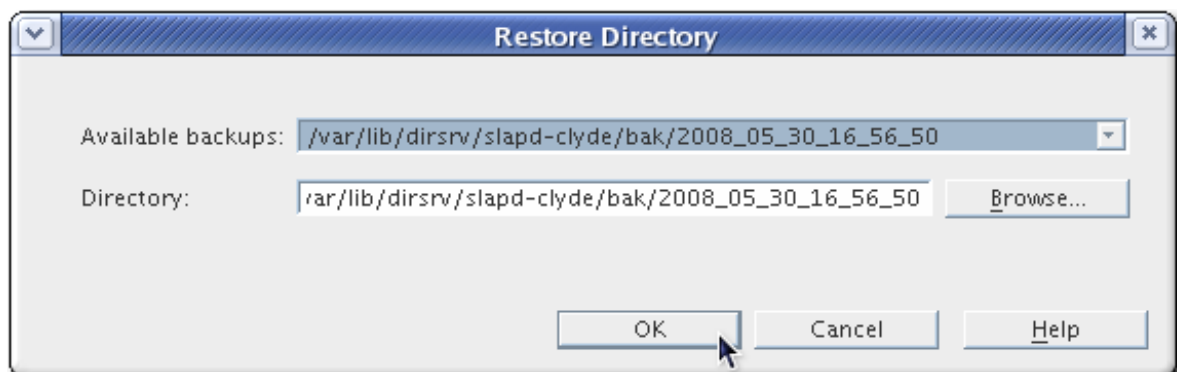
Therefore, **stop all replication processes before attempting to restore a database.**

To restore databases from a previously created backup:

1. In the Directory Server Console, select the **Tasks** tab.
2. Click **Restore Directory Server**.



3. Select the backup from the **Available Backups** list, or enter the full path to a valid backup in the **Directory** text box.



The **Available Backups** list shows all backups located in the default directory, `/var/lib/dirsrv/slapd-instance_name/bak/backup_directory`. *backup_directory* is the directory of the most recent backup, in the form *serverID-YYYY_MM_DD_hhmmss*.

4.3.3.2. Restoring Databases from the Command Line

There are three ways to restore databases from the command line:

- Using the **bak2db** command-line script. This script requires the server to be shut down.

- Using the **bak2db.pl** Perl script. This script works while the server is running.
- Creating a temporary entry under **cn=restore,cn=tasks,cn=config**. This method can also be run while the server is running.



IMPORTANT

While restoring databases, the server must be running (with the exception of running the **bak2db** command-line script). However, the databases will be unavailable for processing operations during the restore.

Therefore, **stop all replication processes before attempting to restore a database.**

4.3.3.2.1. Using the bak2db Command-Line Script

1. If the Directory Server is running, stop it:

```
[root@server ~]# service dirsrv stop instance
```

2. Run the **bak2db** command-line script. The **bak2db** script requires the full path and name of the input file.

```
[root@server ~]# /etc/dirsrv/slapd-instance_name/bak2db
/var/lib/dirsrv/slapd-instance_name/bak/instance_name-
2017_04_30_11_48_30
```

For more information about using this script, see the *Directory Server Configuration and Command-Line Tool Reference*.

4.3.3.2.2. Using bak2db.pl Perl Script

Run the **bak2db.pl** Perl script.

```
[root@server ~]# /usr/lib[64]/dirsrv/slapd-example/bak2db.pl -D
"cn=Directory Manager" -w secret -a
/var/lib/dirsrv/slapd-instance_name/bak/instance_name-2017_04_30_11_48_30
```

For more information on using this Perl script, see the *Directory Server Configuration and Command-Line Tool Reference*.



IMPORTANT

While restoring databases, the server must be running. However, the databases will be unavailable for processing operations during the restore.

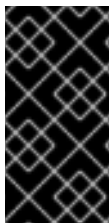
Therefore, **stop all replication processes before attempting to restore a database.**

Option	Description
-a	Defines the full path and name of the input file.

Option	Description
-D	Specifies the DN of the administrative user.
-w	Specifies the password of the administrative user.

4.3.3.2.3. Restoring the Database through the `cn=tasks` Entry

The `cn=tasks, cn=config` entry in the Directory Server configuration is a container entry for temporary entries that the server uses to manage tasks. Several common directory tasks have container entries under `cn=tasks, cn=config`. Temporary task entries can be created under `cn=restore, cn=tasks, cn=config` to initiate a restore operation.



IMPORTANT

While restoring databases, the server must be running. However, the databases will be unavailable for processing operations during the restore.

Therefore, **stop all replication processes before attempting to restore a database.**

The restore task entry requires three attributes, the same as the backup task:

- A unique name (`cn`).
- The directory from which to retrieve the backup file (`nsArchiveDir`).
- The type of database (`nsDatabaseType`); the only option is **ldbm database**.

The entry is simply added using `ldapmodify`, as described in [Section 3.2.4, “Adding and Modifying Entries Using ldapmodify”](#). For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=example restore,cn=restore,cn=tasks,cn=config
changetype: add
objectclass: extensibleObject
cn: example restore
nsArchiveDir: /export/backups/
nsDatabaseType: ldbm database
```

As soon as the task is completed, the entry is removed from the directory configuration.

The *Directory Server Configuration and Command-Line Tool Reference* has more information on the available attributes for running database restore tasks under the `cn=tasks` entries.

4.3.4. Restoring a Single Database

It is possible to restore a single database through the command line, but not in the Directory Server Console. To restore a single database:

1. Stop the Directory Server if it is running.

```
service dirsrv stop instance
```

2. Restore the back end from the `/var/lib/dirsrv/slapd-instance_name/bak` archives with the **bak2db** script, using the **-n** parameter to specify the database name. For example:

```
bak2db /var/lib/dirsrv/slapd-instance_name/bak/backup_file -n  
userRoot
```

3. Restart the Directory Server.

```
service dirsrv start instance
```



NOTE

If the Directory Server fails to start, remove the database transaction log files in `/var/lib/dirsrv/slapd-instance_name/db/log.###`, then retry starting the server.

4.3.5. Restoring Databases That Include Replicated Entries

Several situations can occur when a supplier server is restored:

- The consumer servers are also restored.

If all databases are restored from backups taken at the same time (so that the data are in sync), the consumers remain synchronized with the supplier, and it is not necessary to do anything else. Replication resumes without interruption.

- Only the supplier is restored.

If only the supplier is restored or if the consumers are restored from backups taken at a different times, reinitialize the consumers for the supplier to update the data in the database. If only the supplier is restored or if the consumers are restored from backups taken at a different times, reinitialize the consumers for the supplier to update the data in the database.

- Changelog entries have not yet expired on the supplier server.

If the supplier's changelog has not expired since the database backup was taken, then restore the local consumer and continue with normal operations. This situation occurs only if the backup was taken within a period of time that is shorter than the value set for the maximum changelog age attribute, **nsslapd-changelogmaxage**, in the **cn=changelog5,cn=config** entry. For more information about this option, see the *Directory Server Configuration and Command-Line Tool Reference*.

Directory Server automatically detects the compatibility between the replica and its changelog. If a mismatch is detected, the server removes the old changelog file and creates a new, empty one.

- Changelog entries have expired on the supplier server since the time of the local backup.

If changelog entries have expired, reinitialize the consumer. For more information on reinitializing consumers, see [Section 11.15, “Initializing Consumers”](#).

The changelog associated with the restored database will be erased during the restore operation. A message will be logged to the supplier servers' log files indicating that reinitialization is required.

For information on managing replication, see [Chapter 11, *Managing Replication*](#).

4.3.6. Restoring the `dse.ldif` Configuration File

The directory creates two backup copies of the `dse.ldif` file in the `/etc/dirsrv/slapd-instance_name` directory. The `dse.ldif.startOK` file records a copy of the `dse.ldif` file at server start up. The `dse.ldif.bak` file contains a backup of the most recent changes to the `dse.ldif` file. Use the version with the most recent changes to restore the directory.

To restore the `dse.ldif` configuration file:

1. Stop the server.

```
service dirsrv stop instance
```

2. Restore the database as outlined in [Section 4.3.4, “Restoring a Single Database”](#) to copy the backup copy of the `dse.ldif` file into the directory.

3. Restart the server.

```
service dirsrv restart instance
```

CHAPTER 5. MANAGING ATTRIBUTES AND VALUES

Red Hat Directory Server provides several different mechanisms for dynamically and automatically maintaining some types of attributes on directory entries. These plug-ins and configuration options simplify managing directory data and expressing relationships between entries.

Part of the characteristic of entries are their *relationships* to each other. Obviously, a manager has an employee, so those two entries are related. Groups are associated with their members. There are less apparent relationships, too, like between entries which share a common physical location.

Red Hat Directory Server provides several different ways that these relationships between entries can be maintained smoothly and consistently. There are several plug-ins can apply or generate attributes automatically as part of the data within the directory, including classes of service, linking attributes, and generating unique numeric attribute values.

5.1. ENFORCING ATTRIBUTE UNIQUENESS

Attribute uniqueness means that the Directory Server requires that all new or edited attributes always have unique values. Attribute uniqueness is enforced through a plug-in. A new instance of the Attribute Uniqueness Plug-in must be created for every attribute for which values must be unique.

5.1.1. Attribute Uniqueness Plug-in Syntax

Configuration information for the Attribute Uniqueness Plug-in is specified in an entry under **cn=plugins, cn=config** entry. There are two possible syntaxes for **nsslapd-pluginarg** attributes.



NOTE

To enforce uniqueness of another attribute than the ones in these example, copy and paste the default Attribute Uniqueness Plug-in entry, and being care to change only the attributes described here.

Use the following syntax to perform the uniqueness check under a suffix or subtree:

```
dn: cn=descriptive_plugin_name,cn=plugins,cn=config
...
nsslapd-pluginEnabled: state
nsslapd-pluginarg0: attribute_name
nsslapd-pluginarg1: dn1
nsslapd-pluginarg2: dn2
...
```

- Any value can be given to the **cn** attribute to name the plug-in. The name should be descriptive.
- The **cn** attribute does not contain the name of the attribute which is checked for uniqueness.
- Only one attribute can be specified on which the uniqueness check will be performed.
- It is possible to specify several DN's of suffixes or subtrees in which to perform the uniqueness check by incrementing the **nsslapd-pluginarg** attribute suffix by one each time.

The variable components of the Attribute Uniqueness Plug-in syntax are described in [Table 5.1, "Attribute Uniqueness Plug-in Variables"](#).

Use the following syntax to specify to perform the uniqueness check below an entry containing a specified object class:

```
dn: cn=descriptive_plugin_name,cn=plugins,cn=config
...
nsslapd-pluginEnabled: state
nsslapd-pluginarg0: attribute=attribute_name
nsslapd-pluginarg1: markerObjectClass=objectclass1
nsslapd-pluginarg2: requiredObjectClass=objectclass2
...
```

- Any value can be given to the **cn** attribute to name the plug-in. The name should be descriptive.
- The **cn** attribute does not contain the name of the attribute which is checked for uniqueness.
- Only one attribute can be specified on which the uniqueness check will be performed.
- If the **nsslapd-pluginarg0** attribute begins with **attribute=attribute_name**, then the server expects the **nsslapd-pluginarg1** attribute to include a **markerObjectClass** value.

The variable components of the attribute uniqueness plug-in syntax are described in [Table 5.1, “Attribute Uniqueness Plug-in Variables”](#).

Table 5.1. Attribute Uniqueness Plug-in Variables

Variable	Definition
<i>descriptive_plugin_name</i>	Specifies the name of this instance of the Attribute Uniqueness Plug-in. It is not required that the name of the attribute for which to ensure uniqueness be included, but it is advisable. For example, cn=mail uniqueness, cn=plugins, cn=config .
<i>state</i>	Defines whether the plug-in is enabled or disabled. Acceptable values are on or off .
<i>attribute_name</i>	The name of the attribute for which to ensure unique values. Only one attribute can be named.
<i>dn</i>	The DN of the suffix or subtree in which to ensure attribute uniqueness. To specify several suffixes or subtrees, increment the suffix of the nsslapd-pluginarg attribute by one for each additional suffix or subtree.
attribute= <i>attribute_name</i>	The name of the attribute for which to ensure unique values. Only one attribute can be named.
markerObjectClass= <i>objectclass1</i>	Attribute uniqueness will be checked under the entry belonging to the DN of the updated entry that has the object class specified in the markerObjectClass keyword. Do not include a space before or after the equals sign.

Variable	Definition
requiredObjectClass= <i>objectclass2</i>	<i>Optional.</i> When using the markerObjectClass keyword to specify the scope of the uniqueness check instead of a DN, it is also possible to specify to perform the check only if the updated entry contains the objectclass specified in the requiredObjectClass keyword. Do not include a space before or after the equals sign.

5.1.2. Creating an Instance of the Attribute Uniqueness Plug-in

To ensure that a particular attribute in the directory always has unique values, create an instance of the Attribute Uniqueness Plug-in for the attribute to check. For example, to ensure that every entry in the directory that includes a **mail** attribute has a unique value for that attribute, create a mail uniqueness plug-in.

To create an instance of the Attribute Uniqueness Plug-in, modify the Directory Server configuration to add an entry for the new plug-in under the **cn=plugins,cn=config** entry. The format of the new entry must conform to the syntax described in [Section 5.1.1, “Attribute Uniqueness Plug-in Syntax”](#).



NOTE

Red Hat strongly encourages you to copy and paste an existing Attribute Uniqueness Plug-in entry and only modify the attributes listed below.

For example, to create an instance the Attribute Uniqueness Plug-in for the mail attribute:

1. Stop the Directory Server. Changes to the **dse.ldif** file are not saved if it is edited while the server is running.

```
service dirsrv stop instance_name
```

2. In the **dse.ldif** file, locate the entry for the Attribute Uniqueness Plug-in, **cn=attribute uniqueness,cn=plugins,cn=config**.
3. Copy the entire entry. The entry ends in an empty line; copy the empty line, too.
4. Paste the copied Attribute Uniqueness Plug-in entry at the end of the file.
5. Modify the Attribute Uniqueness Plug-in entry attributes for the new attribute information:

```
dn: cn=mail uniqueness,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: mail
nsslapd-pluginarg1: dc=example,dc=com
...
```

6. Restart the Directory Server.

```
service dirsrv start instance_name
```

In this example, the uniqueness check will be performed on every entry in the **dc=example,dc=com** entry that includes the **mail** attribute.

5.1.3. Configuring Attribute Uniqueness

This section explains how to use Directory Server Console to view the plug-ins configured for the directory and how to modify the configuration of the Attribute Uniqueness Plug-ins.

5.1.3.1. Configuring Attribute Uniqueness Plug-ins from the Directory Server Console

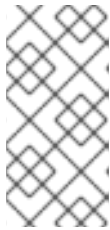
1. In the Directory Server Console, select the **Configuration** tab; then, in the navigation tree, expand the **Plug-ins** folder, and select the Attribute Uniqueness Plug-in to modify.

The configuration parameters for the plug-in are displayed in the right pane.

2. To add a suffix or subtree, click **Add**, and type a DN in the blank text field.

To delete a suffix from the list, place the cursor in the text field to delete, and click **Delete**.

To avoid using a DN, enter the **markerObjectClass** keyword. With this syntax, it is possible to click **Add** again to specify a **requiredObjectClass**, as described in [Section 5.1.1, “Attribute Uniqueness Plug-in Syntax”](#).



NOTE

Do *not* add an attribute name to the list. To check the uniqueness of other attributes, create a new instance of the Attribute Uniqueness Plug-in for the attribute to check. For information, see [Section 5.1.2, “Creating an Instance of the Attribute Uniqueness Plug-in”](#).

3. Click **Save**.

5.1.3.2. Configuring Attribute Uniqueness Plug-ins from the Command Line

This section provides information about configuring the plug-in from the command line.

- [Section 5.1.3.2.1, “Specifying a Suffix or Subtree”](#)
- [Section 5.1.3.2.2, “Using the markerObjectClass and requiredObjectClass Keywords”](#)

5.1.3.2.1. Specifying a Suffix or Subtree

The suffix or subtrees which the plug-in checks to ensure attribute uniqueness are defined using the **nsslapd-pluginarg** attribute in the entry defining the plug-in.

To specify the subtree or subtrees, use **ldapmodify** to send LDIF update statements. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=mail uniqueness,cn=plugins,cn=config
changetype: modify
```

```
add: nsslapd-pluginarg2 nsslapd-pluginarg3
nsslapd-pluginarg2: ou=Engineering,dc=example,dc=com
nsslapd-pluginarg3: ou=Sales,dc=example,dc=com
```

This example LDIF statement modified the Attribute Uniqueness Plug-in to check the uniqueness of the **mail** attribute under the subtrees **dc=example,dc=com, ou=Engineering,dc=example,dc=com**, and **ou=Sales,dc=example,dc=com**.

Whenever this type of configuration change is made, restart the server.

```
service dirsrv restart instance_name
```

For information on restarting the server, see [Section 1.3, “Starting and Stopping Servers”](#).

5.1.3.2.2. Using the **markerObjectClass** and **requiredObjectClass** Keywords

Instead of specifying a suffix or subtree in the configuration of an Attribute Uniqueness Plug-in, perform the check under the entry belonging to the DN of the updated entry that has the object class given in the **markerObjectClass** keyword.

To specify to perform the uniqueness check under the entry in the DN of the updated entry that contains the organizational unit (**ou**) object class, copy and paste an existing Attribute Uniqueness Plug-in entry, and change the following attributes:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x

dn: cn=mail uniqueness,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: attribute=mail
nsslapd-pluginarg1: markerObjectClass=organizationalUnit
...
```

If the server should not check every entry in the organization unit, limit the scope by setting the check to be performed only if the updated entry contains a specified object class.

For example, if the uniqueness of the **mail** attribute is checked, it is probably only necessary to perform the check when adding or modifying entries with the **person** or **inetorgperson** object class.

Restrict the scope of the check by using the **requiredObjectClass** keyword, as shown in the following example:

```
dn: cn=mail uniqueness,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: attribute=mail
nsslapd-pluginarg1: markerObjectClass=organizationalUnit
nsslapd-pluginarg2: requiredObjectClass=person
...
```

The **markerObjectClass** or **requiredObjectClass** keywords *cannot* be repeated by incrementing the counter in the **nsslapd-pluginarg** attribute suffix. These keywords can only be used once per Attribute Uniqueness Plug-in instance.

**NOTE**

The ***nsslapd-pluginarg0*** attribute always contains the name of the attribute for which to ensure uniqueness.

5.1.4. Attribute Uniqueness Plug-in Syntax Examples

This section contains examples of Attribute Uniqueness Plug-in syntax in the **dse.ldif** file.

- [Section 5.1.4.1, “Specifying One Attribute and One Subtree”](#)
- [Section 5.1.4.2, “Specifying One Attribute and Multiple Subtrees”](#)

5.1.4.1. Specifying One Attribute and One Subtree

This example configures the plug-in to ensure the uniqueness of the **mail** attribute under the **dc=example,dc=com** subtree.

```
dn: cn=mail uniqueness,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: mail
nsslapd-pluginarg1: dc=example,dc=com
...
```

5.1.4.2. Specifying One Attribute and Multiple Subtrees

It is possible use a single plug-in instance to check for the uniqueness of an attribute within multiple subtrees, which means that the attribute value must be unique *within* each subtree but not unique across all subtrees. This example configures the Attribute Uniqueness Plug-in to ensure the uniqueness of the **mail** attribute for separate subtrees, **l=Chicago,dc=example,dc=com** and **l=Boston,dc=example,dc=com**.

```
dn: cn=mail uniqueness,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: mail
nsslapd-pluginarg1: l=Chicago,dc=example,dc=com
nsslapd-pluginarg2: l=Boston,dc=example,dc=com
...
```

**NOTE**

The ***nsslapd-pluginarg0*** attribute always contains the name of the attribute for which to ensure uniqueness. All other occurrences of the ***nsslapd-pluginarg***, such as ***nsslapd-pluginarg1***, contain DNs.

With this configuration, the plug-in allows an instance of a value for the **mail** attribute to exist once under the **l=Chicago,dc=example,dc=com** subtree and once under the **l=Boston,dc=example,dc=com** subtree. For example, the following two attribute-value settings are allowed:

```
mail=bjensen,l=Chicago,dc=example,dc=com  
mail=bjensen,l=Boston,dc=example,dc=com
```

To ensure that only one instance of a value exists under both subtrees, configure the plug-in to ensure uniqueness for the entire **dc=example, dc=com** subtree.

5.2. ASSIGNING CLASS OF SERVICE

A *class of service definition* (CoS) shares attributes between entries in a way that is transparent to applications. CoS simplifies entry management and reduces storage requirements.

Clients of the Directory Server read the attributes in a user's entry. With CoS, some attribute values may not be stored within the entry itself. Instead, these attribute values are generated by class of service logic as the entry is sent to the client application.

Each CoS is comprised of two types of entry in the directory:

- *CoS definition entry.* The CoS definition entry identifies the type of CoS used. Like the role definition entry, it inherits from the **LDAPsubentry** object class. The CoS definition entry is below the branch at which it is effective.
- *Template entry.* The CoS template entry contains a list of the shared attribute values. Changes to the template entry attribute values are automatically applied to all the entries within the scope of the CoS. A single CoS might have more than one template entry associated with it.

The CoS definition entry and template entry interact to provide attribute information to their target entries, any entry within the scope of the CoS.

5.2.1. About the CoS Definition Entry

The CoS definition entry is an instance of the **cosSuperDefinition** object class. The CoS definition entry also contains one of three object class that specifies the type of template entry it uses to generate the entry. The target entries which interact with the CoS share the same parent as the CoS definition entry.

There are three types of CoS, defined using three types of CoS definition entries:

- *Pointer CoS.* A pointer CoS identifies the template entry using the template DN only.
- *Indirect CoS.* An indirect CoS identifies the template entry using the value of one of the target entry's attributes. For example, an indirect CoS might specify the **manager** attribute of a target entry. The value of the **manager** attribute is then used to identify the template entry.

The target entry's attribute must be single-valued and contain a DN.

- *Classic CoS.* A classic CoS identifies the template entry using a combination of the template entry's base DN and the value of one of the target entry's attributes.

For more information about the object classes and attributes associated with each type of CoS, see [Section 5.2.11, "Managing CoS from the Command Line"](#).

If the CoS logic detects that an entry contains an attribute for which the CoS is generating values, the CoS, by default, supplies the client application with the attribute value in the entry itself. However, the CoS definition entry can control this behavior.

5.2.2. About the CoS Template Entry

The CoS template entry contains the value or values of the attributes generated by the CoS logic. The CoS template entry contains a general object class of **cosTemplate**. The CoS template entries for a given CoS are stored in the directory tree along with the CoS definition.

The relative distinguished name (RDN) of the template entry is determined by one of the following:

- The DN of the template entry alone. This type of template is associated with a pointer CoS definition.
- The value of one of the target entry's attributes. The attribute used to provide the relative DN to the template entry is specified in the CoS definition entry using the **cosIndirectSpecifier** attribute. This type of template is associated with an indirect CoS definition.
- By a combination of the DN of the subtree where the CoS performs a one level search for templates and the value of one of the target entry's attributes. This type of template is associated with a classic CoS definition.

5.2.3. How a Pointer CoS Works

An administrator creates a pointer CoS that shares a common postal code with all of the entries stored under **dc=example,dc=com**. The three entries for this CoS appear as illustrated in [Figure 5.1, "Sample Pointer CoS"](#).

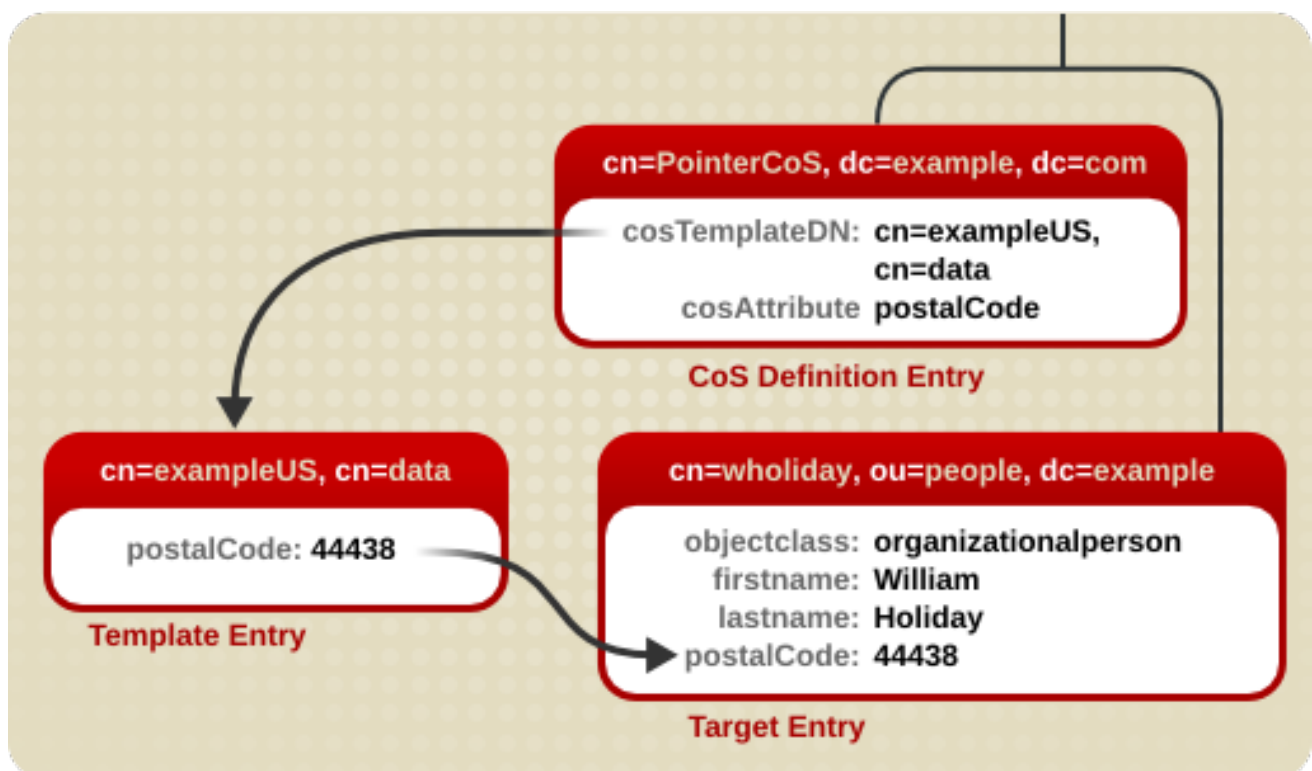


Figure 5.1. Sample Pointer CoS

In this example, the template entry is identified by its DN, **cn=exampleUS, cn=data**, in the CoS definition entry. Each time the **postalCode** attribute is queried on the entry **cn=wholiday, ou=people, dc=example, dc=com**, the Directory Server returns the value available in the template entry **cn=exampleUS, cn=data**.

5.2.4. How an Indirect CoS Works

An administrator creates an indirect CoS that uses the *manager* attribute of the target entry to identify the template entry. The three CoS entries appear as illustrated in [Figure 5.2, “Sample Indirect CoS”](#).

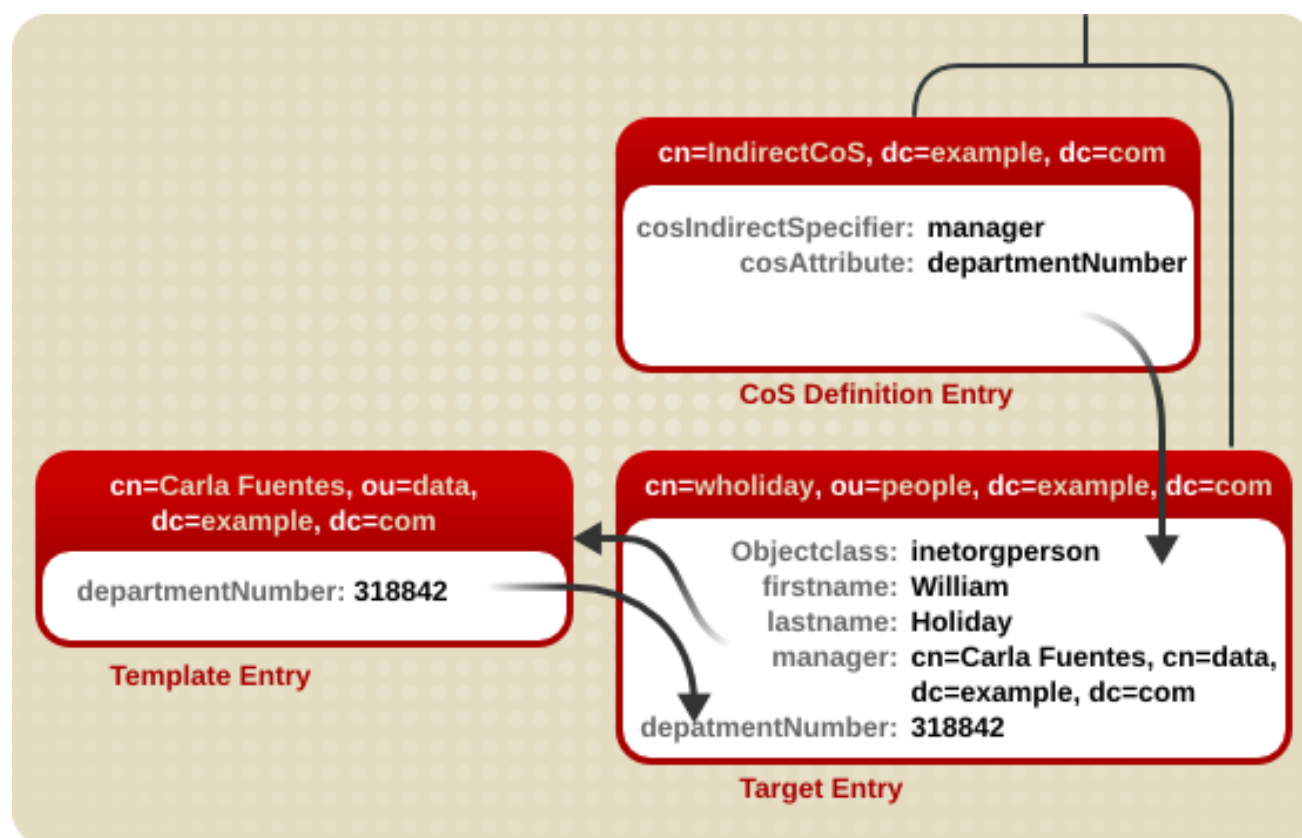


Figure 5.2. Sample Indirect CoS

In this example, the target entry for William Holiday contains the indirect specifier, the *manager* attribute. William's manager is Carla Fuentes, so the *manager* attribute contains a pointer to the DN of the template entry, `cn=Carla Fuentes, ou=people, dc=example, dc=com`. The template entry in turn provides the *departmentNumber* attribute value of `318842`.

5.2.5. How a Classic CoS Works

An administrator creates a classic CoS that uses a combination of the template DN and a CoS specifier to identify the template entry containing the postal code. The three CoS entries appear as illustrated in [Figure 5.3, “Sample Classic CoS”](#):

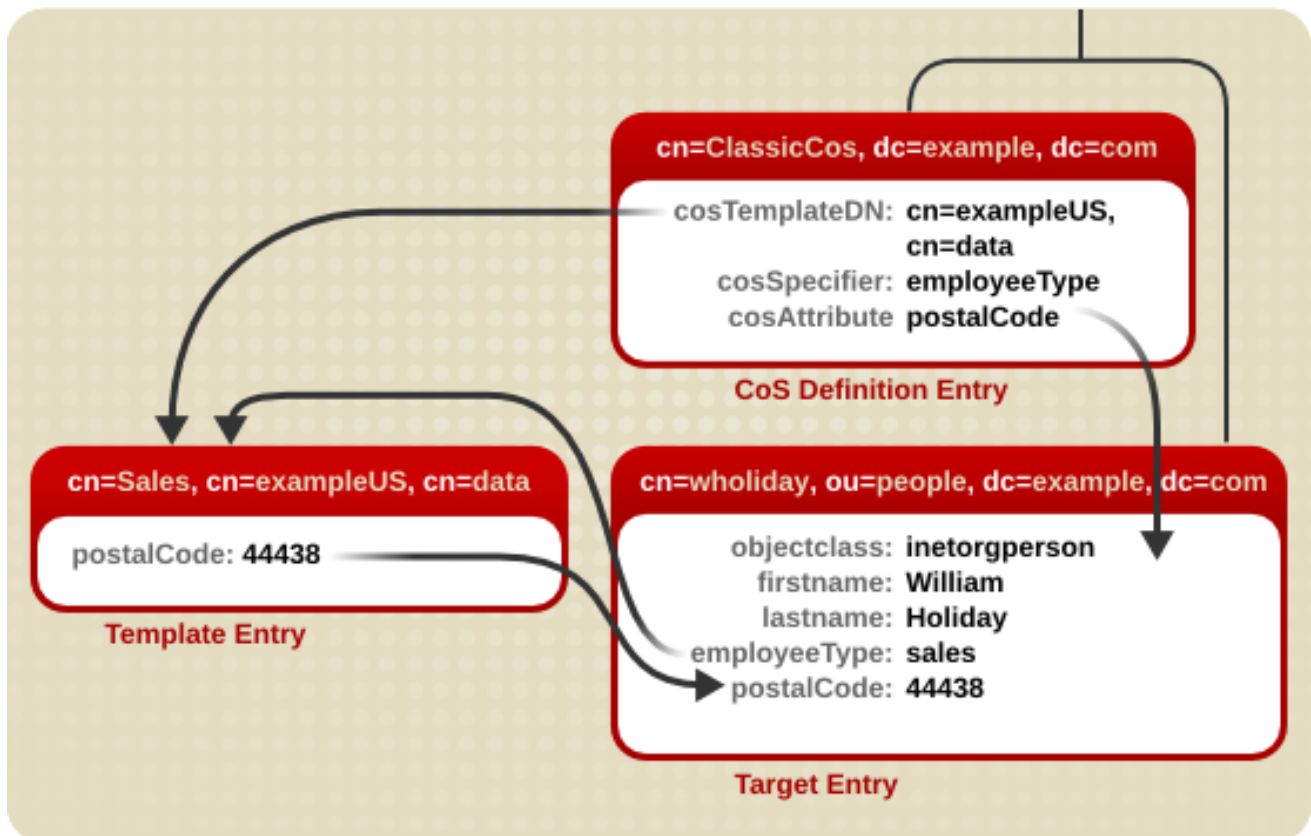


Figure 5.3. Sample Classic CoS

In this example, the CoS definition entry's **cosSpecifier** attribute specifies the **employeeType** attribute. This attribute, in combination with the template DN, identify the template entry as **cn=sales,cn=exampleUS,cn=data**. The template entry then provides the value of the **postalCode** attribute to the target entry.


5.2.6. Handling Physical Attribute Values

The **cosAttribute** attribute contains the name of another attribute which is governed by the class of service. This attribute allows an *override* qualifier (after the attribute value) which sets how the CoS handles existing attribute values on entries when it generates attribute values.

```
cosAttribute: attribute_name override
```

There are four *override* qualifiers.

Override Qualifier	Description
default	Only returns a generated value if there is no corresponding attribute value stored with the entry.
override	Always returns the value generated by the CoS, even when there is a value stored with the entry.

Override Qualifier	Description
operational	<p>Returns a generated attribute only if it is explicitly requested in the search. Operational attributes do not need to pass a schema check in order to be returned. When operational is used, it also overrides any existing attribute values.</p> <div>  <div> <p>NOTE</p> <p>An attribute can only be made operational if it is defined as operational in the schema. For example, if the CoS generates a value for the description attribute, it is not possible to use the operational qualifier because this attribute is not marked operational in the schema.</p> </div> </div>
operational-default	<p>Only returns a generated value if there is no corresponding attribute value stored with the entry and if it is explicitly requested in the search.</p>

If no qualifier is set, **default** is assumed.

For example, this pointer CoS definition entry indicates that it is associated with a template entry, **cn=exampleUS,ou=data,dc=example,dc=com**, that generates the value of the **postalCode** attribute. The **override** qualifier indicates that this value will take precedence over the value stored by the entries for the **postalCode** attribute:

```
dn: cn=pointerCoS,dc=example,dc=com
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=exampleUS,ou=data,dc=example,dc=com
cosAttribute: postalCode override
```



NOTE

If an entry contains an attribute value generated by a CoS, the value of the attribute *cannot* be manually updated if it is defined with the operational or override qualifiers.

For more information about the CoS attributes, see the *Directory Server Configuration and Command-Line Tool Reference*.

5.2.7. Handling Multi-valued Attributes with CoS

Any attribute can be generated using a class of service — including multi-valued attributes. That introduces the potential for confusion. Which CoS supplies a value? Any of them or all of them? How is the value selected from competing CoS templates? Does the generated attribute use a single value or multiple values?

There are two ways to resolve this:

- Creating a rule to merge multiple CoS-generated attributes into the target entry. This results in multiple values in the target entry.
- Setting a priority to select one CoS value out of competing CoS definitions. This generates one single value for the target entry.



NOTE

Indirect CoS do not support the ***cosPriority*** attribute.

The way that the CoS handles multiple values for a CoS attribute is defined in whether it uses a *merge-schemes* qualifier.

```
cosAttribute: attribute override merge-schemes
```



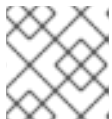
NOTE

The *merge-schemes* qualifier does not affect how the CoS handles physical attribute values or the *override* qualifier. If there are multiple competing CoS templates or definitions, then the same *merge-schemes* and *override* qualifiers have to be set on every ***cosAttribute*** for every competing CoS definition. Otherwise, one combination is chosen arbitrarily from all possible CoS definitions.

Using the *merge-schemes* qualifier tells the CoS that it will, or can, generate multiple values for the managed attribute. There are two possible scenarios for having a multi-valued CoS attribute:

- One CoS template entry contains multiple instances of the managed CoS attribute, resulting in multiple values on the target entry. For example:

```
dn: cn=server access template,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
accessTo: mail.example.com
accessTo: irc.example.com
```



NOTE

This method only works with classic CoS.

- Multiple CoS definitions may define a class of service for the same target attribute, so there are multiple template entries. For example:

```
dn: cn=mail template,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
accessTo: mail.example.com

dn: cn=chat template,dc=example,dc=com
```

```

objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
accessTo: irc.example.com

```

However, it may be that even if there are multiple CoS definitions, only one value should be generated for the attribute. If there are multiple CoS definitions, then the value is chosen arbitrarily. This is an unpredictable and unwieldy option. The way to control which CoS template to use is to set a ranking on the template — a *priority* — and the highest prioritized CoS always "wins" and provides the value.

It is fairly common for there to be multiple templates competing to provide a value. For example, there can be a multi-valued ***cosSpecifier*** attribute in the CoS definition entry. The template priority is set using the ***cosPriority*** attribute. This attribute represents the global priority of a particular template. A priority of zero is the highest priority.

For example, a CoS template entry for generating a department number appears as follows:

```

dn: cn=data,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
departmentNumber: 71776
cosPriority: 0

```

This template entry contains the value for the ***departmentNumber*** attribute. It has a priority of zero, meaning this template takes precedence over any other conflicting templates that define a different ***departmentNumber*** value.

Templates that contain no ***cosPriority*** attribute are considered the lowest priority. Where two or more templates are considered to supply an attribute value and they have the same (or no) priority, a value is chosen arbitrarily.



NOTE

The behavior for negative ***cosPriority*** values is not defined in Directory Server; do not enter negative values.

5.2.8. Searches for CoS-Specified Attributes

CoS definitions provide values for attributes in entries. For example, a CoS can set the ***postalCode*** attribute for every entry in a subtree. Searches against those CoS-defined attributes, however, do not behave like searches against regular entries.

If the CoS-defined attribute is indexed with any kind of index (including presence), then any attribute with a value set by the CoS is not returned with a search. For example:

- The ***postalCode*** attribute for Ted Morris is defined by a CoS.
- The ***postalCode*** attribute for Barbara Jensen is set in her entry.
- The ***postalCode*** attribute is indexed.

If an ***ldapsearch*** command uses the filter (***postalCode=****), then Barbara Jensen's entry is returned, while Ted Morris's is not.

If the CoS-defined attribute is *not* indexed, then every matching entry is returned in a search, regardless of whether the attribute value is set locally or with CoS. For example:

- The **postalCode** attribute for Ted Morris is defined by a CoS.
- The **postalCode** attribute for Barbara Jensen is set in her entry.
- The **postalCode** attribute is *not* indexed.

If an **ldapsearch** command uses the filter (**postalCode=***), then both Barbara Jensen's and Ted Morris's entries are returned.

CoS allows for an *override*, an identifier given to the **cosAttribute** attribute in the CoS entry, which means that local values for an attribute can override the CoS value. If an override is set on the CoS, then an **ldapsearch** operation will return a value for an entry even if the attribute is indexed, as long as there is a local value for the entry. Other entries which possess the CoS but do not have a local value will still not be returned in the **ldapsearch** operation.

Because of the potential issues with running LDAP search requests on CoS-defined attributes, take care when deciding which attributes to generate using a CoS.

5.2.9. Access Control and CoS

The server controls access to attributes generated by a CoS in exactly the same way as regular stored attributes. However, access control rules depending upon the value of attributes generated by CoS will not work. This is the same restriction that applies to using CoS-generated attributes in search filters.

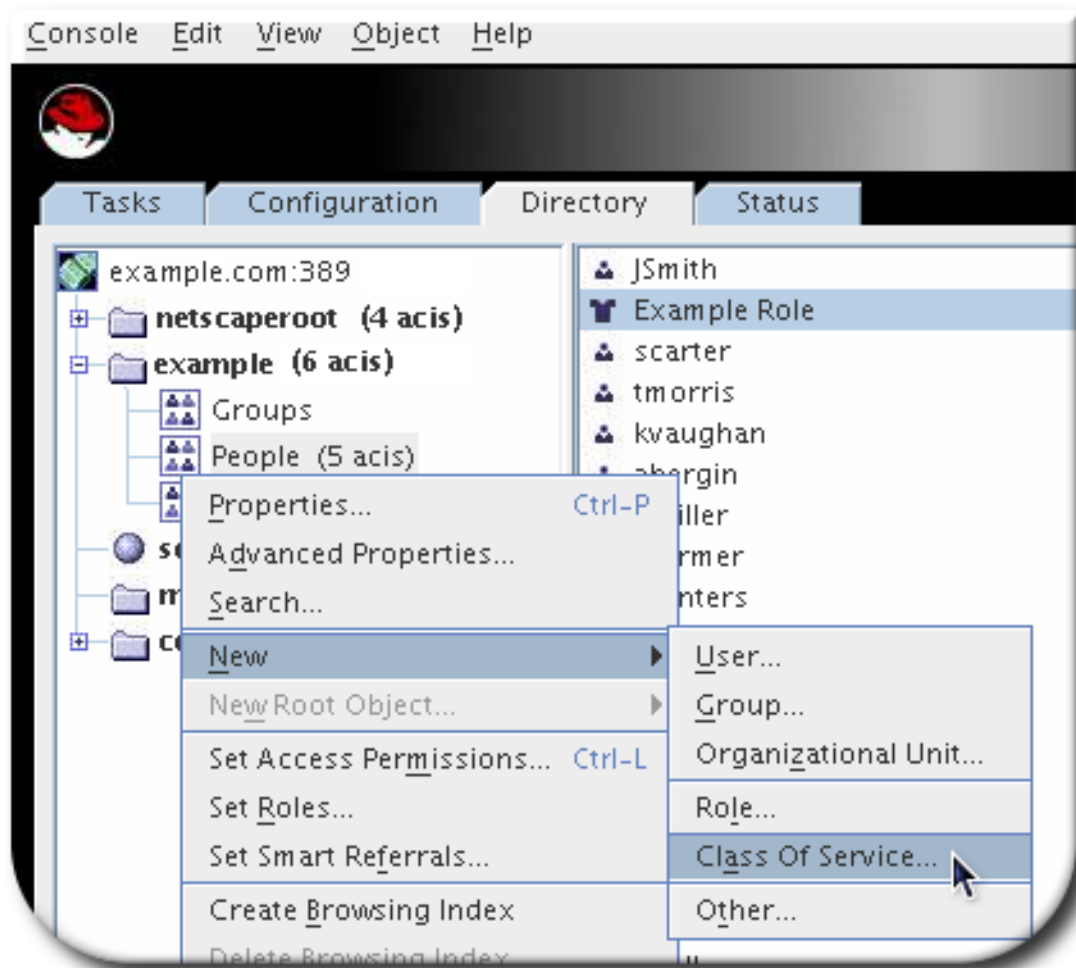
5.2.10. Managing CoS Using the Console

This section describes creating and editing CoS through the Directory Server Console:

- [Section 5.2.10.1, “Creating a New CoS”](#)
- [Section 5.2.10.2, “Creating the CoS Template Entry”](#)

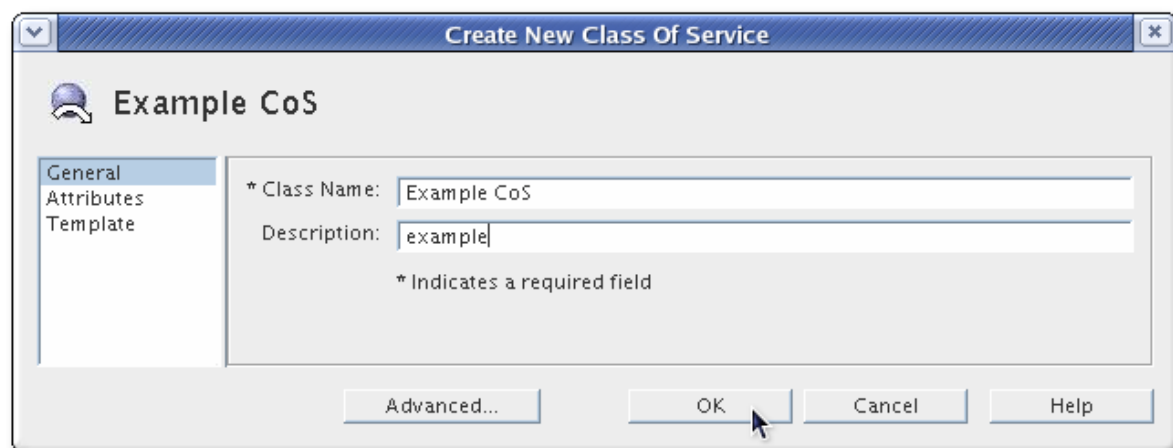
5.2.10.1. Creating a New CoS

1. In the Directory Server Console, select the **Directory** tab.
2. Browse the tree in the left navigation pane, and select the parent entry for the new class of service.
3. Go to the **Object** menu, and select **New > Class of Service**.



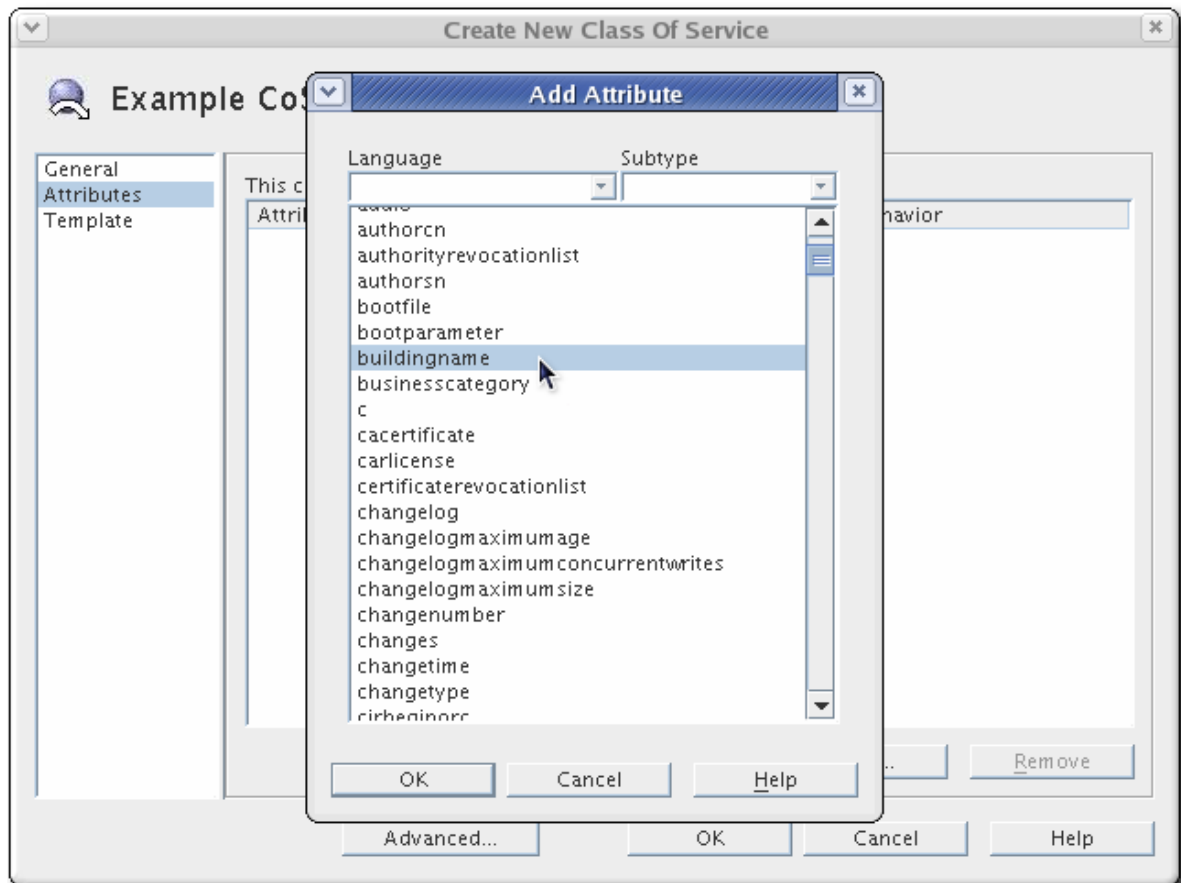
Alternatively, right-click the entry and select **New > Class of Service**.

4. Select **General** in the left pane. In the right pane, enter the name of the new class of service in the **Class Name** field. Enter a description of the class in the **Description** field.

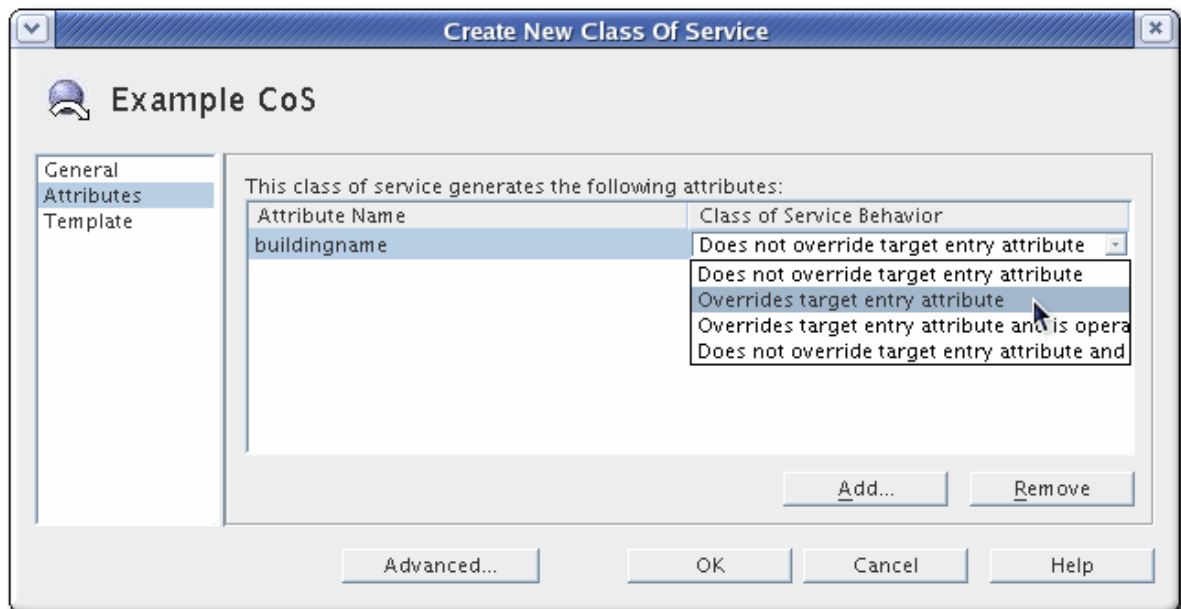


5. Click **Attributes** in the left pane. The right pane displays a list of attributes generated on the target entries.

Click **Add** to browse the list of possible attributes and add them to the list.



6. After an attribute is added to the list, a drop-down list appears in the **Class of Service Behavior** column.



- Select **Does not override target entry attribute** to tell the directory to only return a generated value if there is no corresponding attribute value stored with the entry.
- Select **Overrides target entry attribute** to make the value of the attribute generated by the CoS override the local value.

- Select **Overrides target entry attribute and is operational** to make the attribute override the local value and to make the attribute operational, so that it is not visible to client applications unless explicitly requested.
- Select **Does not override target entry attribute and is operational** to tell the directory to return a generated value only if there is no corresponding attribute value stored with the entry and to make the attribute operational (so that it is not visible to client applications unless explicitly requested).



NOTE

An attribute can only be made operational if it is also defined as operational in the schema. For example, if a CoS generates a value for the **description** attribute, you cannot select **Overrides target entry attribute and is operational** because this attribute is not marked operational in the schema.

7. Click **Template** in the left pane. In the right pane, select how the template entry is identified.

- *By its DN.* To have the template entry identified by only its DN (a pointer CoS), enter the DN of the template in the **Template DN** field. Click **Browse** to locate the DN on the local server. This will be an exact DN, such as **cn=CoS template, ou=People, dc=example, dc=com**.
- *Using the value of one of the target entry's attribute.* To have the template entry identified by the value of one of the target entry's attributes (an indirect CoS), enter the attribute name in the **Attribute Name** field. Click **Change** to select a different attribute from the list of available attributes.
- *Using both its DN and the value of one of the target entry's attributes.* To have the template entry identified by both its DN and the value of one of the target entry's attributes (a classic CoS), enter both a template DN and an attribute name. The template DN in a classic CoS is

more general than for a pointer CoS; it references the suffix or subsuffix where the template entries will be. There can be more than one template for a classic CoS.

8. Click **OK**.

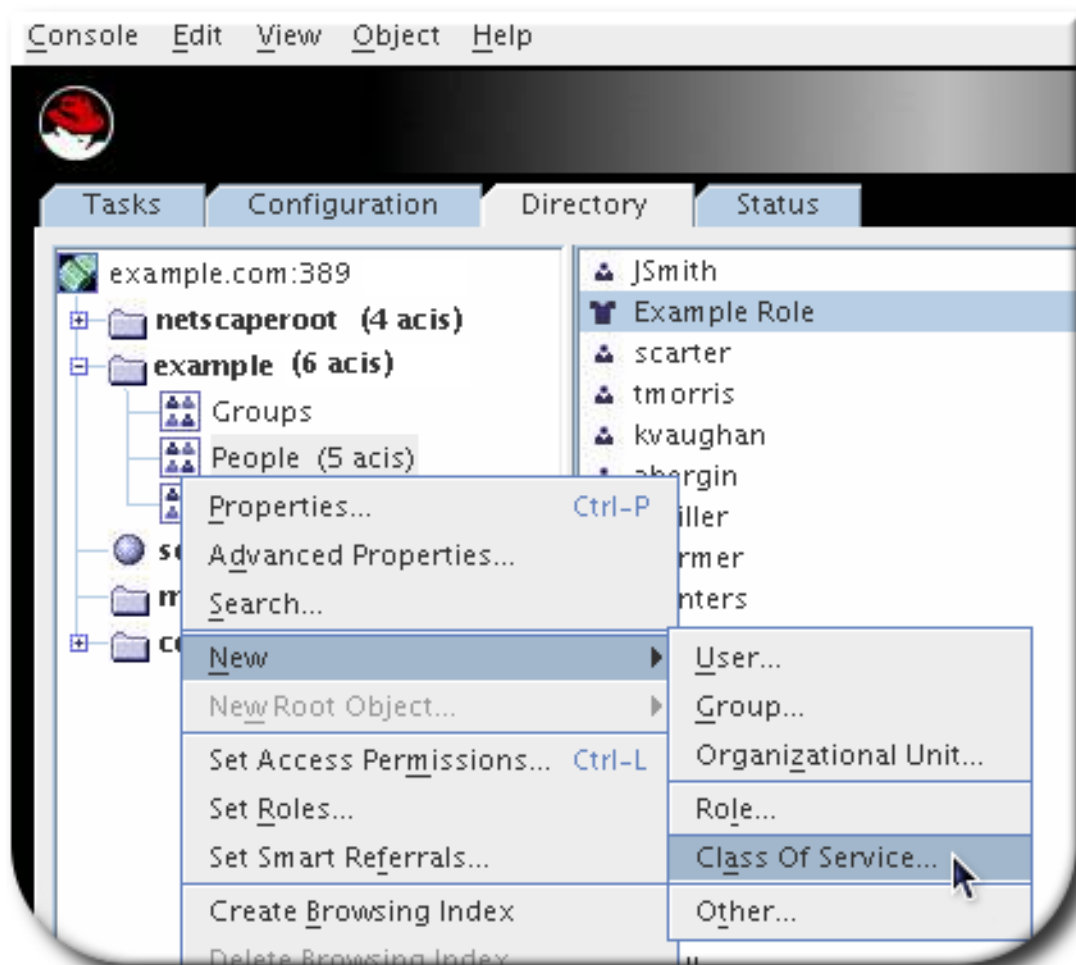
5.2.10.2. Creating the CoS Template Entry

For a pointer CoS or a classic CoS, there must be a template entry, according to the template DN set when the class of service was created. Although the template entries can be placed anywhere in the directory as long as the *cosTemplateDn* attribute reflects that DN, it is best to place the template entries under the CoS itself.

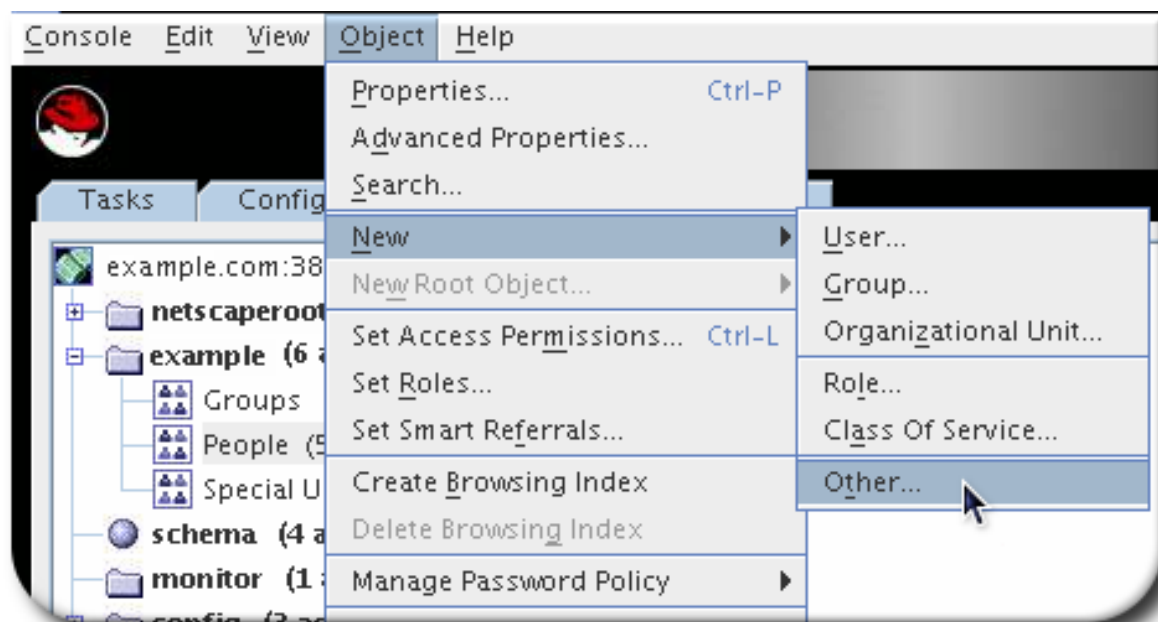
- For a pointer CoS, make sure that this entry reflects the exact DN given when the CoS was created.
- For a classic CoS, the template DN should be recursive, pointing back to the CoS entry itself as the base suffix for the template.

1. In the Directory Server Console, select the **Directory** tab.
2. Browse the tree in the left navigation pane, and select the parent entry that contains the class of service.

The CoS appears in the right pane with other entries.

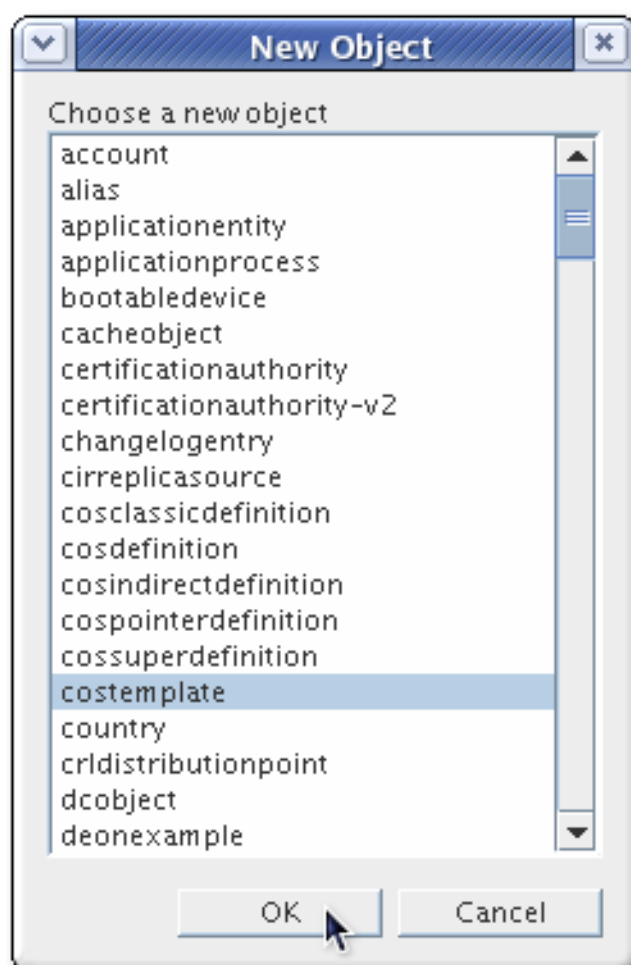


3. Right-click the CoS, and select **New > Other**.



Alternatively, select the CoS in the right pane, click **Object** in the menu at the top, and select **New > Other**.

4. Select **cosTemplate** from the list of object classes.

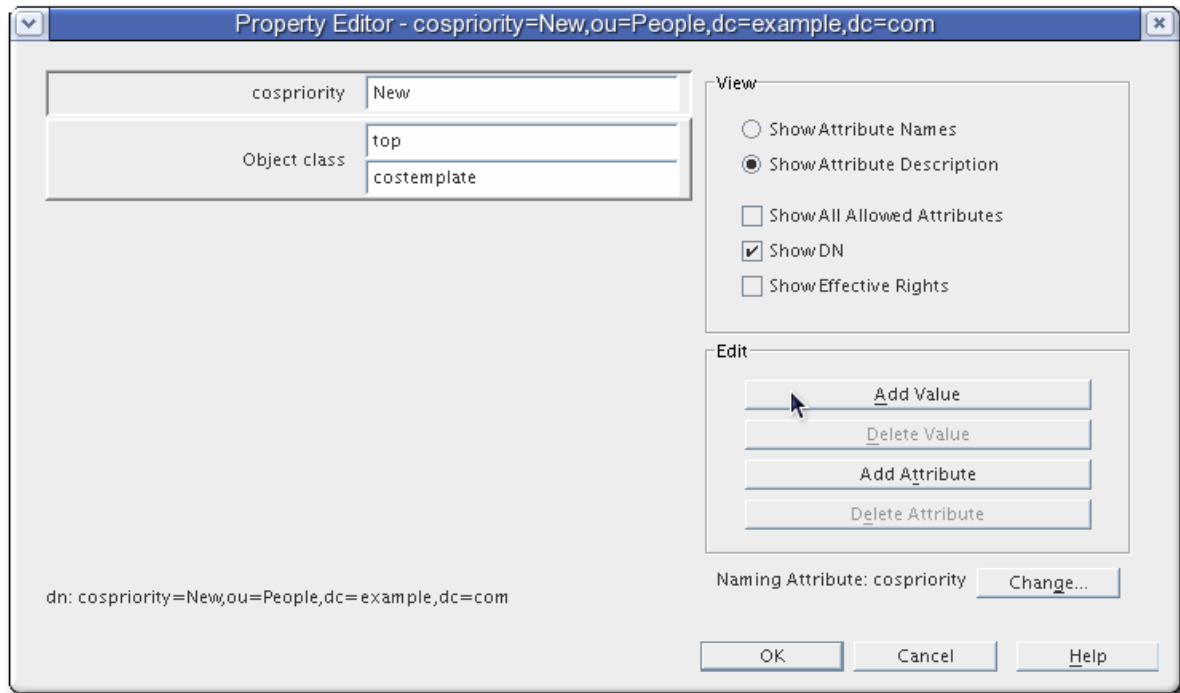




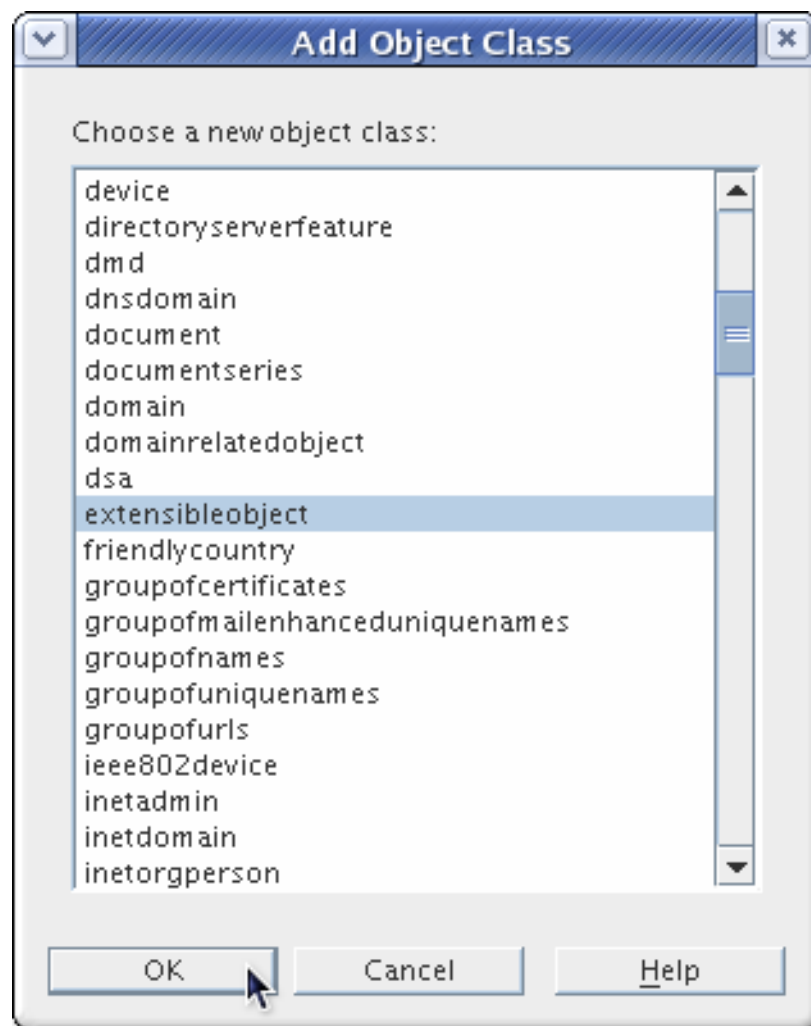
NOTE

The **LDAPsubentry** object class can be added to a new template entry. Making the CoS template entry an instance of the **LDAPsubentry** object class allows ordinary searches to be performed unhindered by the configuration entries. However, if the template entry already exists and is used for something else (for example, if it is a user entry), the **LDAPsubentry** object class does not need to be added to the template entry.

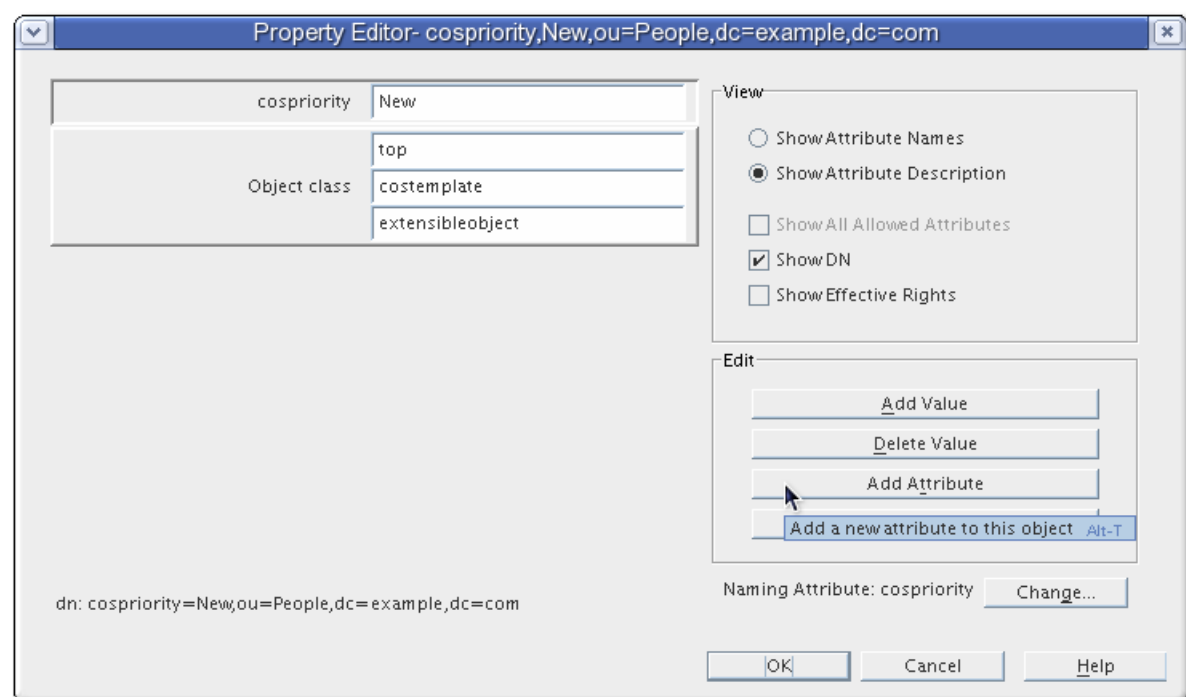
5. Select the object classes attribute, and click **Add Value**.



6. Add the **extensibleObject** object class. This makes it possible to add any attribute available in the directory.

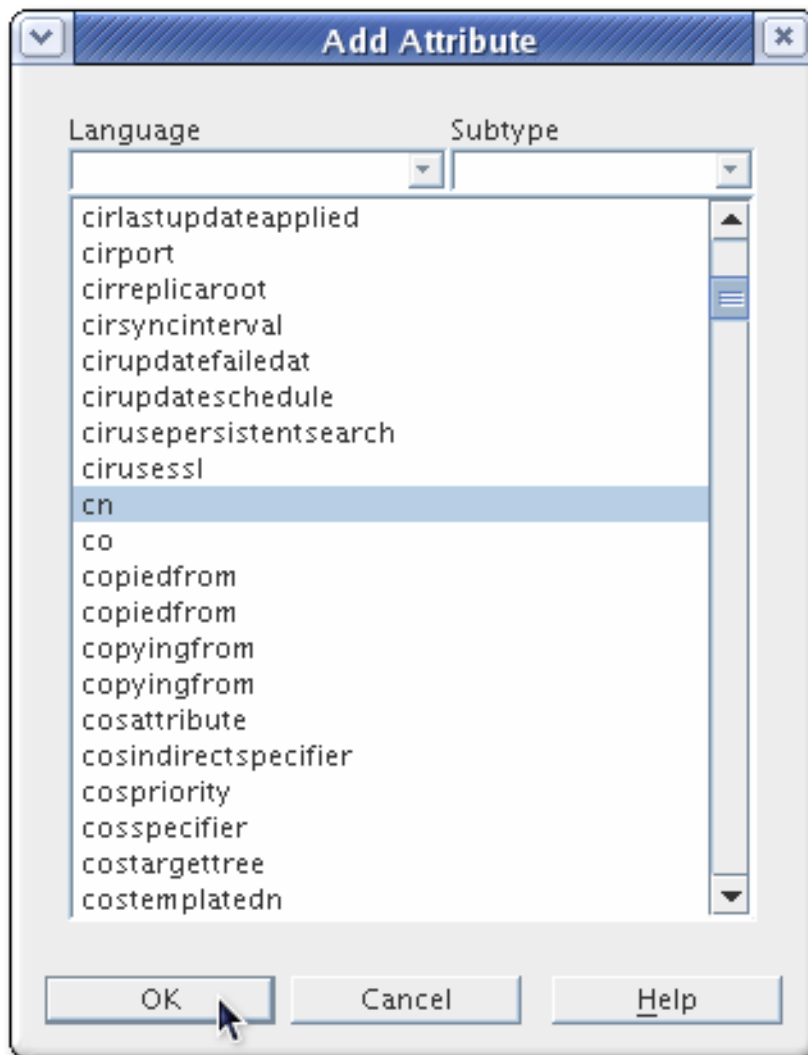


7. Click the **Add Attribute** button.

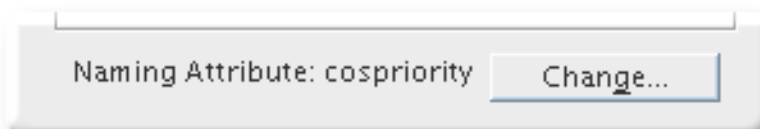


8. Add the **cn** attribute, and give it a value that corresponds to the attribute value in the target entry. For example, if the **manager** attribute is used to set the value for a classic CoS, give the **cn** a

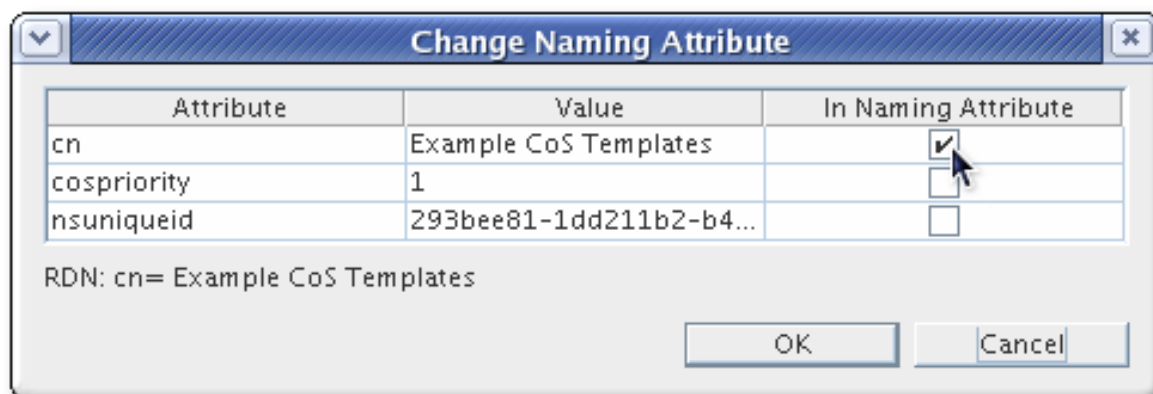
value of a manager's DN, such as **uid=bparker,ou=people,dc=example,dc=com**. Alternatively, set it to a role, such as **cn=QA Role,dc=example,dc=com** or a regular attribute value. For example, if the **employeeType** attribute is selected, it can be **full time** or **temporary**.



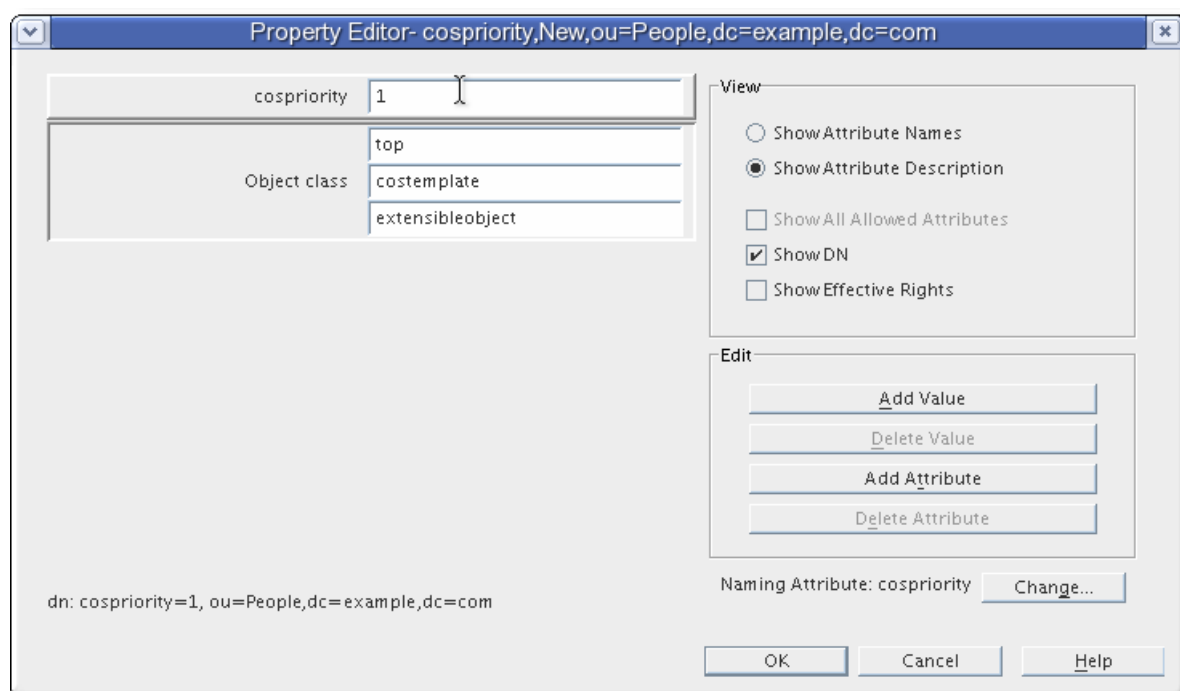
9. Click the **Change** button in the lower right corner to change the naming attribute.



10. Use the **cn** of the entry as the naming attribute instead of **cospriority**.



11. Click the **Add Attribute** button, and add the attributes listed in the CoS. The values used here will be used throughout the directory in the targeted entries.
12. Set the **cospriority**. There may be more than one CoS that applies to a given attribute in an entry; the **cospriority** attribute ranks the importance of that particular CoS. The higher **cospriority** will take precedence in a conflict. The highest priority is 0.



Templates that contain no **cosPriority** attribute are considered the lowest priority. In the case where two or more templates could supply an attribute value and they have the same (or no) priority, a value is chosen arbitrarily.



NOTE

The behavior for negative **cosPriority** values is not defined in Directory Server; do not enter negative values.



NOTE

The **cosPriority** attribute is not supported by indirect CoS.

The CoS is visible in the left navigation pane once there are entries beneath it. For classic CoS, there can be multiple entries, according to the different potential values of the attribute specifier.

To edit the description or attributes generated on the target entry of an existing CoS, simply double-click the CoS entry listed in the **Directory** tab, and make the appropriate changes in the editor window.

5.2.11. Managing CoS from the Command Line

Because all configuration information and template data is stored as entries in the directory, standard LDAP tools can be used for CoS configuration and management.

- [Section 5.2.11.1, “Creating the CoS Definition Entry from the Command Line”](#)
- [Section 5.2.11.2, “Creating the CoS Template Entry from the Command Line”](#)
- [Section 5.2.11.3, “Example of a Pointer CoS”](#)
- [Section 5.2.11.4, “Example of an Indirect CoS”](#)
- [Section 5.2.11.5, “Example of a Classic CoS”](#)
- [Section 5.2.11.6, “Searching for CoS Entries”](#)

5.2.11.1. Creating the CoS Definition Entry from the Command Line

Each type of CoS requires a particular object class to be specified in the definition entry. All CoS definition object classes inherit from the **LDAPSubentry** object class and the **cosSuperDefinition** object class.

A pointer CoS uses the **cosPointerDefinition** object class. This object class identifies the template entry using an entry DN value specified in the **cosTemplateDn** attribute, as shown in [Example 5.1, “An Example Pointer CoS Entry”](#).

Example 5.1. An Example Pointer CoS Entry

```
dn: cn=pointerCoS,dc=example,dc=com
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: DN_string
cosAttribute: list_of_attributes_qualifier
cn: pointerCoS
```

An indirect CoS uses the **cosIndirectDefinition** object class. This type of CoS identifies the template entry based on the value of one of the target entry's attributes, as specified in the **cosIndirectSpecifier** attribute. This is illustrated in [Example 5.2, “An Example Indirect CoS Entry”](#).

Example 5.2. An Example Indirect CoS Entry

```
dn: cn=indirectCoS,dc=example,dc=com
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosIndirectDefinition
```

```

cosIndirectSpecifier:attribute_name
cosAttribute:list_of_attributes qualifier
cn: indirectCoS

```

A classic CoS uses the **cosClassicDefinition** object class. This identifies the template entry using both the template entry's DN (set in the **cosTemplateDn** attribute) and the value of one of the target entry's attributes (set in the **cosSpecifier** attribute). This is illustrated in [Example 5.3, “An Example Classic CoS Entry”](#).

Example 5.3. An Example Classic CoS Entry

```

dn: cn=classicCoS,dc=example,dc=com
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosClassicDefinition
cosTemplateDn:DN_string
cosSpecifier:attribute_name
cosAttribute:list_of_attributes qualifier
cn: classicCoS

```

For a class of service, the object class defines the type of CoS, and the supporting attributes identify which directory entries are affected by defining the CoS template. Every CoS has one additional attribute which can be defined for it: **cosAttribute**. The purpose of a CoS is to supply attribute values across multiple entries; the **cosAttribute** attribute defines which attribute the CoS generates values for.

5.2.11.2. Creating the CoS Template Entry from the Command Line

Each template entry is an instance of the **cosTemplate** object class.



NOTE

Consider adding the **LDAPsubentry** object class to a new template entry. Making the CoS template entry an instance of the **LDAPsubentry** object classes allows ordinary searches to be performed unhindered by the configuration entries. However, if the template entry already exists and is used for something else, such as a user entry, the **LDAPsubentry** object class does not need to be added to the template entry.

The CoS template entry also contains the attribute generated by the CoS (as specified in the **cosAttribute** attribute of the CoS definition entry) and the value for that attribute.

For example, a CoS template entry that provides a value for the **postalCode** attribute follows:

```

dn:cn=exampleUS,ou=data,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
postalCode: 44438

```

The following sections provide examples of template entries along with examples of each type of CoS definition entry.

- [Section 5.2.11.3, “Example of a Pointer CoS”](#)
- [Section 5.2.11.4, “Example of an Indirect CoS”](#)
- [Section 5.2.11.5, “Example of a Classic CoS”](#)

5.2.11.3. Example of a Pointer CoS

Example Corporation's administrator is creating a pointer CoS that shares a common postal code with all entries in the **dc=example,dc=com** tree.

1. Add a new pointer CoS definition entry to the **dc=example,dc=com** suffix using **ldapmodify**:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

2. Next, add the pointer CoS definition to the **dc=example,dc=com** root suffix.

```
dn: cn=pointerCoS,dc=example,dc=com
changetype: add
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=exampleUS,ou=data,dc=example,dc=com
cosAttribute: postalCode
```

3. Create the template entry.

```
dn: cn=exampleUS,ou=data,dc=example,dc=com
changetype: add
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
postalCode: 44438
```

The CoS template entry (**cn=exampleUS,ou=data,dc=example,dc=com**) supplies the value stored in its **postalCode** attribute to any entries located under the **dc=example,dc=com** suffix. These entries are the target entries.

5.2.11.4. Example of an Indirect CoS

This indirect CoS uses the **manager** attribute of the target entry to identify the CoS template entry, which varies depending on the different values of the attribute.

1. Add a new indirect CoS definition entry to the **dc=example,dc=com** suffix:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

```
dn: cn=indirectCoS,dc=example,dc=com
changetype: add
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosIndirectDefinition
```

```
cosIndirectSpecifier: manager
cosAttribute: departmentNumber
```

If the directory or modify the manager entries already contain the **departmentNumber** attribute, then no other attribute needs to be added to the manager entries. The definition entry looks in the target suffix (the entries under **dc=example,dc=com**) for entries containing the **manager** attribute because this attribute is specified in the **cosIndirectSpecifier** attribute of the definition entry). It then checks the **departmentNumber** value in the manager entry that is listed. The value of the **departmentNumber** attribute will automatically be relayed to all of the manager's subordinates that have the **manager** attribute. The value of **departmentNumber** will vary depending on the department number listed in the different manager's entries.

5.2.11.5. Example of a Classic CoS

The Example Corporation administrator is creating a classic CoS that automatically generates postal codes using a combination of the template DN and the attribute specified in the **cosSpecifier** attribute.

1. Add a new classic CoS definition entry to the **dc=example,dc=com** suffix.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=classicCoS,dc=example,dc=com
changetype: add
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosClassicDefinition
cosTemplateDn: cn=classicCoS,dc=example,dc=com
cosSpecifier: businessCategory
cosAttribute: postalCode override
```

2. Create the template entries for the sales and marketing departments. Add the CoS attributes to the template entry. The **cn** of the template sets the value of the **businessCategory** attribute in the target entry, and then the attributes are added or overwritten according to the value in the template:

```
dn: cn=sales,cn=classicCoS,dc=example,dc=com
changetype: add
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
postalCode: 44438

dn: cn=marketing,cn=classicCoS,dc=example,dc=com
changetype: add
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
postalCode: 99111
```

The classic CoS definition entry applies to all entries under the **dc=example,dc=com** suffix. Depending upon the combination of the **businessCategory** attribute found in the entry and the **cosTemplateDn**, it can arrive at one of two templates. One, the sales template, provides a postal code specific to

employees in the sales department. The marketing template provides a postal code specific to employees in the marketing department.

5.2.11.6. Searching for CoS Entries

CoS definition entries are *operational* entries and are not returned by default with regular searches. This means that if a CoS is defined under **ou=People,dc=example,dc=com**, for example, the following **ldapsearch** command will not return them:

```
ldapsearch -x -s sub -b ou=People,dc=example,dc=com "(objectclass=*)"
```

To return the CoS definition entries, add the **ldapSubEntry** object class to the CoS definition entries. For example:

```
dn: cn=pointerCoS,ou=People,dc=example,dc=com
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
objectclass: ldapSubEntry
cosTemplateDn: cn=exampleUS,ou=data,dc=example,dc=com
cosAttribute: postalCode override
```

Then use a special search filter, **(objectclass=ldapSubEntry)**, with the search. This filter can be added to any other search filter using OR (|):

```
ldapsearch -x -s sub -b ou=People,dc=example,dc=com "(|(objectclass=*)
(objectclass=ldapSubEntry))"
```

This search returns all regular entries in addition to CoS definition entries in the **ou=People,dc=example,dc=com** subtree.



NOTE

The Console automatically shows CoS entries.

5.2.12. Creating Role-Based Attributes

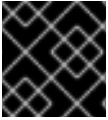
Classic CoS schemes generate attribute values for an entry based on the role possessed by the entry. For example, role-based attributes can be used to set the server look-through limit on an entry-by-entry basis.

To create a role-based attribute, use the **nsRole** attribute as the **cosSpecifier** in the CoS definition entry of a classic CoS. Because the **nsRole** attribute can be multi-valued, CoS schemes can be defined that have more than one possible template entry. To resolve the ambiguity of which template entry to use, include the **cosPriority** attribute in the CoS template entry.

For example, this CoS allows members of the manager role to exceed the standard mailbox quota. The manager role entry is:

```
dn: cn=ManagerRole,ou=people,dc=example,dc=com
objectclass: top
objectclass: nsRoleDefinition
objectclass: nsComplexRoleDefinition
```

```
objectclass: nsFilteredRoleDefinition
cn: ManagerRole
nsRoleFilter: ou=managers
Description: filtered role for managers
```



IMPORTANT

The ***nsRoleFilter*** attribute cannot accept virtual attribute values.

The classic CoS definition entry looks like:

```
dn: cn=managerCOS,dc=example,dc=com
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosClassicDefinition
cosTemplateDn: cn=managerCOS,dc=example,dc=com
cosSpecifier: nsRole
cosAttribute: mailboxquota override
```

The ***cosTemplateDn*** attribute provides a value that, in combination with the attribute specified in the ***cosSpecifier*** attribute (in the example, the ***nsRole*** attribute of the target entry), identifies the CoS template entry. The CoS template entry provides the value for the ***mailboxquota*** attribute. An additional qualifier of ***override*** tells the CoS to override any existing ***mailboxquota*** attributes values in the target entry.

The corresponding CoS template entry looks as follows:

```
dn:cn=cn=ManagerRole\,ou=people\,dc=example\,dc=com,cn=managerCOS,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
mailboxquota: 1000000
```

The template provides the value for the ***mailboxquota*** attribute, **1000000**.



NOTE

The role entry and the CoS definition and template entries should be located at the same level in the directory tree.

5.3. LINKING ATTRIBUTES TO MANAGE ATTRIBUTE VALUES

A class of service dynamically supplies attribute values for entries which all have attributes with the *same value*, like building addresses, postal codes, or main office numbers. These are shared attribute values, which are updated in a single template entry.

Frequently, though, there are relationships between entries where there needs to be a way to express linkage between them, but the values (and possibly even the attributes) that express that relationship are different. Red Hat Directory Server provides a way to link specified attributes together, so that when one attribute in one entry is altered, a corresponding attribute on a related entry is automatically updated.

(The link and managed attributes both have DN values. The value of the link attribute contains the DN of the entry for the plug-in to update; the managed attribute in the second entry has a DN value which points back to the original link entry.)

5.3.1. About Linking Attributes

The Linked Attributes Plug-in, allows multiple instances of the plug-in. Each instance configures one attribute which is manually maintained by the administrator (**linkType**) and one attribute which is automatically maintained by the plug-in (**managedType**).

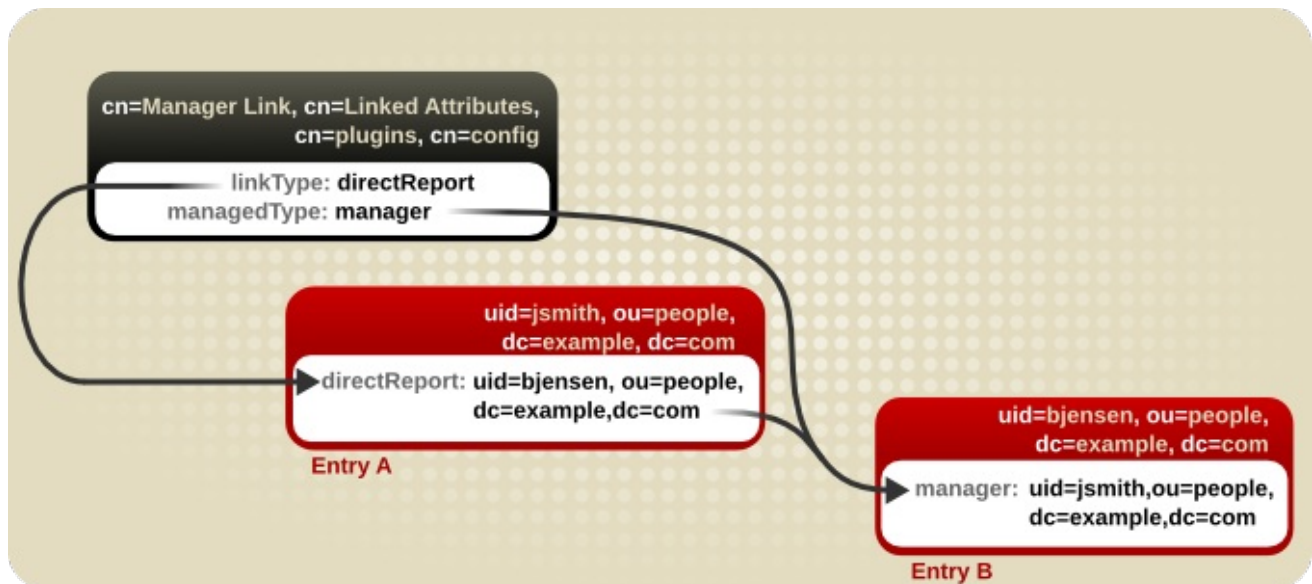


Figure 5.4. Basic Linked Attribute Configuration



NOTE

To preserve data consistency, only the plug-in process should maintain the managed attribute. Consider creating an ACI that will restrict all write access to any managed attribute. See [Section 13.5.2, “Creating a New ACI”](#) for information on setting ACIs.

A Linked Attribute Plug-in instance can be restricted to a single subtree within the directory. This can allow more flexible customization of attribute combinations and affected entries. If no scope is set, then the plug-in operates in the entire directory.

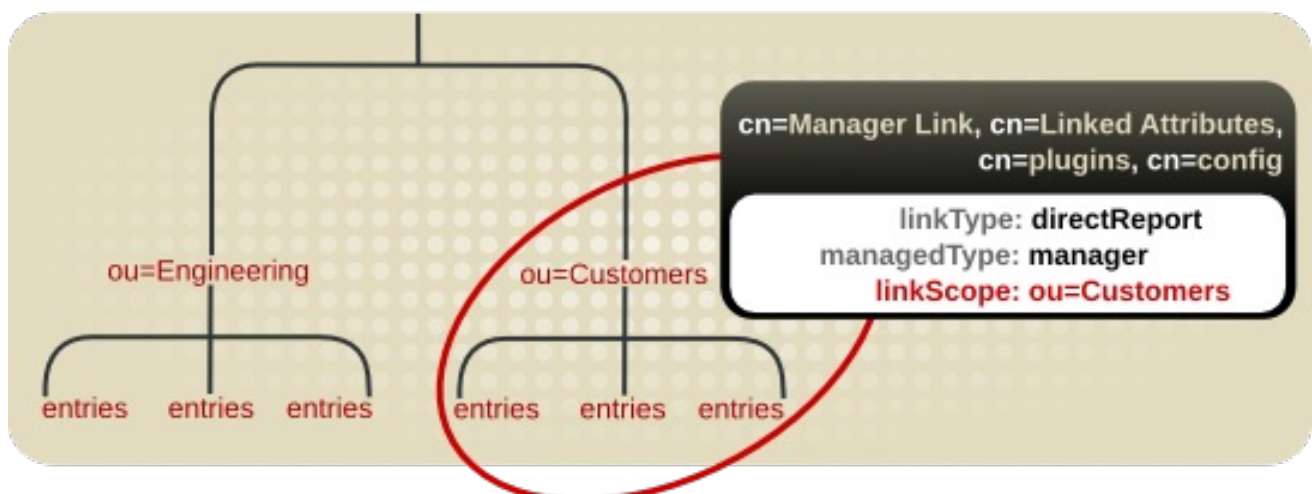


Figure 5.5. Restricting the Linked Attribute Plug-in to a Specific Subtree

When configuring the Linked Attribute Plug-in instance, certain configurations are required:

- Both the managed attribute and linked attribute must require the Distinguished Name syntax in their attribute definitions. The linked attributes are essentially managed cross-references, and the way that the plug-in handles these cross-references is by pulling the DN of the entry from the attribute value.

For information on planning custom schema elements, see [Chapter 8, *Managing the Directory Schema*](#).

- Each Linked Attribute Plug-in instance must be local and any *managed* attributes must be blocked from replication using fractional replication.

Any changes that are made on one supplier will automatically trigger the plug-in to manage the values on the corresponding directory entries, so the data stay consistent across servers. However, the managed attributes must be maintained by the plug-in instance for the data to be consistent between the linked entries. This means that managed attribute values should be maintained solely by the plug-in processes, not the replication process, even in a multi-master replication environment.

For information on using fractional replication, see [Section 11.1.7, “Replicating a Subset of Attributes with Fractional Replication”](#).

5.3.2. Looking at the Linking Attributes Plug-in Syntax

The default Linked Attributes Plug-in entry is a container entry for each plug-in instance, similar to the password syntax plug-ins or the DNA Plug-in in the next section. Each entry beneath this container entry defines a different link-managed attribute pair.

To create a new linking attribute pair, then, create a new plug-in instance beneath the container entry. A basic linking attribute plug-in instance required defining two things:

- The attribute that is managed manually by administrators, in the ***linkType*** attribute
- The attribute that is created dynamically by the plug-in, in the ***managedType*** attribute
- Optionally, a scope that restricts the plug-in to a specific part of the directory tree, in the ***linkScope*** attribute

Example 5.4. Example Linked Attributes Plug-in Instance Entry

```
dn: cn=Manager Link,cn=Linked Attributes,cn=plugins,cn=config
objectClass: top
objectClass: extensibleObject
cn: Manager Link1
cn: Manager Link
linkType: directReport
managedType: manager
linkScope: ou=people,dc=example,dc=com
```

All of the attributes available for an instance of the Linked Attributes Plug-in instance are listed in [Table 5.2, “Linked Attributes Plug-in Instance Attributes”](#).

Table 5.2. Linked Attributes Plug-in Instance Attributes

Plug-in Attribute	Description
cn	Gives a unique name for the plug-in instance.
linkScope	Contains the DN of a suffix to which to restrict the function of the plug-in instance.
linkType	Gives the attribute which is maintained by an administrator. This attribute is manually maintained and is used as the reference for the plug-in. This attribute must have a DN value format. When the attribute is added, modified, or deleted, then its value contains the DN of the target entry for the plug-in to update.
managedType	Gives the attribute which is maintained by the plug-in. This attribute is created and updated on target entries. This attribute must have a DN value format. When the attribute is added to the entry, its value will point back as a cross-reference to the managed entry.

5.3.3. Configuring Attribute Links



NOTE

The Linked Attribute Plug-in instance can be created in the Directory Server Console, but only through the Advanced Property Editor for the directory entry, by manually adding all of the required attributes, the same as creating the entry manually through the command line.

1. If it is not already enabled, enable the Linked Attributes Plug-in, as described in [Section 1.8.1, “Enabling Plug-ins in the Directory Server Console”](#). For example:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x

dn: cn=Linked Attributes,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
```

2. Create the plug-in instance. Both the **managedType** and **linkType** attributes are required. The plug-in syntax is covered in [Section 5.3.2, “Looking at the Linking Attributes Plug-in Syntax”](#). For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=Manager Link,cn=Linked Attributes,cn=plugins,cn=config
changetype: add
objectClass: top
```

```
objectClass: extensibleObject
cn: Manager Link
linkType: directReport
managedType: manager
```

3. Restart the server to apply the new plug-in instance.

```
service dirsrv restart
```

5.3.4. Cleaning up Attribute Links

The managed-linked attributes can get out of sync. For instance, a linked attribute could be imported or replicated over to a server, but the corresponding managed attribute was not because the link attribute was not properly configured. The managed-linked attribute pairs can be fixed by running a script (**fixup-linkedattrs.pl**) or by launching a fix-up task.

The fixup task removes any managed attributes (attributes managed by the plug-in) that do not have a corresponding link attribute (attributes managed by the administrator) on the referenced entry. Conversely, the task adds any missing managed attributes if the link attribute exists in an entry.

5.3.4.1. Regenerating Linked Attributes Using **fixup-linkedattrs.pl**

The **fixup-linkedattrs.pl** script launches a special task to regenerate all of the managed-link attribute pairs on directory entries. One or the other may be lost in certain situations. If the link attribute exists in an entry, the task traces the cross-referenced DN in the available attribute and creates the corresponding configured managed attribute on the referenced entry. If a managed attribute exists with no corresponding link attribute, then the managed attribute value is removed.

To repair all configured link attribute pairs for the entire scope of the plug-in, then simply run the command as the Directory Manager:

```
/usr/lib64/dirsrv/instance_name/fixup-linkedattrs.pl -D "cn=Directory
Manager" -w password
```

It is also possible to limit the fixup task to a single link-managed attribute pair, using the **-l** option to specify the target plug-in instance DN:

```
/usr/lib64/dirsrv/instance_name/fixup-linkedattrs.pl -D "cn=Directory
Manager" -w password -l "cn=Manager Link,cn=Linked
Attributes,cn=plugins,cn=config"
```

The **fixup-linkedattrs.pl** tool is described in more detail in the *Configuration and Command-Line Tool Reference*.

5.3.4.2. Regenerating Linked Attributes Using **ldapmodify**

Repairing linked attributes is one of the tasks which can be managed through a special task configuration entry. Task entries occur under the **cn=tasks** configuration entry in the **dse.ldif** file, so it is also possible to initiate a task by adding the entry using **ldapmodify**. When the task is complete, the entry is removed from the directory.

This task is the same one created automatically by the **fixup-linkedattrs.pl** script when it is run.

To initiate a linked attributes fixup task, add an entry under the **cn=fixup linked attributes,cn=tasks,cn=config** entry. The only required attribute is the **cn** for the specific task, though it also allows the **ttl** attribute to set a timeout period.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=example,cn=fixup linked attributes,cn=tasks,cn=config
changetype: add
cn:example
ttl: 5
```

Once the task is completed, the entry is deleted from the **dse.ldif** configuration, so it is possible to reuse the same task entry continually.

The **cn=fixup linked attributes** task configuration is described in more detail in the [Configuration and Command-Line Tool Reference](#).

5.4. ASSIGNING AND MANAGING UNIQUE NUMERIC ATTRIBUTE VALUES

Some entry attributes require having a unique number, such as **uidNumber** and **gidNumber**. The Directory Server can automatically generate and supply unique numbers for specified attributes using the Distributed Numeric Assignment (DNA) Plug-in.



NOTE

Attribute uniqueness is not necessarily preserved with the DNA Plug-in. The plug-in only assigns non-overlapping ranges, but it does allow manually-assigned numbers for its managed attributes, and it does not verify or require that the manually-assigned numbers are unique.

The issue with assigning unique numbers is not with generating the numbers but in effectively avoiding replication conflicts. The DNA Plug-in assigns unique numbers across a *single* back end. For multi-master replication, when each master is running a local DNA Plug-in instance, there has to be a way to ensure that each instance is using a truly unique set of numbers. This is done by assigning different *ranges* of numbers to each server to assign.

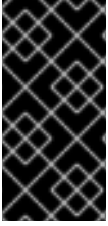
5.4.1. About Dynamic Number Assignments

The DNA Plug-in for a server assigns a range of available numbers that that instance can issue. The range definition is very simple and is set by two attributes: the server's next available number (the low end of the range) and its maximum value (the top end of the range). The initial bottom range is set when the plug-in instance is configured. After that, the bottom value is updated by the plug-in. By breaking the available numbers into separate ranges on each replica, the servers can all continually assign numbers without overlapping with each other.

5.4.1.1. Filters, Searches, and Target Entries

The server performs a sorted search, internally, to see if the next specified range is already taken, requiring the managed attribute to have an equality index with the proper ordering matching rule (as described in [Section 9.2, "Creating Standard Indexes"](#)).

The DNA Plug-in is applied, always, to a specific area of the directory tree (the *scope*) and to specific entry types within that subtree (the *filter*).



IMPORTANT

The DNA Plug-in only works on a single back end; it cannot manage number assignments for multiple databases. The DNA plug-in uses the sort control when checking whether a value has already been manually allocated outside of the DNA Plug-in. This validation, using the sort control, only works on a single back end.

5.4.1.2. Ranges and Assigning Numbers

There are several different ways that the Directory Server can handle generating attribute values:

- In the simplest case, a user entry is added to the directory with an object class which requires the unique-number attribute, but without the attribute present. Adding an entry with no value for the managed attribute triggers the DNA Plug-in to assign a value. This option only works if the DNA Plug-in has been configured to assign unique values to a single attribute.
- A similar and more manageable option is to use a *magic number*. This magic number is a template value for the managed attribute, something outside the server's range, a number or even a word, that the plug-in recognizes it needs to replace with a new assigned value. When an entry is added with the magic value and the entry is within the scope and filter of the configured DNA Plug-in, then using the magic number automatically triggers the plug-in to generate a new value. For example, this uses 0 as a magic number:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: posixAccount
uid: jsmith
cn: John Smith
uidNumber: 0
gidNumber: 0
....
```

The DNA Plug-in only generates new, unique values. If an entry is added or modified to use a specific value for an attribute controlled by the DNA Plug-in, the specified number is used; the DNA Plug-in will not overwrite it.

5.4.1.3. Multiple Attributes in the Same Range

The DNA Plug-in can assign unique numbers to a single attribute type or across multiple attribute types from a single range of unique numbers.

This provides several options for assigning unique numbers to attributes:

- A single number assigned to a single attribute type from a single range of unique numbers.
- The same unique number assigned to two attributes for a single entry.

- Two different attributes assigned two different numbers from the same range of unique numbers.

In many cases, it is sufficient to have a unique number assigned per attribute type. When assigning an **employeeID** to a new employee entry, it is important each employee entry is assigned a unique **employeeID**.

However, there are cases where it may be useful to assign unique numbers from the same range of numbers to multiple attributes. For example, when assigning a **uidNumber** and a **gidNumber** to a **posixAccount** entry, the DNA Plug-in will assign the same number to both attributes. To do this, then pass both managed attributes to the modify operation, specifying the magic value.

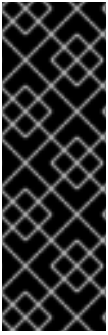
```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: modify
add: uidNumber
uidNumber: 0
-
add: gidNumber
gidNumber: 0
```

When multiple attributes are handled by the DNA Plug-in, the plug-in can assign a unique value to only one of those attributes if the object class only allows one of them. For example, the **posixGroup** object class does not allow a **uidNumber** attribute but it does allow **gidNumber**. If the DNA Plug-in manages both **uidNumber** and **gidNumber**, then when a **posixGroup** entry is created, a unique number for **gidNumber** is assigned from the same range as the **uidNumber** and **gidNumber** attributes. Using the same pool for all attributes managed by the plug-in keeps the assignment of unique numbers aligned and prevents situations where a **uidNumber** and a **gidNumber** on different entries are assigned from different ranges and result in the same *unique* number.

If multiple attributes are handled by the DNA Plug-in, then the same value will be assigned to all of the given managed attributes in an entry in a single modify operation. To assign *different* numbers from the same range, then you must perform separate modify operations. For example:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: modify
add: uidNumber
uidNumber: 0
^D

[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: modify
add: employeeId
employeeId: magic
```



IMPORTANT

When the DNA Plug-in is configured to assign unique numbers to multiple attributes, it is necessary to specify the magic value for each attribute that requires the unique number. While this is not necessary when the DNA plug-in has been configured to provide unique numbers for a single attribute, it is necessary for multiple attributes. There may be instances where an entry does not allow each type of attribute defined for the range, or, more important, an entry allow all of the attributes types defined, but only a subset of the attributes require the unique value.

Example 5.5. DNA and Unique Bank Account Numbers

Example Bank wants to use the same unique number for a customer's **primaryAccount** and **customerID** attributes. The Example Bank administrator configured the DNA Plug-in to assign unique values for both attributes from the same range.

The bank also wants to assign numbers for secondary accounts from the same range as the customer ID and primary account numbers, but these numbers cannot be the same as the primary account numbers. The Example Bank administrator configures the DNA Plug-in to also manage the **secondaryAccount** attribute, but will only add the **secondaryAccount** attribute to an entry *after* the entry is created and the **primaryAccount** and **customerID** attributes are assigned. This ensures that **primaryAccount** and **customerID** share the same unique number, and any **secondaryAccount** numbers are entirely unique but still from the same range of numbers.

5.4.2. Looking at the DNA Plug-in Syntax

The DNA Plug-in itself is a container entry, similar to the Password Storage Schemes Plug-in. Each DNA entry underneath the DNA Plug-in entry defines a new managed range for the DNA Plug-in.

To set new managed ranges for the DNA Plug-in, create entries beneath the container entry.

The most basic configuration is to set up distributed numeric assignments on a single server, meaning the ranges will not be shared or transferred between servers. A basic DNA configuration entry defines four things:

- The attribute that value is being managed, is set in the **dnaType** attribute
- The entry DN to use as the base to search for entries, set in the **dnaScope** attribute
- The search filter to use to identify entries to manage, set in the **dnaFilter** attribute
- The next available value to assign, set in the **dnaNextValue** attribute (after the entry is created, this is handled by the plug-in)

All of the attributes available for a DNA Plug-in instance are listed in [Table 5.3, "DNA Entry Attributes"](#).

To configure distributed numeric assignment on a single server for a single attribute type:

```
dn: cn=Account UIDs,cn=Distributed Numeric Assignment
Plugin,cn=plugins,cn=config
objectClass: top
objectClass: dnaPluginConfig
cn: Account UIDs
dnatype: uidNumber
```



```
dnafilter: (objectclass=posixAccount)
dnascope: ou=people,dc=example,dc=com
dnaNextValue: 1
```

If multiple suppliers are configured for distributed numeric assignments, then the entry must contain the required information to transfer ranges:

- The maximum number that the server can assign; this sets the upward bound for the range, which is logically required when multiple servers are assigning numbers. This is set in the ***dnaMaxValue*** attribute.
- The threshold where the range is low enough to trigger a range transfer, set in the ***dnaThreshold*** attribute. If this is not set, the default value is **1**.
- A timeout period so that the server does not hang waiting for a transfer, set in the ***dnaRangeRequestTimeout*** attribute. If this is not set, the default value is **10**, meaning 10 seconds.
- A configuration entry DN which is shared among all supplier servers, which stores the range information for each supplier, set in the ***dnaSharedCfgDN*** attribute.

The specific number range which could be assigned by the server is defined in the ***dnaNextRange*** attribute. This shows the next available range for transfer and is managed automatically by the plug-in, as ranges are assigned or used by the server. This range is just "on deck." It has not yet been assigned to another server and is still available for its local Directory Server to use.

```
dn: cn=Account UIDs,cn=Distributed Numeric Assignment
Plugin,cn=plugins,cn=config
objectClass: top
objectClass: dnaPluginConfig
cn: Account UIDs
dnatype: uidNumber
dnafilter: (objectclass=posixAccount)
dnascope: ou=People,dc=example,dc=com
dnanextvalue: 1
dnaMaxValue: 1300
dnasharedcfgdn: cn=Account UIDs,ou=Ranges,dc=example,dc=com
dnathreshold: 100
dnaRangeRequestTimeout: 60
dnaNextRange: 1301-2301
```

The ***dnaNextRange*** attribute should be set explicitly only if a separate, specific range has to be assigned to other servers. Any range set in the ***dnaNextRange*** attribute must be unique from the available range for the other servers to avoid duplication. If there is no request from the other servers and the server where ***dnaNextRange*** is set explicitly has reached its set ***dnaMaxValue***, the next set of values (part of the ***dnaNextRange***) is allocated from this deck.

The ***dnaNextRange*** allocation is also limited by the ***dnaThreshold*** attribute that is set in the DNA configuration. Any range allocated to another server for ***dnaNextRange*** cannot violate the threshold for the server, even if the range is available on the deck of ***dnaNextRange***.

**NOTE**

If the ***dnaNextRange*** attribute is handled internally if it is not set explicitly. When it is handled automatically, the ***dnaMaxValue*** attribute serves as upper limit for the next range.

Each supplier keeps a track of its current range in a separate configuration entry which contains information about its range and its connection settings. This entry is a child of the location in ***dnasharedcfdn***. The configuration entry is replicated to all of the other suppliers, so each supplier can check that configuration to find a server to contact for a new range. For example:

```
dn: dnaHostname=ldap1.example.com+dnaPortNum=389,cn=Account
UIDs,ou=Ranges,dc=example,dc=com
objectClass: dnaSharedConfig
objectClass: top
dnahostname: ldap1.example.com
dnaPortNum: 389
dnaSecurePortNum: 636
dnaRemainingValues: 1000
```

Table 5.3. DNA Entry Attributes

Plug-in Attribute	Description
dnaPluginConfig (object class)	The object class for instances of the DNA Plug-in.
cn	Gives a unique name for the plug-in instance.
dnaType	<p>Contains the name of the attributes for which unique numbers are assigned.</p> <p>If a prefix will be prepended to the generated value, then be sure to use an attribute which allows the syntax of the combined attribute value, such as a custom attribute which allows alphanumeric strings. Otherwise, syntax validation will enforce the defined syntax for the value, such as integer for <i>uidNumber</i> and <i>gidNumber</i>, and the DNA operations will fail with syntax violations.</p>
dnaScope	Sets the base DN to use to search for entries to which to apply the managed ranges.
dnaFilter	Gives an LDAP filter to use to specify the kinds of entries for the plug-in to manage.
dnaNextValue	Gives the next available number to assign. This is initially set manually when the entry is created; afterward, it is managed by the plug-in.

Plug-in Attribute	Description
<code>dnaMaxValue</code>	Optionally, the upper limit of the range that the server can assign. Defining the range is required when there are multiple servers assigning numbers to entries. The default value is -1 , which is the same as the highest 64-bit integer.
<code>dnaInterval</code>	Optionally, sets an interval to use to increment through numbers in a range. Essentially, this skips numbers at a predefined rate. If the interval is 3 and the first number in the range is 1, then the next number used in the range is 4, then 7, then 10, incrementing by three for every new number assignment.
<code>dnaThreshold</code>	Sets a limit on the amount of remaining available numbers before the server requests a new range.
<code>dnaSharedCfgDN</code>	Specifies the DN of a container entry that each supplier server shares. The plug-in automatically creates an entry for the individual instances underneath this entry which contains their available ranges. The plug-in can use this information to request and transfer ranges as servers consume their available range.
<code>dnaNextRange</code>	Shows the next range of numbers which are available to be transferred. This attribute can be set automatically by the plug-in according to the threshold and shared configuration information; this can also be set manually for an administrator to specifically assign an additional range of values to a server. This attribute is always limited by the <code>dnaThreshold</code> settings.
<code>dnaRangeRequestTimeout</code>	Sets a timeout period for a range request so that a server does not hang indefinitely waiting for a transfer.
<code>dnaMagicRegen</code>	Sets a word or number (outside of the assigned range) which automatically triggers the plug-in to assign a number to an attribute. This is a useful default to use for importing entries.

Plug-in Attribute	Description
<code>dnaPrefix</code>	<p>Sets a string to insert in front of whatever number is assigned. For example, if the prefix is user and the assigned number for the attribute is 1001, then the final assigned value is user1001.</p> <p><i>dnaPrefix</i> can hold any kind of string. However, some possible values for <i>dnaType</i> (such as <i>uidNumber</i> and <i>gidNumber</i>) require integer syntax for attribute values. To use a prefix string, consider using a custom attribute for <i>dnaType</i> which allows the syntax of the prefix plus the generated number assignment.</p>
<code>dnaSharedConfig</code> (object class)	The object class for shared configuration entries with host information, for servers in multi-master replication.
<code>dnaHostname</code>	Identifies the host name of a server in a shared range, as part of the DNA range configuration for that specific host in multi-master replication.
<code>dnaPortNum</code>	Gives the standard port number to use to connect to the host identified in <i>dnaHostname</i> .
<code>dnaSecurePortNum</code>	Gives the secure (SSL) port number to use to connect to the host identified in <i>dnaHostname</i> .
<code>dnaRemainingValues</code>	Contains the number of values that are remaining and available to a server to assign to entries.

5.4.3. Configuring Unique Number Assignments

The unique number distribution is configured by creating different instances of the DNA Plug-in. These DNA Plug-in instances can only be created through the command line, but they can be edited through the Directory Server Console.

5.4.3.1. Configuring Unique Number Assignments



NOTE

Any attribute which has a unique number assigned to it must have an equality index set for it. The server must perform a sorted search, internally, to see if the ***dnaNextvalue*** is already taken, which requires an equality index on an integer attribute, with the proper ordering matching rule.

Creating indexes is described in [Section 9.2, “Creating Standard Indexes”](#).

**NOTE**

Set up the DNA Plug-in on every supplier server, and be careful not to overlap the number range values.

1. Create the shared container entry in the replicated subtree. For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

```
dn: ou=Ranges,dc=example,dc=coma
changetype: add
objectclass: top
objectclass: extensibleObject
objectclass: organizationalUnit
ou: Ranges
```

```
dn: cn=Account UIDs,ou=Ranges,dc=example,dc=coma
changetype: add
objectclass: top
objectclass: extensibleObject
cn: Account UIDs
```

2. Enable the DNA Plug-in. By default, the plug-in entry (which is the container entry) is disabled.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x
```

```
dn: cn=Distributed Numeric Assignment Plugin,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
```

3. Create the new DNA Plug-in instance beneath the container entry. [Table 5.3, “DNA Entry Attributes”](#) lists the possible plug-in attributes.

**NOTE**

The plug-in attribute which sets which entry attributes have unique number assignments, ***dnaType***, is multi-valued. If multiple attributes are set in the same plug-in instance, then their number assignments are taken from the same range. To use different ranges, configure different plug-in instances.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

```
dn: cn=Account UIDs,cn=Distributed Numeric Assignment
Plugin,cn=plugins,cn=config
changetype: add
objectClass: top
objectClass: dnaPluginConfig
cn: Account UIDs
```

```

dnatype: uidNumber
dnafilter: (objectclass=posixAccount)
dnascope: ou=People,dc=example,dc=com
dnanextvalue: 1
dnaMaxValue: 1300
dnasharedcfgdn: cn=Account UIDs,ou=Ranges,dc=example,dc=com
dnathreshold: 100
dnaRangeRequestTimeout: 60
dnaMagicRegen: magic

```

4. For servers in multi-master replication, create a configuration entry for the host, which specifies its connection information and range.

The DN of the entry is a combination of the host name and the port number (**`dnahostname+dnaportnum`**).

```

ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

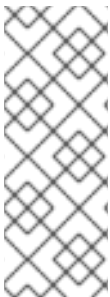
dn: dnahostname=ldap1.example.com+dnaportnum=389,cn=Account
UIDs,ou=Ranges,dc=example,dc=com
objectClass: dnaSharedConfig
objectClass: top
dnahostname: ldap1.example.com
dnaportnum: 389
dnaSecurePortNum: 636
dnaRemainingValues: 1000

```

5. Restart the server to load the new plug-in instance.

```
service dirsrv restart instance_name
```

5.4.3.2. Editing the DNA Plug-in in the Console



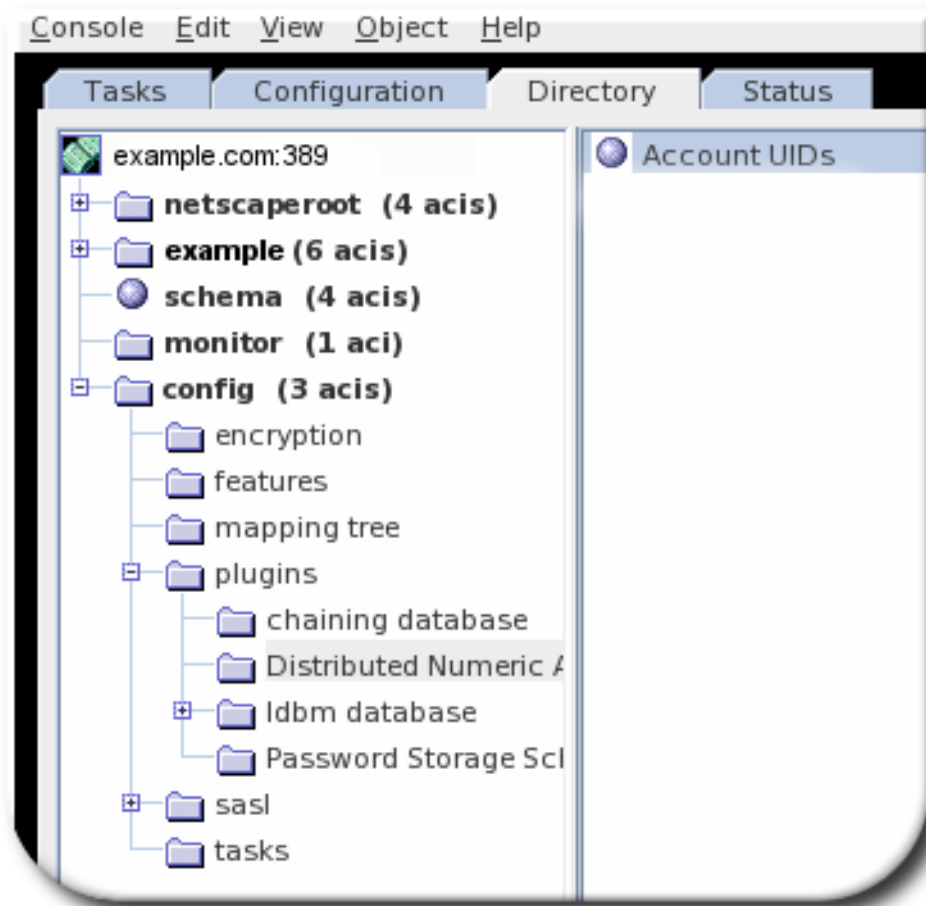
NOTE

Any attribute which has a unique number assigned to it must have an equality index set for it. The server must perform a sorted search, internally, to see if the ***dnanextvalue*** is already taken, which requires an equality index on an integer attribute, with the proper ordering matching rule.

Creating indexes is described in [Section 9.2, “Creating Standard Indexes”](#).

The Directory Server Console can be used to edit the DNA Plug-in instances.

1. Click the **Directory** tab.
2. Open the **config** folder, and then expand the **plugins** folder.
3. Click the **Distributed Numeric Assignment** plug-in folder. All of the DNA Plug-in instances are listed in the main window.



4. Highlight the DNA instance entry, and right-click on the **Advanced** link to open the property editor.
5. Edit the DNA-related attributes.

Property Editor - cn=Account UIDs, cn=Distributed Numeric Assignment Plugin

createtimestamp	20090204152503Z
creatorsname	cn=directory manager
dnafilter	(objectclass=posixAccount)
dnamaxvalue	1300
dnanextvalue	1
dnascope	ou=People, dc=example, dc=com
dnatype	uidNumber
Full name	Account UIDs
hassubordinates	FALSE
modifiersname	cn=directory manager
modifytimestamp	20090204152503Z
nsuniqueid	f4c9c981-f2cf11dd-97a5a226-356308f1
numsubordinates	0
Object class	top

dn: cn=Account UIDs, cn=Distributed Numeric Assignment Plugin, cn=plugins, cn=c...

5.4.4. Distributed Number Assignment Plug-in Performance Notes

There can be thread locking issues as DNA configuration is changed dynamically, so that new operations which access the DNS configuration (such as a DNA task or additional changes to the DNA configuration) will access the old configuration because the thread with the new configuration has not yet been released. This can cause operations to use old configuration or simply cause operations to hang.

To avoid this, preserve an interval between dynamic DNA configuration changes of 35 seconds. This means have a sleep or delay between both DNA configuration changes and any directory entry changes which would trigger a DNA plug-in operation.

CHAPTER 6. ORGANIZING AND GROUPING ENTRIES

Entries contained within the directory can be grouped in different ways to simplify the management of user accounts. Red Hat Directory Server supports a variety of methods for grouping entries and sharing attributes between entries. To take full advantage of the features offered by roles and class of service, determine the directory topology when planning the directory deployment.

6.1. USING GROUPS

Groups are a mechanism for associating entries for ease of administration. Groups do not have the flexibility or utility of roles. However, there are certain clients and applications where groups are useful. Groups also offer compatibility with older LDAP clients and directory services.



NOTE

Managing groups is significantly easier by using the *memberOf* attribute to identify in user entries to what groups a user belongs. The *memberOf* attribute is maintained by the Directory Server and updated automatically on entries as group membership changes. See [Section 6.1.4, “Listing Group Membership in User Entries”](#) for information on using the *memberOf* attribute.

6.1.1. Creating Static Groups in the Console

Static groups organize entries by specifying the same group value in the DN attribute of any number of users.

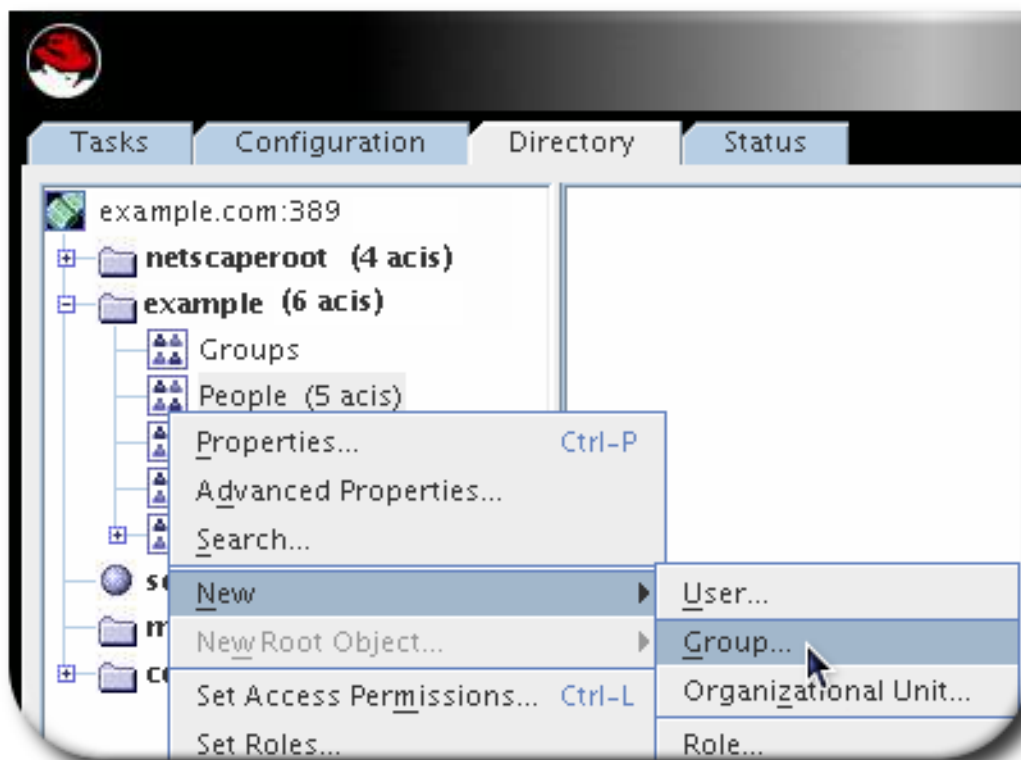


NOTE

If a user has an entry on a remote Directory Server (for example, in a chained database), different from the Directory Server which has the entry that defines the static group, then use the Referential Integrity plug-in to ensure that deleted user entries are automatically deleted from the static group.

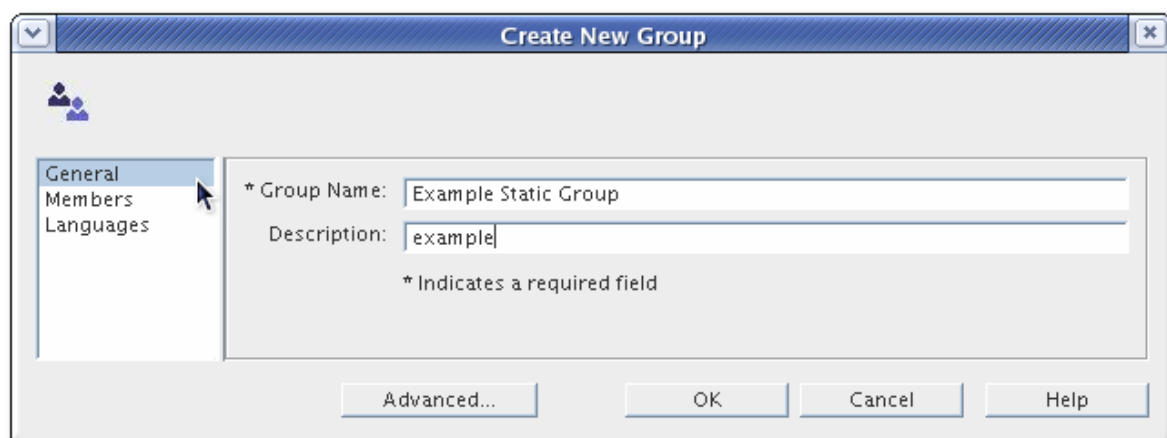
There are some performance and access control considerations with the Referential Integrity plug-in. For more information about using referential integrity with chaining, see [Section 2.3.2, “Configuring the Chaining Policy”](#).

1. In the Directory Server Console, select the **Directory** tab.
2. In the left pane, right-click the entry under which to add a new group, and select **New > Group**.

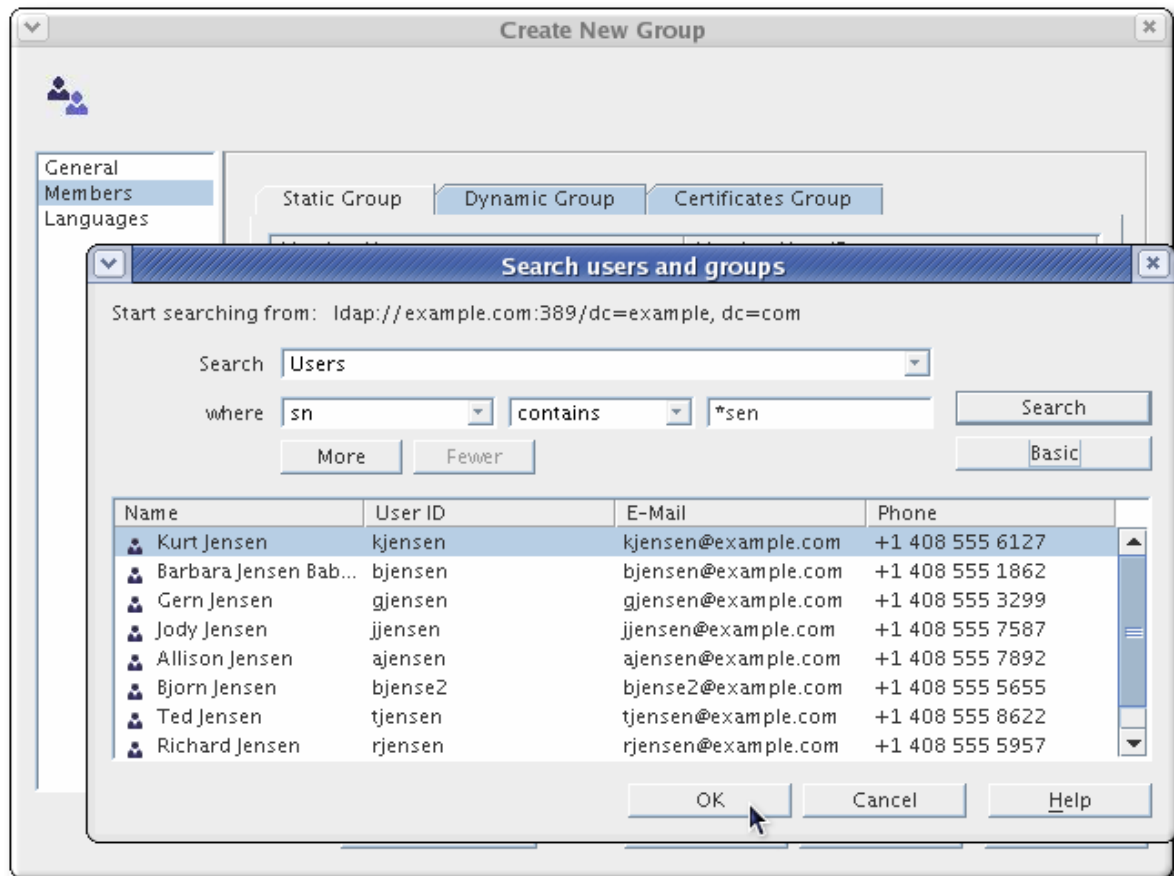


Alternatively, go to the **Object** menu, and select **New > Group**.

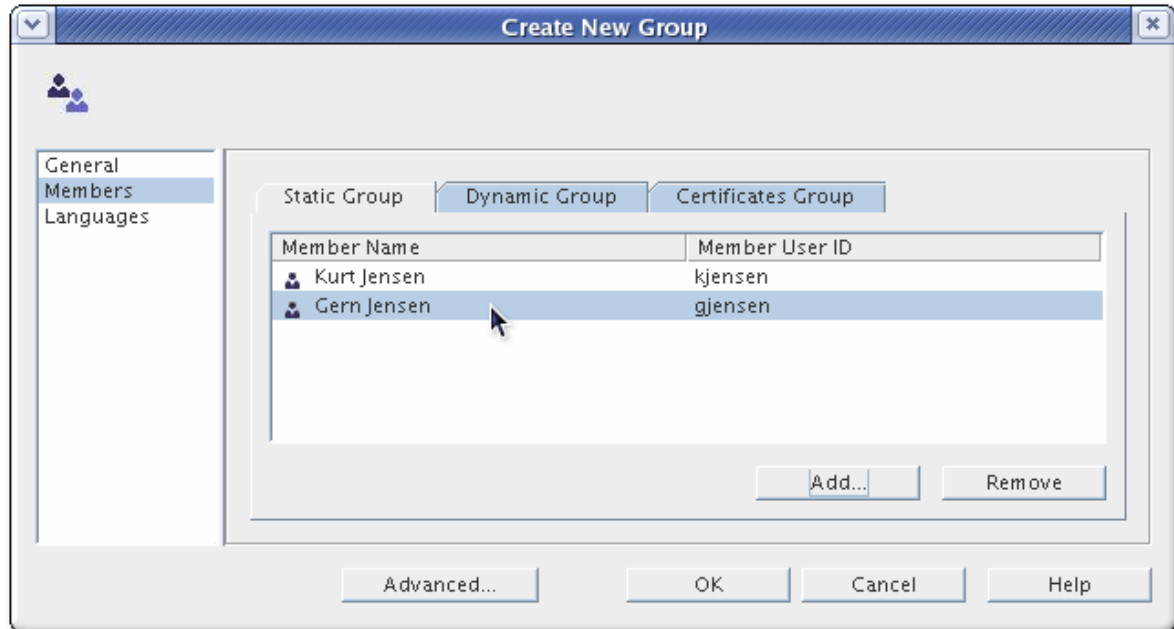
- Click **General** in the left pane. Type a name for the new group in the **Group Name** field (the name is required), and enter a description of the new group in the **Description** field.



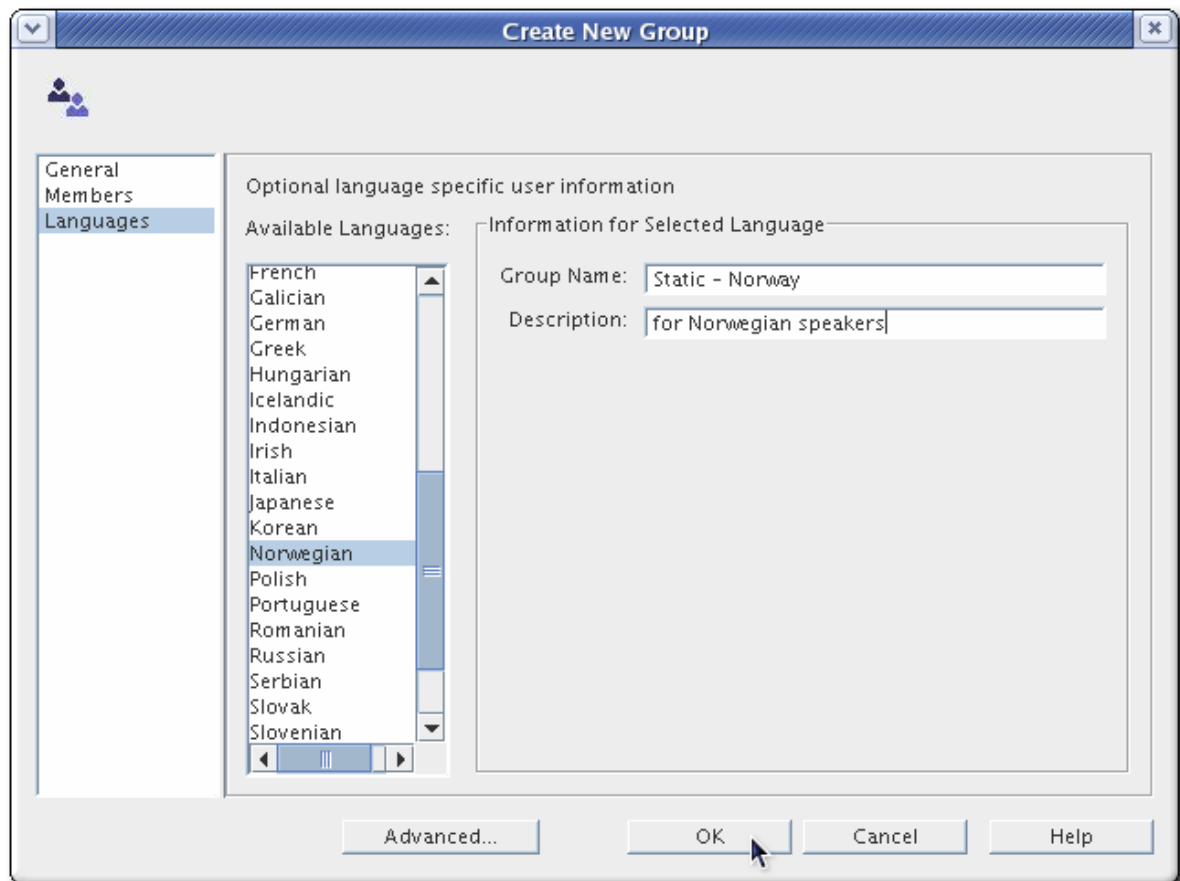
- Click **Members** in the left pane. In the right pane, select the **Static Group** tab. Click **Add** to add new members to the group.
- In the **Search** drop-down list, select what sort of entries to search for (users, groups, or both) then click **Search**.



6. Select the members from the returned entries, and click **OK**.



7. Click **Languages** in the left pane to add language-specific information for the group.



8. Click **OK** to create the new group. It appears in the right pane.

To edit a static group, double-click the group entry, and make the changes in the editor window. To view the changes, go to the **View** menu, and select **Refresh**.



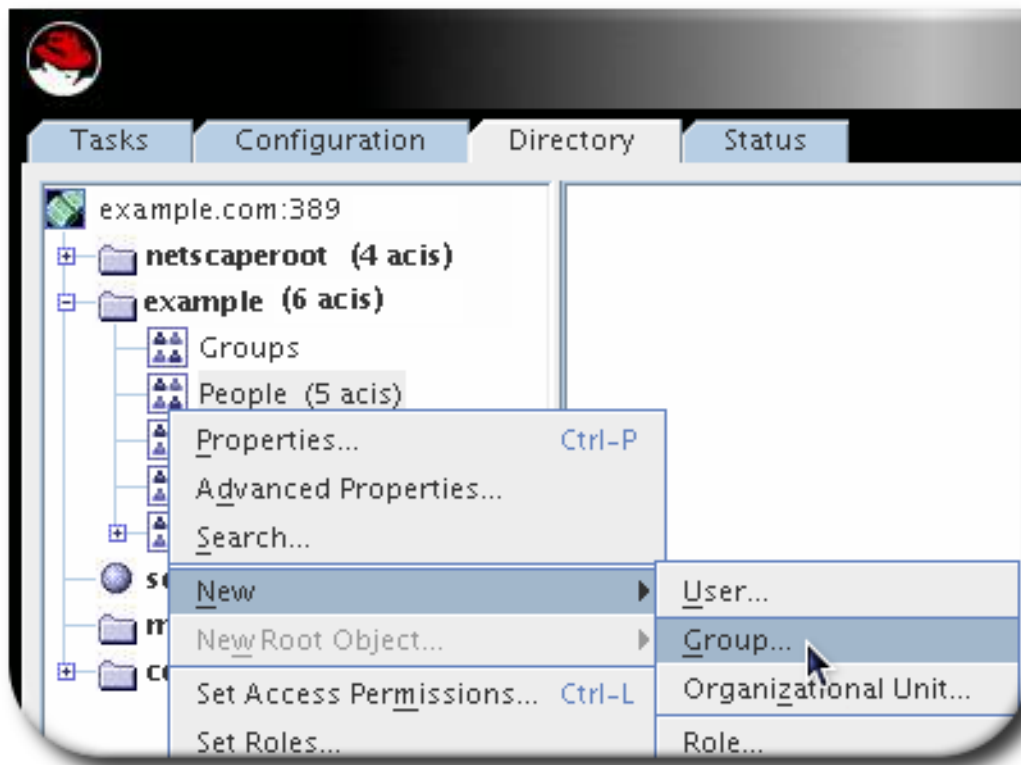
NOTE

The Console for managing static groups may not display all possible selections during a search operation if there is no VLV index for users' search. This problem occurs only when the number of users is 1000 or more and there is no VLV index for search. To work around the problem, create a VLV index for the users suffix with the filter **(objectclass=person)** and scope **sub-tree**.

6.1.2. Creating Dynamic Groups in the Console

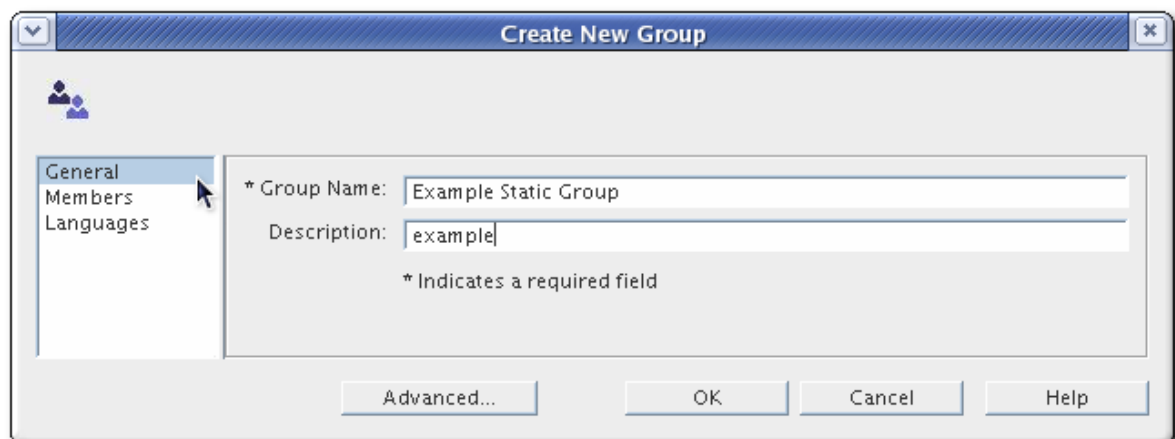
Dynamic groups filter users based on their DN and include them in a single group.

1. In the Directory Server Console, select the **Directory** tab.
2. In the left pane, right-click the entry under which to add a new group, and select **New > Group**.

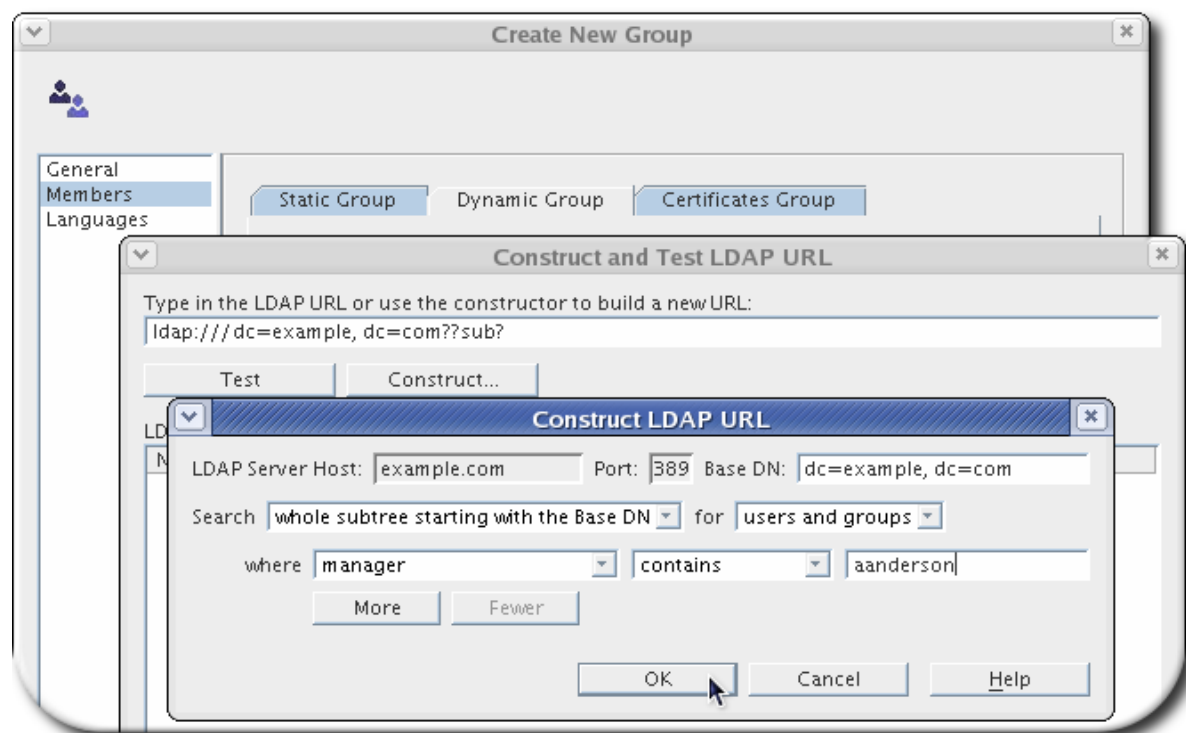


Alternatively, go to the **Object** menu, and select **New > Group**.

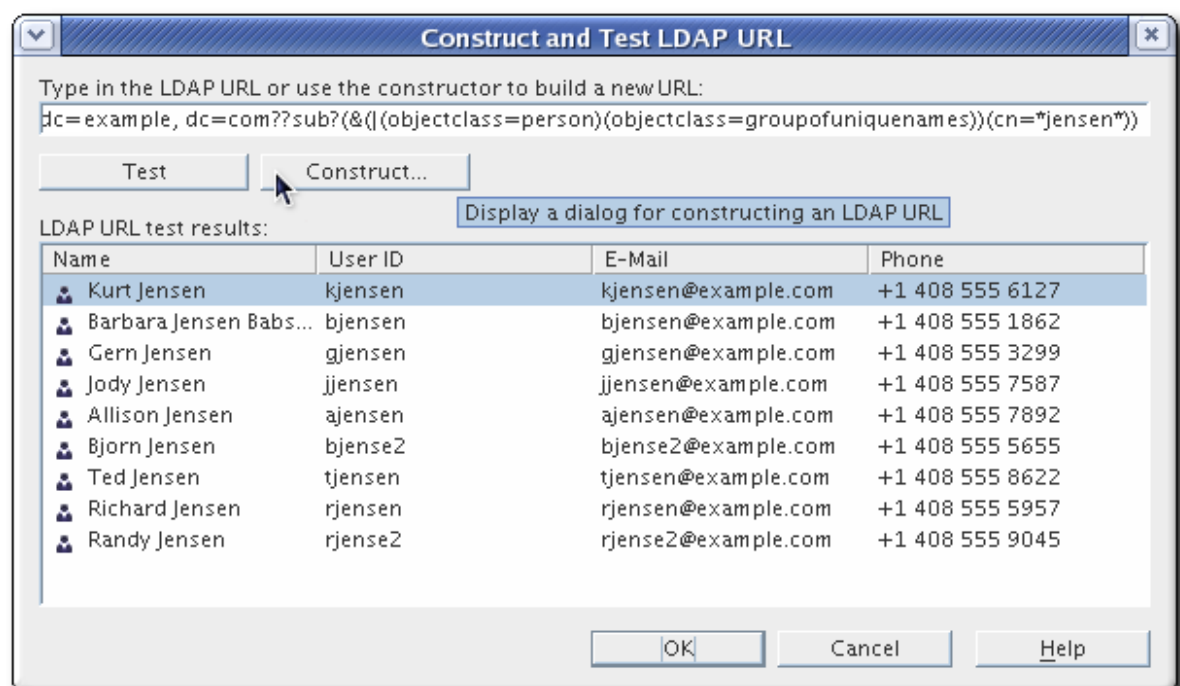
3. Click **General** in the left pane. Type a name for the new group in the **Group Name** field (the name is required), and enter a description of the new group in the **Description** field.



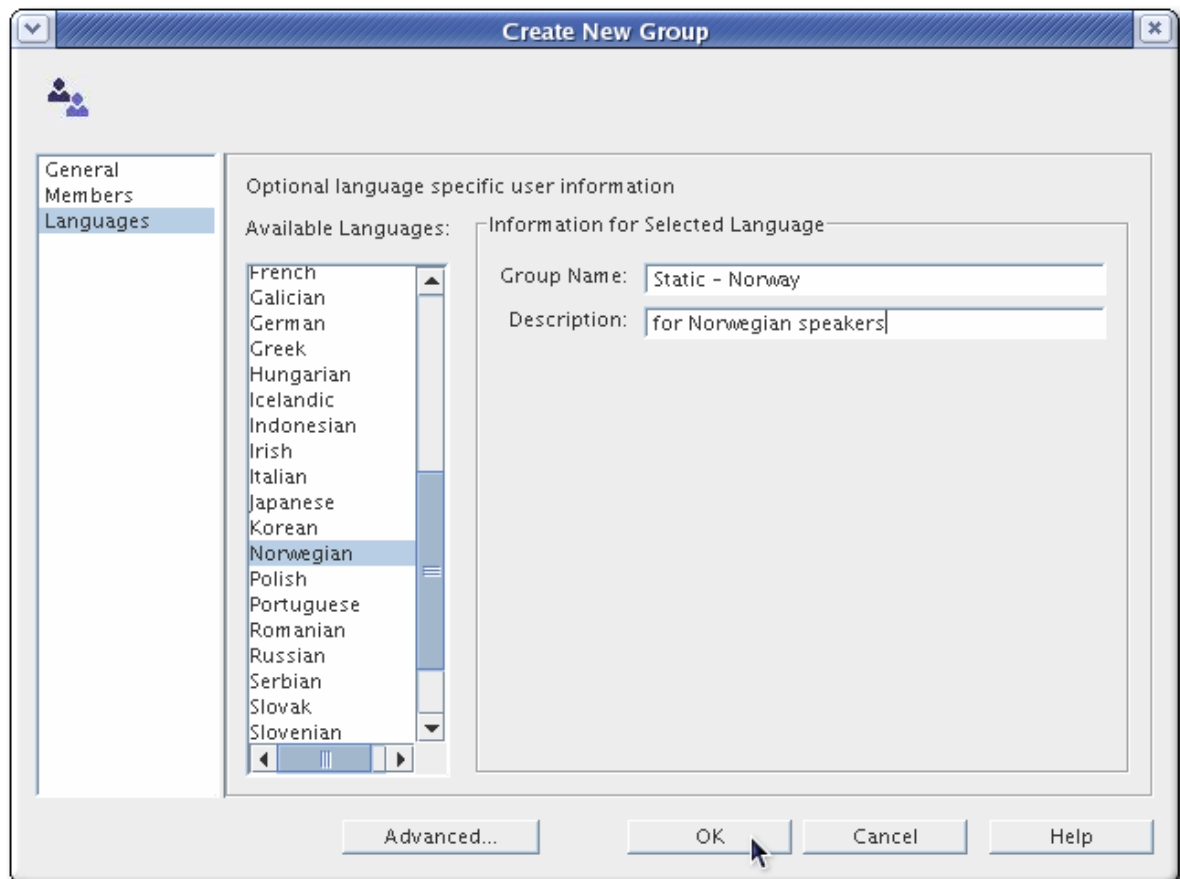
4. Click **Members** in the left pane. In the right pane, select the **Dynamic Group** tab. Click **Add** to create a LDAP URL for querying the database.
5. Enter an LDAP URL in the text field or select **Construct** to be guided through the construction of an LDAP URL.



The results show the current entries (group members) which correspond to the filter.



- Click **Languages** in the left pane to add language-specific information for the group.



7. Click **OK**. The new group appears in the right pane.

To edit a dynamic group, double-click the group entry to open the editor window, and make whatever changes to the dynamic group. To view the changes to the group, go to the **View** menu, and select **Refresh**.



NOTE

The Console for managing dynamic groups may not display all possible selections during a search operation if there is no VLV index for users' search. This problem can occur when the number of users is 1000 or more and there is no VLV index for search. To work around the problem, create a VLV index for the users suffix with the filter **(objectclass=person)** and scope **sub-tree**.

6.1.3. Creating Groups in the Command Line

Creating both static and dynamic groups from the command line is a similar process. A group entry contains the group name, the type of group, and a members attribute.

There are several different options for the type of group; these are described in more detail in the [Red Hat Directory Server 9 Configuration, Command, and File Reference](#). The *type of group* in this case refers to the type of defining member attribute it has:

- **groupOfNames** is a simple group, that allows any entry to be added. The attribute used to determine members for this is **member**.
- **groupOfUniqueNames**, like **groupOfNames**, simply lists user DNs as members, but the members must be unique. This prevents users being added more than once as a group member, which is one way of preventing self-referential group memberships. The attribute used

to determine members for this is *uniqueMember*.

- **groupOfURLs** uses a list of LDAP URLs to filter and generate its membership list. This object class is required for any dynamic group and can be used in conjunction with **groupOfNames** and **groupOfUniqueNames**.
- **groupOfCertificates** is similar to **groupOfURLs** in that it uses an LDAP filter to search for and identify certificates (or, really, certificate names) to identify group members. This is useful for group-based access control, since the group can be given special access permissions. The attribute used to determine members for this is *memberCertificate*.

Table 6.1, “Dynamic and Static Group Schema” lists the default attributes for groups as they are created from the command line.

Table 6.1. Dynamic and Static Group Schema

Type of Group	Group Object Classes	Member Attributes
Static	groupOfUniqueNames	uniqueMember
Dynamic	<div>groupOfUniqueNames</div> <div>groupOfURLs</div>	memberURL

A static group entry lists the specific members of the group. For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x

dn: cn=static group,ou=Groups,dc=example,dc=com
changetype: add
objectClass: top
objectClass: groupOfUniqueNames
cn: static group
description: Example static group.
uniqueMember: uid=mwhite,ou=People,dc=example,dc=com
uniqueMember: uid=awhite,ou=People,dc=example,dc=com
```

A dynamic group uses at least one LDAP URL to identify entries belonging to the group and can specify multiple LDAP URLs or, if used with another group object class like **groupOfUniqueNames**, can explicitly list some group members along with the dynamic LDAP URL.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x

dn: cn=dynamic group,ou=Groups,dc=example,dc=com
changetype: add
objectClass: top
objectClass: groupOfUniqueNames
objectClass: groupOfURLs
cn: dynamic group
description: Example dynamic group.
memberURL: ldap:///dc=example,dc=com??sub?(&(objectclass=person)
(cn=*sen*))
```


6.1.4. Listing Group Membership in User Entries

The entries which belong to a group are defined, in some way, in the group entry itself. This makes it very easy to look at a group and see its members and to manage group membership centrally. However, there is no good way to find out what groups a single user belongs to. There is nothing in a user entry which indicates its memberships, as there are with roles.

The MemberOf Plug-in correlates group membership lists to the corresponding user entries.

The MemberOf Plug-in analyzes the member attribute in a group entry and automatically writes a corresponding **memberOf** attribute in the member's entry. (By default, this checks the **member** attribute, but multiple attribute instances can be used to support multiple different group types.)

As membership changes, the plug-in updates the **memberOf** attributes on the user entries. The MemberOf Plug-in provides a way to view the groups to which a user belongs simply by looking at the entry, including nested group membership. It can be very difficult to backtrack memberships through nested groups, but the MemberOf Plug-in shows memberships for all groups, direct and indirect.

The MemberOf Plug-in manages member attributes for static groups, not dynamic groups or circular groups.

6.1.4.1. Directory Topology Considerations with the MemberOf Plug-in

memberOf and Multi-Master Replication

The **memberOf** attributes for user entries *should not be replicated in multi-master environments*. Make sure that the **memberOf** attribute is excluded from replication in the replication agreement. (Fractional replication is described in [Section 11.1.7, “Replicating a Subset of Attributes with Fractional Replication”](#).)

Each server must maintain its own MemberOf Plug-in independently. To make sure that the **memberOf** attributes for entries are the same across servers, simply configure the MemberOf Plug-in the same on all servers.

With single-master replication, it is perfectly safe to replicate **memberOf** attributes. Configure the MemberOf Plug-in for the supplier, then replicate the **memberOf** attributes to the consumers.

memberOf and Distributed Databases

It is possible, as outlined in [Section 2.2.1, “Creating Databases”](#), to distribute suffixes and directory data across different databases.

By default, the MemberOf Plug-in only looks for potential members for users who are in the same database as the group. If users are stored in a different database than the group, then the user entries will not be updated with **memberOf** attributes because the plug-in cannot ascertain the relationship between them.

The plug-in can be configured to search across all databases by enabling the **memberOfAllBackends** in the plug-in configuration.

6.1.4.2. Required Object Classes by the memberOf Plug-In

To enable the **memberOf** plug-in to add the **memberOf** attribute, the user entry must contain the **inetUser** or **inetAdmin** object class to support this attribute. If you configure the **memberOf** plug-in to use a different attribute, make sure that the user entry contains an object class that supports this attribute.

To configure nested groups, the group must use the **extensibleObject** object class.



NOTE

If directory entries do not contain an object class that supports the required attributes, operations fail with the following error:

```
LDAP: error code 65 - Object Class Violation
```

6.1.4.3. The MemberOf Plug-in Syntax

The MemberOf Plug-in instance defines two attributes, one for the group member attribute to poll (*memberOfGroupAttr*) and the other for the attribute to create and manage in the member's user entry (*memberOfAttr*).

The *memberOfGroupAttr* attribute is multi-valued. Because different types of groups use different member attributes, using multiple *memberOfGroupAttr* attributes allows the plug-in to manage multiple types of groups.

The plug-in instance also gives the plug-in path and function to identify the MemberOf Plug-in and contains a state setting to enable the plug-in, both of which are required for all plug-ins. The default MemberOf Plug-in is shown in [Example 6.1, “Default MemberOf Plug-in Entry”](#) and the different parameters are described in [Table 6.2, “MemberOf Syntax”](#).

Example 6.1. Default MemberOf Plug-in Entry

```
dn: cn=MemberOf Plugin,cn=plugins,cn=config
objectClass: top
objectClass: nsSlapdPlugin
objectClass: extensibleObject
cn: MemberOf Plugin
nsslapd-pluginPath: libmemberof-plugin
nsslapd-pluginInitfunc: memberof_postop_init
nsslapd-pluginType: postoperation
nsslapd-pluginEnabled: on
nsslapd-plugin-depends-on-type: database
memberOfGroupAttr: member
memberOfGroupAttr: uniqueMember
memberOfAttr: memberOf
memberOfAllBackends: on
nsslapd-pluginId: memberOf
nsslapd-pluginVersion: 9.0.4
nsslapd-pluginVendor: Red Hat, Inc.
nsslapd-pluginDescription: memberOf plugin
```

Table 6.2. MemberOf Syntax

Plug-in Attribute	Description
cn	Gives a unique name for the plug-in instance.

Plug-in Attribute	Description
memberOfGroupAttr	Gives the attribute in the group entry to poll to identify member DNs. By default, this is the member attribute, but it can be any attribute used to identify group members, such as uniqueMember . This is a multi-valued attribute, so if multiple types of groups will be used with the MemberOf Plug-in, multiple member type attributes can be set.
memberOfAttr	Gives the attribute for the plug-in to create and manage on the user entry. By default, this is the memberOf attribute.
memberOfAllBackends	Sets whether the plug-in should evaluate user entries only within the local suffix (off) or whether it should evaluate all configured databases (on).
memberOfEntryScope	Sets on which suffixes the plug-in works on. If not set, the plug-in works on all suffixes.
memberOfEntryScopeExcludeSubtree	Sets what suffixes the plug-in excludes.
nsslapd-pluginEnabled	Sets whether the plug-in instance is enabled (active) or disabled. The default MemberOf Plug-in instance is disabled by default.
nsslapd-pluginPath	Gives the name of the specific plug-in to load. This must be libmemberof-plugin .
nsslapd-pluginInitfunc	Gives the name of the function to call to initialize the plug-in. This must be memberof_postop_init .

NOTE

To maintain backwards compatibility with older versions of Directory Server, which only allowed a single member attribute (by default, **member**), it may be necessary to include the **member** group attribute or whatever previous member attribute was used, in addition any new member attributes used in the plug-in configuration.

```
memberOfGroupAttr: member
memberOfGroupAttr: uniqueMember
```

See [Example 6.1, “Default MemberOf Plug-in Entry”](#).

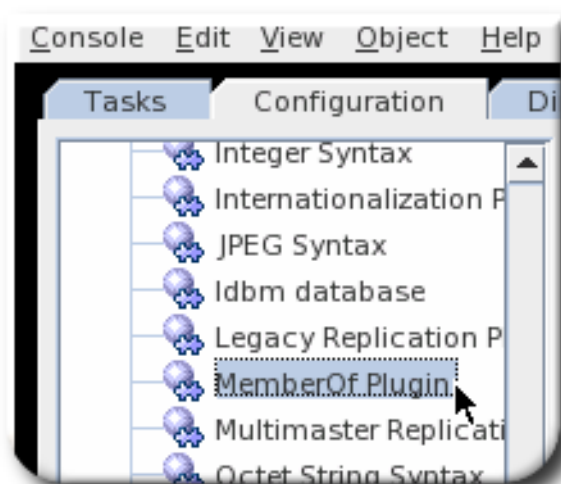
6.1.4.4. Configuring an Instance of the MemberOf Plug-in

The attributes defined in the MemberOf Plug-in can be changed, depending on the types of groups used in the directory.

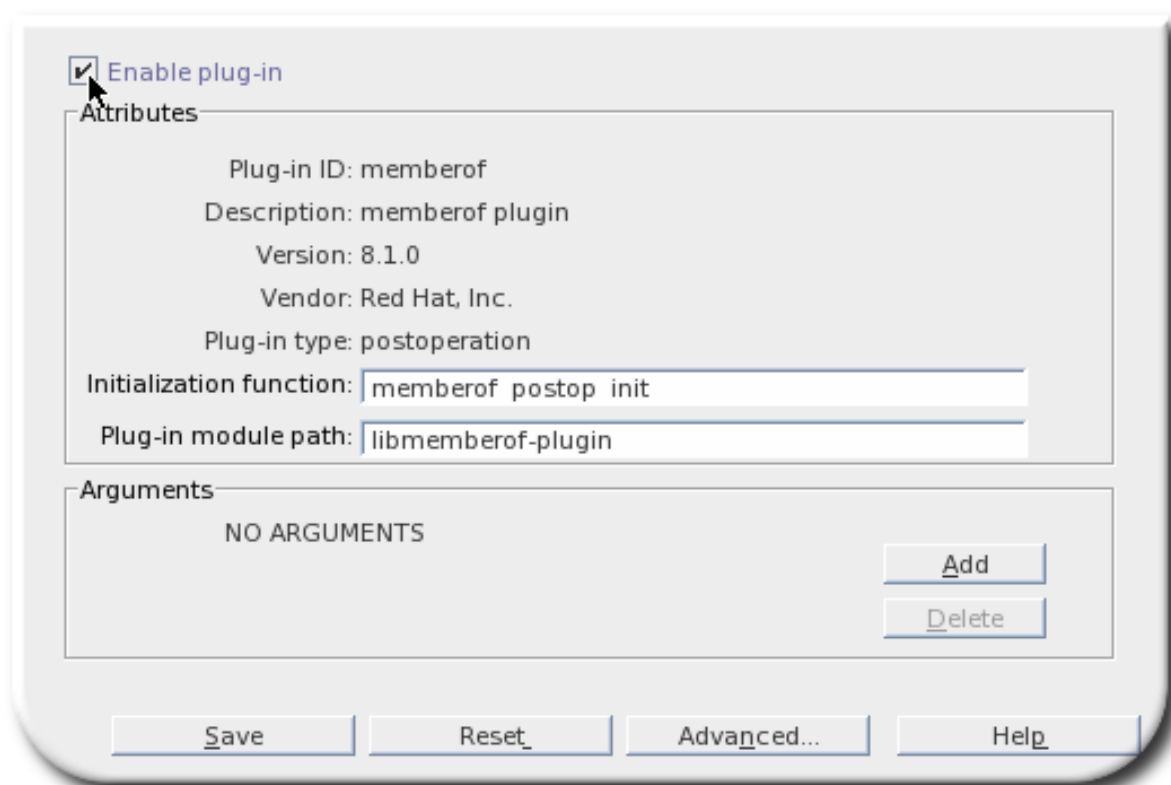
6.1.4.4.1. Editing the MemberOf Plug-in from the Console

1. Select the **Configuration** tab, and expand to the **Plugins** folder.

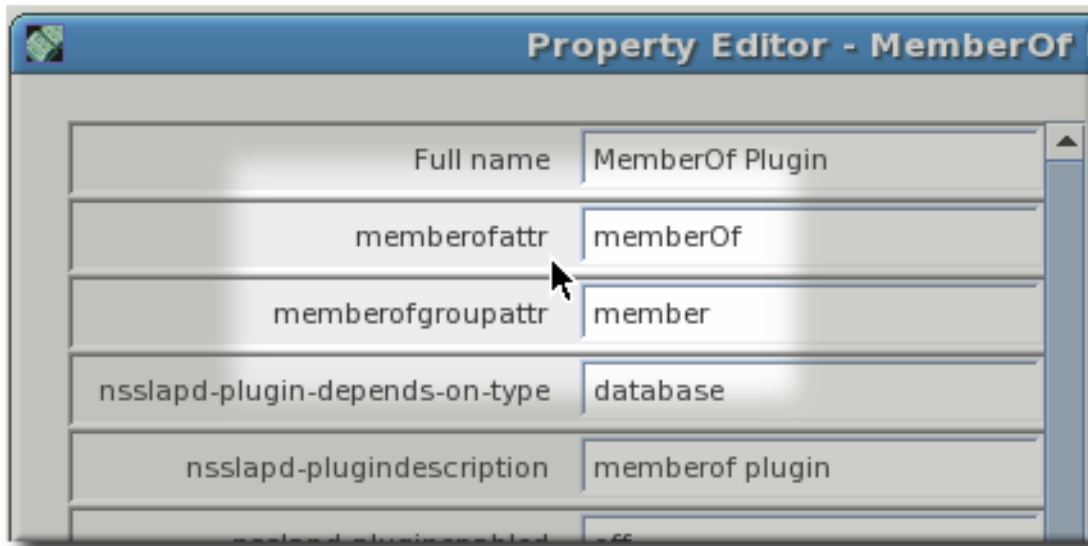
2. Scroll to the **Memberof Plugin** entry.



3. Make sure that the plug-in is enabled. This is disabled by default.



4. Click the **Advanced** button to open the **Advanced Properties Editor**.
5. The **memberOfGroupAttr** attribute sets the attribute in the group entry which the server uses to identify member entries; this attribute can be used multiple times for different group/member types. The **memberOfAttr** attribute sets the attribute which the plug-in creates and manages on user entries.



6. Save the changes.
7. Restart the server to update the plug-in. For example, open the **Tasks** tab and click the **Restart server** task.

6.1.4.4.2. Editing the MemberOf Plug-in from the Command Line

1. Enable the MemberOf Plug-in.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=MemberOf Plugin,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
-
```

2. Set the attribute to use for the group member entry attribute. The default attribute is *member*, which can be changed using the **replace** command, or, since the *memberOfGroupAttr* attribute is multi-valued, additional member types can be added to the definition. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=MemberOf Plugin,cn=plugins,cn=config
changetype: modify
add: memberOfGroupAttr
memberOfGroupAttr: uniqueMember

add: memberOfGroupAttr
memberOfGroupAttr: customMember-
```

3. Set the attribute to set on the user entries to show group membership. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x
```

```
dn: cn=MemberOf Plugin,cn=plugins,cn=config
changetype: modify
replace: memberOfAttr
memberOfAttr: memberOf
-
```

4. *Optional.* If the deployment uses distributed databases, then enable the ***memberOfAllBackends*** attribute to search through all databases, not just the local one, for user entries.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=MemberOf Plugin,cn=plugins,cn=config
changetype: modify
replace: memberOfAllBackends
memberOfAllBackends: on
-
```

5. Restart the Directory Server to load the modified new plug-in instance.

```
service dirsrv restart instance_name
```

6.1.4.5. Setting the Scope of the MemberOf Plug-in

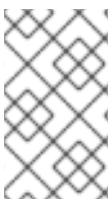
If you configured several back ends or multiple-nested suffixes, you can use the ***memberOfEntryScope*** and ***memberOfEntryScopeExcludeSubtree*** parameters to set what suffixes the **MemberOf** plug-in works on.

If you add a user to a group, the **MemberOf** plug-in only adds the ***memberOf*** attribute to the group if both the user and the group are in the plug-in's scope. For example, to configure the **MemberOf** plug-in to work on all entries in ***dc=example, dc=com***, but to exclude entries in ***ou=private, dc=example, dc=com***, set:

```
memberOfEntryScope: dc=example,dc=com
memberOfEntryScopeExcludeSubtree: ou=private,dc=example,dc=com
```

If you moved a user entry out of the scope set in the ***memberOfEntryScope*** parameter:

- The membership attribute, such as ***member***, is updated in the group entry to remove the user DN value.
- The ***memberOf*** attribute is updated in the user entry to remove the group DN value.



NOTE

The value set in the ***memberOfEntryScopeExcludeSubtree*** parameter has a higher priority than values set in ***memberOfEntryScope***. If the scopes set in both parameters overlap, the **MemberOf** plug-in only works on the non-overlapping directory entries.

6.1.4.6. Synchronizing memberOf Values

The MemberOf Plug-in automatically manages the *memberOf* attribute on group member entries, based on the configuration in the group entry itself. However, the *memberOf* attribute can be edited on a user entry directly (which is improper) or new entries can be imported or replicated over to the server that have a *memberOf* attribute already set. These situations create inconsistencies between the *memberOf* configuration managed by the server plug-in and the actual memberships defined for an entry.

Directory Server has a *memberOf* repair task which manually runs the plug-in to make sure the appropriate *memberOf* attributes are set on entries. There are three ways to trigger this task:

- In the Directory Server Console
- Using the **fixup-memberof.pl** script
- Running a **cn=memberOf task,cn=tasks,cn=config** tasks entry



NOTE

The memberOf regeneration tasks are run locally, even if the entries themselves are replicated. This means that the *memberOf* attributes for the entries on other servers are not updated until the updated entry is replicated.

6.1.4.6.1. Initializing and Regenerating memberOf Attributes Using fixup-memberof.pl

The **fixup-memberof.pl** script launches a special task to regenerate all of the *memberOf* attributes on user entries based on the defined member attributes in the group entries. This is a clean-up task which synchronizes the membership defined in group entries and the corresponding user entries and overwrites any accidental or improper edits on the user entries.

1. Open the tool directory for the Directory Server instance,
/usr/lib/dirsrv/slapd-*instance_name*/.
2. Run the script, binding as the Directory Manager.

```
./fixup-memberof.pl -D "cn=Directory Manager" -w password
```

The **fixup-memberof.pl** command is described in more detail in the [Configuration and Command-Line Tool Reference](#).

6.1.4.6.2. Initializing and Regenerating memberOf Attributes Using ldapmodify

Regenerating *memberOf* attributes is one of the tasks which can be managed through a special task configuration entry. Task entries occur under the **cn=tasks** configuration entry in the **dse.ldif** file, so it is also possible to initiate a task by adding the entry using **ldapmodify**. As soon as the task is complete, the entry is removed from the directory.

The **fixup-memberof.pl** script creates a special task entry in a Directory Server instance which regenerates the *memberOf* attributes.

To initiate a memberOf fixup task, add an entry under the **cn=memberOf task,cn=tasks,cn=config** entry. The only required attribute is the **cn** for the specific task.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=example memberOf,cn=memberOf task,cn=tasks,cn=config
```

```
changetype: add  
cn:example memberOf
```

As soon as the task is completed, the entry is deleted from the **dse.ldif** configuration, so it is possible to reuse the same task entry continually.

The **cn=memberof task** configuration is described in more detail in the [Configuration and Command-Line Tool Reference](#).

6.1.5. Automatically Adding Entries to Specified Groups

- [Section 6.1.5.1, “Looking at the Structure of an Automembership Rule”](#)
- [Section 6.1.5.2, “Examples of Automembership Rules”](#)
- [Section 6.1.5.3, “Creating Automembership Definitions”](#)

Group management can be a critical factor for managing directory data, especially for clients which use Directory Server data and organization or which use groups to apply functionality to entries. Groups make it easier to apply policies consistently and reliably across the directory. Password policies, access control lists, and other rules can all be based on group membership.

Being able to assign new entries to groups, automatically, at the time that an account is created ensures that the appropriate policies and functionality are immediately applied to those entries — without requiring administrator intervention.

Dynamic groups are one method of creating groups and assigning members automatically because any matching entry is automatically included in the group. For applying Directory Server policies and settings, this is sufficient. However, LDAP applications and clients commonly need a static and explicit list of group members in order to perform whatever operation is required. And all of the members in static groups have to be manually added to those groups.

The static group itself cannot search for members like a dynamic group, but there is a way to allow a static group to have members added to it automatically — *the Auto Membership Plug-in*.

Automembership essentially allows a static group to act like a dynamic group, at least for adding new members to the group. Different automembership definitions create searches that are automatically run on all new directory entries. The automembership rules search for and identify matching entries — much like the dynamic search filters — and then explicitly add those entries as members to the static group.



NOTE

Automembership assignments are only made automatically when an entry is added to the Directory Server.

For existing entries or entries who are edited to match an automember group rule, there is a fix-up task which can be run that updates the group membership.

The Auto Membership Plug-in can target any type of object stored in the directory: users, machines and network devices, customer data, or other assets.



NOTE

The Auto Membership Plug-in adds a new entry to an existing group based on defined criteria. It does not create a group for the new entry.

To create a corresponding group entry when a new entry of a certain type is created, use the Managed Entries Plug-in. This is covered in [Section 6.3, “Automatically Creating Dual Entries”](#).

6.1.5.1. Looking at the Structure of an Automembership Rule

The Auto Membership Plug-in itself is a container entry in **cn=plugins,cn=config**. Group assignments are defined through child entries.

6.1.5.1.1. The Automembership Configuration Entry

Automembership assignments are created through a main definition entry, a child of the Auto Membership Plug-in entry. Each definition entry defines three elements:

- An LDAP search to identify entries, including both a search scope and a search filter (***autoMemberScope*** and ***autoMemberFilter***)
- A default group to which to add the member entries (***autoMemberDefaultGroup***)
- The member entry format, which is the attribute in the group entry, such as ***member***, and the attribute value, such as ***dn (autoMemberGroupingAttr)***

The definition is the basic configuration for an automember rule. It identifies all of the required information: what a matching member entry looks like and a group for that member to belong to.

For example, this definition assigns all Windows users to the **cn=windows-users** group:

```
dn: cn=Windows Users,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberDefinition
autoMemberScope: ou=People,dc=example,dc=com
autoMemberFilter: objectclass=ntUser
autoMemberDefaultGroup: cn=windows-group,cn=groups,dc=example,dc=com
autoMemberGroupingAttr: member:dn
```

The attributes for the definition entry are listed in [Table 6.3, “Automember Definition Attributes”](#).

Table 6.3. Automember Definition Attributes

Attribute	Description
autoMemberDefinition (required object class)	Identifies the entry as an automember definition. This entry must be a child of the Auto Membership Plug-in, cn=Auto Membership Plugin,cn=plugins,cn=config .
autoMemberScope	Sets the subtree DN to search for entries. This is the search base.

Attribute	Description
autoMemberFilter	Sets a standard LDAP search filter to use to search for matching entries. Examples of search filters are in Section 10.4, “LDAP Search Filters” .
autoMemberDefaultGroup	Sets a default or fallback group to add the entry to as a member. If the definition does not use any regular expression conditions, then this is the primary group to which entries are added. If the automember definition does have defined regular expression conditions, then an entry is added to those specified groups first, and the autoMemberDefaultGroup group is used as a fallback for entries which match the autoMemberFilter but do not match a regular expression.
autoMemberGroupingAttr	Sets the name of the member attribute in the group entry and the attribute in the object entry that supplies the member attribute value. This structures how the Auto Membership Plug-in adds a member to the group, depending on the group configuration. For example, for a groupOfUniqueNames user group, each member is added as a uniqueMember attribute. The value of uniqueMember is the DN of the user entry. In essence, each group member is identified by the attribute-value pair of uniqueMember: user_entry_DN . The member entry format, then, is uniqueMember: dn .

6.1.5.1.2. Additional Regular Expression Entries

For something like a users group, where more than likely all matching entries should be added as members, a simple definition is sufficient. However, there can be instances where entries that match the LDAP search filter should be added to different groups, depending on the value of some other attribute. For example, machines may need to be added to different groups depending on their IP address or physical location; users may need to be in different groups depending on their employee ID number.

The automember definition can use regular expressions to provide additional conditions on what entries to include or exclude from a group, and then a new, specific group to add those selected entries to.

For example, an automember definition sets all machines to be added to a generic host group.

Example 6.2. Automember Definition for a Host Group

```
dn: cn=Hostgroups,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberDefinition
cn: Hostgroups
autoMemberScope: dc=example,dc=com
autoMemberFilter: objectclass=ipHost
autoMemberDefaultGroup:
cn=systems,cn=hostgroups,ou=groups,dc=example,dc=com
autoMemberGroupingAttr: member:dn
```

A regular expression rule is added so that any machine with a fully-qualified domain name within a given range is added to a web server group.

Example 6.3. Regular Expression Condition for a Web Server Group

```
dn: cn=webservers,cn=Hostgroups,cn=Auto Membership
Plugin,cn=plugins,cn=config
objectclass: autoMemberRegexRule
description: Group for webservers
cn: webservers
autoMemberTargetGroup: cn=webservers,cn=hostgroups,dc=example,dc=com
autoMemberInclusiveRegex: fqdn=^www\.web[0-9]+\\.example\.com
```

So, any host machine added with a fully-qualified domain name that matches the expression `^www\.web[0-9]+\\.example\.com`, such as `www.web1.example.com`, is added to the `cn=webservers` group, defined for that exact regular expression. Any other machine entry, which matches the LDAP filter `objectclass=ipHost` but with a different type of fully-qualified domain name, is added to the general host group, `cn=systems`, defined in the main definition entry.

The group in the definition, then, is a fallback for entries which match the general definition, but do not meet the conditions in the regular expression rule.

Regular expression rules are child entries of the automember definition.

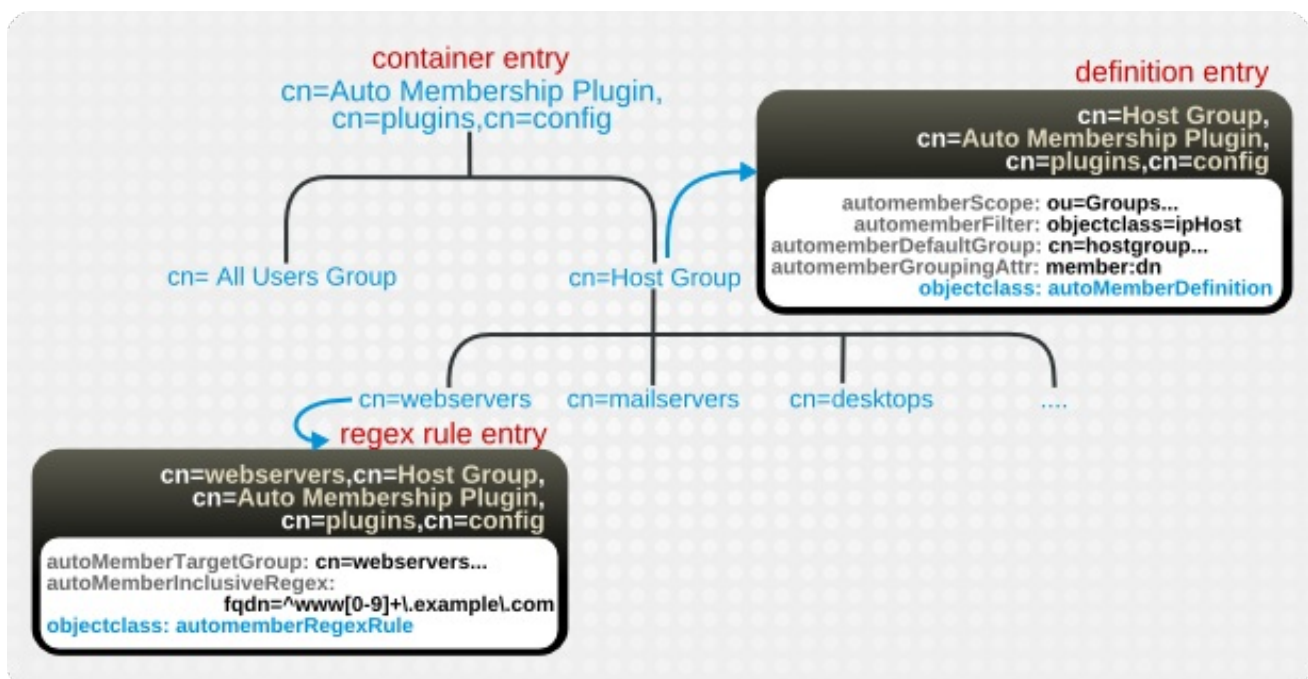



Figure 6.1. Regular Expression Conditions

Each rule can include multiple inclusion and exclusion expressions. (Exclusions are evaluated first.) If an entry matches any inclusion rule, it is added to the group.

There can be only one target group given for the regular expression rule.

Table 6.4. Regular Expression Condition Attributes

Attribute	Description
autoMemberRegexRule (required object class)	Identifies the entry as a regular expression rule. This entry must be a child of an automember definition (objectclass: autoMemberDefinition).
autoMemberInclusiveRegex	<p>Sets a regular expression to use to identify entries to include. Only matching entries are added to the group. Multiple regular expressions could be used, and if an entry matches any one of those expressions, it is included in the group.</p> <p>The format of the expression is a Perl-compatible regular expression (PCRE). For more information on PCRE patterns, see the pcreyntax(3) man page.</p> <p>This is a multi-valued attribute.</p>
autoMemberExclusiveRegex	<p>Sets a regular expression to use to identify entries to exclude. If an entry matches the exclusion condition, then it is <i>not</i> included in the group. Multiple regular expressions could be used, and if an entry matches any one of those expressions, it is excluded in the group.</p> <p>The format of the expression is a Perl-compatible regular expression (PCRE). For more information on PCRE patterns, see the pcreyntax(3) man page.</p> <p>This is a multi-valued attribute.</p> <div>  <div> <p>NOTE</p> <p>Exclude conditions are evaluated first and take precedence over include conditions.</p> </div> </div>
autoMemberTargetGroup	Sets which group to add the entry to as a member, if it meets the regular expression conditions.

6.1.5.2. Examples of Automembership Rules

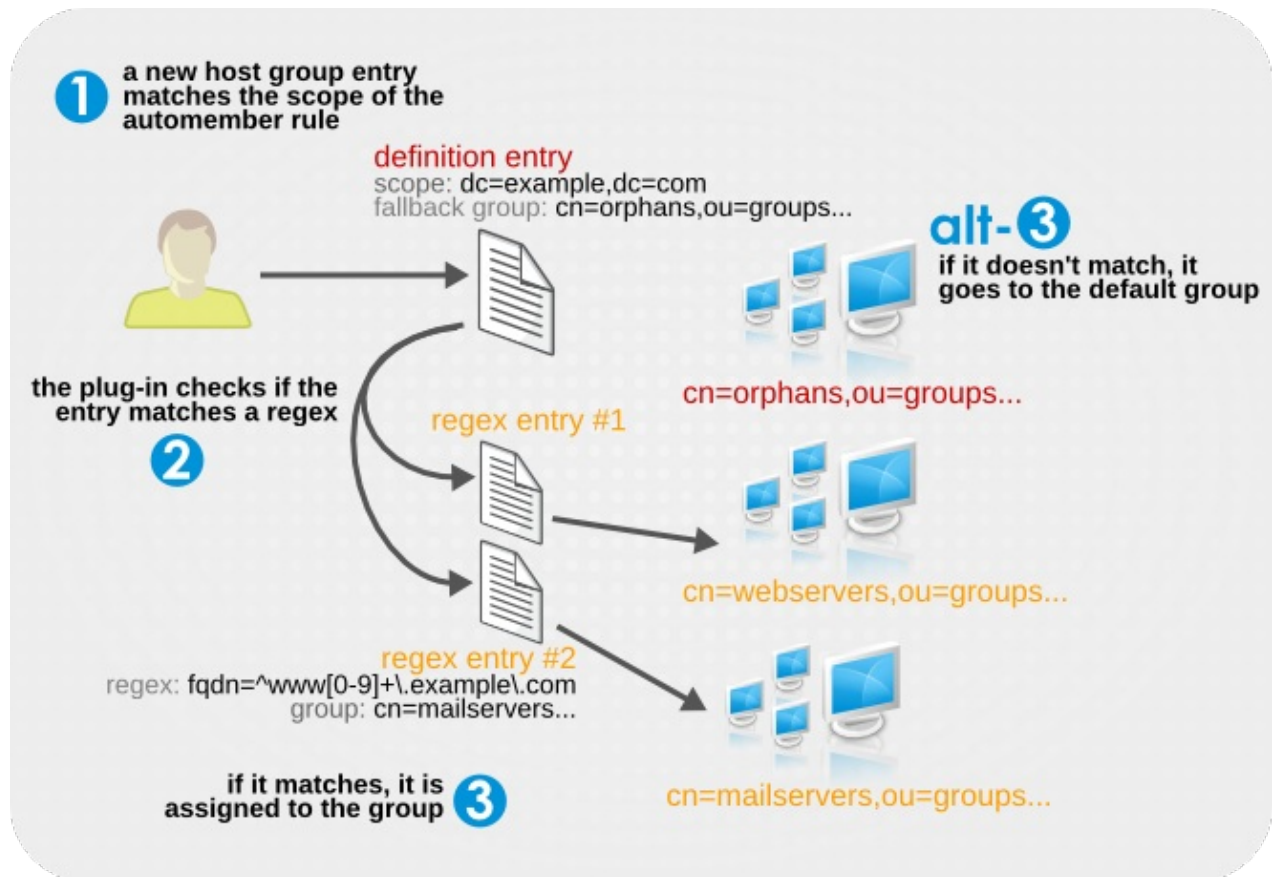
Automembership rules are usually going to applied to users and to machines (although they can be applied to any type of entry). There are a handful of examples that may be useful in planning automembership rules:

- Different host groups based on IP address
- Windows user groups
- Different user groups based on employee ID

Example 6.4. Host Groups by IP Address

The automember rule first defines the scope and target of the rule. The example in [Section 6.1.5.1.2, “Additional Regular Expression Entries”](#) uses the configuration group to define the fallback group and a regular expression entry to sort out matching entries.

The scope is used to find *all* host entries. The plug-in then iterates through the regular expression entries. If an entry matches an inclusive regular expression, then it is added to that host group. If it does not match any group, it is added to the default group.



The actual plug-in configuration entries are configured like this, for the definition entry and two regular expression entries to filter hosts into a web servers group or a mail servers group.

configuration entry

```
dn: cn=Hostgroups,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberDefinition
cn: Hostgroups
autoMemberScope: dc=example,dc=com
autoMemberFilter: objectclass=bootableDevice
autoMemberDefaultGroup: cn=orphans,cn=hostgroups,dc=example,dc=com
autoMemberGroupingAttr: member:dn
```

regex entry #1

```
dn: cn=webservers,cn=Hostgroups,cn=Auto Membership
Plugin,cn=plugins,cn=config
objectclass: autoMemberRegexRule
description: Group placement for web servers
cn: webservers
autoMemberTargetGroup: cn=webservers,cn=hostgroups,dc=example,dc=com
autoMemberInclusiveRegex: fqdn=^www[0-9]+\..example\..com
autoMemberInclusiveRegex: fqdn=^web[0-9]+\..example\..com
autoMemberExclusiveRegex: fqdn=^www13\..example\..com
autoMemberExclusiveRegex: fqdn=^web13\..example\..com
```

regex entry #2

```
dn: cn=mailservers,cn=Hostgroups,cn=Auto Membership
Plugin,cn=plugins,cn=config
objectclass: autoMemberRegexRule
description: Group placement for mailservers
cn: mailservers
autoMemberTargetGroup: cn=mailservers,cn=hostgroups,dc=example,dc=com
autoMemberInclusiveRegex: fqdn=^mail[0-9]+\.\example\.\com
autoMemberInclusiveRegex: fqdn=^smtp[0-9]+\.\example\.\com
autoMemberExclusiveRegex: fqdn=^mail13\.\example\.\com
autoMemberExclusiveRegex: fqdn=^smtp13\.\example\.\com
```

Example 6.5. Windows User Group

The basic users group shown in [Section 6.1.5.1.1, “The Automembership Configuration Entry”](#) uses the ***posixAccount*** attribute to identify all new users. All new users created within Directory Server are created with the ***posixAccount*** attribute, so that is a safe catch-all for new Directory Server users. However, when user accounts are synced over from the Windows domain to the Directory Server, the Windows user accounts are created *without* the ***posixAccount*** attribute.

Windows users are identified by the ***ntUser*** attribute. The basic, all-users group rule can be modified to target Windows users specifically, which can then be added to the default all-users group or to a Windows-specific group.

```
dn: cn=Windows Users,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberDefinition
autoMemberScope: dc=example,dc=com
autoMemberFilter: objectclass=ntUser
autoMemberDefaultGroup: cn=Windows Users,cn=groups,dc=example,dc=com
autoMemberGroupingAttr: member:dn
```

Example 6.6. User Groups by Employee Type

The Auto Membership Plug-in can work on custom attributes, which can be useful for entries which are managed by other applications. For example, a human resources application may create and then reference users based on the employee type, in a custom ***employeeType*** attribute.

Much like [Example 6.4, “Host Groups by IP Address”](#), the user type rule uses two regular expression filters to sort full time and temporary employees, only this example uses an explicit value rather than a true regular expression. For other attributes, it may be more appropriate to use a regular expression, like basing the filter on an employee ID number range.

configuration entry

```
dn: cn=Employee groups,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberDefinition
cn: Hostgroups
autoMemberScope: ou=employees,ou=people,dc=example,dc=com
autoMemberFilter: objectclass=inetorgperson
autoMemberDefaultGroup: cn=general,cn=employee
groups,ou=groups,dc=example,dc=com
autoMemberGroupingAttr: member:dn
```



```

regex entry #1
dn: cn=full time,cn=Employee groups,cn=Auto Membership
Plugin,cn=plugins,cn=config
objectclass: autoMemberRegexRule
description: Group for full time employees
cn: full time
autoMemberTargetGroup: cn=full time,cn=employee
groups,ou=groups,dc=example,dc=com
autoMemberInclusiveRegex: employeeType=full

regex entry #2
dn: cn=temporary,cn=Employee groups,cn=Auto Membership
Plugin,cn=plugins,cn=config
objectclass: autoMemberRegexRule
description: Group placement for interns, contractors, and seasonal
employees
cn: temporary
autoMemberTargetGroup: cn=temporary,cn=employee
groups,ou=groups,dc=example,dc=com
autoMemberInclusiveRegex: employeeType=intern
autoMemberInclusiveRegex: employeeType=contractor
autoMemberInclusiveRegex: employeeType=seasonal

```

6.1.5.3. Creating Automembership Definitions

1. If necessary, enable the Auto Membership Plug-in.

```

ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=Auto Membership Plugin,cn=plugins,cn=config
changetype: replace
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
-

```

2. Create the new plug-in instance below the **cn=Auto Membership Plugin,cn=plugins,cn=config** container entry. This entry must belong to the **autoMemberDefinition** object class.

```

ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=Example Automember Definition,cn=Auto Membership
Plugin,cn=plugins,cn=config
objectclass: autoMemberDefinition

```

The required attributes for the definition are listed in [Table 6.3, “Automember Definition Attributes”](#).

3. Set the scope and filter for the definition. This is used for the initial search for matching entries.

For example, for new entries added to the **ou=People** subtree and containing the **ntUser** attribute:

```
autoMemberScope: ou=People,dc=example,dc=com
autoMemberFilter: objectclass=ntUser
```

4. Set the group to which to add matching entries (as the default or fallback group) and the format of the member entries for that group type.

```
autoMemberDefaultGroup: cn=windows-group,cn=groups,dc=example,dc=com
autoMemberGroupingAttr: member:dn
```

5. *Optional.* Create inclusive or exclusive regular expression filters and set a group to use for entries matching those filters.

The attributes for the regular expression condition are listed in [Table 6.4, “Regular Expression Condition Attributes”](#).

Regular expression conditions are added as children of the automember definition. These conditions must belong to the **autoMemberRegexRule** object class.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=Example Regex,cn=Example Automember Definition,cn=Auto
Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberRegexRule
```

Then add the target group name and any inclusive or exclusive regular expressions. Both include and exclude conditions can be used, and multiple expressions of both types can be used.

```
autoMemberTargetGroup: cn=webserver,cn=hostgroups,dc=example,dc=com
autoMemberInclusiveRegex: fqdn=^www\.\web[0-9]+\.\example\.
```

If a new entry matches a regular expression condition, it is added to that group *instead of* the default group set in the automember definition.

6. Restart the Directory Server to load the modified new plug-in instance.

```
service dirsrv restart instance_name
```

6.1.5.4. Updating Existing Entries for Automembership Definitions

The Auto Member Plug-in only runs when new entries are added to the directory. The plug-in ignores existing entries or entries which are edited to match an automembership rule.

There is a directory task operation which can be run to check existing entries against automembership rules and then update group membership accordingly. This task (**cn=automember rebuild membership**) requires three elements to run, based on LDAP search parameters to identify which existing entries to process:

- The search filter

- The search scope
- The base DN from which to begin the search

The specific task run also needs a name.

The task entry can be created using **ldapmodify**; when the task completes, the entry is automatically removed. For example:

```
[root@server ~]# ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=my rebuild task, cn=automember rebuild
membership,cn=tasks,cn=config
objectClass: top
objectClass: extensibleObject
cn: my rebuild task
basedn: dc=example,dc=com
filter: (uid=*)
scope: sub
```

6.1.5.5. Testing Automembership Definitions

Because each instance of the Auto Member Plug-in is a set of related-but-separate entries for the definition and regular expression, it can be difficult to see exactly how users are going to be mapped to groups. This becomes even more difficult when there are multiple rules which target different subsets of users.

There are two dry-run tasks which can be useful to determine whether all of the different Auto Member Plug-in definitions are assigning groups properly as designed.

Testing with Existing Entries

cn=automember export updates runs against *existing entries* in the directory and exports the results of what users would have been added to what groups, based on the rules. This is useful for testing existing rules against existing users to see how your real deployment are performing.

This task requires the same information as the **cn=automember rebuild membership** task — the base DN to search, search filter, and search scope — and has an additional parameter to specify an export LDIF file to record the proposed entry updates.

```
[root@server ~]# ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=test export, cn=automember export updates,cn=tasks,cn=config
objectClass: top
objectClass: extensibleObject
cn: test export
basedn: dc=example,dc=com
filter: (uid=*)
scope: sub
ldif: /tmp/automember-updates.ldif
```

Testing with an Import LDIF

cn=automember map updates takes an *import LDIF* of new users and then runs the new users against the current automembership rules. This can be very useful for testing a new rule, before applying it to (real) new or existing user entries.

This is called a map task because it maps or relates changes for proposed new entries to the existing rules.

This task only requires two attributes: the location of the input LDIF (which must contain at least some user entries) and an output LDIF file to which to write the proposed entry updates. Both the input and output LDIF files are absolute paths on the local machine.

For example:

```
[root@server ~]# ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

```
dn: cn=test mapping, cn=automember map updates,cn=tasks,cn=config
objectClass: top
objectClass: extensibleObject
cn: test mapping
ldif_in: /tmp/entries.ldif
ldif_out: /tmp/automember-updates.ldif
```

6.2. USING ROLES

Roles are an entry grouping mechanism that unify the static and dynamic groups described in the previous sections. Roles are designed to be more efficient and easier to use for applications. For example, an application can get the list of roles of which an entry is a member by querying the entry itself, rather than selecting a group and browsing the members list of several groups.

6.2.1. About Roles

Red Hat has two kinds of groups. *Static groups* have a finite and defined list of members. *Dynamic groups* used filters to recognize which entries are members of the group, so the group membership is constantly changed as the entries which match the group filter change. (Both kinds of groups are described in [Section 6.1, “Using Groups”](#).)

Roles are a sort of hybrid group, behaving as both a static and a dynamic group. With a group, entries are added to a group entry as members. With a role, the role attribute is added to an entry and then that attribute is used to identify members in the role entry automatically.

Role *members* are entries that possess the role. Members can be specified either explicitly or dynamically. How role membership is specified depends upon the type of role. Directory Server supports three types of roles:

- *Managed roles* have an explicit enumerated list of members.
- *Filtered roles* are assigned entries to the role depending upon the attribute contained by each entry, specified in an LDAP filter. Entries that match the filter possess the role.
- *Nested roles* are roles that contain other roles.

Managed roles can do everything that can normally be done with static groups. The role members can be filtered using filtered roles, similarly to the filtering with dynamic groups. Roles are easier to use than groups, more flexible in their implementation, and reduce client complexity.

When a role is created, determine whether a user can add themselves or remove themselves from the role. See [Section 6.2.10, “Using Roles Securely”](#) for more information about roles and access control.



NOTE

Evaluating roles is more resource-intensive for the Directory Server than evaluating groups because the server does the work for the client application. With roles, the client application can check role membership by searching for the **nsRole** attribute. The **nsRole** attribute is a computed attribute, which identifies to which roles an entry belongs; the **nsRole** attribute is not stored with the entry itself. From the client application point of view, the method for checking membership is uniform and is performed on the server side.

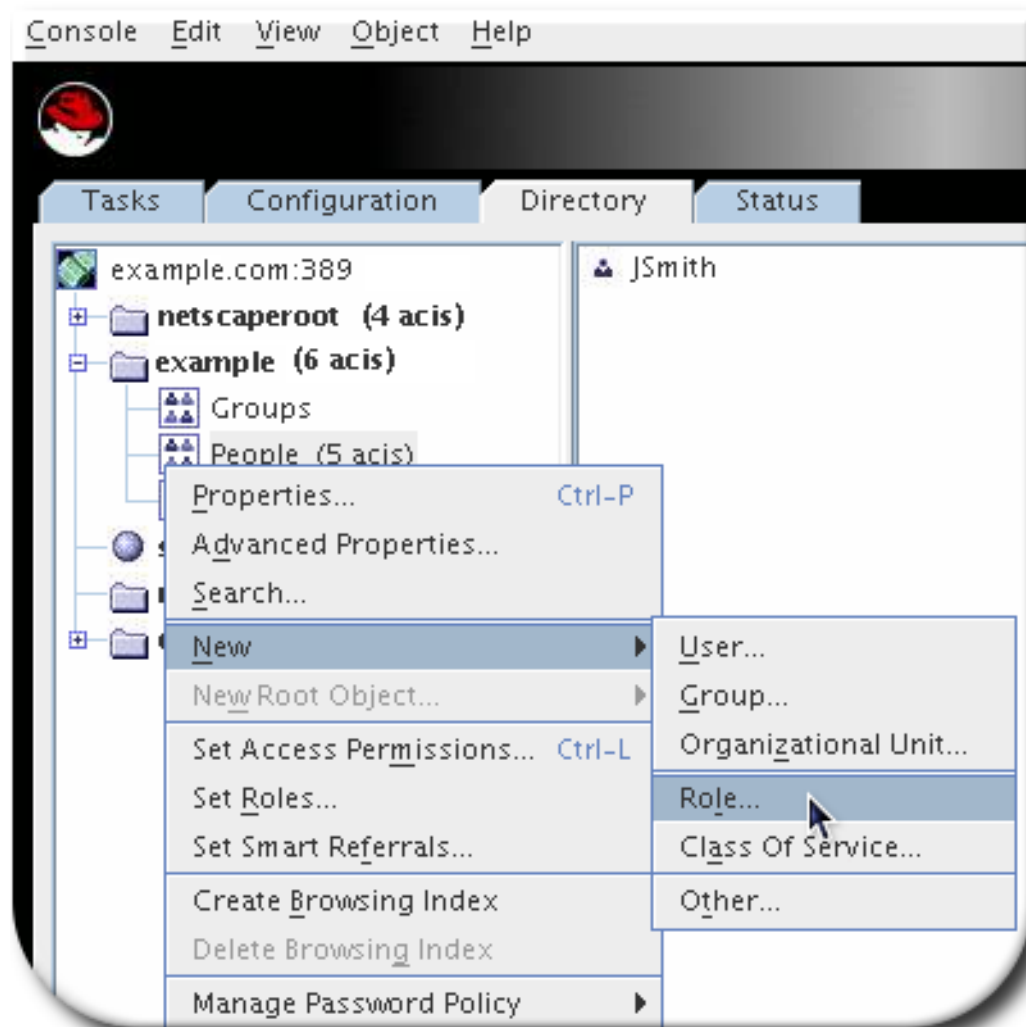
Considerations for using roles are covered in the *Directory Server Deployment Guide*.

6.2.2. Creating a Managed Role

Managed roles have an explicit enumerated list of members. Managed roles are added to entries by adding the **nsRoleDN** attribute to the entry.

6.2.2.1. Creating a Managed Role in the Console

1. In the Directory Server Console, select the **Directory** tab.
2. Browse the tree in the left navigation pane, and select the parent entry for the new role.
3. Go to the **Object** menu, and select **New > Role**.



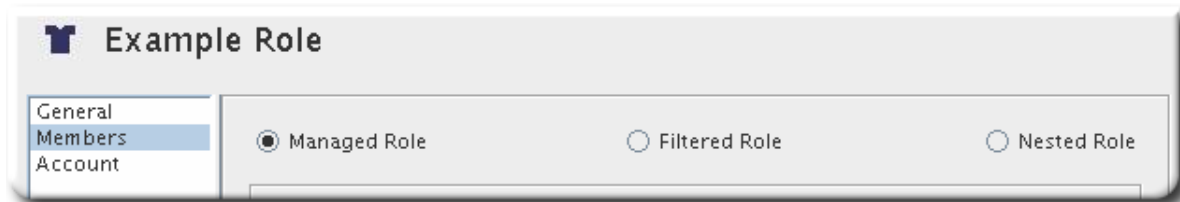
Alternatively, right-click the entry and select **New > Role**.

- Click **General** in the left pane. Type a name for the new role in the **Role Name** field. The role name is required.

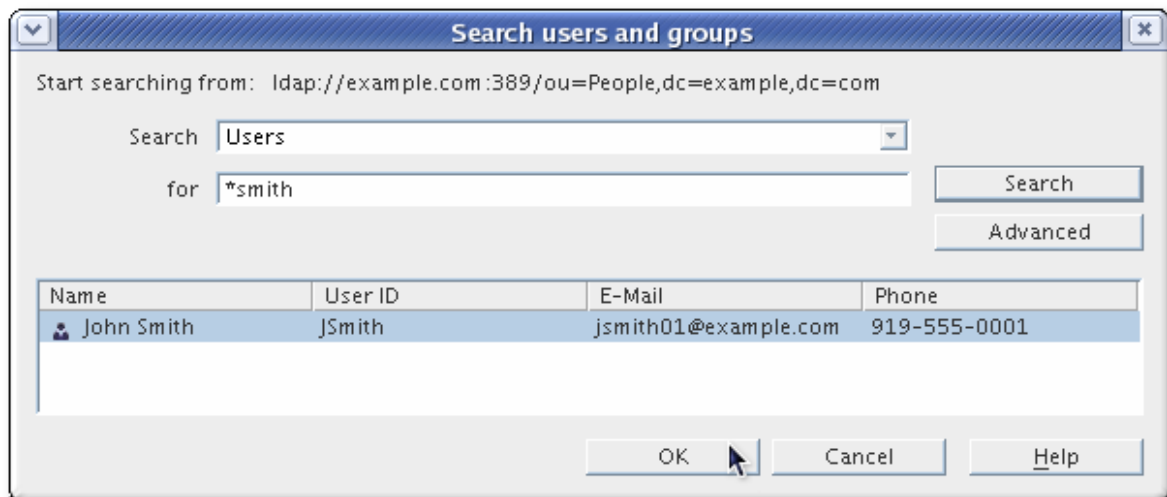


- Enter a description of the new role in the **Description** field.
- Click **Members** in the left pane.

7. In the right pane, select **Managed Role**. Click **Add** to add new entries to the list of members.



8. In the **Search** drop-down list, select **Users** from the **Search** drop-down list, then click **Search**. Select one of the entries returned, and click **OK**.



9. After adding all of the entries, click **OK**.

6.2.2.2. Creating Managed Roles through the Command Line

Roles inherit from the **ldapsubentry** object class, which is defined in the ITU X.509 standard. In addition, each managed role requires two object classes that inherit from the **nsRoleDefinition** object class:

- **nsSimpleRoleDefinition**
- **nsManagedRoleDefinition**

A managed role also allows an optional **description** attribute.

Members of a managed role have the **nsRoleDN** attribute in their entry.

This example creates a role which can be assigned to the marketing department.

1. Use **ldapmodify** with the **-a** option to add the managed role entry. The new entry must contain the **nsManagedRoleDefinition** object class, which in turn inherits from the **LdapSubEntry**, **nsRoleDefinition**, and **nsSimpleRoleDefinition** object classes.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

```
dn: cn=Marketing,ou=people,dc=example,dc=com
objectclass: top
```

```
objectclass: LdapSubEntry
objectclass: nsRoleDefinition
objectclass: nsSimpleRoleDefinition
objectclass: nsManagedRoleDefinition
cn: Marketing
description: managed role for marketing staff
```

2. Assign the role to the marketing staff members, one by one, using **ldapmodify**:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=Bob,ou=people,dc=example,dc=com
changetype: modify
add: nsRoleDN
nsRoleDN: cn=Marketing,ou=people,dc=example,dc=com
```

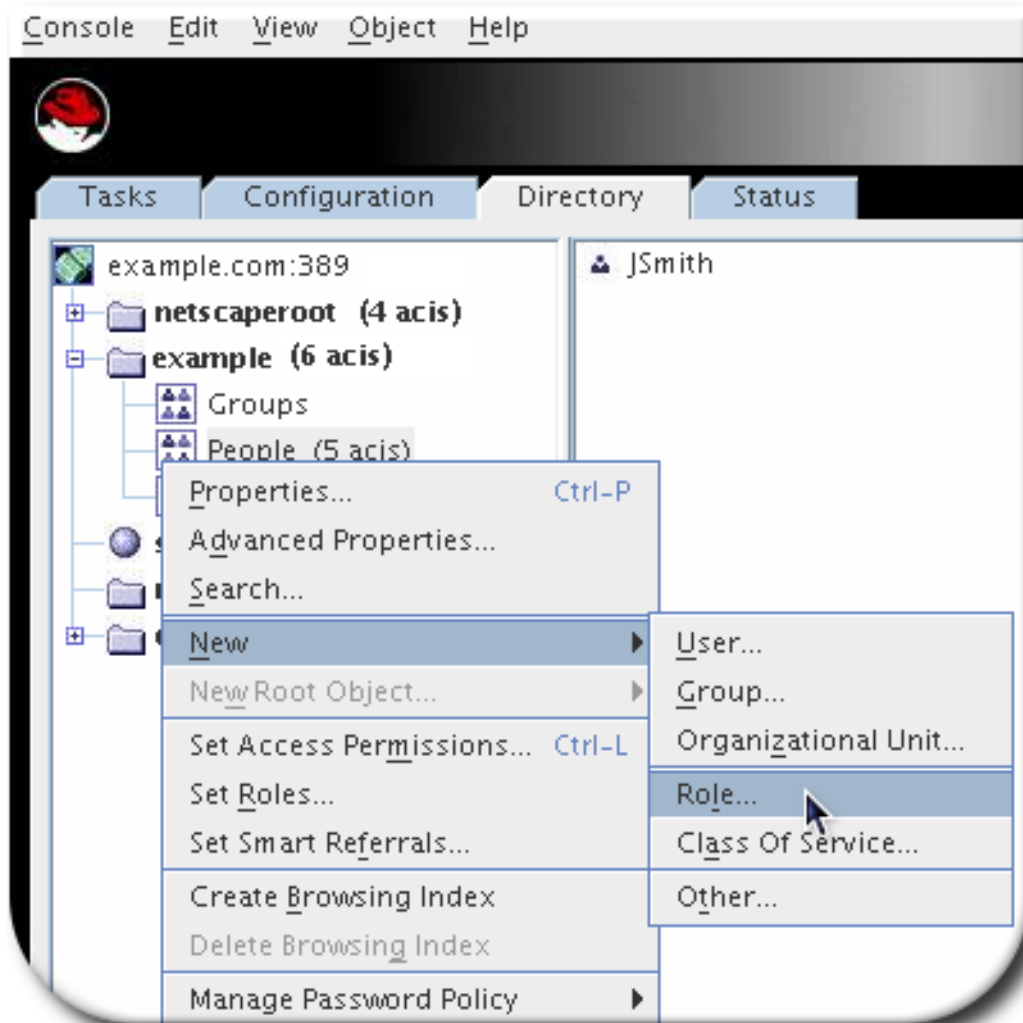
The **nsRoleDN** attribute in the entry indicates that the entry is a member of a managed role, **cn=Marketing,ou=people,dc=example,dc=com**.

6.2.3. Creating a Filtered Role

Entries are assigned to a filtered role depending whether the entry possesses a specific attribute defined in the role. The role definition specifies an LDAP filter for the target attributes. Entries that match the filter possess (are members of) the role.

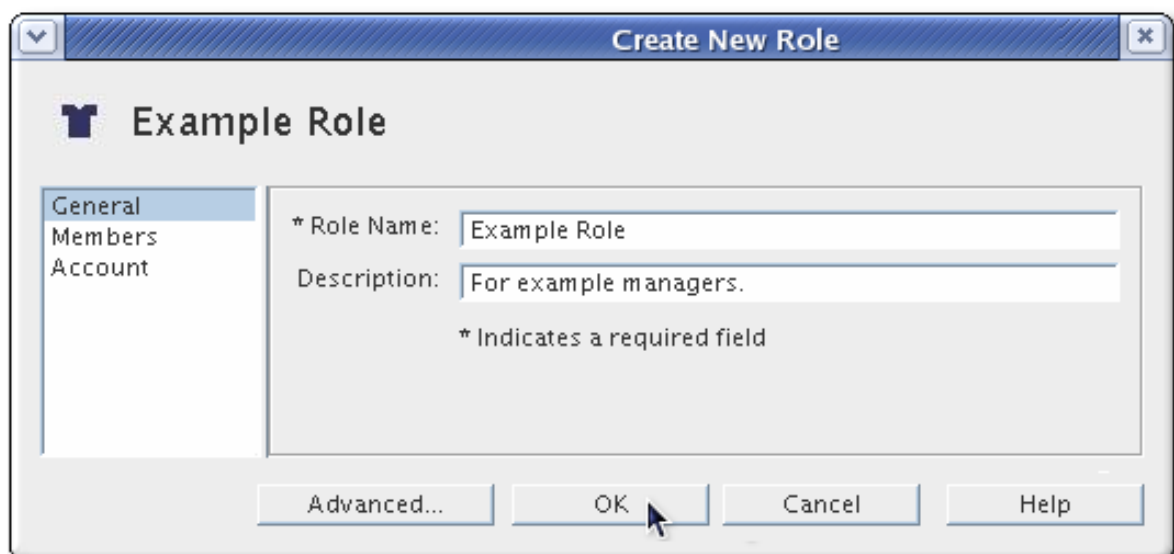
6.2.3.1. Creating a Filtered Role in the Console

1. In the Directory Server Console, select the **Directory** tab.
2. Browse the tree in the left navigation pane, and select the parent entry for the new role.
3. Go to the **Object** menu, and select **New > Role**.



Alternatively, right-click the entry and select **New > Role**.

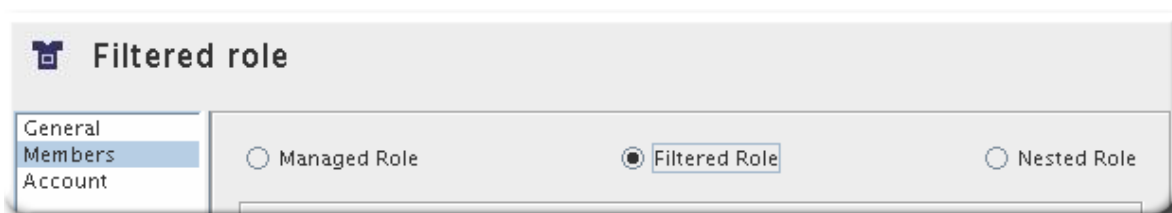
- Click **General** in the left pane. Type a name for the new role in the **Role Name** field. The role name is required.



- Enter a description of the new role in the **Description** field.
- Click **Members** in the left pane.

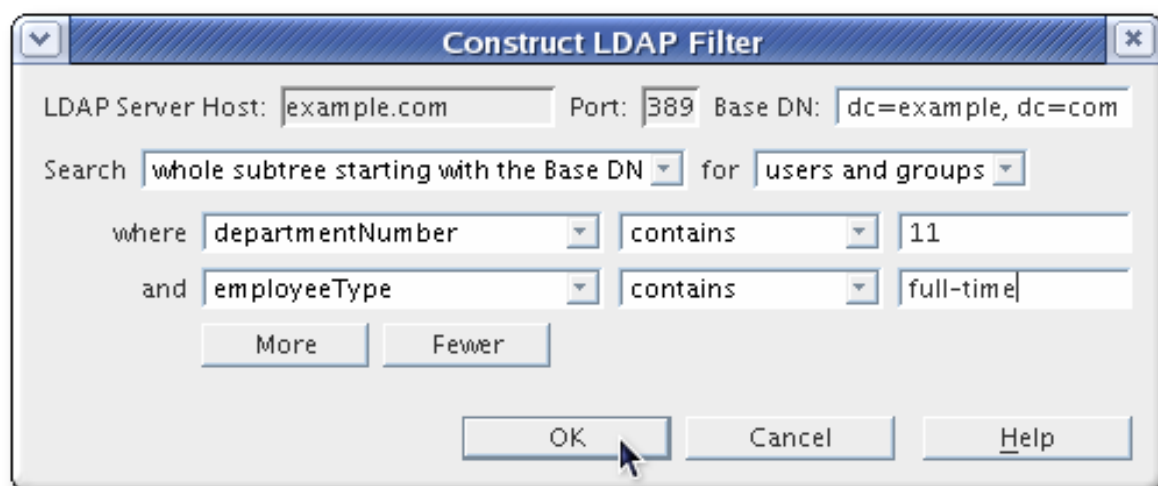
A search dialog box appears briefly.

- In the right pane, select **Filtered Role**.

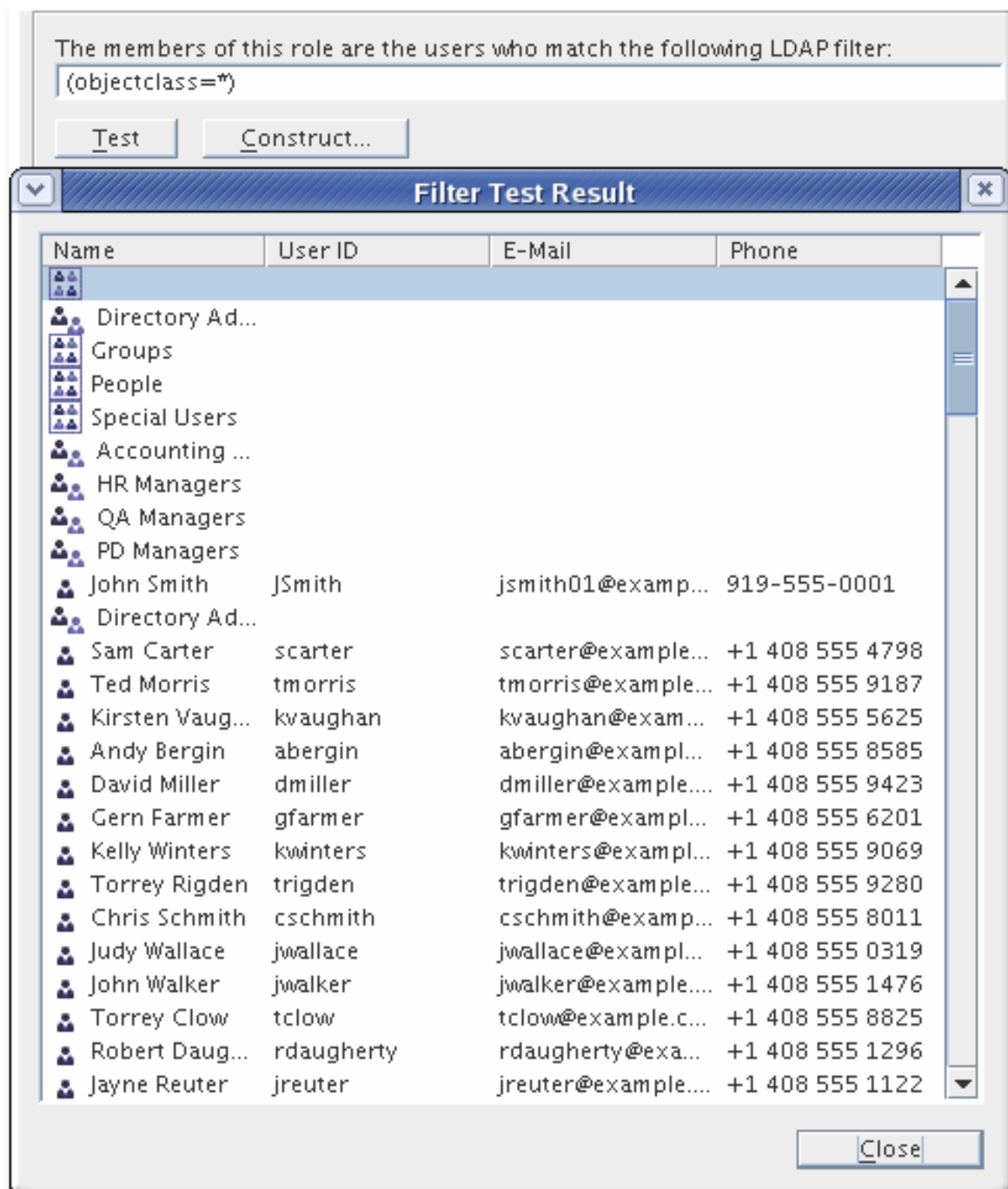


- Enter an LDAP filter in the text field, or click **Construct** to be guided through the construction of an LDAP filter.

The **Construct** opens the standard LDAP URL construction dialog. Ignore the fields for **LDAP Server Host**, **Port**, **Base DN**, and **Search** (since the search scope cannot be set filtered role definitions).



- Select the types of entries to filter from the **For** drop-down list. The entries can be users, groups, or both.
 - Select an attribute from the **Where** drop-down list. The two fields following it refine the search by selecting one of the qualifiers from the drop-down list, such as **contains**, **does not contain**, **is**, or **is not**. Enter an attribute value in the text box. To add additional filters, click **More**. To remove unnecessary filters, click **Fewer**.
- Click **Test** to try the filter.



10. Click **OK**.

6.2.3.2. Creating a Filtered Role through the Command Line

Roles inherit from the **ldapsubentry** object class, which is defined in the ITU X.509 standard. In addition, each filtered role requires two object classes that inherit from the **nsRoleDefinition** object class:

- **nsComplexRoleDefinition**
- **nsFilteredRoleDefinition**

A filtered role entry also requires the **nsRoleFilter** attribute to define the LDAP filter to determine role members. Optionally, the role can take a **description** attribute.

Members of a filtered role are entries that match the filter specified in the ***nsRoleFilter*** attribute.

This example creates a filtered role which is applied to all sales managers.

1. Run **ldapmodify** with the **-a** option to add a new entry:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

2. Create the filtered role entry.

The role entry has the **nsFilteredRoleDefinition** object class, which inherits from the **LdapSubEntry**, **nsRoleDefinition**, and **nsComplexRoleDefinition** object classes.

The **nsRoleFilter** attribute sets a filter for **o** (organization) attributes that contain a value of **sales managers**.

```
dn: cn=SalesManagerFilter,ou=people,dc=example,dc=com
changetype: add
objectclass: top
objectclass: LDAPsubentry
objectclass: nsRoleDefinition
objectclass: nsComplexRoleDefinition
objectclass: nsFilteredRoleDefinition
cn: SalesManagerFilter
nsRoleFilter: o=sales managers
Description: filtered role for sales managers
```

The following entry matches the filter (possesses the **o** attribute with the value **sales managers**), and, therefore, it is a member of this filtered role automatically:

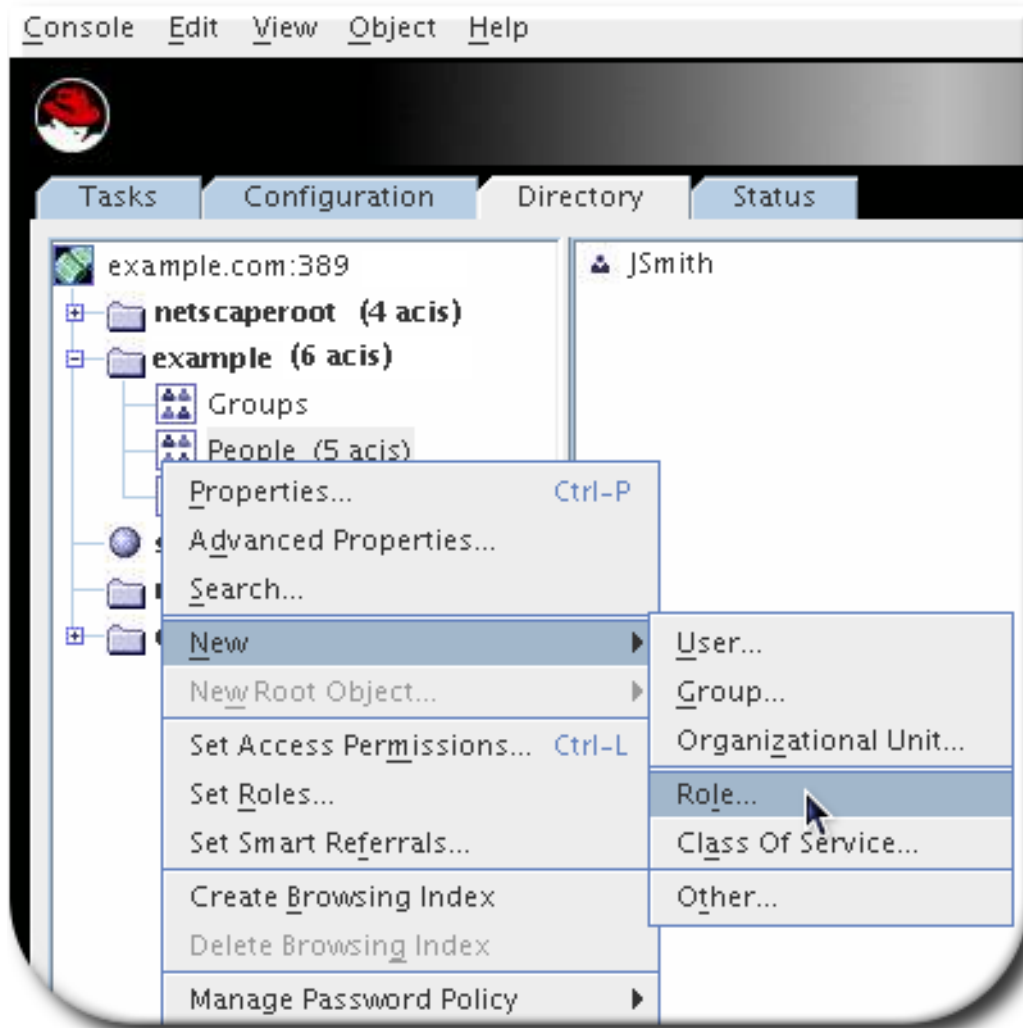
```
dn: cn=Pat Smith,ou=people,dc=example,dc=com
objectclass: person
cn: Pat
sn: Smith
userPassword: secret
o: sales managers
```

6.2.4. Creating a Nested Role

Nested roles are roles that contain other roles. Before it is possible to create a nested role, another role must exist. When a nested role is created, the Console displays a list of the roles available for nesting. The roles nested within the nested role are specified using the ***nsRoleDN*** attribute.

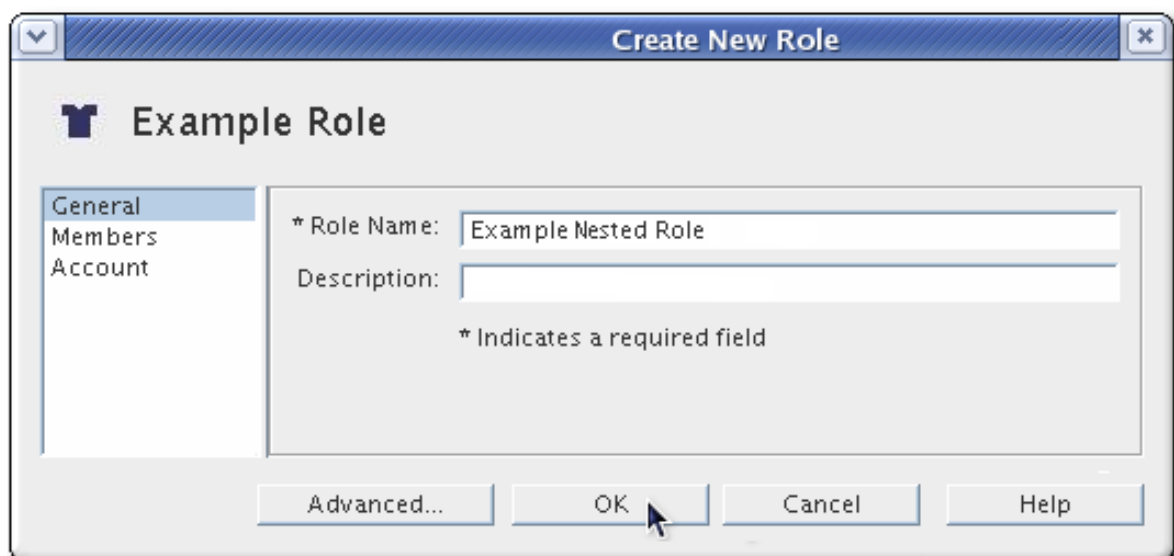
6.2.4.1. Creating a Nested Role in the Console

1. In the Directory Server Console, select the **Directory** tab.
2. Browse the tree in the left navigation pane, and select the parent entry for the new role.
3. Go to the **Object** menu, and select **New > Role**.

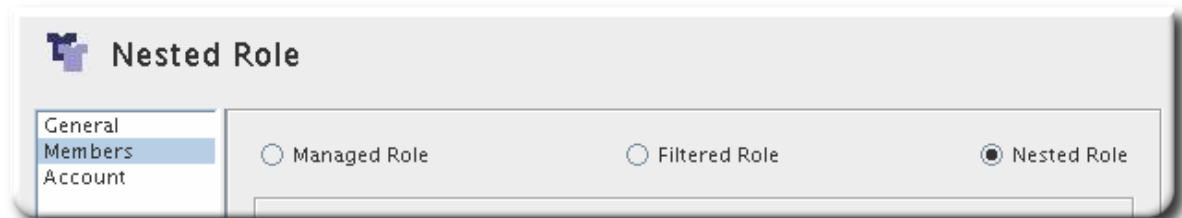


Alternatively, right-click the entry and select **New > Role**.

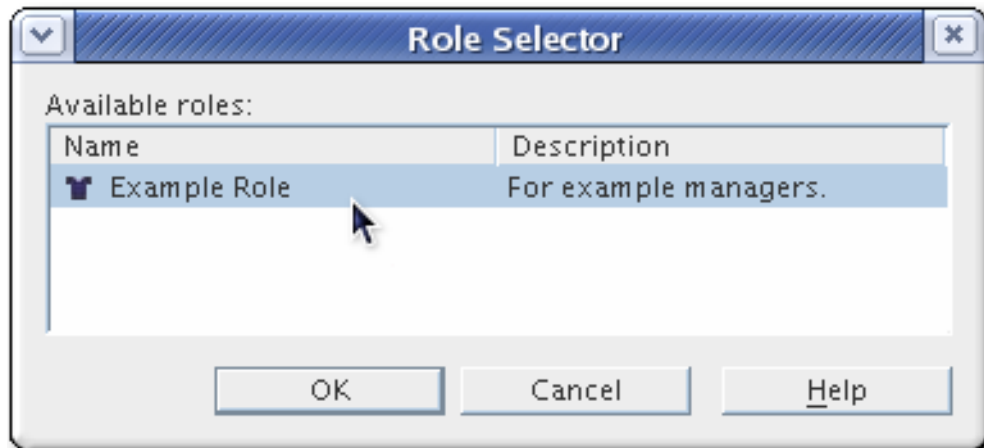
- Click **General** in the left pane. Type a name for the new role in the **Role Name** field. The role name is required.



- Click **Members** in the left pane.
- In the right pane, select **Nested Role**.



7. Click **Add** to add roles to the list. The members of the nested role are members of other existing roles.
8. Select a role from the **Available roles** list, and click **OK**.



6.2.4.2. Creating Nested Role through the Command Line

Roles inherit from the **ldapsubentry** object class, which is defined in the ITU X.509 standard. In addition, each nested role requires two object classes that inherit from the **nsRoleDefinition** object class:

- nsComplexRoleDefinition
- nsNestedRoleDefinition

A nested role entry also requires the **nsRoleDN** attribute to identify the roles to nest within the container role. Optionally, the role can take a **description** attribute.

Members of a nested role are members of the roles specified in the **nsRoleDN** attributes of the nested role definition entry.

This example creates a single role out of the managed marketing role and filtered sales manager role.

1. Run **ldapmodify** with the **-a** option to add a new entry:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

2. Create the nested role entry. The nested role has four object classes:

- nsNestedRoleDefinition
- LDAPsubentry (inherited)

- **nsRoleDefinition** (inherited)
- **nsComplexRoleDefinition** (inherited)

The **nsRoleDN** attributes contain the DN's for both the marketing managed role and the sales managers filtered role.

```
dn: cn=MarketingSales,ou=people,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: nsRoleDefinition
objectclass: nsComplexRoleDefinition
objectclass: nsNestedRoleDefinition
cn: MarketingSales
nsRoleDN: cn=SalesManagerFilter,ou=people,dc=example,dc=com
nsRoleDN: cn=Marketing,ou=people,dc=example,dc=com
```

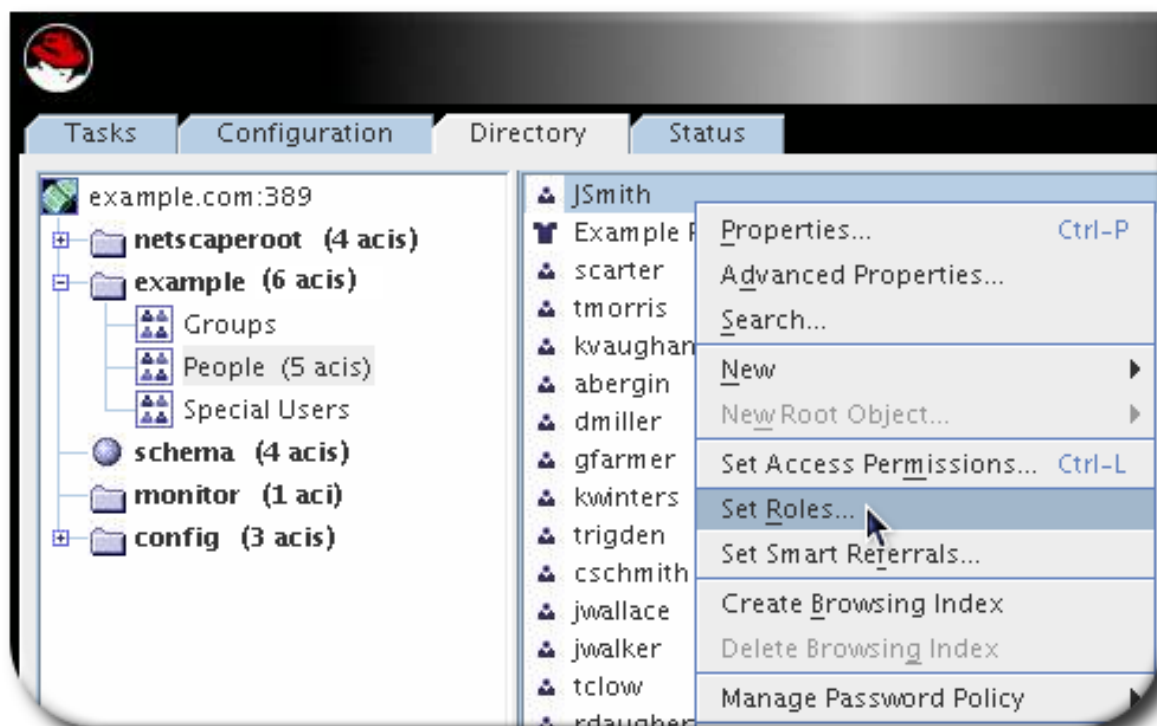
Both of the users in the previous examples, Bob and Pat, are members of this new nested role.

6.2.5. Editing and Assigning Roles to an Entry

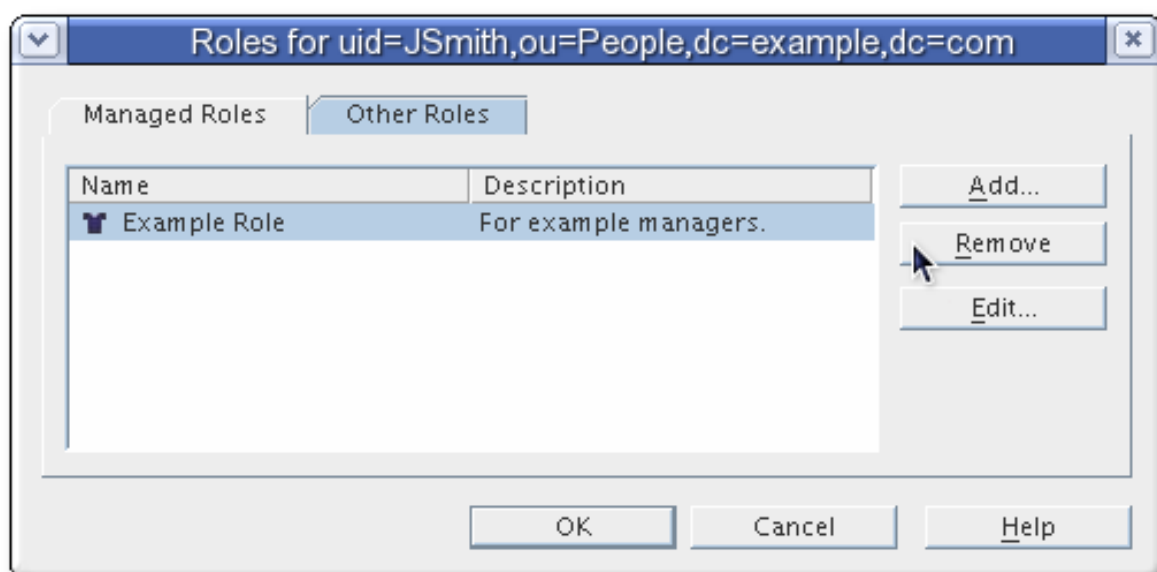
The entries which belong to a role are assigned on the role entry itself. For managed roles, user entries are added explicitly; for filtered roles, they are added through the results of an LDAP filter.

User entries are assigned to the role through the command line by editing the role entry, either by adding the entry as a member or adjusting the filter so it is included. In the Directory Server Console, however, there is a shortcut to add entries to a role by apparently editing the desired user entry (but, functionally, this really edits the role entry).

1. Select the **Directory** tab.
2. In the left navigation pane, browse the tree, and select the entry for which to view or edit a role.
3. Select **Set Roles** from the **Object** menu.



4. Select the **Managed Roles** tab to display the managed roles to which this entry belongs.



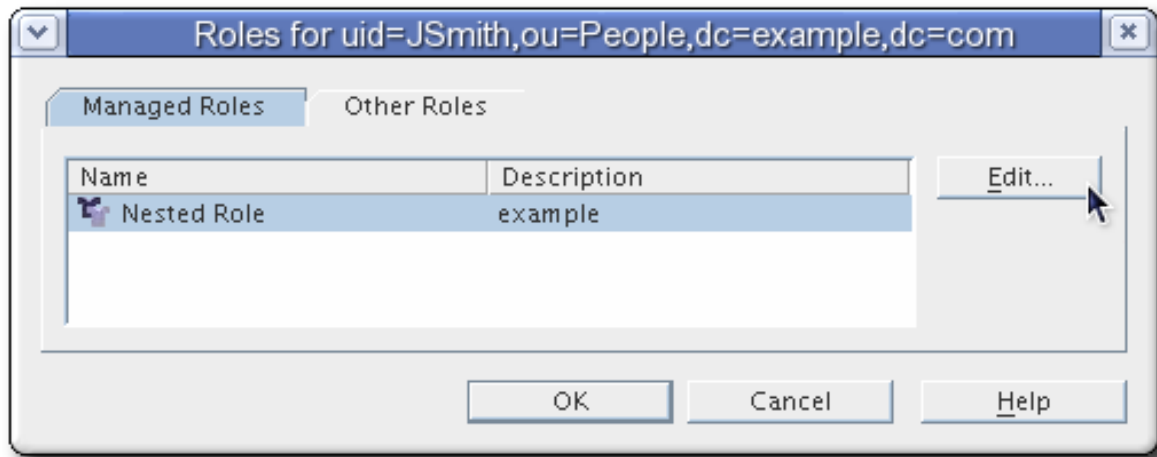
5. To add a new managed role, click **Add**, and select an available role from the **Role Selector** window.



NOTE

The configuration for a managed role associated with an entry can be edited by clicking the **Edit** button. The **Edit Entry** dialog box opens, and the general information or members for the role can be changed.

6. Select the **Other Roles** tab to view the filtered or nested roles to which this entry belongs.



Click **Edit** to make changes to any filtered or nested roles associated with the entry.

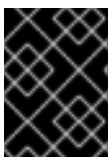
6.2.6. Viewing Roles for an Entry through the Command Line

Role assignments are always visible for an entry when it is displayed in the Directory Server Console. Role assignments are not returned automatically through the command line, however.

The `nsRole` attribute is an operational attribute. In LDAP, operational attributes must be requested explicitly in the search attributes list; they are not returned by default with the regular attributes in the schema of the entry. For example, this **ldapsearch** command returns the list of roles of which **uid=scarter** is a member, in addition to the regular attributes for the entry:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -b "dc=example,dc=com" -s sub -x "(uid=scarter)" \*
nsRole
```

```
dn: uid=scarter,ou=people,dc=example,dc=com
objectClass: inetorgperson
objectClass: top
objectClass: person
objectClass: organizationalPerson
uid: scarter
cn: Sam Carter
sn: Carter
givenName: Sam
mail: scarter@example.com
userPassword: {SSHA}6BE31mhTfcYyIQF60kWlnEL8sIvPZ59hvFTRKw==
manager: uid=lbrown,ou=people,dc=example,dc=com
nsRole: cn=Role for Managers,dc=example,dc=com
nsRole: cn=Role for Accounting,dc=example,dc=com
```



IMPORTANT

Be sure to use the ***nsRole*** attribute, not the ***nsRoleDN*** attribute, to evaluate role membership.

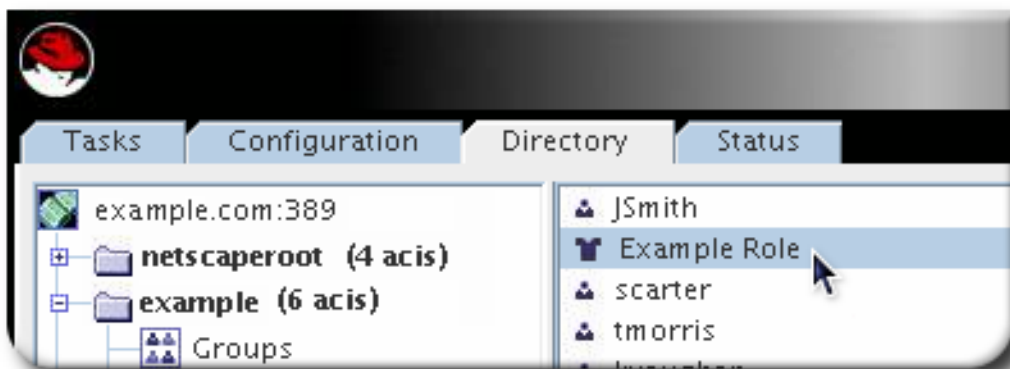
6.2.7. Making a Role Inactive or Active

The concept of activating/inactivating roles allows entire groups of entries to be activated or inactivated in just one operation. That is, the members of a role can be temporarily disabled by inactivating the role to which they belong.

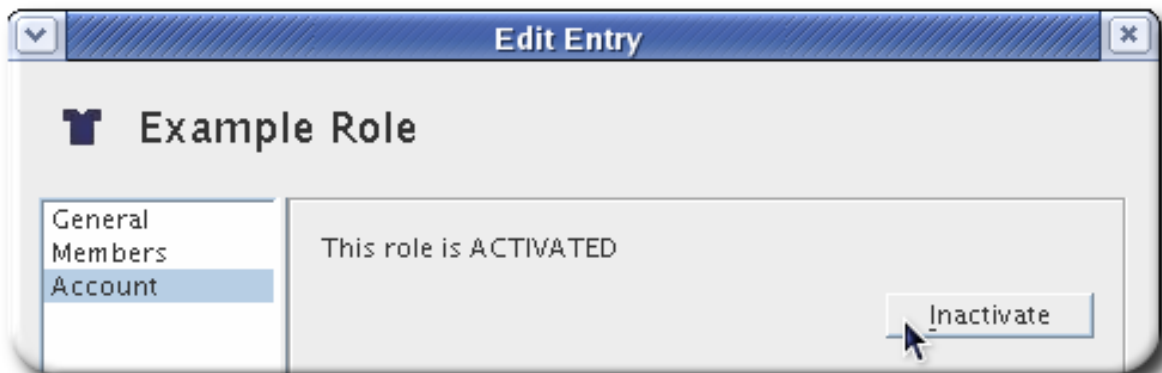
When a role is inactivated, it does not mean that the user cannot bind to the server using that role entry. The meaning of an inactivated role is that the user cannot bind to the server using any of the entries that belong to that role; the entries that belong to an inactivated role will have the **nsAccountLock** attribute set to **true**.

Members of a role can be temporarily disabled by inactivating the role to which they belong. Inactivating a role inactivates the entries possessed by the role, not the role itself.

1. Select the **Directory** tab.
2. Browse the navigation tree in the left pane to locate the base DN for the role. Roles appear in the right pane with other entries.



3. Double-click the role, open the **Account** tab, and click the **Inactivate** button.



Alternatively, select the role. Right-click the role and select **Inactivate** from the menu.

The role is inactivated.

To reactivate a disabled role, re-open the role configuration or open the **Object** menu, and select **Activate**. All of the members of the role are re-enabled.

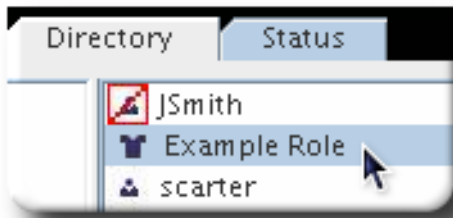
6.2.8. Viewing the Activation Status for Entries

When a nested role is inactivated, a user cannot bind to the server if it is a member of any role within the nested role. All the entries that belong to a role that directly or indirectly are members of the nested role have **nsAccountLock** set to **true**. There can be several layers of nested roles, and inactivating a

nested role at any point in the nesting will inactivate all roles and users beneath it.

The Directory Server Console automatically shows the active or inactive status of entries.

To see the inactivated entries, select **Inactivation State** from the **View** menu. Members of an inactivated role have a red slash through them. For example, John Smith is a member of the inactive Example Role.



The ***nsAccountLock*** attribute is an operational attribute and must be explicitly requested in the search command in the list of search attributes. For example:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -b "dc=example,dc=com" -s sub -x "(uid=scarter)"
nsAccountLock
```

6.2.9. About Deleting Roles

Deleting a role deletes the role entry but does not delete the ***nsRoleDN*** attribute for each role member. To delete the ***nsRoleDN*** attribute for each role member, enable the Referential Integrity plug-in, and configure it to manage the ***nsRoleDN*** attribute. For more information on the Referential Integrity plug-in, see [Section 3.6, “Maintaining Referential Integrity”](#).

6.2.10. Using Roles Securely

Not every role is suitable for use in a security context. When creating a new role, consider how easily the role can be assigned to and removed from an entry. Sometimes it is appropriate for users to be able to add or remove themselves easily from a role. For example, if there is an interest group role called **Mountain Biking**, interested users should be able to add themselves or remove themselves easily.

However, it is inappropriate to have such open roles for some security situations. One potential security risk is inactivating user accounts by inactivating roles. Inactive roles have special ACIs defined for their suffix. If an administrator allows users to add and remove themselves from roles freely, then in some circumstance, they may be able to remove themselves from an inactive role to prevent their accounts from being locked.

For example, user A possesses the managed role, **MR**. The **MR** role has been locked using account inactivation. This means that user A cannot bind to the server because the ***nsAccountLock*** attribute is computed as **true** for that user. However, if user A was already bound to Directory Server and noticed that he is now locked through the MR role, the user can remove the ***nsRoleDN*** attribute from his entry and unlock himself if there are no ACIs preventing him.

To prevent users from removing the ***nsRoleDN*** attribute, use the following ACIs depending upon the type of role being used.

- *Managed roles.* For entries that are members of a managed role, use the following ACI to prevent users from unlocking themselves by removing the appropriate ***nsRoleDN***:

```
aci: (targetattr="nsRoleDN") (targetattrfilters= add=nsRoleDN:(!  
(nsRoleDN=cn=AdministratorRole,dc=example,dc=com)), del=nsRoleDN:(!  
(nsRoleDN=cn=nsManagedDisabledRole,dc=example,dc=com)))  
(version3.0;acl "allow mod of nsRoleDN by self but not to critical  
values"; allow(write) userdn=ldap:///self;)
```

- *Filtered roles.* The attributes that are part of the filter should be protected so that the user cannot relinquish the filtered role by modifying an attribute. The user should not be allowed to add, delete, or modify the attribute used by the filtered role. If the value of the filter attribute is computed, then all attributes that can modify the value of the filter attribute should be protected in the same way.
- *Nested roles.* A nested role is comprised of filtered and managed roles, so both ACIs should be considered for modifying the attributes (*nsRoleDN* or something else) of the roles that comprise the nested role.

For more information about account inactivation, see [Section 14.11, “Manually Inactivating Users and Roles”](#).

6.3. AUTOMATICALLY CREATING DUAL ENTRIES

Some clients and integration with Red Hat Directory Server require dual entries. For example, both Posix systems typically have a group for each user. The Directory Server's *Managed Entries Plug-in* creates a new managed entry, with accurate and specific values for attributes, automatically whenever an appropriate origin entry is created.

6.3.1. About Managed Entries

The basic idea behind the Managed Entries Plug-in is that there are situations when Entry A is created and there should automatically be an Entry B with related attribute values. For example, when a Posix user (**posixAccount** entry) is created, a corresponding group entry (**posixGroup** entry) should also be created. An instance of the Managed Entries Plug-in identifies what entry (the *origin entry*) triggers the plug-in to automatically generate a new entry (the *managed entry*).

The plug-in works within a defined scope of the directory tree, so only entries within that subtree and that match the given search filter trigger a Managed Entries operation.

Much like configuring a class of service, a managed entry is configured through two entries:

- A definition entry, that identifies the scope of the plug-in instance and the template to use
- A template entry, that models what the final managed entry will look like

6.3.1.1. About the Instance Definition Entry

As with the Linked Attributes and DNA Plug-ins, the Managed Entries Plug-in has a container entry in **cn=plugins,cn=config**, and each unique configuration instance of the plug-in has a definition entry beneath that container.

An instance of the Managed Entries Plug-in defines three things:

- The search criteria to identify the origin entries (using a search scope and a search filter)
- The subtree under which to create the managed entries (the new entry location)

- The template entry to use for the managed entries

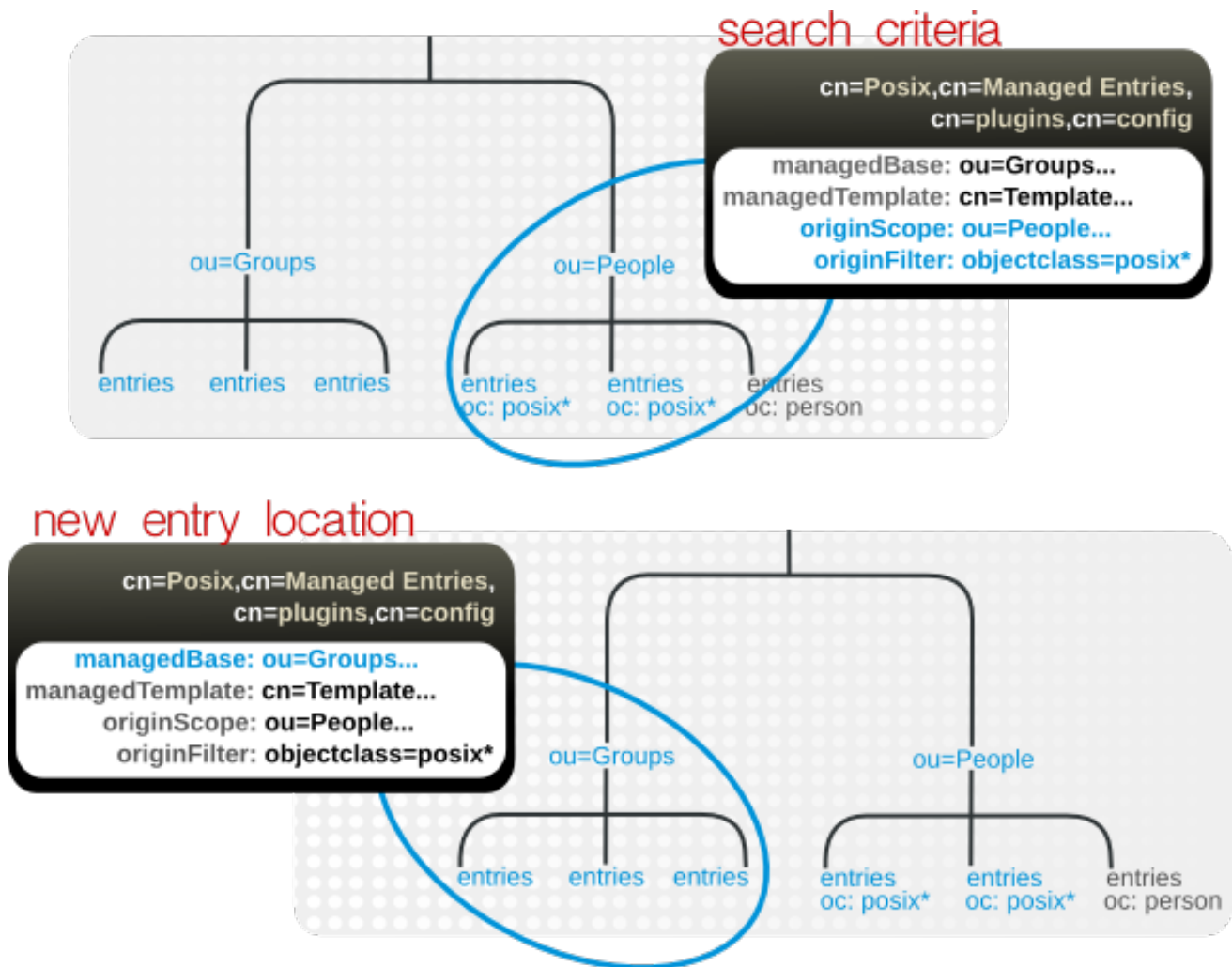


Figure 6.2. Defining Managed Entries

For example:

```
dn: cn=Posix User-Group,cn=Managed Entries,cn=plugins,cn=config
objectclass: extensibleObject
cn: Posix User-Group
originScope: ou=people,dc=example,dc=com
originFilter: objectclass=posixAccount
managedBase: ou=groups,dc=example,dc=com
managedTemplate: cn=Posix User-Group
Template,ou=Templates,dc=example,dc=com
```

The origin entry does not have to have any special configuration or settings to create a managed entry; it simply has to be created within the scope of the plug-in and match the given search filter.

6.3.1.2. About the Template Entry

Each instance of the plug-in uses a template entry which defines the managed entry configuration. The template effectively lays out the entry, from the object classes to the entry values.



NOTE

Since the template is referenced in the definition entry, it can be located anywhere in the directory. However, it is recommended that the template entry be under the replicated suffix so that any other masters in multi-master replication all use the same template for their local instances of the Managed Entries Plug-in.

The concept of a template entry is similar to the templates used in CoS, but there are some important differences. The managed entry template is slightly different than the type of template used for a class of service. For a class of service, the template contains a single attribute with a specific value that is fed into all of the entries which belong to that CoS. Any changes to the class of service are immediately reflected in the associated entries, because the CoS attributes in those entries are virtual attributes, not truly attributes set on the entry.

The template entry for the Managed Entries Plug-in, on the other hand, is not a central entry that supplies values to associated entries. It is a true template — it lays out what is in the entry. The template entry can contain both static attributes (ones with pre-defined values, similar to a CoS) and mapped attributes (attributes that pull their values or parts of values from the origin entry). The template is referenced when the managed entry is created and then any changes are applied to the managed entry *only* when the origin entry is changed and the template is evaluated again by the plug-in to apply those updates.

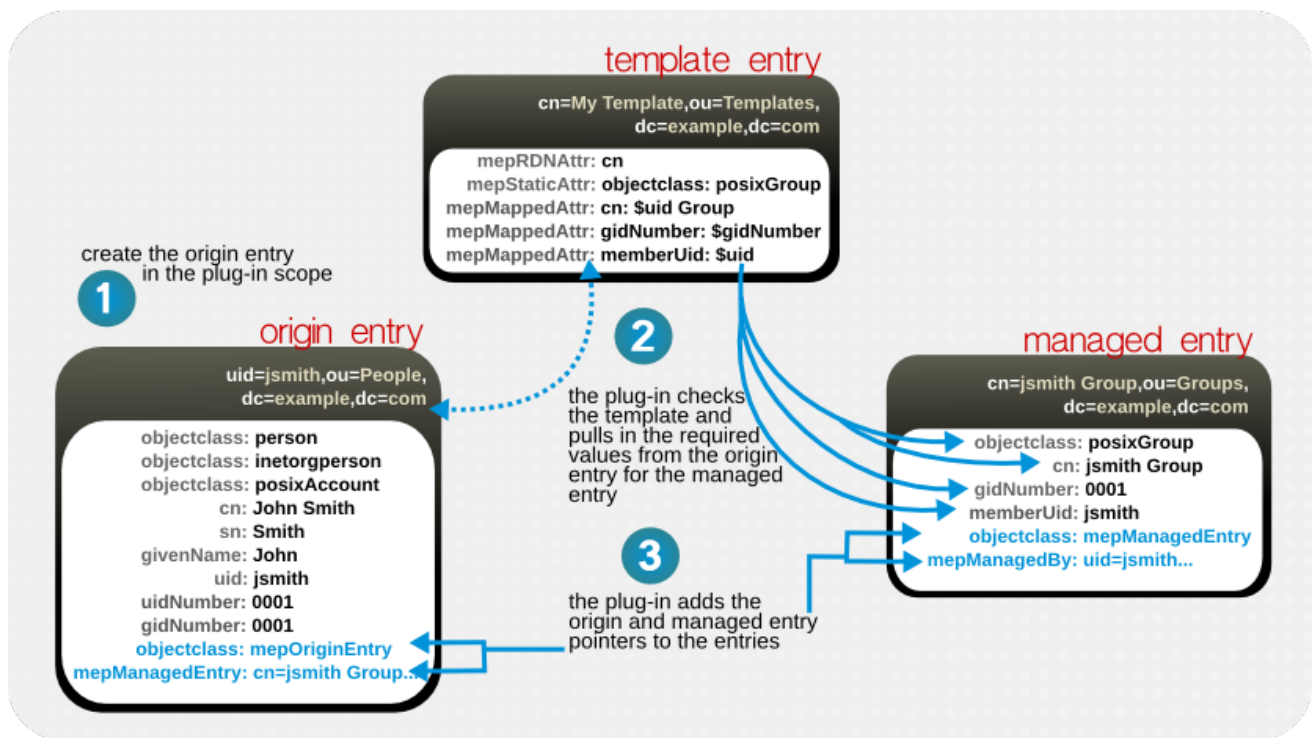


Figure 6.3. Templates, Managed Entries, and Origin Entries

The template can provide a specific value for an attribute in the managed entry by using a *static* attribute in the template. The template can also use a value that is derived from some attribute in the origin entry, so the value may be different from entry to entry; that is a *mapped* attribute, because it references the attribute type in the origin entry, not a value.

A mapped value use a combination of token (dynamic values) and static values, but it can only use *one token in a mapped attribute*.

```
dn: cn=Posix User-Group Template,ou=Templates,dc=example,dc=com
objectclass: mepTemplateEntry
cn: Posix User-Group Template
```

```
mepRDNAttr: cn
mepStaticAttr: objectclass: posixGroup
mepMappedAttr: cn: $cn Group Entry
mepMappedAttr: gidNumber: $gidNumber
mepMappedAttr: memberUid: $uid
```

The mapped attributes in the template use tokens, prepended by a dollar sign (\$), to pull in values from the origin entry and use it in the managed entry. (If a dollar sign is actually in the managed attribute value, then the dollar sign can be escaped by using two dollar signs in a row.)

A mapped attribute definition can be quoted with curly braces, such as **Attr: \${cn}test**. Quoting a token value is not required if the token name is not immediately followed by a character that is valid in an attribute name, such as a space or comma. For example, **\$cn test** is acceptable in an attribute definition because a space character immediately follow the attribute name, but **\$cntest** is not valid because the Managed Entries Plug-in attempts to look for an attribute named **cntest** in the origin entry. Using curly braces identifies the attribute token name.



NOTE

Make sure that the values given for static and mapped attributes comply with the required attribute syntax.

6.3.1.3. Entry Attributes Written by the Managed Entries Plug-in

Both the origin entry and the managed entry have special managed entries attributes which indicate that they are being managed by an instance of the Managed Entries Plug-in. For the origin entry, the plug-in adds links to associated managed entries.

```
dn: uid=jsmith,ou=people,dc=example,dc=com
objectclass: mepOriginEntry
objectclass: posixAccount
...
sn: Smith
mail: jsmith@example.com
mepManagedEntry: cn=jsmith Posix Group,ou=groups,dc=example,dc=com
```

On the managed entry, the plug-in adds attributes that point back to the origin entry, in addition to the attributes defined in the template.

```
dn: cn=jsmith Posix Group,ou=groups,dc=example,dc=com
objectclass: mepManagedEntry
objectclass: posixGroup
...
mepManagedBy: uid=jsmith,ou=people,dc=example,dc=com
```

Using special attributes to indicate managed and origin entries makes it easy to identify the related entries and to assess changes made by the Managed Entries Plug-in.

6.3.1.4. Managed Entries Plug-in and Directory Server Operations

The Managed Entries Plug-in has some impact on how the Directory Server carries out common operations, like add and delete operations.

Table 6.5. Managed Entries Plug-in and Directory Server Operations

Operation	Effect by the Managed Entries Plug-in
Add	<p>With every add operation, the server checks to see if the new entry is within the scope of any Managed Entries Plug-in instance. If it meets the criteria for an origin entry, then a managed entry is created and managed entry-related attributes are added to both the origin and managed entry.</p>
Modify	<p>If an origin entry is modified, it triggers the plug-in to update the managed entry. Changing a <i>template</i> entry, however, does not update the managed entry automatically. Any changes to the template entry are not reflected in the managed entry until after the next time the origin entry is modified.</p> <p>The mapped managed attributes <i>within</i> a managed entry cannot be modified manually, only by the Managed Entry Plug-in. Other attributes in the managed entry (including static attributes added by the Managed Entry Plug-in) can be modified manually.</p>
Delete	<p>If an origin entry is deleted, then the Managed Entries Plug-in will also delete any managed entry associated with that entry. There are some limits on what entries can be deleted.</p> <ul style="list-style-type: none">• A template entry cannot be deleted if it is currently referenced by a plug-in instance definition.• A managed entry cannot be deleted except by the Managed Entries Plug-in.

Operation	Effect by the Managed Entries Plug-in
Rename	<p>If an origin entry is renamed, then plug-in updates the corresponding managed entry. If the entry is moved <i>out</i> of the plug-in scope, then the managed entry is deleted, while if an entry is moved <i>into</i> the plug-in scope, it is treated like an add operation and a new managed entry is created. As with delete operations, there are limits on what entries can be renamed or moved.</p> <ul style="list-style-type: none"> • A configuration definition entry cannot be moved out of the Managed Entries Plug-in container entry. If the entry is removed, that plug-in instance is inactivated. • If an entry is moved <i>into</i> the Managed Entries Plug-in container entry, then it is validated and treated as an active configuration definition. • A template entry cannot be renamed or moved if it is currently referenced by a plug-in instance definition. • A managed entry cannot be renamed or moved except by the Managed Entries Plug-in.
Replication	<p>The Managed Entries Plug-in operations <i>are not initiated by replication updates</i>. If an add or modify operation for an entry in the plug-in scope is replicated to another replica, that operation does not trigger the Managed Entries Plug-in instance on the replica to create or update an entry. The only way for updates for managed entries to be replicated is to replicate the final managed entry over to the replica.</p>

6.3.2. Creating the Managed Entries Template Entry

The first entry to create is the template entry. The template entry must contain all of the configuration required for the generated, managed entry. This is done by setting the attribute-value assertions in static and mapped attributes in the template:

```
mepStaticAttr: attribute: specific_value
mepMappedAttr: attribute: $token_value
```

The static attributes set an explicit value; mapped attributes pull some value from the originating entry is used to supply the given attribute. The values of these attributes will be tokens in the form *attribute: \$attr*. As long as the syntax of the expanded token of the attribute does not violate the required attribute syntax, then other terms and strings can be used in the attribute. For example:

```
mepMappedAttr: cn: Managed Group for $cn
```

There are some syntax rules that must be followed for the managed entries:

- A mapped value use a combination of token (dynamic values) and static values, but it can only use *one token per mapped attribute*.
- The mapped attributes in the template use tokens, prepended by a dollar sign (\$), to pull in values from the origin entry and use it in the managed entry. (If a dollar sign is actually in the managed attribute value, then the dollar sign can be escaped by using two dollar signs in a row.)
- A mapped attribute definition can be quoted with curly braces, such as **Attr: \${cn}test**. Quoting a token value is not required if the token name is not immediately followed by a character that is valid in an attribute name, such as a space or comma. For example, **\$cn test** is acceptable in an attribute definition because a space character immediately follow the attribute name, but **\$cntest** is not valid because the Managed Entries Plug-in attempts to look for an attribute named **cntest** in the origin entry. Using curly braces identifies the attribute token name.
- Make sure that the values given for static and mapped attributes comply with the required attribute syntax.



NOTE

Make sure that the values given for static and mapped attributes comply with the required attribute syntax. For example, if one of the mapped attributes is **gidNumber**, then the mapped value should be an integer.

Table 6.6. Attributes for the Managed Entry Template

Attribute	Description
mepTemplateEntry (object class)	Identifies the entry as a template.
cn	Gives the common name of the entry.
mepMappedAttr	Contains an attribute-token pair that the plug-in uses to create an attribute in the managed entry with a value taken from the originating entry.
mepRDNAAttr	Specifies which attribute to use as the naming attribute in the managed entry. The attribute used as the RDN must be a mapped attribute for the configuration to be valid.
mepStaticAttr	Contains an attribute-value pair that will be used, with that specified value, in the managed entry.

To create a template entry:

1. Run **ldapmodify** to add the entry. This entry can be located anywhere in the directory tree.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```



```
dn: cn=Posix User Template,ou=templates,dc=example,dc=com
cn: Posix User Template
```

You can also use the Directory Server Console to create the entry, as described in [Section 3.1.2, “Creating Directory Entries”](#).

2. Give it the **mepTemplateEntry** object class to indicate that it is a template entry.

```
objectClass: top
objectclass: mepTemplateEntry
```

3. Set the attributes for the entry; these are described in [Table 6.6, “Attributes for the Managed Entry Template”](#). The RDN attribute (**mepRDNAAttr**) is required. The attribute parameters are optional and the values depend on the type of entry that the plug-in will create. Make sure that whatever attribute you use for the naming attribute is also contained in the template entry as a mapped attribute.



NOTE

Attributes which will be the same for each managed entry — like the object class for the entries — should use the **mepStaticAttr** attribute to set the values manually.

```
mepRDNAAttr: cn
mepStaticAttr: objectclass: posixGroup
mepMappedAttr: cn: $cn Group Entry
mepMappedAttr: gidNumber: $gidNumber
mepMappedAttr: memberUid: $uid
```

6.3.3. Creating the Managed Entries Instance Definition

Once the template entry is created, then it is possible to create a definition entry that points to that template. The definition entry is an instance of the Managed Entries Plug-in.



NOTE

When the definition is created, the server checks to see if the specified template entry exists. If the template does not exist, then the server returns a warning that the definition configuration is invalid.

The definition entry must define the parameters to recognize the potential origin entry and the information to create the managed entry. The attributes available for the plug-in instance are listed in [Table 6.7, “Attributes for the Managed Entries Definition Entry”](#).

Table 6.7. Attributes for the Managed Entries Definition Entry

Attribute Name	Description
----------------	-------------

Attribute Name	Description
originFilter	The search filter to use to search for and identify the entries within the subtree which require a managed entry. The syntax is the same as a regular search filter.
originScope	The base subtree which contains the potential origin entries for the plug-in to monitor.
managedTemplate	Identifies the template entry to use to create the managed entry. This entry can be located anywhere in the directory tree.
managedBase	The subtree under which to create the managed entries.



NOTE

The Managed Entries Plug-in is enabled by default. If this plug-in is disabled, then re-enable it as described in [Section 1.8.1, “Enabling Plug-ins in the Directory Server Console”](#).

To create an instance:

1. Create the new plug-in instance below the **cn=Managed Entries, cn=plugins, cn=config** container entry.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

```
dn: cn=instance_name,cn=Managed Entries,cn=plugins,cn=config
```

2. Set the scope and filter for the origin entry search, the location of the new managed entries, and the template entry to use. These required attributes are listed in [Table 6.7, “Attributes for the Managed Entries Definition Entry”](#).

```
objectClass: top
objectClass: extensibleObject
cn: Posix User-Group
originScope: ou=people,dc=example,dc=com
originFilter: objectclass=posixAccount
managedBase: ou=groups,dc=example,dc=com
managedTemplate: cn=Posix User-Group
Template,ou=Templates,dc=example,dc=com
```

3. Restart the Directory Server to load the modified new plug-in instance.

```
service dirsrv restart instance_name
```

6.3.4. Putting Managed Entries Plug-in Configuration in a Replicated Database

As [Section 6.3.1, “About Managed Entries”](#) highlights, different instances of the Managed Entries Plug-in are created as children beneath the container plug-in entry in **cn=plugins,cn=com**. (This is common for plug-ins which allow multiple instances.) The drawback to this is that the configuration entries in **cn=plugins,cn=com** are not replicated, so the configuration has to be re-created on each Directory Server instance.

The Managed Entries Plug-in entry allows the **nsslapd-pluginConfigArea** attribute. This attribute points to another container entry, in the main database area, which contains the plug-in instance entries. This container entry can be in a replicated database, which allows the plug-in configuration to be replicated.

1. Create a container entry in a subtree that is replicated.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x

dn: cn=managed entries container,ou=containers,dc=example,dc=com
objectclass: top
objectClass: extensibleObject
objectClass: nsContainer
cn: managed entries container
```

2. Add the **nsslapd-pluginConfigArea** attribute to the Managed Entries Plug-in entry that points back to the container entry.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -D
"cn=directory manager" -w secret -p 389 -h server.example.com -x

dn: cn=Managed Entries,cn=plugins,cn=config
changetype: modify
add: nsslapd-pluginConfigArea
nsslapd-pluginConfigArea: cn=managed entries
container,ou=containers,dc=example,dc=com
```

3. Move or create the definition ([Section 6.3.3, “Creating the Managed Entries Instance Definition”](#)) and template ([Section 6.3.2, “Creating the Managed Entries Template Entry”](#)) entries under the new container entry.

6.4. USING VIEWS

Virtual directory tree views, or *views*, create a virtual directory hierarchy, so it is easy to navigate entries, without having to make sure those entries physically exist in any particular place. The view uses information about the entries to place them in the view hierarchy, similarly to members of a filtered role or a dynamic group. Views superimpose a DIT hierarchy over a set of entries, and to client applications, views appear as ordinary container hierarchies.

6.4.1. About Views

Views create a directory tree similar to the regular hierarchy, such as using organizational unit entries for subtrees, but views entries have an additional object class (**nsview**) and a filter attribute (**nsviewfilter**) that set up a filter for the entries which belong in that view. Once the view container entry is added, all of the entries that match the view filter instantly populate the view. The target entries

only *appear* to exist in the view; their true location never changes. For example, a view may be created as **ou=Location Views**, and a filter is set for **l=Mountain View**. Every entry, such as **cn=Jane Smith,l=Mountain View,ou=People,dc=example,dc=com**, is immediately listed under the **ou=Location Views** entry, but the real **cn=Jane Smith** entry remains in the **ou=People,dc=example,dc=com** subtree.

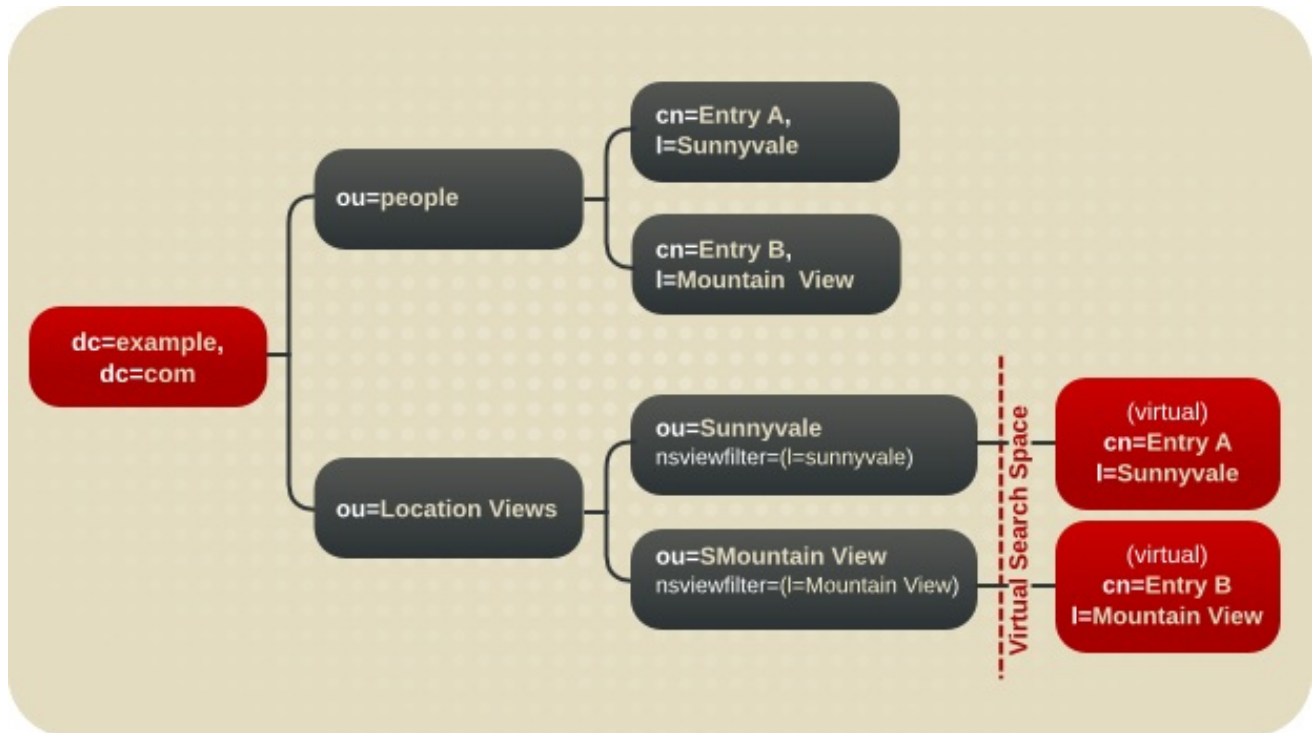


Figure 6.4. A Directory Tree with a Virtual DIT View hierarchy

Virtual DIT views behave like normal DITs in that a subtree or a one-level search can be performed with the expected results being returned.



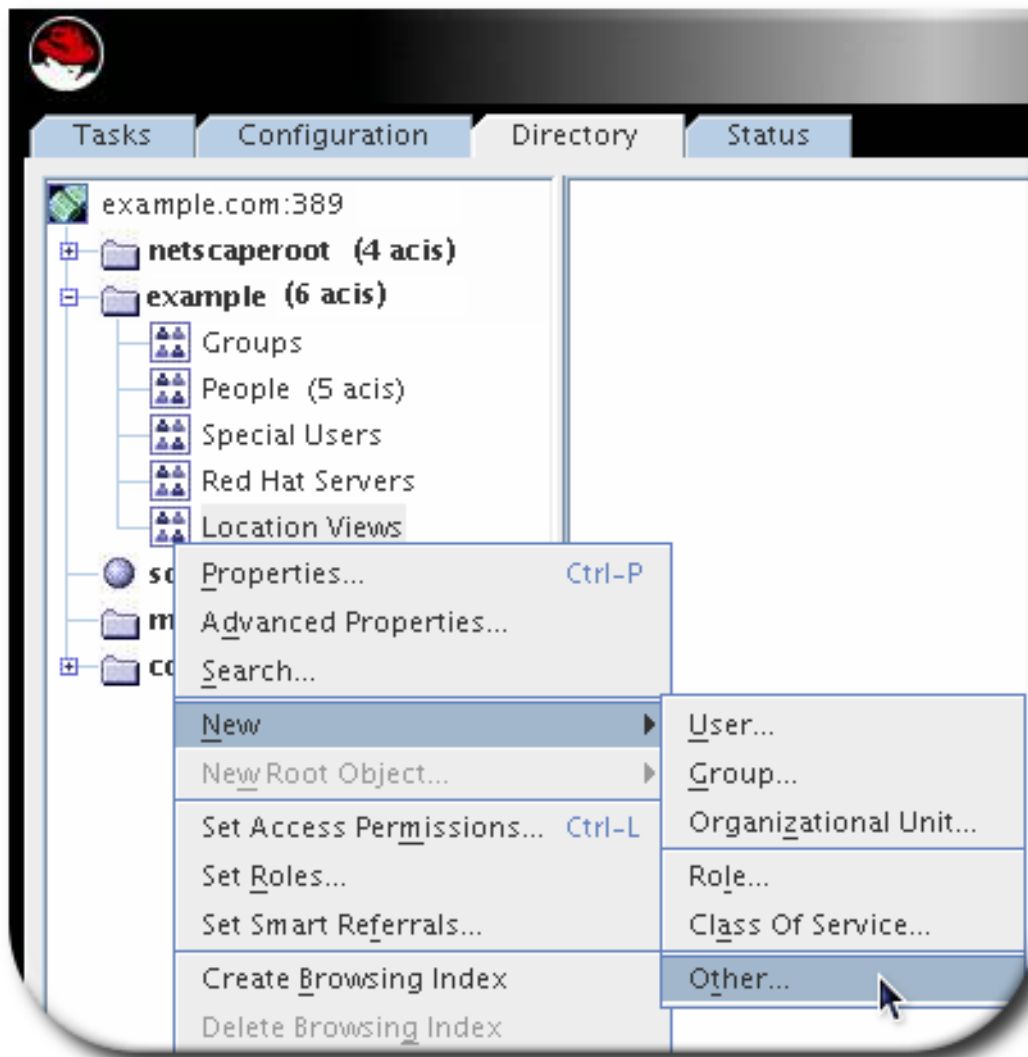
NOTE

There is a sample LDIF file with example views entries, **Example-views.ldif**, installed with Directory Server. This file is in the **/usr/share/dirsrv/data** directory on Red Hat Enterprise Linux 6 (64-bit).

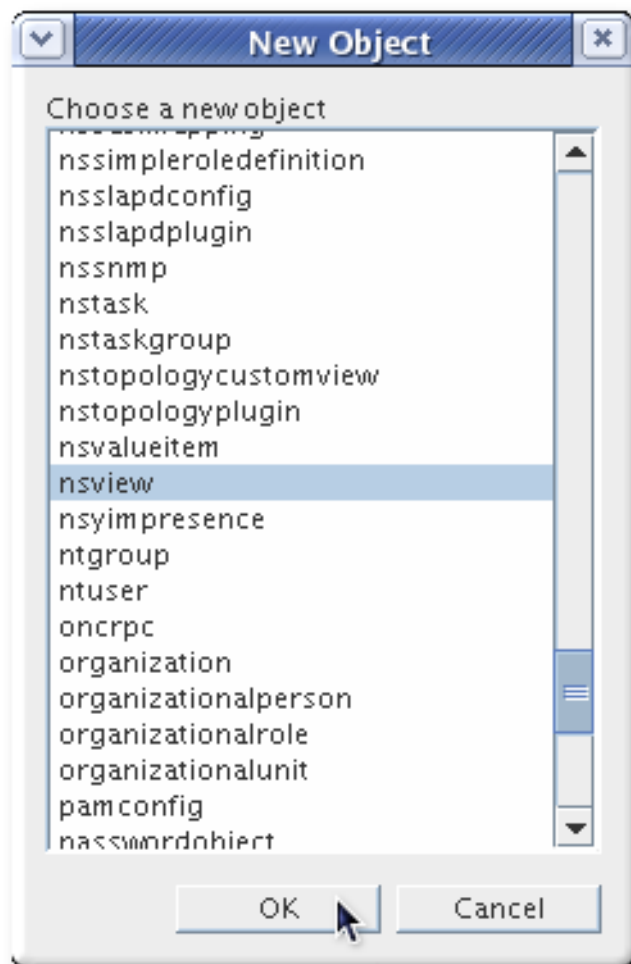
The *Directory Server Deployment Guide* has more information on how to integrate views with the directory tree hierarchy.

6.4.2. Creating Views in the Console

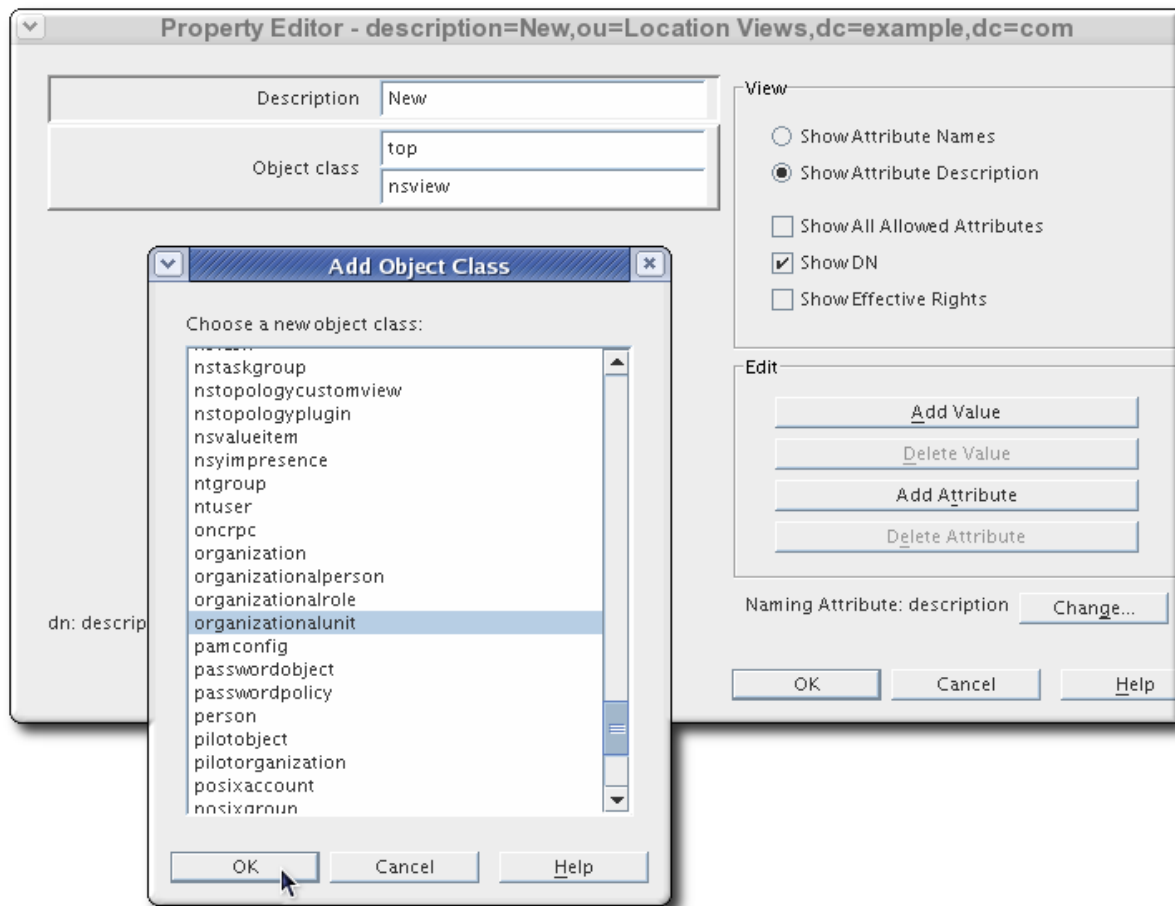
1. Select the **Directory** tab.
2. In the left navigation tree, create an organizational unit suffix to hold the views. For instance, for views based on the locality (**l**) attribute, name this organizational unit **Location Views**. Creating sub suffixes is described in [Section 2.1.1.2, “Creating a New Sub Suffix Using the Console”](#).
3. Right-click **ou=Location Views**, and select **New > Other**.



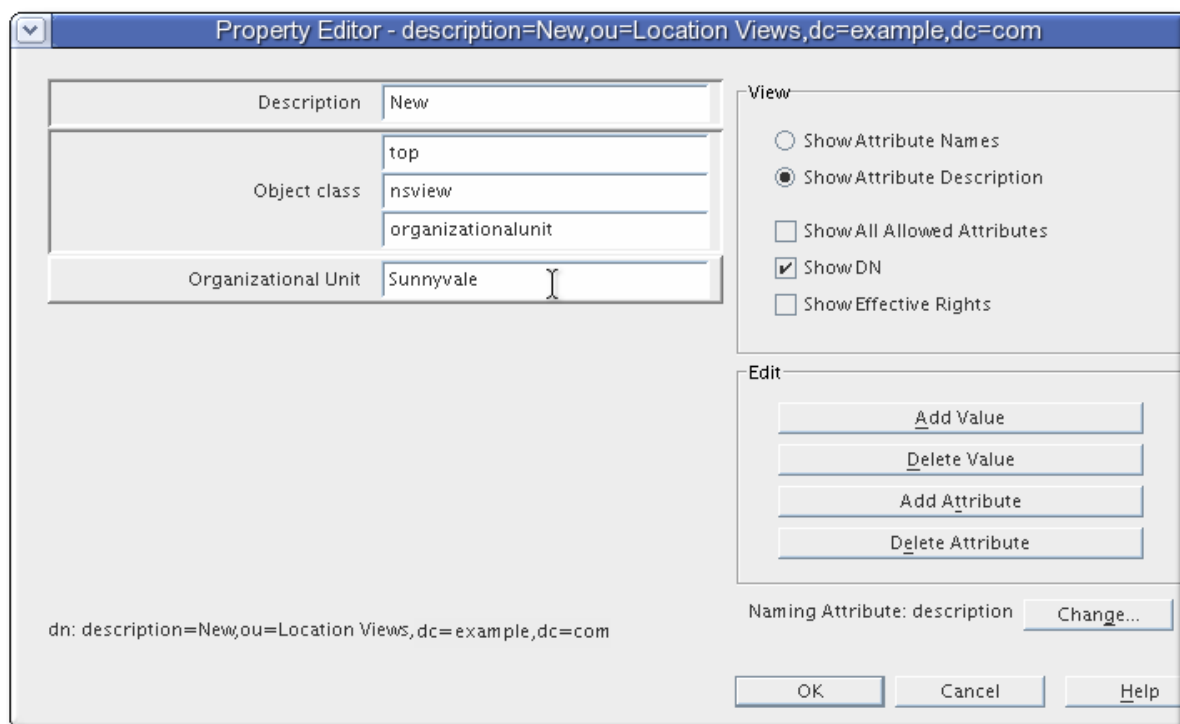
4. Select **nsview** from the **New Object** menu, and click **OK**.



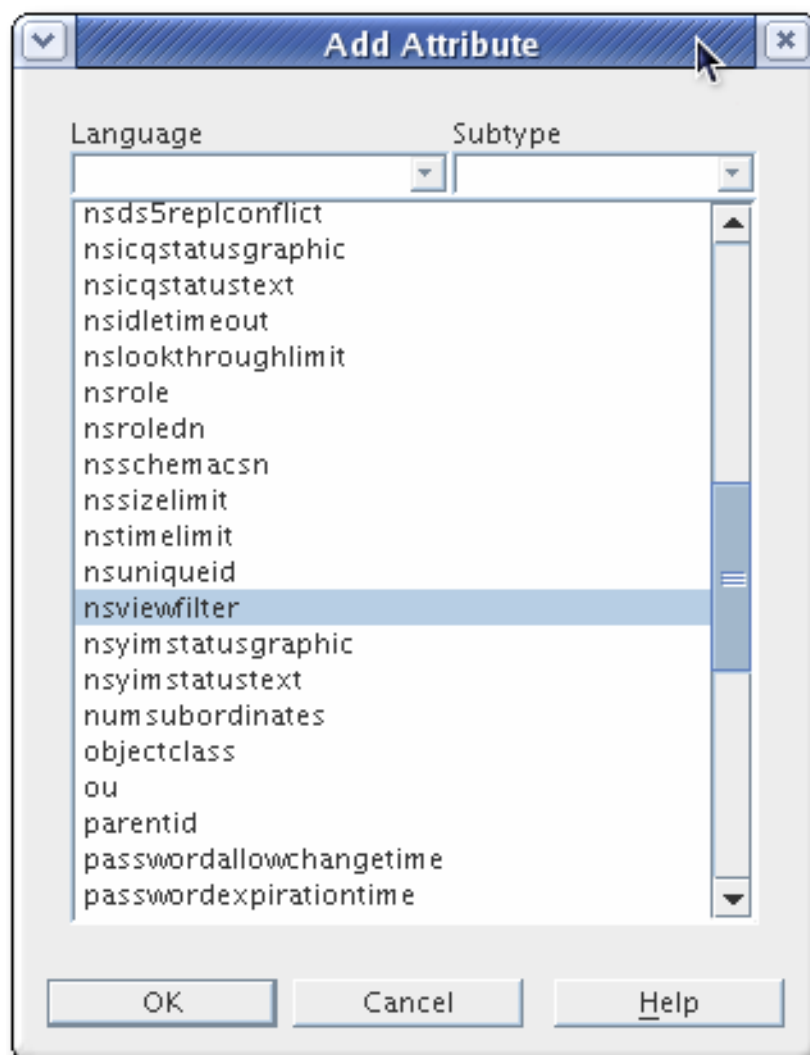
5. In the **Property Editor** window, click the **Add Value** button, and add the organization unit object class.



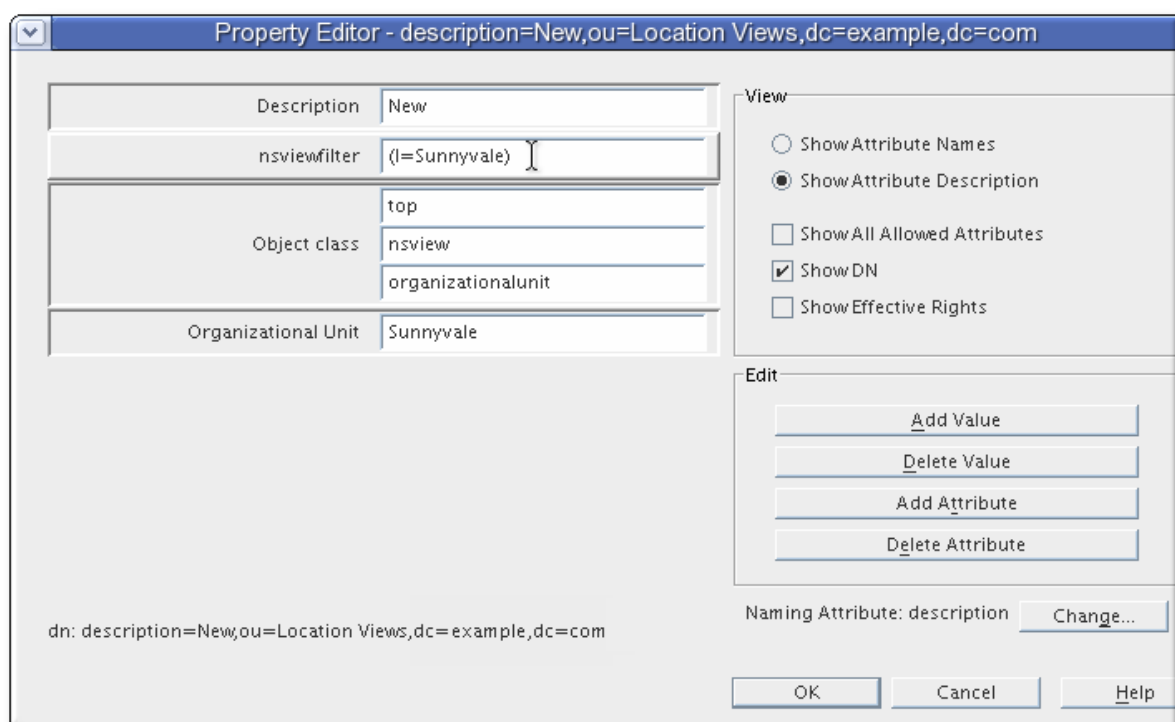
6. Name the organization unit according to how to organize the views. For instance, **ou=Sunnyvale**. Make the **ou** attribute the naming attribute.



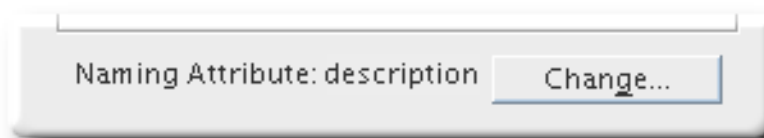
7. Click the **Add Attribute** button, and add the **nsviewfilter** attribute.



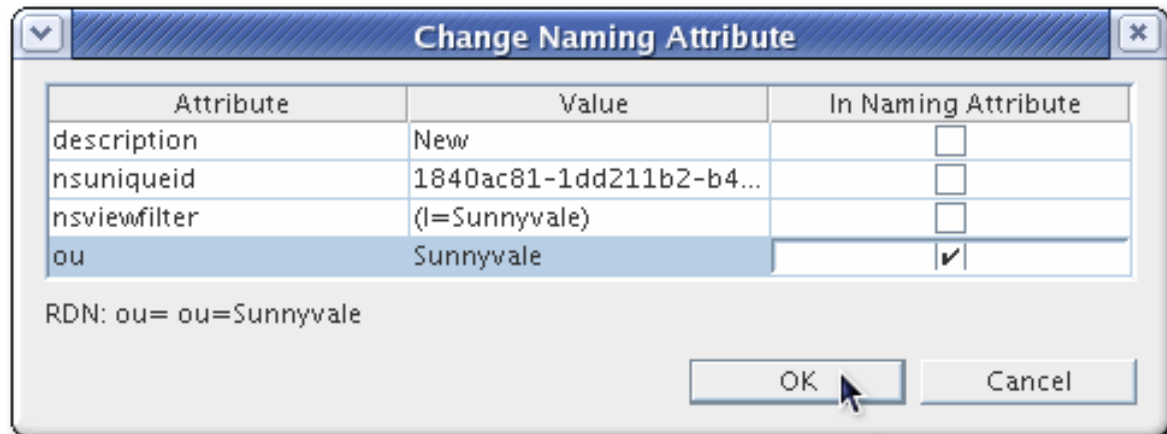
8. Create a filter that reflects the views, such as **(1=Sunnyvale)**.



- Click the **Change** button in the lower right corner to change the naming attribute.



Use the **ou** of the entry as the naming attribute instead of *description*.



- Click **OK** to close the attributes box, and click **OK** again to save the new view entry.

The new view is immediately populated with any entries matching the search filter, and any new entries added to directory are automatically included in the view.

6.4.3. Creating Views from the Command Line

- Use the **ldapmodify** utility to bind to the server and prepare it to add the new view entry to the configuration file.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

- Add the new views container entry, in this example, under the **dc=example,dc=com** root suffix. This entry must have the **nsview** object class and the **nsViewFilter** attribute. The **nsViewFilter** attribute sets the attribute-value which identifies entries that belong in the view.

```
dn: ou=Example View,dc=example,dc=com
changetype: add
objectClass: top
objectClass: organizationalUnit
objectClass: nsview
ou: Example View
nsViewFilter: l=Mountain View
description: Example View
```

6.4.4. Improving Views Performance

As [Section 6.4.1, “About Views”](#) describes, views are derived from search results based on a given filter. Part of the filter is the attribute defined in the ***nsViewFilter*** attribute; the rest of the filter is based on the entry hierarchy, looking for the ***entryid*** and ***parentid*** of the real entries included in the view.

```
( |(parentid=search_base_id)(entryid=search_base_id)
```

If any of the searched-for attributes — ***entryid***, ***parentid***, or the attribute set in ***nsViewFilter*** — are not indexed, then the views search becomes an unindexed search because the views operation searches the entire tree for matching entries.

To improve views performance, create equality indexes for *entryid*, *parentid*, and the attribute set in *nsViewFilter*.

Creating equality indexes is covered in [Section 9.2, “Creating Standard Indexes”](#), and updating existing indexes to include new attributes is covered in [Section 9.3, “Applying New Indexes to Existing Databases”](#).

CHAPTER 7. CONFIGURING SECURE CONNECTIONS

By default, clients and users connect to the Red Hat Directory Server over a standard connection. Standard connections do not use any encryption, so information is sent back and forth between the server and client in the clear.

Directory Server supports SSL/TLS connections, Start TLS connection, and SASL authentication, which provide layers of encryption and security that protect directory data from being read even if it is intercepted.

7.1. REQUIRING SECURE CONNECTIONS

The Directory Server has two different ways of encrypting data: SSL/TLS (either through an initial SSL connection or using Start TLS to establish encryption on a previously unencrypted connection) and SASL. If the server is configured to use it, any client can connect to the server using either encryption method to protect data.

For additional security, the Directory Server can be configured to require a certain level of encryption before it allows a connection. The Directory Server can define and require a specific *security strength factor* for any connection. The SSF sets a minimum encryption level, defined by its key strength, for any connection or operation.

To require a minimum SSF for *any* and all directory operations, set the ***nsslapd-minssf*** configuration attribute. When enforcing a minimum SSF, the Directory Server looks at each available encryption type for an operation — SSL/TLS or SASL — and determines which has the higher SSF value and then compares the higher value to the minimum SSF. (It is possible for both SASL authentication and SSL/TLS to be configured for some server-to-server connections, such as replication.)



NOTE

Alternatively, use the ***nsslapd-minssf-exclude-rootdse*** configuration attribute. This sets a minimum SSF setting for all connections to the Directory Server *except* for queries against the root DSE. A client may need to obtain information about the server configuration, like its default naming context, before initiating an operation. The ***nsslapd-minssf-exclude-rootdse*** attribute allows the client to get that information without having to establish a secure connection first.

The SSF for a connection is evaluated when the first operation is initiated on a connection. This allows Start TLS and SASL binds to succeed, even though those two connections initially open a regular connection. After the TLS or SASL session is opened, then the SSF is evaluated. Any connection which does not meet the SSF requirements is closed with an LDAP unwilling to perform error.

Setting a minimum SSF can be used to effectively disable all standard or insecure connections to a directory.

The default ***nsslapd-minssf*** attribute value is 0, which means there is no minimum SSF for server connections. The value can be set to any reasonable positive integer. The value represents the required key strength for any secure connection.

1. Add the ***nsslapd-minssf*** attribute to the **cn=config** entry:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=config
```

```
changetype: modify
replace: nsslapd-minssf
nsslapd-minssf: 128
```

2. Restart the server.

```
service dirsrv restart
```



NOTE

An ACI can be set to require an SSF for a specific type of operation, as in [Section 13.9.10, “Setting an ACI to Require a Certain Security Strength Factor for Some Operations”](#).

Secure connections can be required for bind operations by turning on the **nsslapd-require-secure-binds** attribute, as in [Section 14.8.1, “Requiring Secure Binds”](#).

7.2. DISABLING SSL AND REQUIRING TLS

TLS is enabled in the **nsTLS1** parameter, which is enabled by default.

By default, SSLv3^[2] is also enabled. If both TLS and SSLv3 are enabled, then a client can use either protocol to connect to the Directory Server. In some environments, SSLv3 should be disabled so that only TLSv1 connections are allowed.

In the Directory Server configuration, SSLv3 is enabled in the **nsSSL3** parameter. There are two ways to change the value of this setting and disable SSLv3:

- Change the **nsSSL3** attribute to off. The SSLv3 attribute must be explicitly set to off; if the parameter is missing or has any other value, SSLv3 is implicitly enabled.

For example:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -D
"cn=directory manager" -w secret -p 636 -h server.example.com -x

dn: cn=encryption,cn=config
changetype: modify
replace: nsSSL3
nsSSL3: off
```

- Use **modutil** to enable FIPS mode, which automatically disables SSLv3. If FIPS mode is enabled, it overrides whatever the **nsSSL3** attribute is in order to disable SSLv3. For example:

```
modutil -dbdir /etc/dirsrv/slapd-instance_name -chkfips true
FIPS mode enabled.
```

Enabling FIPS mode may have other security and configuration implications, however, so it may be easier to disable the SSLv3 parameter.

7.3. MANAGING CERTIFICATES USED BY THE DIRECTORY SERVER

7.3.1. Obtaining and Installing Server Certificates

Before the Directory Server can be set to run in TLS/SSL, server and CA certificates must be properly configured in the Directory Server. If a server certificate has already been generated for the Directory Server instance and the issuing certificate authority (CA) is already trusted by the Directory Server, begin setting up TLS/SSL as described in [Section 7.4, “Setting up TLS/SSL”](#).

Obtaining and installing certificates consists of the following steps:

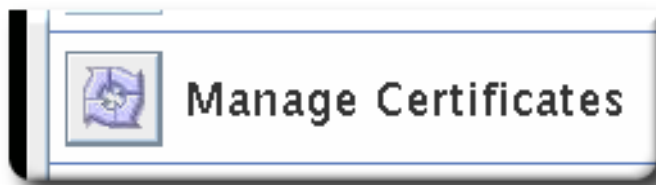
1. Generate a certificate request.
2. Send the certificate request to a certificate authority.
3. Install the server certificate.
4. Set the Directory Server to trust the certificate authority.
5. Confirm that the certificates are installed.

Two wizards automate the process of creating a certificate database and of installing the key-pair. The **Certificate Request Wizard** in the Directory Server Console can generate a certificate request and send it to a certificate authority. The **Certificate Install Wizard** in the Directory Server Console can then install the server certificate and the CA certificate.

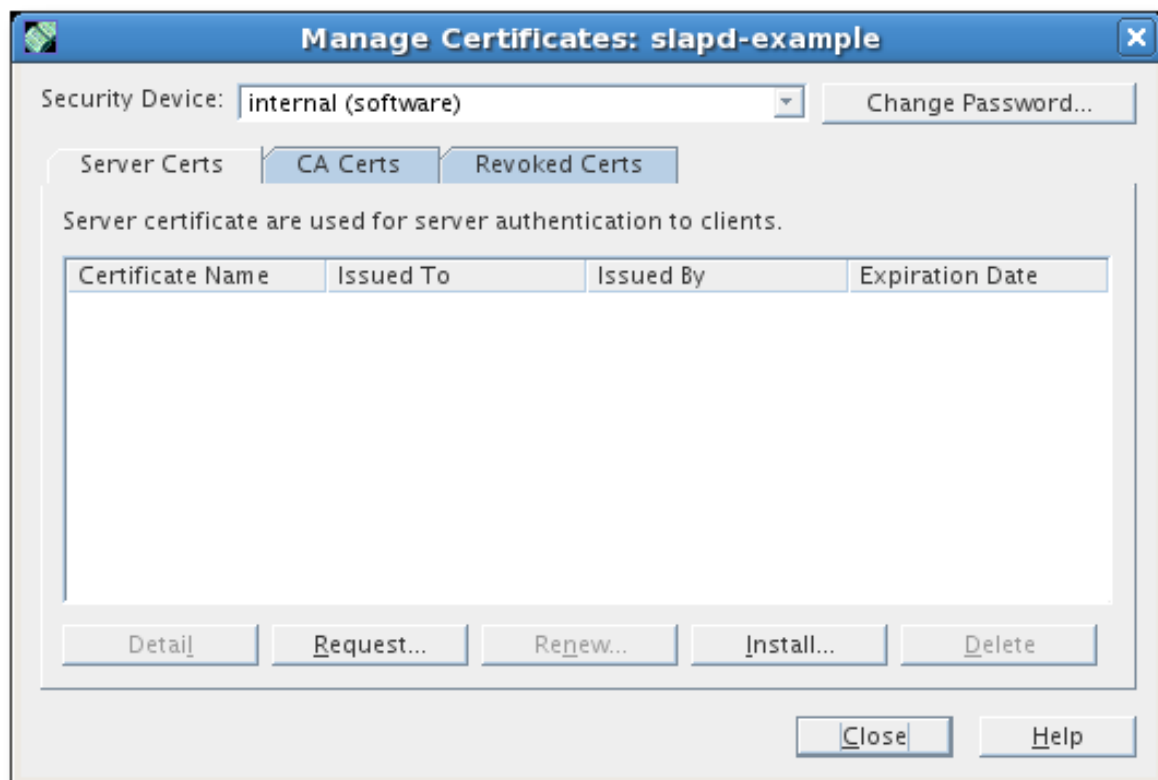
7.3.1.1. Generating a Certificate Request

Generate a certificate request, and send it to a CA. The Directory Server Console has a tool, the **Certificate Request Wizard**, which generates a valid certificate request to submit to any certificate authority (CA).

1. Select the **Tasks** tab, and click **Manage Certificates**.



2. Select the **Server Certs** tab, and click the **Request** button.

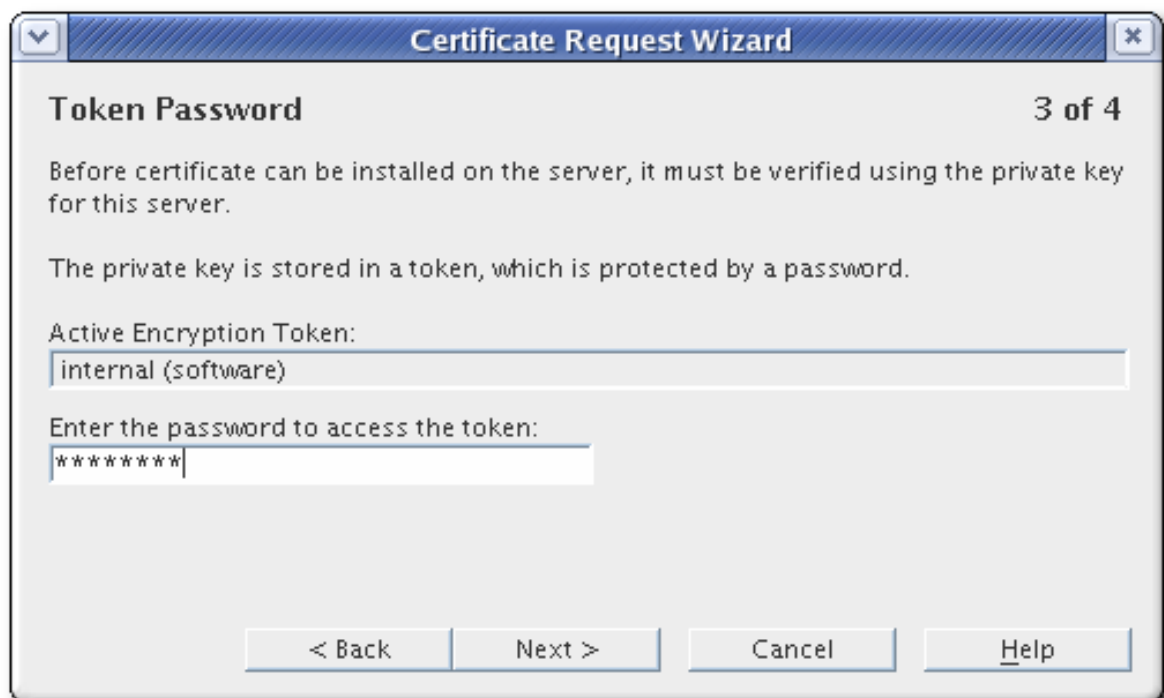


3. Enter the **Requester Information** in the blank text fields, then click **Next**.



- **Server Name.** Enter the fully qualified host name of the Directory Server as it is used in DNS and reverse DNS lookups; for example, **dir.example.com**. The server name is critical for client-side validation to work, which prevents man-in-the-middle attacks.

- *Organization*. Enter the legal name of the company or institution. Most CAs require this information to be verified with legal documents such as a copy of a business license.
 - *Organizational Unit*. *Optional*. Enter a descriptive name for the organization within the company.
 - *Locality*. *Optional*. Enter the company's city name.
 - *State or Province*. Enter the full name of the company's state or province (no abbreviations).
 - *Country*. Select the two-character abbreviation for the country's name (ISO format). The country code for the United States is US.
4. If an external security device is to be used to store the Directory Server certificates, the device is plugged in, and the module has been installed as described in [Section 7.8, “Using Hardware Security Modules”](#), then the module is available in the **Active Encryption Token** menu. The default is to use the software databases with Directory Server, **internal (software)**.
 5. Enter the password that will be used to protect the private key, and click **Next**.



The **Next** button is grayed out until a password is supplied.

6. The **Request Submission** dialog box provides two ways to submit a request: directly to the CA (if there is one internally) or manually. To submit the request manually, select **Copy to Clipboard** or **Save to File** to save the certificate request which will be submitted to the CA.



7. Click **Done** to dismiss the **Certificate Request Wizard**.

After generating the certificate request, send it to the CA.

7.3.1.2. Sending a Certificate Request

After the certificate request is generated, send it to a certificate authority (CA); the CA will generate and return a server certificate.

1. Most certificate requests are emailed to the CA, so open a new message.
2. Copy the certificate request information from the clipboard or the saved file into the body of the message.

```
-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBrjCCARcCAQAwbjELMAkGA1UEBhMCVXMxEzARBgNVBAGTCkNBTElGT1J
OSUEXLDAqBgVBaoTI25ldHNjYXB1IGNvbW11bm1jYXRpb25zIGNvcnBvcmlF
0aw9uMRwwGgYDVQQDEXNtZWxs24ubmV0c2NhcnGUuY29tMIGfMA0GCSqGSI
b3DQEBAQUAA4GNADCBiQKBgQCwAbskGh6SKY0gHy+UCSLnm3ok3X3u83Us7
ug0EfgSLR0f+K41eNqqRftGR83emqPLD0f0ZLTLjVGJaH4Jn4l1gG+Jdf/n
/zMyahxtV7+mT8G0FFigFfuxaxMjr2j7IvELlxQ4IfZgWwqCm4qQecv3G+N
9YdbjveMVXW0v4XwIDAQABoAAwDQYK
-----END NEW CERTIFICATE REQUEST-----
```

3. Send the email message to the CA.

After emailing the certificate request, wait for the CA to respond with the server certificate. Response time for requests varies. For example, if the CA is internal to the company, it may only take a day or two to respond to the request. If the selected CA is a third-party, it could take several weeks to respond to the request.

After receiving the certificate, install it in the Directory Server's certificate database. When the CA sends a response, be sure to save the information in a text file. The certificate must be available to install in the Directory Server.

Also, keep a backup of the certificate data in a safe location. If the system ever loses the certificate data, the certificate can be reinstalled using the backup file.

7.3.1.3. Installing the Certificate

NOTE

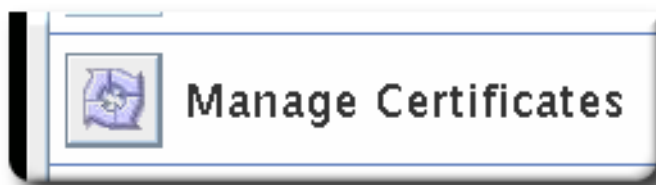
If an existing certificate has the same name as the new certificate you are attempting to install, you cannot use the Directory Server Console to install the certificate. It fails with the error *Internal error: Fail to install certificate -8169*.

You can use **certutil** to install the certificate.

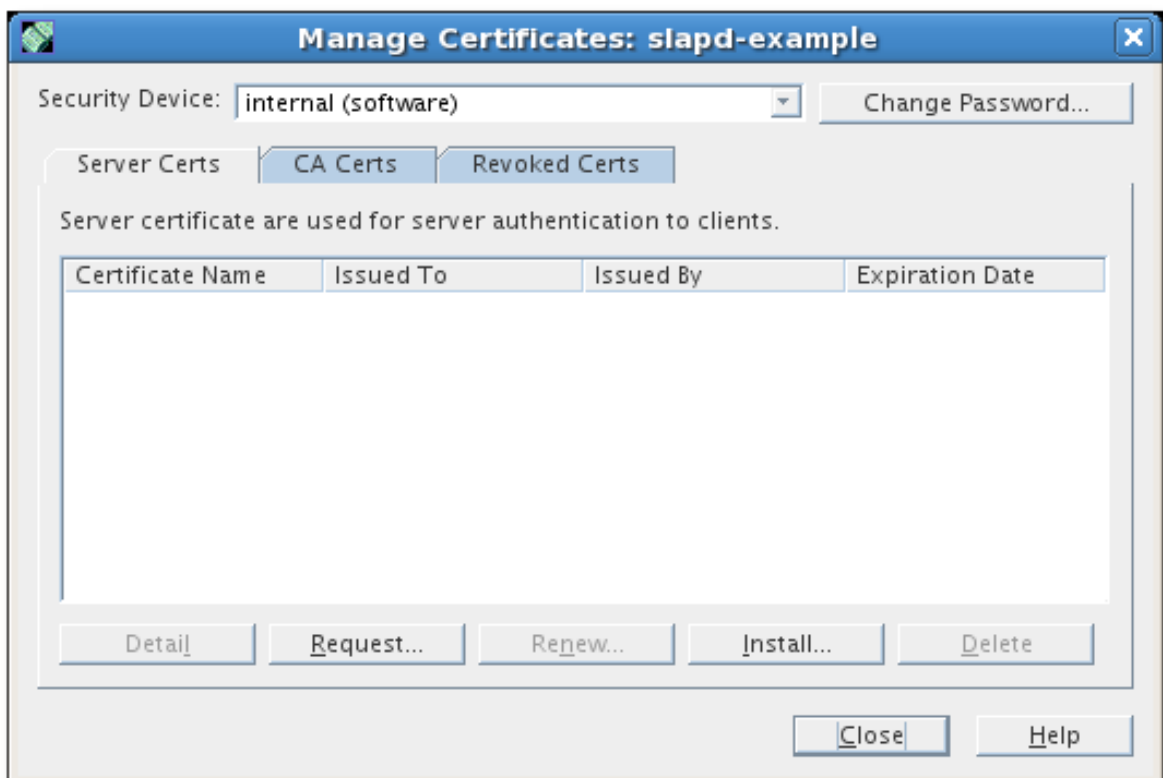
```
certutil -A -n nickname -t trust-settings -d certDB -f
/path/to/certificate_file
```

For more information on using **certutil** to manage certificates, see [Section 7.6, “Using certutil”](#).

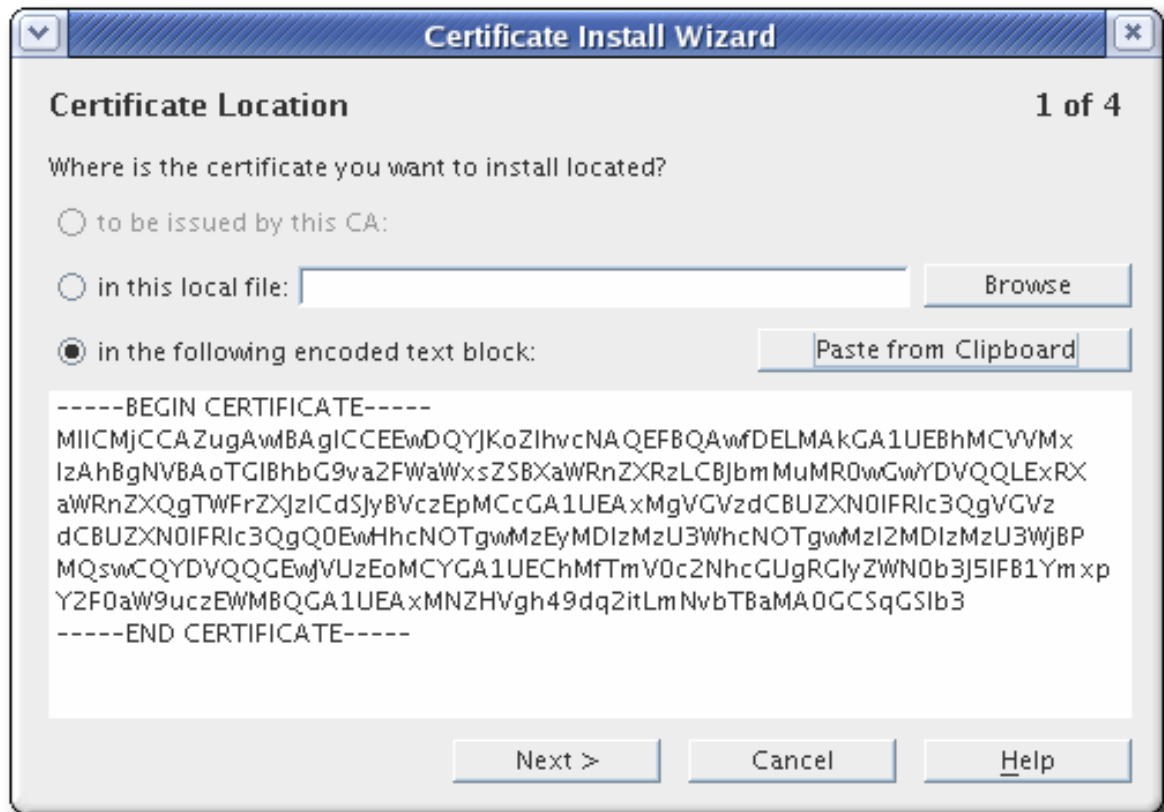
1. Select the **Tasks** tab, and click **Manage Certificates**.



2. Select the **Server Certs** tab, and click **Install**.



3. Give the certificate location or paste the certificate text in the text box, then click **Next**.



- *In this file.* Enter the absolute path to the certificate in this field.
 - *In the following encoded text block.* Copy the text from the CA's email or from the created text file, and paste it in this field.
4. Check that the certificate information displayed is correct, and click **Next**.
 5. Give a name to the certificate, and click **Next**.
 6. Provide the password that protects the private key. This password is the same as the one provided in step 5 in [Section 7.3.1.1, "Generating a Certificate Request"](#).

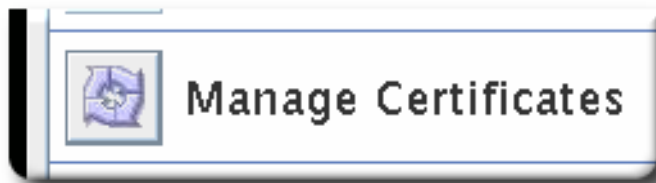
After installing the server certificate, configure the Directory Server to trust the CA which issued the server's certificate, [Section 7.3.2, "Trusting the Certificate Authority"](#).

7.3.2. Trusting the Certificate Authority

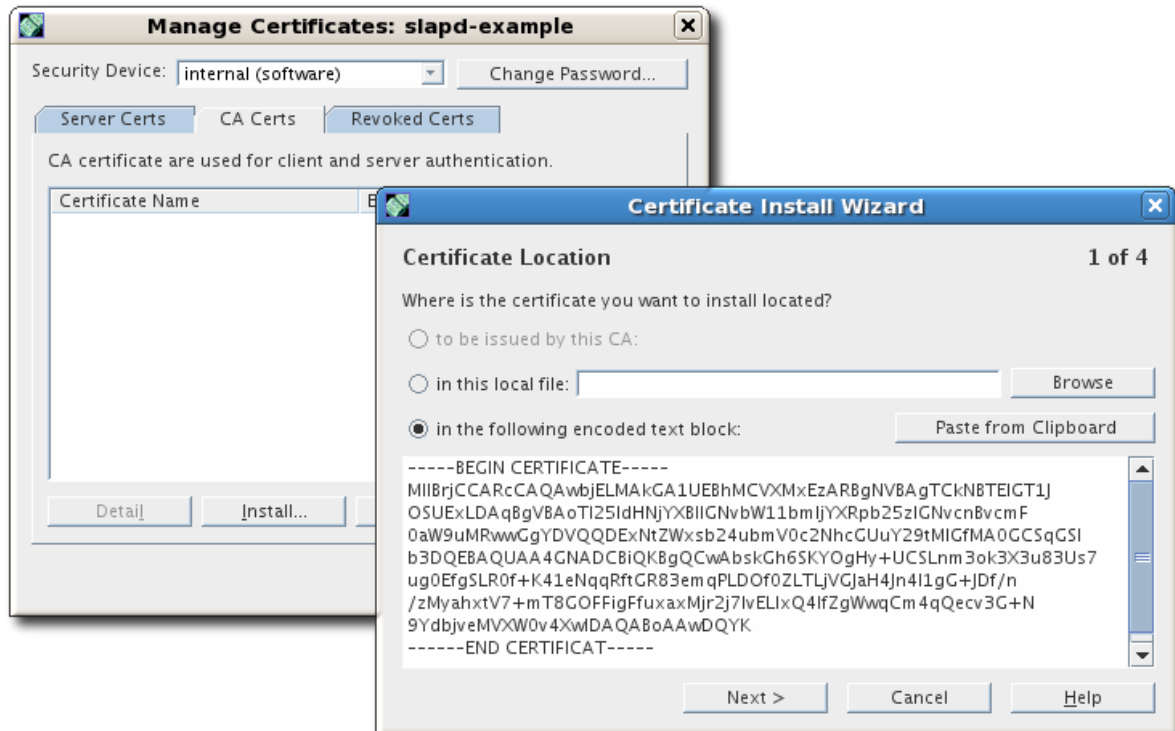
Configuring the Directory Server to trust the certificate authority consists of obtaining the CA's certificate and installing it into the server's certificate database. This process differs depending on the certificate authority. Some commercial CAs provide a web site that allow users to automatically download the certificate. Others will email it back to users.

After receiving the CA certificate, use the **Certificate Install Wizard** to configure the Directory Server to trust the certificate authority.

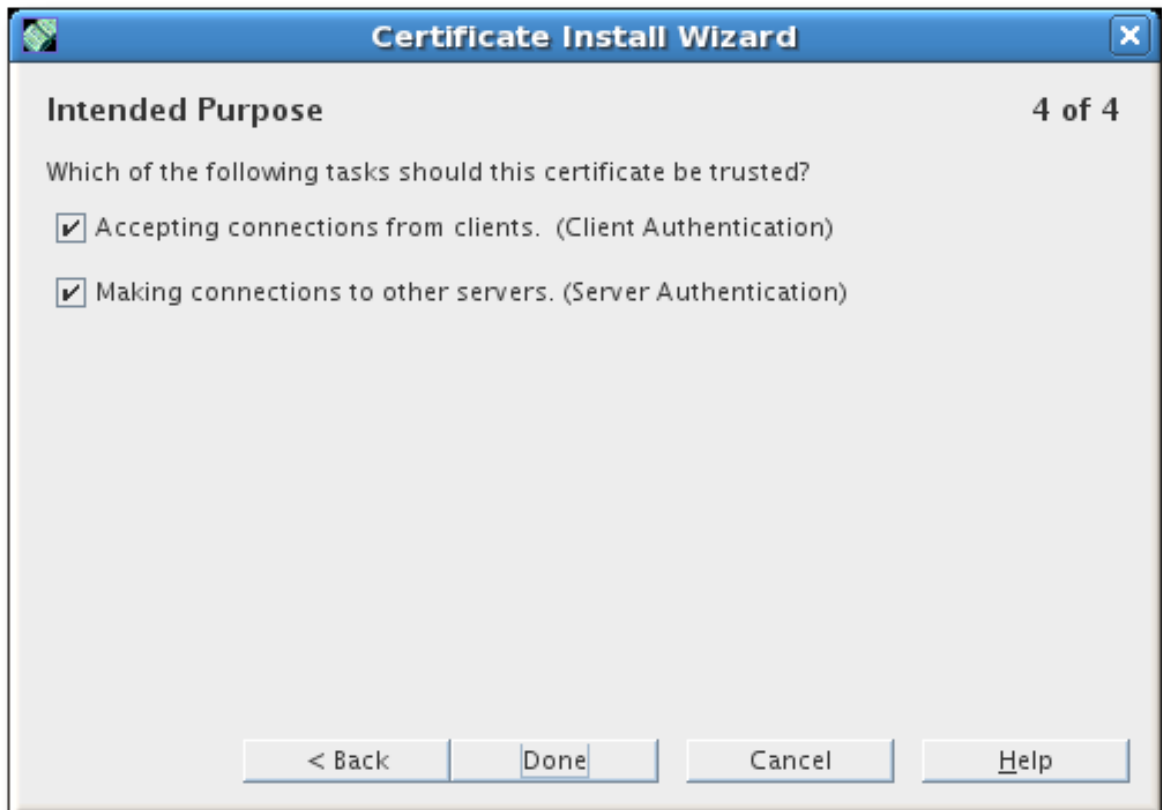
1. Select the **Tasks** tab, and click **Manage Certificates**.



2. Go to the **CA Certs** tab, and click **Install**.



3. If the CA's certificate is saved to a file, enter the path in the field provided. Alternatively, copy and paste the certificate, including the headers, into the text box. Click **Next**.
4. Check that the certificate information that opens is correct, and click **Next**.
5. Name the certificate, and click **Next**.
6. Select the purpose of trusting this certificate authority; it is possible to select both options.



- *Accepting connections from clients (Client Authentication).* The server checks that the client's certificate has been issued by a trusted certificate authority.
- *Accepting connections to other servers (Server Authentication).* This server checks that the directory to which it is making a connection (for replication updates, for example) has a certificate that has been issued by a trusted certificate authority.

Once both the server and CA certificates are installed, it is possible to configure the Directory Server to run in TLS/SSL. However, Red Hat recommends verifying that the certificates have been installed correctly.

7.3.3. Renewing Certificates

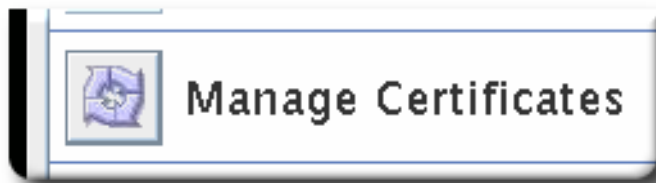
As with any issued identification — drivers' licenses, student IDs — certificates are valid for a predefined period and then expire and must be renewed. To renew a certificate, regenerate a certificate request, using the same information that was used to create the original, submit the request to a CA, and re-install the renewed certificate.



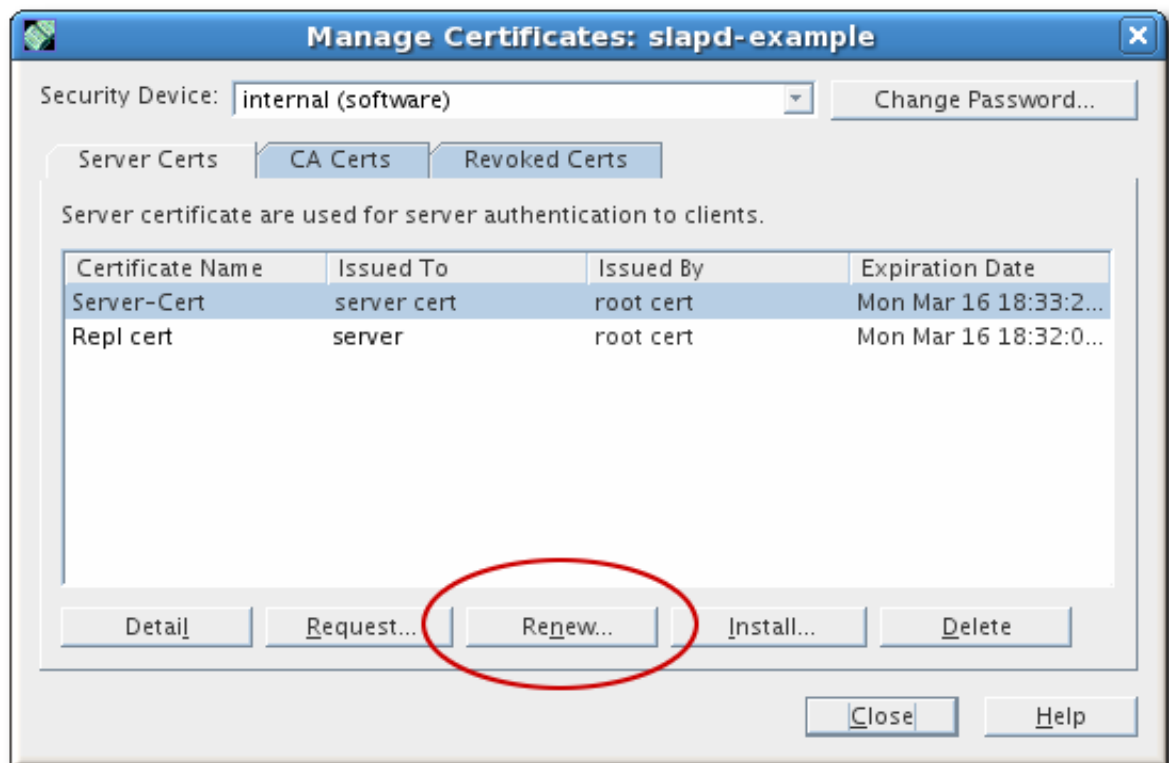
NOTE

When renewing a certificate using the **Certificate Wizard**, the text on the introduction screen does not clearly indicate that the process is renewal and not requesting a new certificate. Also, the requester information is not filled in automatically.

1. Open the Directory Server Console.
2. In the **Tasks** tab, click the **Manage Certificates** button.



3. Click the **Server Certs** tab.
4. Select the certificate to renew from the list of certificates, and click the **Renew** button.

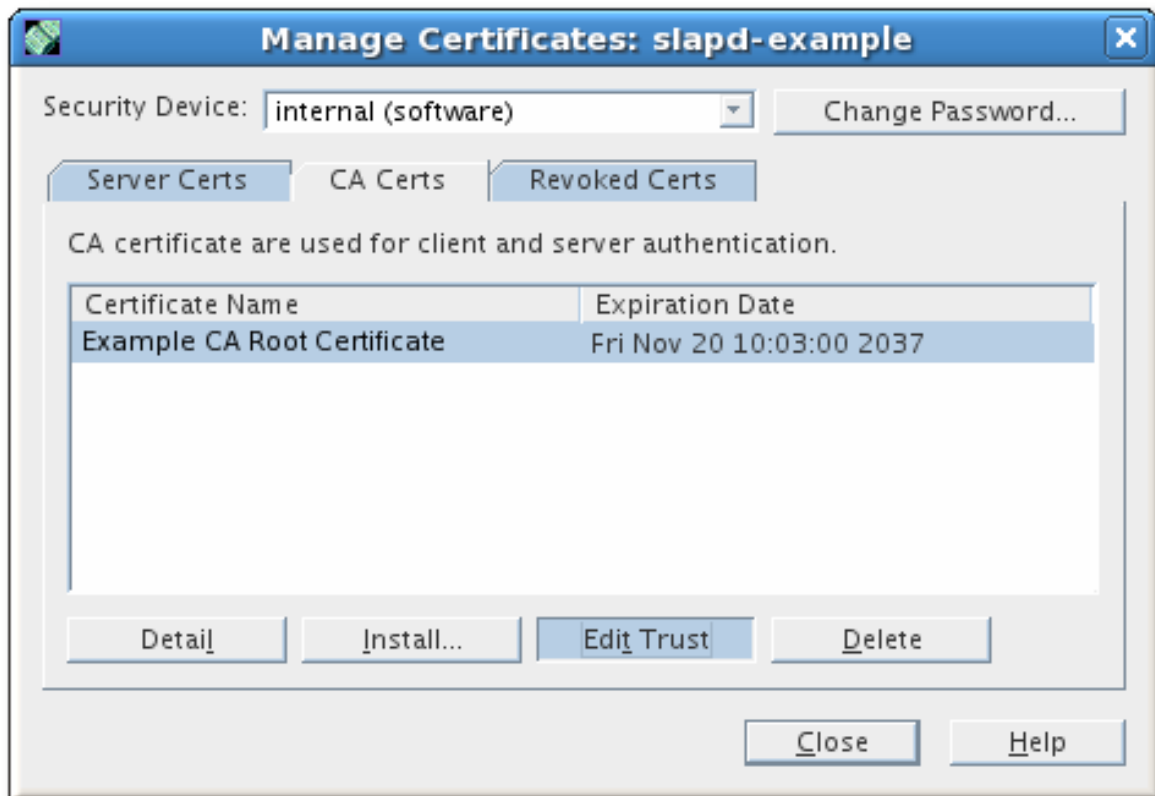


5. Go through the request wizard, using the same information used for requesting the original certificate.
6. Submit the request to a certificate authority.
7. Once the certificate is issued, reinstall it in the Directory Server.
 1. In the **Tasks** tab, click the **Manage Certificates** button.
 2. Click the **Server Certs** tab.
 3. Click the **Install** button.
 4. Paste in the renewed certificate, and continue through the installation wizard.

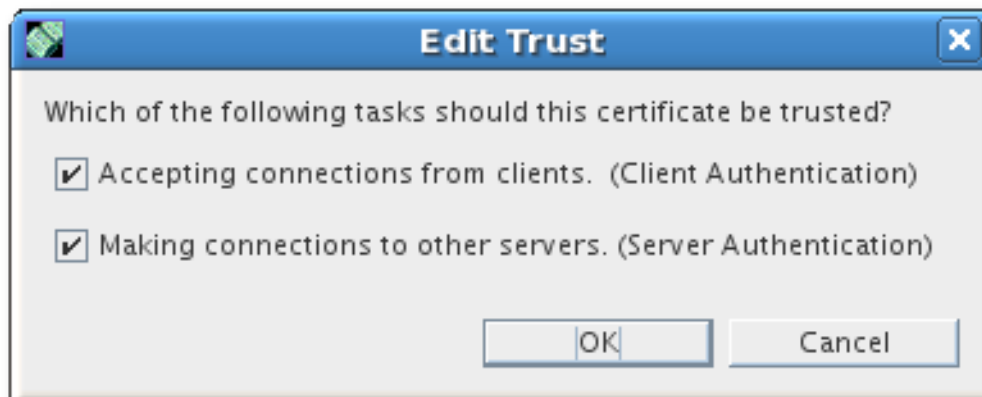
7.3.4. Changing the CA Trust Options

It is sometimes necessary to reject certificates issued by a generally trusted CA. The trust settings on CA certificates installed in the Directory Server can be untrusted, trusted, or change the operations for which it is trusted.

1. In the **Tasks** tab, click the **Manage Certificates** button.
2. Click the **CA Certs** tab.



3. Select the CA certificate to edit.
4. Click the **Edit Trust** button.
5. Set the CA trust options.

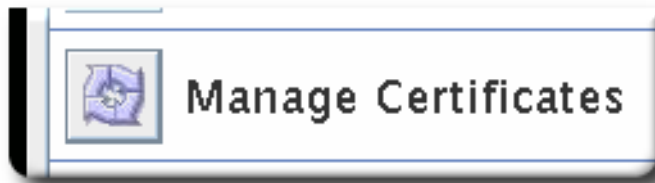


- *Accepting connections from clients (Client Authentication)*. This option sets whether to accept client, or user, certificates issued by the CA.
- *Making connections to other servers (Server Authentication)*. This option sets whether to accept server certificates issued by the CA.
- Click **OK**.

7.3.5. Changing Security Device Passwords

Periodically change the settings for the security databases or devices.

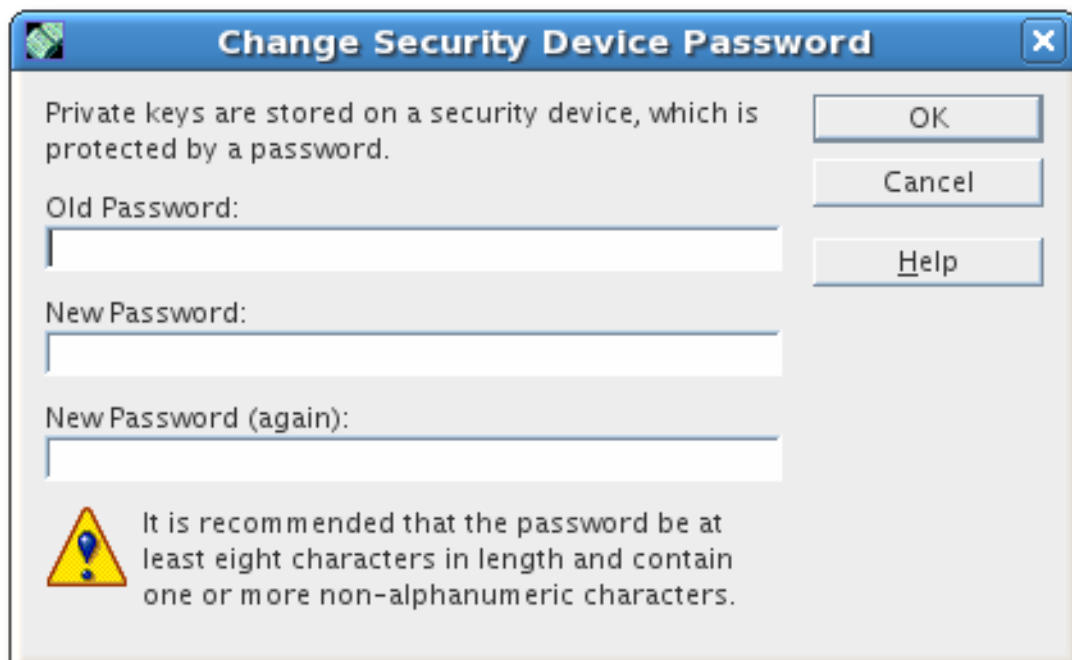
1. In the **Tasks** tab, click the **Manage Certificates** button.



2. In the top of the window, choose a security device from the drop-down list.



3. Click the **Password** button.
4. In the **Change Security Device Password** dialog box, enter the old password, and then enter and confirm the new password.

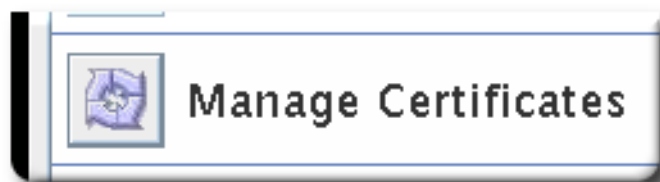


5. Click **OK**.

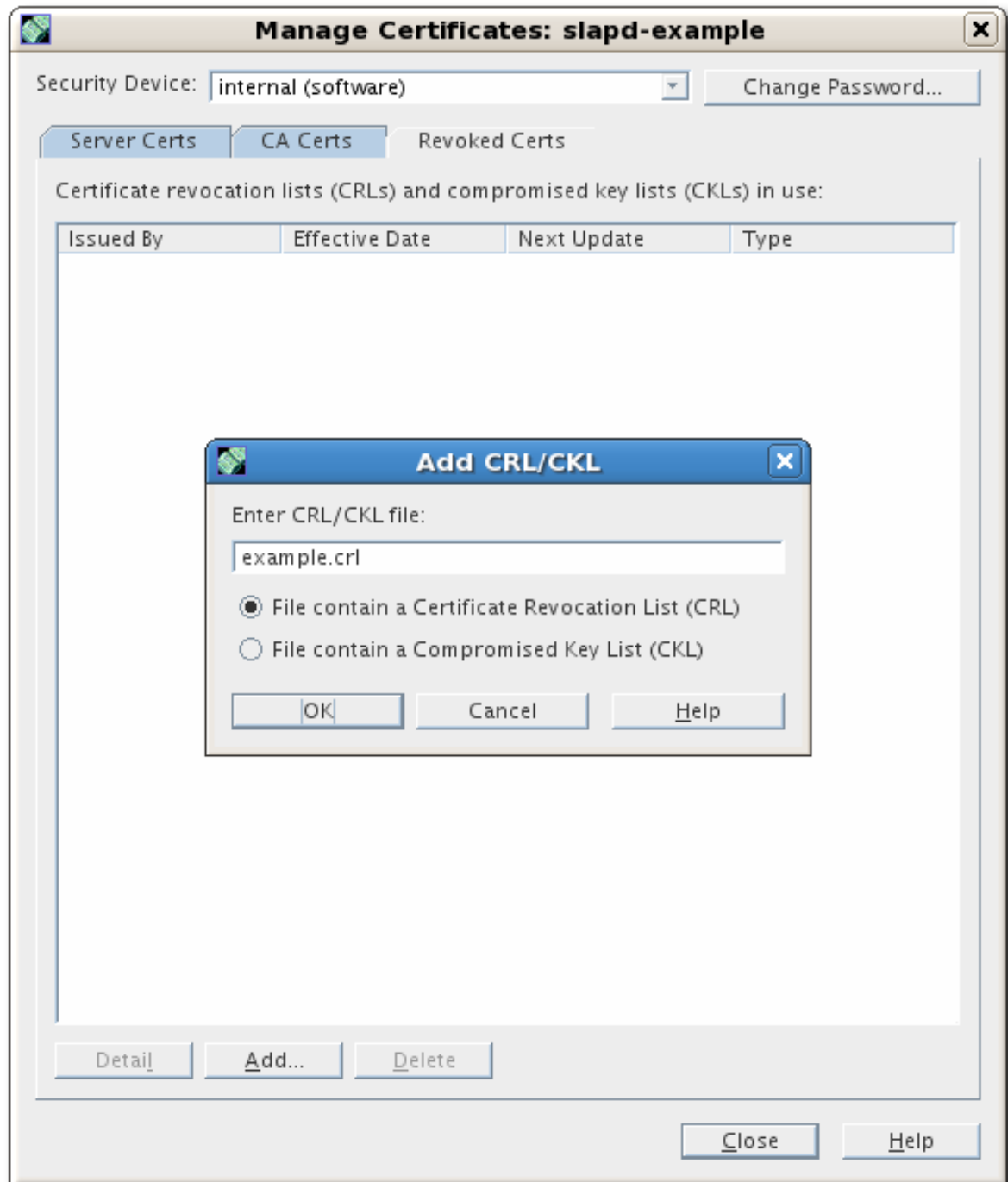
7.3.6. Adding Certificate Revocation Lists

Certificate revocation lists (CRLs) allow CAs to specify certificates that client or server users should no longer trust. If data in a certificate changes, a CA can revoke the certificate and list it in a CRL. CRLs are produced and periodically updated by a CA, so updated CRLs can be added to the Directory Server.

1. Obtain the CRL from the CA; these can usually be downloaded from the CA's website.
2. In the **Tasks** tab, click the **Manage Certificates** button.



3. Select the **Revoked Certs** tab.
4. To add a CRL, click **Add** at the bottom of the window, and enter the full path to the CRL file.



5. Click **OK**.

7.3.7. Managing Certificates Used by the Directory Server Console

The certificates and keys used by the server are stored in NSS security databases in the `/etc/dirsrv/slapd-instance_name` directory. The Directory Server Console itself also uses certificates and keys for SSL/TLS connections; these certificates are stored in a separate database in the user's home directory. If the Directory Server Console is used to connect to multiple instances of Directory Server over SSL, then it is necessary to trust every CA which issued the certificates for every Directory Server instance.

When SSL/TLS is enabled for the Directory Server Console ([Section 7.4.3, "Enabling TLS/SSL in the Directory Server, Admin Server, and Console"](#)), the Directory Server Console must have a copy of the issuing CA certificate for it to trust the server's SSL client certificates. Otherwise, the Console will return errors about not trusting the CA which issued the certificate.

**NOTE**

Only the CA certificates for the CA which issued the server's SSL certificate is required. The Directory Server Console does not require its own SSL client certificate.

The Console's security databases are managed directly using **certutil**. The **certutil** tool is described in more detail in [Section 7.6, “Using certutil”](#) and in the NSS tool manpages.

To list the certificates in the security database:

```
certutil -d $HOME/.redhat-idm-console -L
```

To add a CA certificate to the database:

```
certutil -d $HOME/.redhat-idm-console -A -t CT,, -a -i /path/to/cacert.asc
```

**NOTE**

If you are running the Directory Server Console on Windows,

- the security database is located in the **C:\Documents and Settings\user_name\.389-console** directory.
- change to the **C:\Program Files\Red Hat Identity Management Console** directory to run the **certutil.exe** command.

7.4. SETTING UP TLS/SSL

To provide secure communications over the network, Red Hat Directory Server includes the LDAPS communications protocol. LDAPS is the standard LDAP protocol, running over Transport Layer Security (TLS, formerly Secure Sockets Layer or SSL). Directory Server also allows *spontaneous* secure connections over otherwise-insecure LDAP ports, using the Start TLS LDAP extended operation.

7.4.1. TLS/SSL in Directory Server

The Directory Server supports TLS/SSL to secure communications between LDAP clients and the Directory Server, between Directory Servers that are bound by a replication agreement, or between a database link and a remote database. Directory Server can use TLS/SSL with simple authentication (bind DN and password) or with certificate-based authentication.

Directory Server's cryptographic services are provided by Mozilla Network Security Services (NSS), a library of TLS/SSL and base cryptographic functions. NSS includes a software-based cryptographic token which is FIPS 140-2 certified.

Using TLS/SSL with simple authentication ensures confidentiality and data integrity. There are two major benefits to using a certificate — smart card, token, or software-based — to authenticate to the Directory Server instead of a bind DN and password:

- *Improved efficiency.* When using applications that prompt once for the certificate database password and then use that certificate for all subsequent bind or authentication operations, it is more efficient than continuously providing a bind DN and password.
- *Improved security.* The use of certificate-based authentication is more secure than non-

certificate bind operations because certificate-based authentication uses public-key cryptography. Bind credentials cannot be intercepted across the network. If the certificate or device is lost, it is useless without the PIN, so it is immune from third-party interference like phishing attacks.

The Directory Server is capable of simultaneous TLS/SSL and non-SSL communications. This means that you do not have to choose between TLS/SSL or non-SSL communications for the Directory Server; both can be used at the same time. Directory Server can also utilize the Start TLS extended operation to allow TLS/SSL secure communication over a regular (insecure) LDAP port.

There are four steps to enable TLS/SSL:

1. Obtain and install a certificate for the Directory Server, and configure the Directory Server to trust the certification authority's (CA's) certificate.

For information, see [Section 7.3.1, “Obtaining and Installing Server Certificates”](#).

2. Turn on TLS/SSL in the directory.

For information, see [Section 7.4, “Setting up TLS/SSL”](#).

3. Configure the Admin Server connect to a TLS-enabled Directory Server.

4. Optionally, ensure that each user of the Directory Server obtains and installs a personal certificate for all clients that will authenticate with TLS/SSL.

For information, see [Section 7.10.1, “Configuring Directory Server to Accept Certificate-Based Authentication from LDAP Clients”](#).

Most of the time, the server should run with TLS/SSL enabled. If TLS/SSL is temporarily disabled, re-enable it before processing transactions that require confidentiality, authentication, or data integrity.

Before TLS/SSL can be activated, first create a certificate database, obtain and install a server certificate, and trust the CA's certificate, as described in [Section 7.3.1, “Obtaining and Installing Server Certificates”](#).

With TLS/SSL enabled, when the server restarts, it prompts for the PIN or password to unlock the key database. This is the same password used when the server certificate and key were imported into the database. Restarting the Directory Server without the password prompt is possible by using a hardware crypto device or creating a PIN file ([Section 7.4.4, “Creating a Password File for the Directory Server”](#)).

NOTE

On TLS-enabled servers, be sure to check the file permissions on certificate database files, key database files, and PIN files to protect the sensitive information they contain. Because the server does not enforce read-only permissions on these files, check the file modes to protect the sensitive information contained in these files.

The files must be owned by the Directory Server user, such as the default **nobody**. If you set a different account during the installation, like Red Hat recommends, use this user and group for a better security instead. The key and cert databases should be owned by the Directory Server user and should typically have read/write access for the owner with no access allowed to any other user (mode **0600**). The PIN file should also be owned by the Directory Server user and set to read-only by this user, with no access to anyone other user (mode **0400**).



WARNING

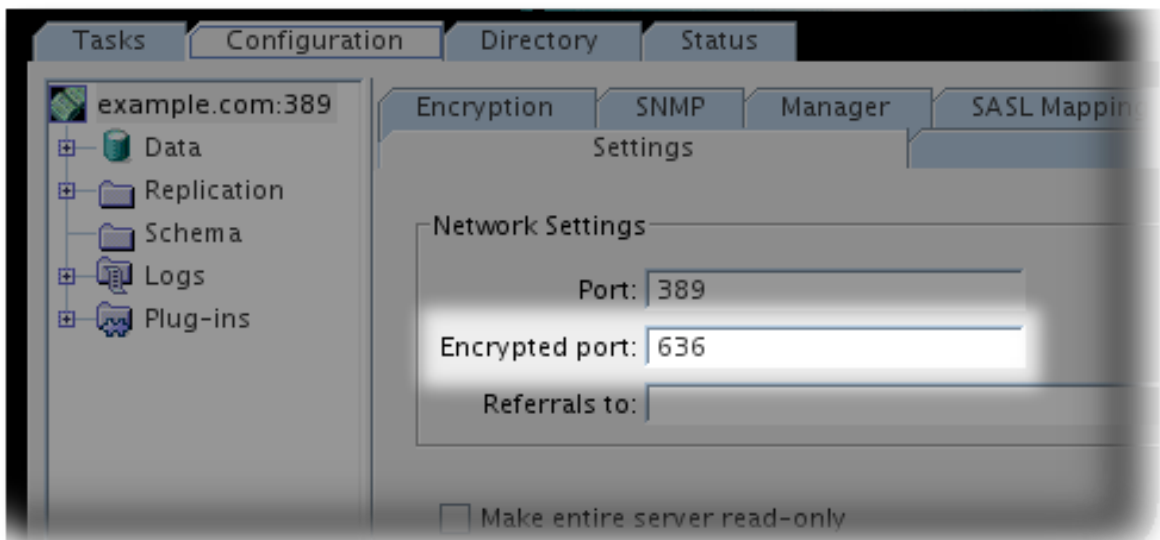
The Directory Server must already be configured to run in SSL ([Section 7.4.2, “Enabling TLS/SSL Only in the Directory Server”](#)) and **the server must already have been restarted** before the Directory Server Console can be configured to use SSL. Configuring SSL/TLS for the server requires a server restart to load the new configuration, including the new secure port assignment. However, enabling SSL/TLS for the Console takes effect immediately. Therefore, if the Console has SSL/TLS enabled before the server is running in SSL/TLS, then the Console loses the connection to the server and cannot reconnect.

To disable SSL/TLS in the Directory Server Console, use `ldapmodify` to edit the `nsServerSecurity` attribute:

```
nsServerSecurity: off
```

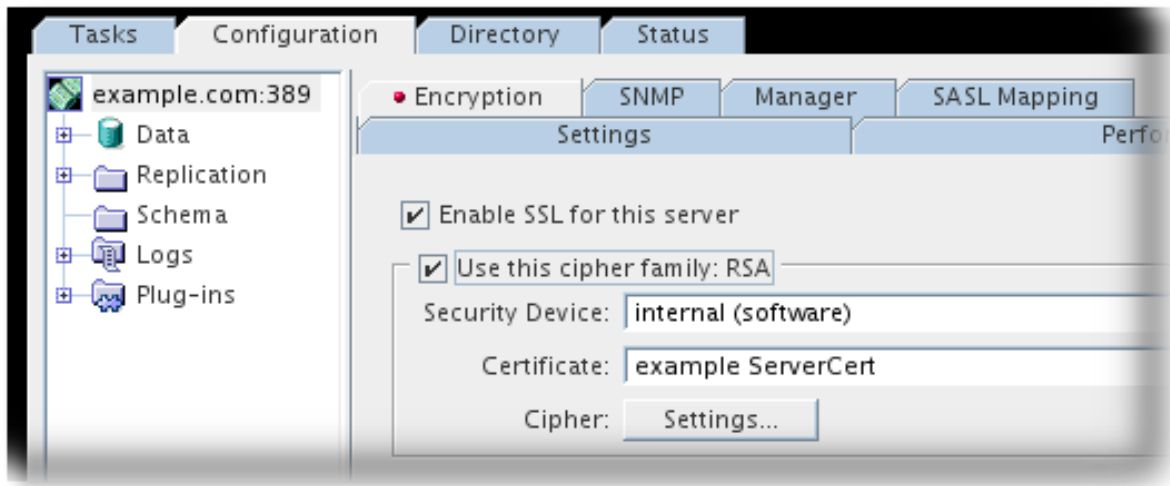
7.4.2. Enabling TLS/SSL Only in the Directory Server

1. Obtain and install CA and server certificates.
2. Set the secure port for the server to use for TLS/SSL communications. In the **Configuration** area, select the **Settings** tab, and enter the value in the **Encrypted Port** field.

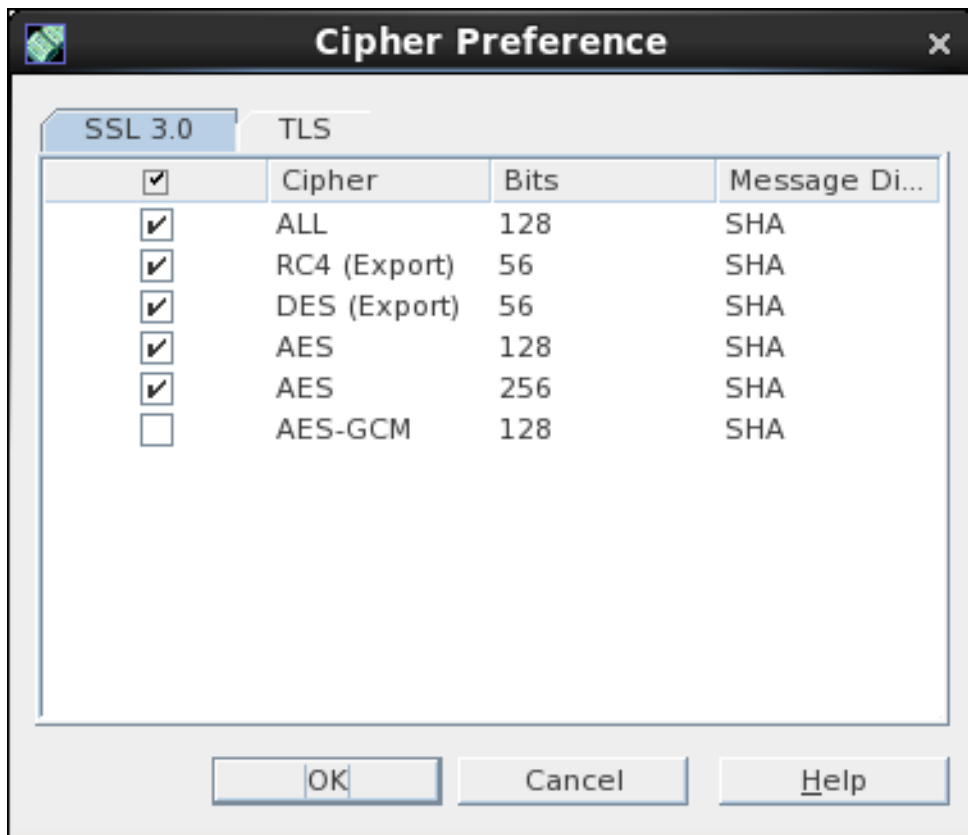


The encrypted port number *must not* be the same port number used for normal LDAP communications. By default, the standard port number is **389**, and the secure port is **636**.

3. Select the **Configuration** tab, and then select the top entry in the navigation tree in the left pane. Select the **Encryption** tab in the right pane.
4. Select the **Enable SSL for this Server** check box.



5. Check the **Use this Cipher Family** check box.
6. Select the certificate to use from the drop-down menu.
7. Click **Cipher Settings**.



By default, **ALL** in the **TLS** tab is selected. When **ALL** is set, the server selects safe ciphers internally.

8. Set the preferences for client authentication.



- *Do not allow client authentication.* With this option, the server ignores the client's certificate. This does not mean that the bind will fail.
- *Allow client authentication.* This is the default setting. With this option, authentication is performed on the client's request. For more information about certificate-based authentication, see [Section 7.10, "Using Client \(Certificate-Based\) Authentication"](#).
- *Require client authentication.* With this option, the server requests authentication from the client.

If TLS/SSL is only enabled in the Directory Server and not the Directory Server Console, do not select **Require client authentication** check box.



NOTE

To use certificate-based authentication with replication, the consumer server must be configured either to allow or to require client authentication.

- To verify the authenticity of requests, select the **Check host name against name in certificate for outbound SSL connections** option. The server does this verification by matching the host name against the value assigned to the common name (**cn**) attribute of the subject name in the being presented for authentication.



By default, this feature is disabled. If it is enabled and if the host name does not match the **cn** attribute of the certificate, appropriate error and audit messages are logged. For example, in a replicated environment, messages similar to these are logged in the supplier server's log files if it finds that the peer server's host name does not match the name specified in its certificate:

```
[DATE] - SSL alert: ldap_sasl_bind("",LDAP_SASL_EXTERNAL) 81
(Netscape runtime error -12276 -
  Unable to communicate securely with peer: requested domain name
  does not match the server's
  certificate.)
[DATE] NSMMReplicationPlugin - agmt="cn=to ultra60 client auth"
(ultra60:1924): Replication
  bind with SSL client authentication failed: LDAP error 81
(Cannot contact LDAP server)
```

Red Hat recommends enabling this option to protect Directory Server's outbound SSL connections against a man-in-the-middle (MITM) attack.

10. Click **Save**.
11. Restart the Directory Server. The Directory Server must be restarted from the command line.

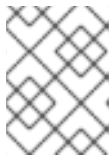
```
service dirsrv restart instance
```

When the server restarts, it prompts for the PIN or password to unlock the key database. This is the same password used when the server certificate and key were imported into the database.

To restart the Directory Server without the password prompt, create a PIN file or use a hardware crypto device. See [Section 7.4.4, “Creating a Password File for the Directory Server”](#) for information on how to create a PIN file.

7.4.3. Enabling TLS/SSL in the Directory Server, Admin Server, and Console

1. Obtain server certificates and CA certs, and install them on the Directory Server. This is described in [Section 7.3.1, “Obtaining and Installing Server Certificates”](#).
2. Obtain and install server and CA certificates on the Admin Server. This is a similar process as for the Directory Server.



NOTE

It is important that the Admin Server and Directory Server have a CA certificate in common so that they can trust the other's certificates.

3. Configure TLS/SSL for the Directory Server as described in [Section 7.4.2, “Enabling TLS/SSL Only in the Directory Server”](#).
4. Require SSL/TLS to connect to the Directory Server Console.



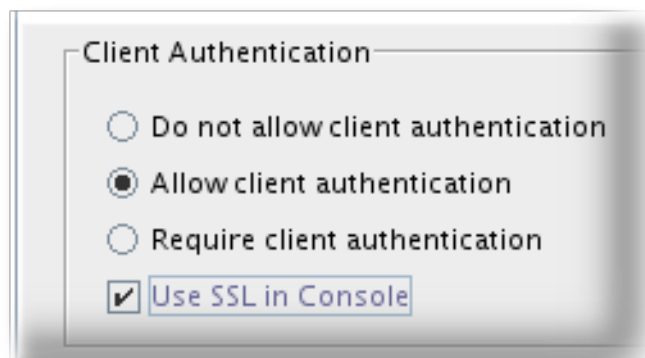
WARNING

The Directory Server must already be configured to run in SSL and **the server must already have been restarted** before the Directory Server Console can be configured to use SSL. Configuring SSL/TLS for the server requires a server restart to load the new configuration, including the new secure port assignment. However, enabling SSL/TLS for the Console takes effect immediately. Therefore, if the Console has SSL/TLS enabled before the server is running in SSL/TLS, then the Console loses the connection to the server and cannot reconnect.

To disable SSL/TLS in the Directory Server Console, use `ldapmodify` to edit the `nsServerSecurity` attribute:

```
nsServerSecurity: off
```

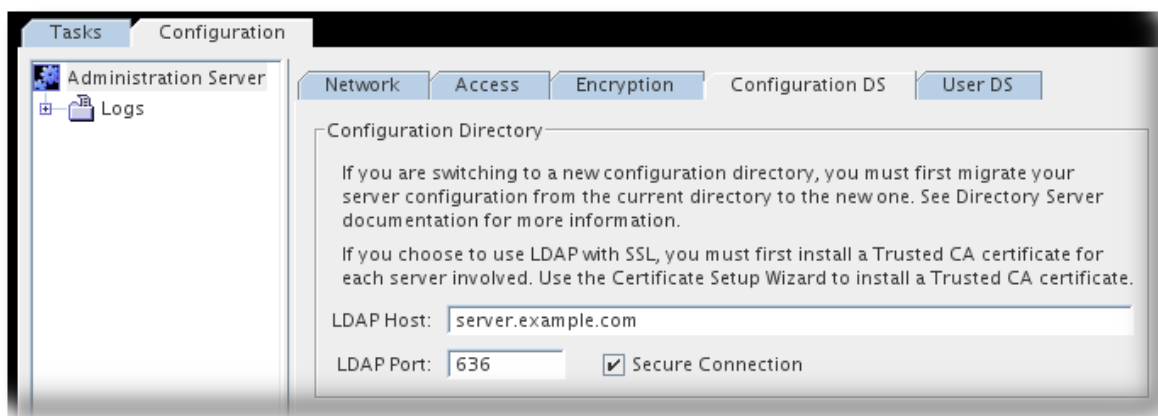
1. Reopen the Directory Server Console.
2. In the **Configuration** tab, select the server, and open the **Encryption** tab.
3. Check the **Use SSL in the Console** box.



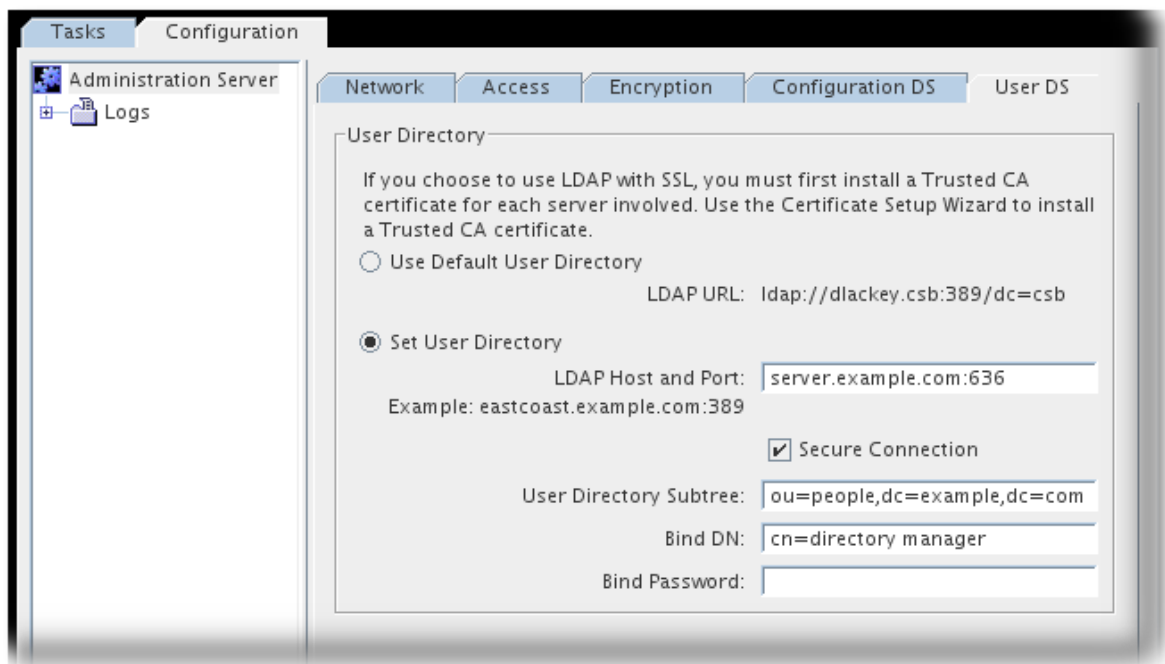
5. In the Admin Server Console, select the **Configuration** tab. Select the **Encryption** tab, check the **Enable SSL** check box, and fill in the appropriate certificate information.



6. In the **Configuration DS** tab, change the port number to the new Directory Server secure port information. See [Section 1.6, “Changing Directory Server Port Numbers”](#) for more information. Do this even if the default port of **636** is used. Check the **Secure Connection** check box.



7. In the **User DS** tab, select the **Set User Directory** radio button, and fill in the Directory Server secure port information, the LDAP URL, and the user database information. Check the **Secure Connection** check box.



8. Save the new TLS/SSL settings and **Configuration DS** and **User DS** information in the Admin Server Console.
9. Restart the Admin Server. The server must be restarted from the command line.

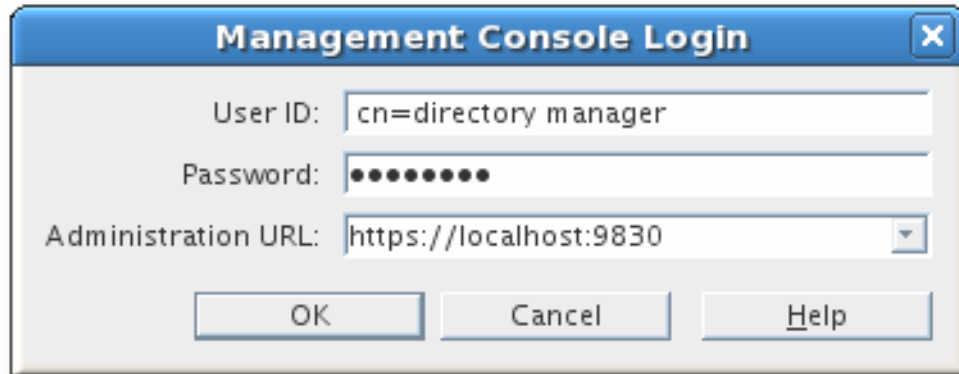
```
service dirsrv-admin restart
```

When the server restarts, it prompts for the PIN or password to unlock the key database. This is the same password used when the server certificate and key were imported into the database.

To restart the Admin Server without the password prompt, create a PIN file or use a hardware crypto device. See [Section E.2.9.4, “Creating a Password File for the Admin Server”](#) for information on how to create a PIN file.

NOTE

When next logging into the Directory Server Console, be certain that the address reads **https**; otherwise, the operation will time out, unable to find the server since it is running on a secure connection. After successfully connecting, a dialog box appears to accept the certificate. Click **OK** to accept the certificate (either only for that current session or permanently).



7.4.4. Creating a Password File for the Directory Server

It is possible to store the certificate password in a password file. By placing the certificate database password in a file, the server can be started from the Directory Server Console and also restarted automatically when running unattended.

**WARNING**

This password is stored in clear text within the password file, so its usage represents a significant security risk. Do not use a password file if the server is running in an unsecured environment.

The password file must be in the same directory where the other key and certificate databases for Directory Server are stored. This is usually the main configuration directory, **/etc/dirsrv/slaped-*instance_name***. The file should be named **pin.txt**.

Include the token name and password in the file. For example:

```
Internal (Software) Token:secret
```

For the NSS software crypto module (the default software database), the token is always called **Internal (Software) Token**.

The PIN file should be owned by the Directory Server user and set to read-only by the Directory Server user, with no access to anyone other user (mode **0400**).

7.4.5. Starting the Directory Server with Expired Certificates

If the Directory Server is configured to run in SSL and its certificate expires, then the Directory Server cannot be started. Because this can cut off access to user and authentication information, as well as other directory data, Directory Server has an option to set how it handles an expired certificate.

The ***nsslapd-validate-cert*** parameter sets how the Directory Server should respond when it attempts to start with an expired certificate:

- **warn** allows the Directory Server to start successfully with an expired certificate, but it sends a warning message that the certificate has expired. This is the default setting.
- **on** validates the certificate and will prevent the server from restarting if the certificate is expired. This sets a hard failure for expired certificates.
- **off** disables all certificate expiration validation, so the server can start with an expired certificate without logging a warning.

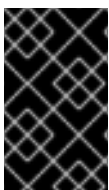
To change the behavior of the expired certificate setting, edit the ***nsslapd-validate-cert*** parameter under **cn=config**:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -D
"cn=directory manager" -w secret -p 636 -h server.example.com -x

dn: cn=config
changetype: modify
replace: nsslapd-validate-cert
nsslapd-validate-cert: on
```

7.5. COMMAND-LINE FUNCTIONS FOR START TLS

LDAP client tools such as **ldapmodify**, **ldapsearch**, and **ldapdelete** can use TLS/SSL when communicating with an SSL-enabled server or to use certificate authentication. Command-line options also specify or enforce Start TLS, which allows a secure connection to be enabled on a clear text port after a session has been initiated.



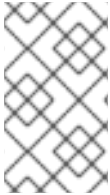
IMPORTANT

For Start TLS to work, the environment variables for the SSL/TLS databases must be configured. This is described in [Section A.2, “Using SSL/TLS and Start TLS with LDAP Client Tools”](#).

In the following example, a network administrator enforces Start TLS for a search for Mike Connor's identification number:

```
ldapsearch -ZZ -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -b "uid=mconnors,ou=people,dc=example,dc=com" "
(attribute=govIdNumber)"
```

-ZZ enforces Start TLS.

**NOTE**

The **-ZZ** option enforces the use of Start TLS, and the server must respond that a Start TLS command was successful. If the **-ZZ** command is used and the server does not support Start TLS, the operation is aborted immediately.

With the **-Z** option, the following errors could occur:

- If there is no certificate database, the operation fails. See [Section 7.3.1, “Obtaining and Installing Server Certificates”](#) for information on using certificates.
- If the server does not support Start TLS, the connection proceeds in clear text. To enforce the use of Start TLS, use the **-ZZ** command option.
- If the certificate database does not have the certificate authority (CA) certificate, the connection proceeds in clear text. See [Section 7.3.1, “Obtaining and Installing Server Certificates”](#) for information on using certificates.

With the **-ZZ** option, the following errors could occur, causing the Start TLS operation to fail:

- If there is no certificate database. See [Section 7.3.1, “Obtaining and Installing Server Certificates”](#) for information on using certificates.
- If the certificate database does not have the certificate authority (CA) certificate. See [Section 7.3.1, “Obtaining and Installing Server Certificates”](#) for information on using certificates.
- The server does not support Start TLS as an extended operation.

For SDK libraries used in client programs, if a session is already in TLS mode and Start TLS is requested, then the connection continues to be in secure mode but prints the error **"DSA is unwilling to perform"**.

7.6. USING CERTUTIL

The Directory Server has a command-line tool, **certutil**, which locally creates self-signed CA and client certificates, certificate databases, and keys. The default location for the Directory Server certutil tool is **/usr/bin**.

certutil can also be downloaded from <https://ftp.mozilla.org/pub/security/nss/releases/>.

7.6.1. Creating Directory Server Certificates through the Command Line

The following steps outline how to make the databases, key, CA certificate, server/client certificate, and convert the certificates into **pkcs12** format.

1. Open the directory where the Directory Server certificate databases are stored.

```
[root@server ~]# cd /etc/dirsrv/slapd-instance_name
```

2. Make a backup copy of all of the files in the directory as a precaution. If something goes awry while managing certificates, the databases can then be restored. For example:

```
[root@server ~]# tar -cf /tmp/db-backup.tar *
```

3. Create a password file for the security token password.

```
[root@server ~]# vi /tmp/pwdfilename
secretpw
```

This password locks the server's private key in the key database and is used when the keys and certificates are first created. The password in this file is also the default password to encrypt PK12 files used by **pk12util**. Because this password is stored in plaintext, the password file should be owned by the user as which Directory Server runs, by default **nobody**. If a different account was set during the installation, like Red Hat recommends, use this account and group instead for a higher security. The password file must be set as read-only for the Directory Server user and allow no access to anyone else (mode **0400**). It's a good idea to have a secure backup of this file.

4. Set the environment variable for the shell to include the **certutil** directory path. For example:

```
[root@server ~]# export PATH=/usr/bin/:$PATH
```

The command varies depending on the shell.

5. Create the key and certificate databases.

```
[root@server ~]# certutil -N -d . -f /tmp/pwdfilename
```

6. Generate the self-signed CA certificate. **certutil** creates the required key pairs and the certificate. This certificate is used to generate the other server certificates and can be exported for use with other servers and clients.

```
[root@server ~]# certutil -S -n "CA certificate" -s "cn=My Org CA
cert,dc=example,dc=com" -2 -x -t "CT,," -m 1000 -v 120 -d . -k rsa -
f /tmp/pwdfilename
```

7. Generate the Directory Server client certificate.

```
[root@server ~]# certutil -S -n "Server-Cert" -s "cn=FQDN" -c "CA
certificate" -t "u,u,u" -m 1001 -v 120 -d . -k rsa -f /tmp/pwdfilename
```

The value of the **-s** argument is very important. The leftmost RDN must be **cn=FQDN** (where *FQDN* is the fully-qualified host and domain name of the Directory Server). For example, to issue a certificate for a server with the name **ldap.example.com**, specify at least **-s "cn=ldap.example.com"**; it is beneficial to have a more descriptive name to help with server identification, such as **"cn=ldap.example.com,ou=DS1"**. The FQDN must be available for DNS and reverse DNS lookups to Directory Server clients because certificate validation may fail if the clients cannot properly resolve the FQDN, and some clients refuse to connect if a server certificate does not have its FQDN in the subject. Additionally, using the format **cn=hostname.domain** is essential for Directory Server clients to protect themselves from man in the middle attacks.

**NOTE**

There should only be one **cn** in a certificate subject name. To add detail to the subject name, use **cn** as the RDN and another attribute — like **ou**, **l**, or **c** — for the other subject name elements.

To provide a `subjectAltName`, as well as the nickname, use the **-8** argument in addition to the **-s** argument.

To use the Directory Server behind a DNS round robin or any other scheme which aliases a single server certificate to multiple host names, see the TLS/SSL information about server name wildcards or `subjectAltName`.

Server certificates for other servers are created using a similar command as for the Directory Server certificate. Make sure that every **-n** option (nickname) and **-m** option (serial number) is unique for every certificate, and make sure that the **-s** option gives the correct FQDN for the server.

**NOTE**

Keep careful track on the numbers set with the **-m** option. The **-m** option sets the unique identifier for the server certificate, and a CA cannot issue two certificates with the same ID. Keep a log of issued serial numbers so that no number is ever duplicated.

8. Export the CA certificate for use with other servers and clients. A client usually requires the CA certificate to validate the server certificate in an TLS/SSL connection. Use **certutil** to export the CA certificate in ASCII/PEM format:

```
[root@server ~]# certutil -d . -L -n "CA certificate" -a >
cacert.asc
```

The way that the CA certificate is imported is different for every client. For example, **certutil** can import a CA certificate into another Directory Server certificate database:

```
[root@server ~]# cd /etc/dirsrv/slapd-otherserver
[root@server ~]# certutil -A -d . -n "CA certificate" -t "CT,," -a -
i cacert.asc
```

9. Use **pk12util** to export other server certificates and keys created with **certutil** so that they can be used on a remote server.

```
[root@server ~]# pk12util -d . -o ldap1.p12 -n Server-Cert -w
/tmp/pwdfilename -k /tmp/pwdfilename
```

The **-w** argument is the password used to encrypt the **.p12** file for transport. The **-k** argument specifies the password for the key database containing the server certificate being exported to **.p12**.

10. If the Directory Server will run with TLS/SSL enabled, then create a password file (**pin.txt**) for the server to use so it will not prompt you for a password every time it restarts. Creating the password file is described in [Section 7.4.4, "Creating a Password File for the Directory Server"](#).

The certificates created by **certutil** are automatically available in the **Encryption** tab of the Console. There is no need to import them because they are already in the certificate database.

7.6.2. certutil Usage

certutil can be used for a variety of tasks to manage certificates and keys, such as generating certificate requests and removing certificates from the certificate database. Some of the most common options are listed in [Table 7.1, “certutil Options”](#). For the full list of commands and arguments, run **certutil -H** from the command line.

Table 7.1. certutil Options

Options	Description
-x	Creates a self-signed CA certificate.
-S	Creates a server or client certificate.
-R	Generates a certificate request.
-N	Creates new security databases.
-L	Lists all of the certificates in the database.
-A	Adds a certificate to the certificate database.
-n	Gives the name of the certificate.
-d	Certificate database directory; this is the directory for the subsystem instance.
-m	The serial number for the certificate.
-k	The key type to use; the only option is rsa .
-g	The key size. The recommended size for RSA keys is 2048.
-s	The subject name of the certificate.
-t	The trust arguments for the certificate, meaning the purposes for which the certificate is allowed to be used.
-v	The validity period, in months.

Options	Description
numbers 1-8	<p>These set the available certificate extensions. Only eight can be specified through the certutil tool:</p> <ul style="list-style-type: none"> • Key Usage: 1 • Basic Constraints: 2 • Certificate Authority Key ID: 3 • CRL Distribution Point: 4 • Netscape Certificate Type: 5 • Extended Key Usage: 6 • Email Subject Alternative Name: 7 • DNS Subject Alternative Name: 8
-a	Outputs the certificate request to an ASCII file instead of binary.
-o	The output file to which to save the certificate request.
-i	An input file containing a certificate.
-f	The path to a password file for the security databases password.

Table 7.2, “[certutil Examples](#)” has some of the common uses for the **certutil** command.

Table 7.2. certutil Examples

Example	Description
<code>certutil -L -d .</code>	Lists the certificates in the database.
<code>certutil -N -d .</code>	Creates new key (key3.db) and certificate (cert8.db) databases.
<code>certutil -S -n "CA certificate" -s "cn=My Org CA cert,dc=example,dc=com" -2 -x -t "CT,," -m 1000 -v 120 -d . -k rsa</code>	Creates a self-signed CA certificate.
<code>certutil -S -n "Server-Cert" -s "cn=FQDN" -c "CA certificate" -t "u,u,u" -m 1001 -v 120 -d . -k rsa</code>	Creates a client certificate.

Example	Description
<code>certutil -L -d . -n "cert_name"</code>	"Pretty prints" the specified certificate; the <i>cert_name</i> can specify either a CA certificate or a client certificate.
<code>certutil -L -d . -n "cert_name" > certfile.asc</code>	Exports the specified certificate out of the database to ASCII (PEM) format.
<code>certutil -L -d . -n "cert_name" -r > certfile.bin</code>	Exports the specified certificate out of the database to binary format; this can be used with Directory Server attributes such as <i>usercertificate;binary</i> .

7.7. UPDATING ATTRIBUTE ENCRYPTION FOR NEW SSL/TLS CERTIFICATES

When TLS/SSL is first configured, there is no problem with attribute encryption. However, if the TLS/SSL certificate is changed, then attribute encryption fails, with messages like these:

```
Apr  4 13:00:35 smtp1 logger: [04/Apr/2017:13:00:34 -0700] -
attrcrypt_unwrap_key: failed to unwrap key for cipher AES
Apr  4 13:00:35 smtp1 logger: [04/Apr/2017:13:00:34 -0700] - Failed to
retrieve key for cipher AES in attrcrypt_cipher_init
Apr  4 13:00:35 smtp1 logger: [04/Apr/2017:13:00:34 -0700] - Failed to
initialize cipher AES in attrcrypt_init
Apr  4 13:00:35 smtp1 logger: [04/Apr/2017:13:00:34 -0700] -
attrcrypt_unwrap_key: failed to unwrap key for cipher AES
Apr  4 13:00:35 smtp1 logger: [04/Apr/2017:13:00:34 -0700] - Failed to
retrieve key for cipher AES in attrcrypt_cipher_init
Apr  4 13:00:35 smtp1 logger: [04/Apr/2017:13:00:34 -0700] - Failed to
initialize cipher AES in attrcrypt_init
```

This is because the previously-generated keys do not work with the new server certificate. To correct these errors, force the server to generate new keys for attribute encryption:

1. Stop the server.

```
service dirsrv stop
```

2. Open the **dse.ldif** file.

```
vim /etc/dirsrv/dse.ldif
```

3. There are special encryption key entries for the encryption ciphers used for attribute encryption under the database configuration. For example:

```
dn: cn=AES,cn=encrypted attribute keys,cn=userRoot,cn=ldbm
database,cn=plugins,cn=config
objectClass: top
objectClass: extensibleObject
```

```
cn: AES
nssymmetrickey::
mSLm/RlCLvPZrSdARHPowedF9zKx+kjVTww5ARE4w01b12Y1YvrIg6mUlSsmzMfdhc1B
BURhFDNwwUDisHwwiMJRIvHXstx5EGstWE9xokIU+xeMZF8cPJrY1udFSPFc0iKyiC0a
PacWpomn/XPavXkQqBwk4vJzrHHhH1o3bNg=
```

Delete these entries.

4. Start the server again.

```
service dirsrv start
```

As soon as the server restarts, it generates new encryption keys for the encrypted attribute keys.

7.8. USING HARDWARE SECURITY MODULES

A security module serves as a medium between the Directory Server and the SSL layer. The module stores the keys and certificates used for encryption and decryption. The standard which defines these modules is Public Key Cryptography Standard (PKCS) #11, so these modules are PKCS#11 modules.

By default, Directory Server uses built-in security databases, **key3.db** and **cert8.db**, to store the keys and certificates used by the servers.

It is also possible to use external security devices to store Directory Server certificates and keys. For Directory Server to use an external PKCS#11 module, the module's drivers must be installed in Directory Server.



NOTE

It is possible to install PKCS #11 modules through the Directory Server Console, but it is recommended that administrators use the command line to add modules. Because of differences in the way manufacturers implement their modules, some modules (such as nCipher external tokens) must be loaded through the command line rather than the Directory Server Console. For all modules, using the command line is an easier, more effective way to add modules.

7.8.1. Installing PKCS#11 Modules Through the Command Line

Installing PKCS#11 modules through the command line requires the NSS **modutil** tool.

1. Connect the external security device, and install its drivers on the server machine.
2. Use the **-add** and **-libfile** arguments with **modutil** to load the PKCS#11 module and its associated libraries. The *module_name* is the name of the module given when the security device's drivers were installed; the *library_file* is the full path to the module's library.

```
[root@server ~]# modutil -dbdir /etc/dirsrv/slapd-instance_name/ -
nocertdb -add module_name -libfile library_file
```

For example, for an nCipher hardware security module:

```
[root@server ~]# modutil -dbdir /etc/dirsrv/slapd-instance_name/ -
nocertdb -add nethsm -libfile
/opt/nfast/toolkits/pkcs11/libcknfast.so
```

3. List the loaded modules to make sure that the new PKCS#11 module was added correctly.

```
[root@server ~]# modutil -list -dbdir
/etc/dirsrv/slapd-instance_name/
```

Listing of PKCS #11 Modules

```
-----
1. NSS Internal PKCS #11 Module
   slots: 2 slots attached
   status: loaded

   slot: NSS Internal Cryptographic Services
   token: NSS Generic Crypto Services

   slot: NSS User Private Key and Certificate Services
   token: NSS Certificate DB

2. nCipher NetHSM PKCS #11 Module
   slots: 2 slots attached
   status: loaded

   slot: nethsm external cryptographic services
   token: nethsm crypto services

   slot: nethsm user private key and certificate services
   token: nethsm certificate db
-----
```

7.8.2. Adding Certificates to an HSM

All of the certificates required for secure connections must be imported into the HSM. Because of SELinux requirements, the best method to import certificates into the Directory Server databases is using the **certutil** command.

At a minimum, you will need to import the Directory Server and Admin Server server certificates, as well as the CA certificate for the CA which issued those certificates. If there are other certificates used by the Directory Server or Admin Server, then those must also be imported.

For example:

```
certutil -A -d /etc/dirsrv/slapd-instance_name -n "CA certificate" -t
"CT,," -a -i /tmp/cacert.asc

certutil -A -d /etc/dirsrv/slapd-instance_name -n "Server-Cert" -t "u,u,u"
-a -i /tmp/cert.asc
```

certutil is described more in [Section 7.6, “Using certutil”](#).

7.8.3. Setting SELinux Policies for HSMs

In most situations, it will be necessary to create an SELinux policy module to allow the Directory Server to use the certificates stored in the HSM. However, the requirements for writing those policies are different depending on the vendor and model of the HSM.

For a basic primer in writing SELinux policies, see [Dan Walsh's article on writing SELinux modules](#). The Red Hat Enterprise Linux documentation contains information on troubleshooting policy problems in [the SELinux Guide](#). For additional help with writing HSM-related policies, contact Red Hat support.

7.9. SETTING ENCRYPTION CIPHERS

The Directory Server supported several different ciphers, and the type of ciphers to use for TLS/SSL communications are set by the user. A *cipher* is the algorithm used in encryption. Some ciphers are more secure, or stronger, than others. Generally speaking, the more bits a cipher uses during encryption, the more difficult it is to decrypt the key.

When a client initiates an TLS/SSL connection with a server, the client tells the server what ciphers it prefers to use to encrypt information. In any two-way encryption process, both parties must use the same ciphers. There are a number of ciphers available. The server needs to be able to use the ciphers that will be used by client applications connecting to the server.

7.9.1. Available Ciphers

This section lists information about the available ciphers for Directory Server encryption. Each cipher has the following information:

- *Directory Server name.* The name of the cipher suite used when configuring the Directory Server. The Directory Server uses this name both internally and in the Directory Server Console.
- *Key exchange.* The key exchange algorithm. DHE stands for Diffie-Hellman; DSS stands for Digital Signature Standard. The 1024 bit ciphers are lower strength ciphers formerly used for export control.
- *Encryption Algorithm.* AES stands for the Advanced Encryption Standard. DES stands for Data Encryption Standard.
- *Symmetric Key Bit Size.* The size in bits of the key used for the actual transport data encryption.
- *Message Authentication.* SHA stands for Secure Hash Algorithm.



NOTE

Directory Server supports ciphers for TLSv1 (recommended) and SSLv3. SSLv2 support is deprecated and not enabled by default in Directory Server.

To get a list of ciphers supported by the available version of the crypto library:

```
$ ldapsearch -xLLL -H ldap://localhost:389 -D "cn=Directory Manager" -W \
    -b 'cn=encryption,cn=config' -s base nsSSLSupportedCiphers -o ldif-
wrap=no

dn: cn=encryption,cn=config
nsSSLSupportedCiphers: TLS::tls_rsa_aes_256_sha::AES::SHA1::256
nsSSLSupportedCiphers: TLS::rsa_aes_256_sha::AES::SHA1::256
...
```

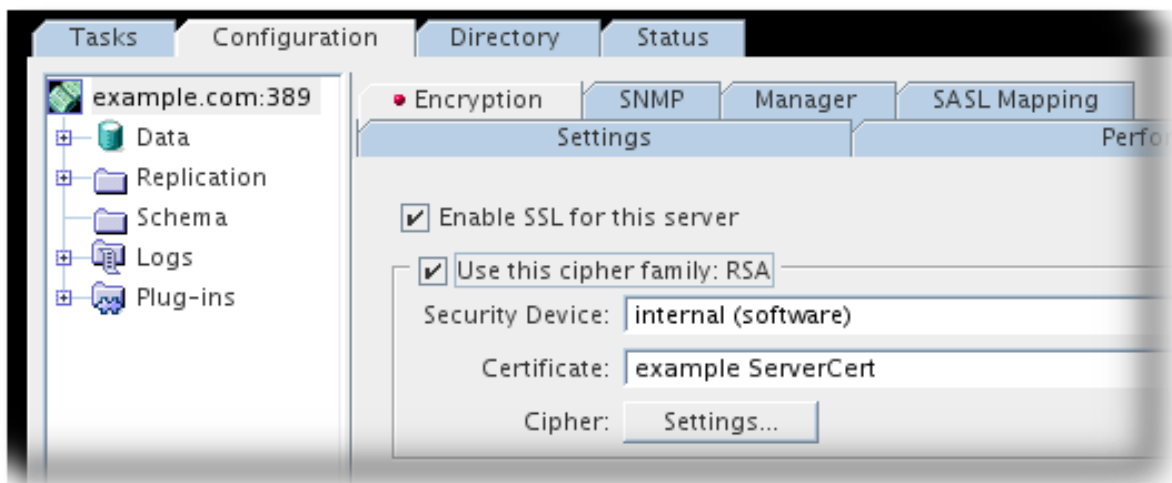
To get a list of ciphers currently configured in **cn=encryption,cn=config** by the *nsSSL3Ciphers* parameter:

```
$ ldapsearch -xLLL -H ldap://localhost:389 -D "cn=Directory Manager" -W \
    -b 'cn=encryption,cn=config' -s base nsSSEnabledCiphers -o ldif-
wrap=no

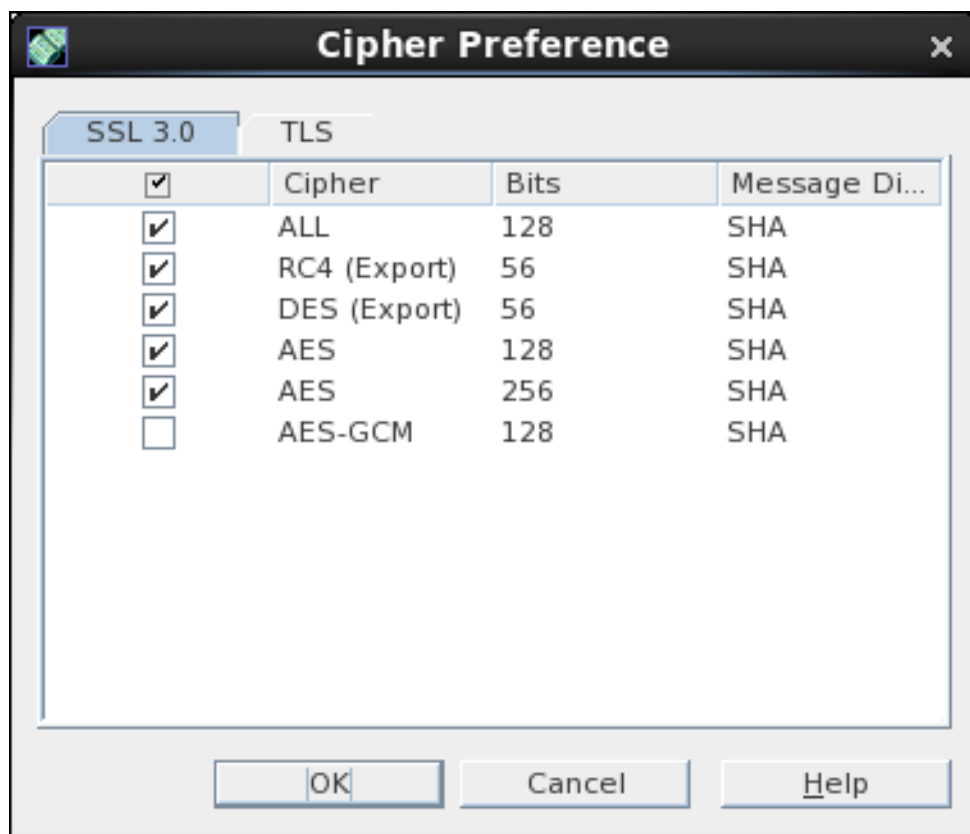
dn: cn=encryption,cn=config
nssslenabledciphers: TLS::tls_dhe_dss_aes_256_sha::AES::SHA1::256
nssslenabledciphers: TLS::tls_dhe_rsa_aes_256_sha::AES::SHA1::256
...
```

7.9.2. Selecting the Encryption Cipher

1. Make sure TLS/SSL is enabled for the server. For instructions on enabling TLS/SSL, see [Section 7.4, “Setting up TLS/SSL”](#).
2. Select the **Configuration** tab, and then select the topmost entry in the navigation tree in the left pane.
3. Select the **Encryption** tab in the right pane.
4. Click the **Cipher Setting** button.



5. In the **Cipher Preference** dialog box, specify which ciphers for the Directory Server to use by selecting them from the list, and click **OK**.



By default, **ALL** in the **TLS** tab is selected. When **ALL** is set, the server selects safe ciphers internally.



WARNING

Avoid selecting the **none**, **MD5** cipher because the server will use this option if no other ciphers are available on the client, instead of refusing the connection. The **none**, **MD5** cipher is not secure because encryption does not occur.

7.10. USING CLIENT (CERTIFICATE-BASED) AUTHENTICATION

Directory Server allows certificate-based authentication for the command-line tools (which are LDAP clients) and for server-to-server connections (replication and chaining). Three things are required for the Directory Server to allow client authentication:

- The server must have SSL turned on. See [Section 7.4, “Setting up TLS/SSL”](#) for more information.
- The Directory Server must trust the CA who issued the certificate to the client, as described in step 6 of [Section 7.3.2, “Trusting the Certificate Authority”](#).
- The subject DN in the certificate must be mapped in the user DN through a mapping in the **certmap.conf** file, as in [Section 7.10.2, “Mapping DN to Certificates”](#).



NOTE

A single configuration parameter, ***nsslapd-certdir***, in ***cn=config*** in ***dse.ldif*** lists the directory containing the key, certificate, and security files. The directory name should be unique and specific to the server. For example, the ***/etc/dirsrv/slapd-instance_name*** directory contains the key and certificate databases only for the Directory Server instance called *instance_name*. That directory will not contain key and certificate databases for any other server or client, nor will any of the key, certificate, or other security-related files for *instance_name* be located in any other directory.

The certificate and key files have been consolidated into a single file, specified in the ***nsslapd-certdir*** parameter, and the key and certificate file is stored in the ***/etc/dirsrv/slapd-instance_name*** directory.

When a server receives a request from a client, it can ask for the client's certificate before proceeding.

After checking that a client certificate chain ends with a trusted CA, the server can optionally determine which user is identified by the client certificate and then look up that user's entry in the directory. Each certificate has the name of the identity it verifies in a subject name, called the *subject DN*. The server authenticates the user by comparing the information in the subject DN with the DN of the user's directory entry.

In order to locate user entries in the directory, a server must know how to interpret the subject names of certificates from different CAs. The mapping between the subject names of the certificates and the user DNs is defined in the ***certmap.conf*** file. This file provides three kinds of information for each listed CA:

- It maps the distinguished name (DN) in the certificate to a branch point in the LDAP directory.
- It specifies which DN values from the certificate (user name, email address, and so on) the server should use for the purpose of searching the directory.
- It specifies whether the server should go through an additional verification process. This process involves comparing the certificate presented for authentication with the certificate stored in the user's directory entry. By comparing the certificate, the server determines whether to allow access or whether to revoke a certificate by removing it from the user's entry.

If more than one directory entry contains the information in the user's certificate, the server can examine all matching entries in order to determine which user is trying to authenticate. When examining a directory entry, the server compares the presented certificate with the one stored in the entry. If the presented certificate does not match any certificates or if the matching entries do not contain certificates, client authentication fails.

After the server finds a matching entry and certificate in the directory, it can determine the appropriate kind of authorization for the client. For example, some servers use information from a user's entry to determine group membership, which in turn can be used during evaluation of ACIs to determine what resources the user is authorized to access.

7.10.1. Configuring Directory Server to Accept Certificate-Based Authentication from LDAP Clients

Client authentication to the Directory Server will require or allow a user to use a certificate to establish its identity, in addition to the server having to present a certification. This is also called *certificate-based authentication*.

1. On the client system, obtain a client certificate from the CA.
2. Install the client certificate on the client system.

Regardless of how the certificate is sent (either in email or on a web page), there should be a link to click to install the certificate.

Record the certificate information that is sent from the CA, especially the subject DN of the certificate because the server must be configured to map it to an entry in the directory. The client certificate resembles the following:

```
-----BEGIN CERTIFICATE-----
MIICMjCCAZugAwIBAgICCEEwDQYJKoZIhvcNAQEFBQAwfDELMakGA1UEBh
MCVVMxIzAhBgNVBAoTG1BhbG9va2FWaWxsZSBXaWRnZXRzLCBJbmMuMR0w
GwYDVQQLExRXaWRnZXQgTWFrZXJzICdSJyBVczEpMCcGA1UEAxMgVGVzdC
BUZXN0IFRlc3QgVGVzdCBUZXN0IFRlc3QgQ0EwHhcNOTgwMzEyMDIzMzU3
WhcNOTgwMzI2MDIzMzU3WjBPMQswCQYDVQQGEwJVUzEoMCYGA1UEChMFTm
V0c2NhcgUGRGlyZWNoY3
-----END CERTIFICATE-----
```

3. Convert the client certificate into binary format using the **certutil** utility.

```
certutil -L -d certdbPath -n userCertName -r > userCert.bin
```

certdbPath is the directory which contains the certificate database; for example, a user certificate for Mozilla Thunderbird is stored in **\$HOME/.thunderbird**. *userCertName* is the name of the certificate, and *userCert.bin* is the name of the output file for binary format.

4. On the server, map the subject DN of the certificate to the appropriate directory entry by editing the **certmap.conf** file.

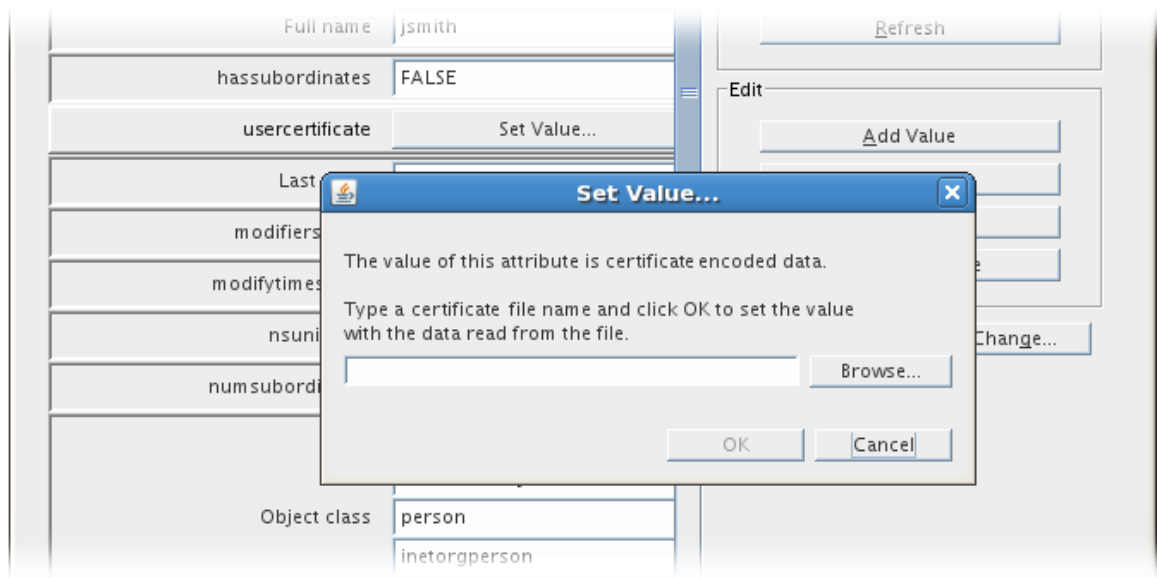


NOTE

Do not map a certificate-based authentication certificate to a distinguished name under **cn=monitor**. Mapping a certificate to a DN under **cn=monitor** causes the bind operation to fail. Map the certificate to a target located elsewhere in the directory information tree. Make sure that the **verifyCert** parameter is set to **on** in the **certmap.conf** file. If this parameter is not set to **on**, Directory Server simply searches for an entry in the directory that matches the information in the **certmap.conf** file. If the search is successful, it grants access without actually checking the value of the **userCertification** and **usercertificate;binary** attributes.

5. In the Directory Server, modify the directory entry for the user or identity (if it is another server) who owns the client certificate to add the **usercertificate** attribute.
 1. Select the **Directory** tab, and navigate to the user entry.
 2. Double-click the user entry, and use the **Property Editor** to add the **usercertificate** attribute, with the **binary** subtype.

When adding this attribute, instead of an editable field, the server provides a **Set Value** button.



3. Click **Set Value**.

A file selector opens. Use it to select the binary file created in step 3.

For information on using the Directory Server Console to edit entries, see [Section 3.1.3, “Modifying Directory Entries”](#).

For information on how to use TLS/SSL with **ldapmodify**, **ldapdelete**, and **ldapsearch**, see [Section A.2, “Using SSL/TLS and Start TLS with LDAP Client Tools”](#).

7.10.2. Mapping DN's to Certificates

When a server performs client authentication, it interprets a certificate, extracts user information, and then searches the directory for that information. In order to process certificates from different CAs, the server uses a file called **certmap.conf**. This file contains instructions on how to interpret different certificates and how to search the directory for the information that those certificates contain.

In the Directory Server, a user entry has a format like the following:

```
dn: uid=jsmith,ou=People,dc=example,dc=com
...
cn: John Smith
mail: jsmith@example.com
```

A subject DN, however, is almost always formatted differently than an LDAP DN. For example:

```
cn=John Smith,e=jsmith@example.com,c=US,o=Example.com
```

The email attribute in the directory is almost always unique within the organization, as is the common name of the user. These attributes are also indexed by default, so they are easily searched, and are common attributes to be used in the subject names of certificates. The **certmap.conf** file can be configured so that the server looks for any mail or common name elements in the subject DN and matches them against the entries in the directory. Much like an **ldapsearch**, the cert mapping defines a search base (**DNComps**) and search filter (**FilterComps**).

```
certmap Example      o=Example.com,c=US
Example:DNComps      dc
Example:FilterComps  mail,cn
```

The **certmap.conf** file is stored in the **/etc/dirsrv/slaped-*instance_name*/** folder. The file contains a default mapping as well as mappings for specific CAs.

The default mapping specifies what the server should do if a client certificate was issued by a CA that is not listed in **certmap.conf**. The mappings for specific CAs specify what the server should do for client certificates issued by those CAs. All mappings define the following:

- Where in the directory the server should begin its search
- What certificate attributes the server should use as search criteria
- Whether the server should verify the certificate with one that is stored in the directory

Mappings have the following syntax:

```
certmap name issuer DN
name:property [value]
name:property [value]
...
```

The first line of a mapping specifies the mapping's name as well as the DN for the issuer of the client certificate. The mapping can have any name, but the *issuerDN* must exactly match the issuer DN of the CA that issued the client certificate. For example, the following two *issuerDN* lines differ only in the number of spaces they contain, but the server would treat these two entries as different:

```
certmap moz ou=Example CA,o=Example,c=US
certmap moz ou=Example CA,o=Example,c=US
```

The second and subsequent lines of a mapping identify the rules that the server should use when searching the directory for information extracted from a certificate. These rules are specified through the use of one or more of the following properties:

- **DNComps**
- **FilterComps**
- **VerifyCert**
- **CmapLdapAttr**
- **Library**
- **InitFn**

DNComps

DNComps is a comma-separated list of relative distinguished name (RDN) keywords used to determine where in the user directory the server should start searching for entries that match the information for the owner of the client certificate. The server gathers values for these keywords from the client certificate and uses the values to form a DN, which determines where the server starts its search in the directory.

For example, if the **DNComps** is set to use the **o** and **c** RDN keywords, the server starts the search from the **o=org, c=country** entry in the directory, where *org* and *country* are replaced with values from the DN in the certificate.

- If there is not a **DNComps** entry in the mapping, the server uses either the **CmapLdapAttr** setting or the entire subject DN in the client certificate to determine where to start searching.
- If the **DNComps** entry is present but has no value, the server searches the entire directory tree for entries matching the filter specified by **FilterComps**.

The following RDN keywords are supported for **DNComps**:

- **cn**
- **ou**
- **o**
- **c**
- **l**
- **st**
- **dc**
- **e** or **mail** (but not both)
- **mail**

Keywords can be in either lower case or upper case.

FilterComps

FilterComps is a comma-separated list of RDN keywords used to create a filter by gathering information from the user's DN in the client certificate. The server uses the values for these keywords to form the search criteria for matching entries in the LDAP directory. If the server finds one or more entries in the directory that match the user's information gathered from the certificate, the search is successful and the server performs a verification (if **verifycert** is set to **on**).

For example, if **FilterComps** is set to use the **e** and **uid** attribute keywords (**FilterComps=e,uid**), the server searches the directory for an entry whose values for **e** and **uid** match the user's information gathered from the client certificate. Email addresses and user IDs are good filters because they are usually unique entries in the directory.

The filter needs to be specific enough to match one and only one entry in the directory.

The following RDN keywords are supported for **FilterComps**:

- **cn**
- **ou**
- **o**
- **c**

- *l*
- *st*
- *dc*
- *e* or *mail* (but not both)
- *mail*

Keywords can be in either lower case or upper case.

VerifyCert

verifycert tells the server whether it should compare the client's certificate with the certificate found in the user's directory entry. The value is either **on** or **off**. Setting the value to **on** ensures that the server will not authenticate the client unless the certificate presented exactly matches the certificate stored in the directory. Setting the value to **off** disables the verification process.

CmapLdapAttr

CmapLdapAttr is the name of the attribute in the directory that contains subject DN's from all certificates belonging to the user. Because this attribute is not a standard LDAP attribute, this attribute must be added to the schema. See [Section 8.4.2, "Creating Attributes"](#) for information on adding schema elements.

If the **CmapLdapAttr** property exists in a **certmap.conf** mapping, the server searches the entire directory for an entry that contains the subject's full DN. The search criteria are the attribute named by **CmapLdapAttr** and the subject's full DN as listed in the certificate. If the search does not yield any entries, the server retries the search using the **DNComps** and **FilterComps** mappings. The search will take place more quickly if the attribute specified by **CmapLdapAttr** is indexed. For more information on indexing attributes, see [Chapter 9, Managing Indexes](#).

Using **CmapLdapAttr** to match a certificate to a directory entry is useful when it is difficult to match entries using **DNComps** and **FilterComps**.

Library

Library is the pathname to a shared library or DLL. Use this property only to extend or replace the standard functions that map information in **certmap.conf** to entries in the directory. This property is typically not necessary unless there are very specialized mapping requirements.

InitFn

InitFn is the name of an **init** function from a custom library. You need to use this property only if you want to extend or replace the functions that map information in **certmap.conf** to entries in the directory. This property is typically not necessary unless you have very specialized mapping requirements.

7.10.3. Editing the certmap.conf File

1. In a text editor, open **/etc/dirsrv/slaped-instance_name/certmap.conf**
2. If necessary, make changes to the default mapping.

For example, change the value for **DNComps** or **FilterComps**. To comment out a line, insert a **#** before it.

3. If desired, create a mapping for a specific CA.

The mapping should take the form **certmap** *mappingName issuerDN*.

For example, to create a mapping named **Example CA** which has the issuer DN **ou=example CA, o=example, c=US**, enter the following:

```
certmap example CA ou=example CA,o=example,c=US
```

4. Add property settings for a specific CA's mapping.

Specify the **Library** and **InitFn** properties before adding any additional properties.

When adding a property, use the form *mappingName:propertyName value*.

For example, add a **DNComps** value of **o, c** for **Example CA** by entering the following line:

```
example CA:DNComps o, c
```

For the **Library** and **InitFn** properties, a complete mapping looks like this:

```
certmap Example CA ou=example CA,o=example,c=US
Example CA:Library /ldapserver/ldap/servers/slapd/plugin.c
Example CA:InitFn plugin_init_dn
Example CA:DNComps o, c
Example CA:FilterComps e, uid
Example CA:VerifyCert on
Example CA:CmapLdapAttr certSubjectDN
```

5. Save the **certmap.conf** file.

7.10.4. Example certmap.conf Mappings

In [Example 7.1, “Default Mapping”](#), the server starts its search at the directory branch point containing the entry **ou=organizationalUnit**, **o=organization**, **c=country**, where the italics represent values from the subject's DN in the client certificate.

Example 7.1. Default Mapping

```
certmap default      default
default:DNComps      ou, o, c
default:FilterComps  e, uid
default:verifycert   on
```

The server then uses the values for **e** (email address) and **uid** (user ID) from the certificate to search for a match in the directory before authenticating the user. When it finds a matching entry, the server verifies the certificate by comparing the certificate the client sent to the certificate stored in the directory.

[Example 7.2, “An Additional Mapping”](#) shows the contents of a sample **certmap.conf** file that defines a default mapping as well as a mapping for MyCA:

Example 7.2. An Additional Mapping

```

certmap default          default
default:DNComps
default:FilterComps     e, uid
certmap MyCA             ou=MySpecialTrust,o=MyOrg,c=US
MyCA:DNComps             ou,o,c
MyCA:FilterComps        e
MyCA:verifycert         on

```

When the server gets a certificate from a CA other than MyCA, the server uses the default mapping, which starts at the top of the directory tree and searches for an entry matching the client's email address (**e**) and user ID (**uid**). If the certificate is from MyCA, the server starts its search at the directory branch containing the organizational unit specified in the subject DN and searches for email addresses (**e**) that match the one specified in the certificate. If the certificate is from MyCA, the server verifies the certificate. If the certificate is from another CA, the server does not verify it.

[Example 7.3, “A Mapping with an Attribute Search”](#) uses the **CmapLdapAttr** property to search the directory for an attribute called **certSubjectDN** whose value exactly matches the entire subject DN in the client certificate:

Example 7.3. A Mapping with an Attribute Search

```

certmap MyCo             ou=My Company Inc,o=MyCo,c=US
MyCo:CmapLdapAttr        certSubjectDN
MyCo:DNComps             o, c
MyCo:FilterComps        mail, uid
MyCo:verifycert         on

```

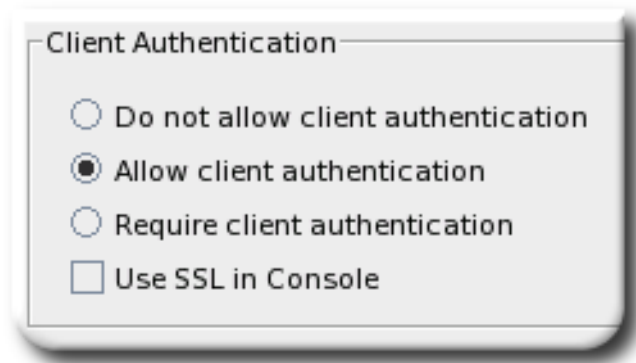
If the subject DN in the client certificate is **uid=jsmith,o=example Inc,c=US**, then the server searches for entries that have **certSubjectDN=uid=jsmith,o=example Inc,c=US**.

If one or more matching entries are found, the server proceeds to verify the entries. If no matching entries are found, the server uses **DNComps** and **FilterComps** to search for matching entries. For the client certificate described above, the server would search for **uid=jsmith** in all entries under **o=example Inc,c=US**.

7.10.5. Allowing and Requiring Client Authentication to the Console

Client authentication must be explicitly set in the Directory Server.

1. Click the **Configuration** tab.
2. With the top server entry highlighted in the left navigation pane, click the **Encryption** tab in the main window.
3. Set whether to require or allow client authentication to the Directory Server.



- *Do not allow client authentication.* With this option, the server ignores the client's certificate. This does not mean that the bind will fail.
- *Allow client authentication.* This is the default setting. With this option, authentication is performed on the client's request.
- *Require client authentication.* With this option, the server requests authentication from the client.

If client authentication is required, then SSL cannot be used with the Console because The Directory Server Console does not support client authentication.



NOTE

To use certificate-based authentication with replication, configure the consumer server either to allow or to require client authentication.



NOTE

The Directory Server must already be configured to run over TLS/SSL or Start TLS for client authentication to be enabled.

4. Save the changes, and restart the server. For example:

```
service dirsrv restart
```

To change the server configuration from requiring client authentication to allowing it through the command line, reset the ***nsSSLClientAuth*** parameter:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -D
"cn=directory manager" -w secret -p 636 -h server.example.com -x

dn: cn=encryption,cn=config
changetype: modify
replace: nsSSLClientAuth
nsSSLClientAuth: allowed
```

The ***nsSSLClientAuth*** parameter can be set to **off**, **allowed**, and **required**.

7.10.6. Forcing SASL/EXTERNAL Mechanisms for Bind Requests

The assumption with certificate-based authentication is that the Directory Server will use the identity in the presented certificate to process the BIND request.

There are two steps that the client takes at the beginning of an SSL session to use certificate authentication. First, the client sends its certificate to the Directory Server. Then, it sends its bind request. Most clients issue the bind request using the SASL/EXTERNAL mechanism, which is important because it signals to the Directory Server that it needs to use the identity in the certificate for the bind, not any credentials in the bind request.

However, some clients use simple authentication or anonymous credentials when they send the bind request. The client still assumes that the Directory Server will use the identity in the certificate for the bind identity, but there is nothing in the bind request to tell the Directory Server that is what it should do. In that case, the SSL session fails with invalid credentials, even if the certificate and the client identity in that certificate are valid.

A configuration option in the Directory Server (***nsslapd-force-sasl-external***) forces clients in certificate-based authentication to use the SASL/EXTERNAL method and to ignore any simple bind method given with the request.

This attribute is off by default, but it can be turned on:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -D
"cn=directory manager" -w secret -p 636 -h server.example.com -x

dn: cn=config
changetype: modify
replace: nsslapd-force-sasl-external
nsslapd-force-sasl-external: on
```

7.11. SETTING UP SASL IDENTITY MAPPING

Red Hat Directory Server supports LDAP client authentication through the Simple Authentication and Security Layer (SASL), an alternative to TLS/SSL and a native way for some applications to share information securely.

Simple Authentication and Security Layer (SASL) is an abstraction layer between protocols like LDAP and authentication methods like GSS-API which allows any protocol which can interact with SASL to utilize any authentication mechanism which can work with SASL. Simply put, SASL is an intermediary that makes authenticating to applications using different mechanisms easier. SASL can also be used to establish an encrypted session between a client and server.

The SASL framework allows different mechanisms to be used to authenticate a user to the server, depending on what mechanism is enabled in both client and server applications. SASL also creates a layer for encrypted (secure) sessions. Using GSS-API, Directory Server utilizes Kerberos tickets to authenticate sessions and encrypt data.

7.11.1. About SASL Identity Mapping

When processing a SASL bind request, the server matches, or maps, the SASL authentication ID used to authenticate to the Directory Server with an LDAP entry stored within the server. When using Kerberos, the SASL user ID usually has the format *userid@REALM*, such as **scarter@EXAMPLE.COM**. This ID must be converted into the DN of the user's Directory Server entry, such as **uid=scarter,ou=people,dc=example,dc=com**.

If the authentication ID clearly corresponds to the LDAP entry for a person, it is possible to configure the

Directory Server to map the authentication ID automatically to the entry DN. Directory Server has some pre-configured default maps which handle most common configurations, and customized maps can be created. During a bind attempt, the first matching mapping rule is applied. If only one user identity is returned, the bind is successful; if none or more than one are returned, then the bind fails.

Be sure to configure SASL maps so that only one mapping rule matches the authentication string.

SASL mappings are configured by entries under a container entry:

```
dn: cn=sasl,cn=config
objectClass: top
objectClass: nsContainer
cn: sasl
```

SASL identity mapping entries are children of this entry:

```
dn: cn=mapping,cn=sasl,cn=config
objectClass: top
objectClass: nsContainer
cn: mapping
```

Mapping entries are defined by three attributes:

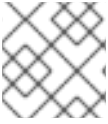
- *nsSaslMapRegexString*, the regular expression which is used to map the elements of the supplied **authid**
- *nsSaslMapFilterTemplate*, a template which applies the elements of the **nsSaslMapRegexString** to create the DN
- *nsSaslMapBaseDNTemplate*, which provides the search base or a specific entry DN to match against the constructed DN

For example:

```
dn: cn=mymap,cn=mapping,cn=sasl,cn=config
objectclass: top
objectclass: nsSaslMapping
cn: mymap
nsSaslMapRegexString: \(.*\)\@*\(.*\)\.\(.*\)
nsSaslMapFilterTemplate: (objectclass=inetOrgPerson)
nsSaslMapBaseDNTemplate: uid=\1,ou=people,dc=\2,dc=\3
```

The **nsSaslMapRegexString** attribute sets variables of the form **\1**, **\2**, **\3** for bind IDs which are filled into the template attributes during a search. This example sets up a SASL identity mapping for any user in the **ou=People, dc=example, dc=com** subtree who belongs to the **inetOrgPerson** object class.

When a Directory Server receives a SASL bind request with **mconnors@EXAMPLE.COM** as the user ID (**authid**), the regular expression fills in the base DN template with **uid=mconnors,ou=people,dc=EXAMPLE,dc=COM** as the user ID, and authentication proceeds from there.

**NOTE**

The **dc** values are not case sensitive, so **dc=EXAMPLE** and **dc=example** are equivalent.

The Directory Server can also use a more inclusive mapping scheme, such as the following:

```
dn: cn=example map,cn=mapping,cn=sasl,cn=config
objectclass: top
objectclass: nsSaslMapping
cn: example map
nsSaslMapRegexString: \(.*\)
nsSaslMapBaseDNTemplate: ou=People,dc=example,dc=com
nsSaslMapFilterTemplate: (cn=\1)
```

This matches any user ID and map it an entry under the **ou=People,dc=example,dc=com** subtree which meets the filter **cn=userId**.

Mappings can be confined to a single realm by specifying the realm in the **nsSaslMapRegexString** attribute. For example:

```
dn: cn=example map,cn=mapping,cn=sasl,cn=config
objectclass: top
objectclass: nsSaslMapping
cn: example map
nsSaslMapRegexString: \(.*\)@US.EXAMPLE.COM
nsSaslMapBaseDNTemplate: ou=People,dc=example,dc=com
nsSaslMapFilterTemplate: (cn=\1)
```

This mapping is identical to the previous mapping, except that it only applies to users authenticating from the **US.EXAMPLE.COM** realm. (Realms are described in [Section 7.12.2.1, “About Principals and Realms”](#).)

When a server connects to another server, such as during replication or with chaining, the default mappings for the will not properly map the identities. This is because the principal (SASL identity) for one server does not match the principal on the server where authentication is taking place, so it does not match the mapping entries.

To allow server to server authentication using SASL, create a mapping for the specific server principal to a specific user entry. For example, this mapping matches the **ldap1.example.com** server to the **cn=replication manager,cn=config** entry. The mapping entry itself is created on the second server, such as **ldap2.example.com**.

```
dn: cn=z,cn=mapping,cn=sasl,cn=config
objectclass: top
objectclass: nsSaslMapping
cn: z
nsSaslMapRegexString: ldap/ldap1.example.com@EXAMPLE.COM
nsSaslMapBaseDNTemplate: cn=replication manager,cn=config
nsSaslMapFilterTemplate: (objectclass=*)
```

Sometimes, the realm name is not included in the principal name in SASL GSS-API configuration. A second mapping can be created which is identical to the first, only without specifying the realm in the principal name. For example:

```
dn: cn=y,cn=mapping,cn=sasl,cn=config
objectclass: top
objectclass: nsSaslMapping
cn: y
nsSaslMapRegexString: ldap/ldap1.example.com
nsSaslMapBaseDNTemplate: cn=replication manager,cn=config
nsSaslMapFilterTemplate: (objectclass=*)
```

Because the realm is not specified, the second mapping is more general (meaning, it has the potential to match more entries than the first. The best practice is to have more specific mappings processed first and gradually progress through more general mappings.

There is no way to specify the order that mappings are processed. However, there is a way to control how SASL mappings are processed: the name. The Directory Server processes SASL mappings in reverse ASCII order. In the past two example, then the **cn=z** mapping (the first example) is processed first. If there is no match, the server processes the **cn=y** mapping (the second example).



NOTE

SASL mappings can be added when an instance is created during a silent installation by specifying the mappings in an LDIF file and adding the LDIF file with the **ConfigFile** directive. Using silent installation is described in the *Installation Guide*.

7.11.2. Default SASL Mappings for Directory Server

The Directory Server has pre-defined SASL mapping rules to handle some of the most common usage.

Kerberos UID Mapping

This matches a Kerberos principal using a two part realm, such as *user@example.com*. The realm is then used to define the search base, and the user ID (**authid**) defines the filter. The search base is **dc=example,dc=com** and the filter of (**uid=user**).

```
dn: cn=Kerberos uid mapping,cn=mapping,cn=sasl,cn=config
objectClass: top
objectClass: nsSaslMapping
cn: Kerberos uid mapping
nsSaslMapRegexString: \(.*\)\@(\.*\)\.(\.*\)
nsSaslMapBaseDNTemplate: dc=\2,dc=\3
nsSaslMapFilterTemplate: (uid=\1)
```

RFC 2829 DN Syntax

This mapping matches an **authid** that is a valid DN (defined in RFC 2829) prefixed by **dn:.** The **authid** maps directly to the specified DN.

```
dn: cn=rfc 2829 dn syntax,cn=mapping,cn=sasl,cn=config
objectClass: top
objectClass: nsSaslMapping
cn: rfc 2829 dn syntax
nsSaslMapRegexString: ^dn:\(.*\)
nsSaslMapBaseDNTemplate: \1
nsSaslMapFilterTemplate: (objectclass=*)
```

RFC 2829 U Syntax

This mapping matches an **authid** that is a UID prefixed by **u:**. The value specified after the prefix defines a filter of (**uid=value**). The search base is hard-coded to be the suffix of the default **userRoot** database.

```
dn: cn=rfc 2829 u syntax,cn=mapping,cn=sasl,cn=config
objectClass: top
objectClass: nsSaslMapping
cn: rfc 2829 u syntax
nsSaslMapRegexString: ^u:\(.*\)
nsSaslMapBaseDNTemplate: dc=example,dc=com
nsSaslMapFilterTemplate: (uid=\1)
```

UID Mapping

This mapping matches an **authid** that is any plain string that does not match the other default mapping rules. It use this value to define a filter of (**uid=value**). The search base is hard-coded to be the suffix of the default **userRoot** database.

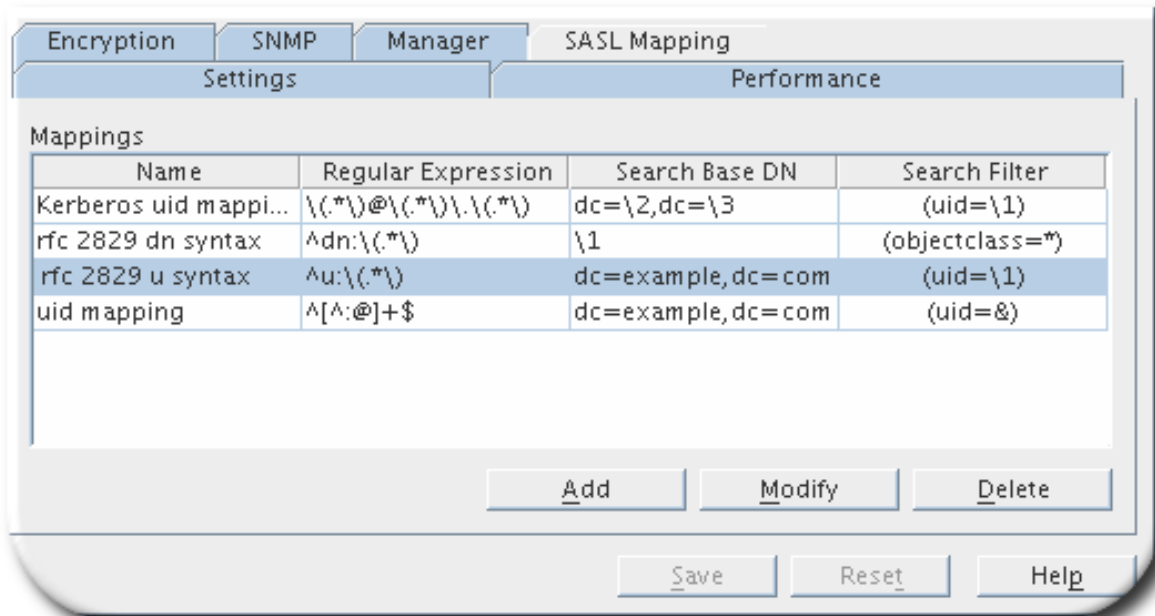
```
dn: cn=uid mapping,cn=mapping,cn=sasl,cn=config
objectClass: top
objectClass: nsSaslMapping
cn: uid mapping
nsSaslMapRegexString: ^[^:@]+$
nsSaslMapBaseDNTemplate: dc=example,dc=com
nsSaslMapFilterTemplate: (uid=&)
```

7.11.3. Configuring SASL Identity Mapping

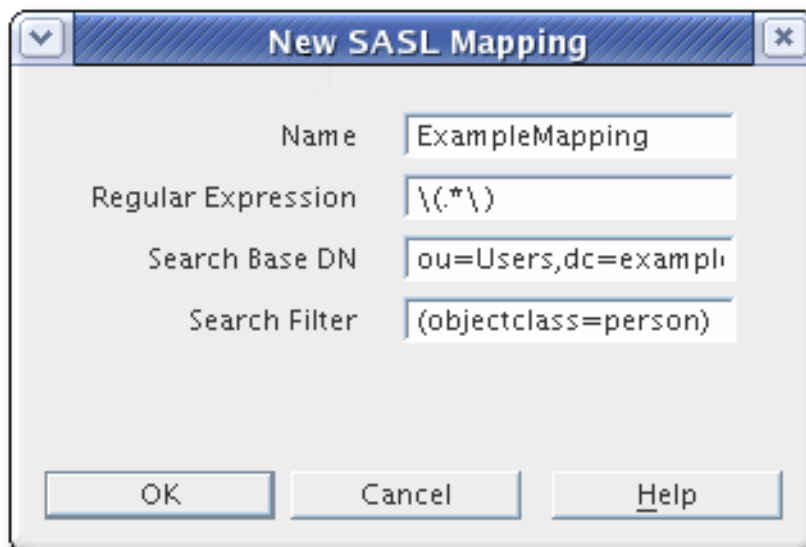
SASL identity mapping can be configured from either the Directory Server or the command line. For SASL identity mapping to work for SASL authentication, the mapping must return one, and only one, entry that matches and Kerberos must be configured on the host machine.

7.11.3.1. Configuring SASL Identity Mapping from the Console

1. In the Directory Server Console, open the **Configuration** tab.
2. Select the **SASL Mapping** tab.



- To add a new SASL identity mapping, select the **Add** button, and fill in the required values.



- *Name*. This field sets the unique name of the SASL mapping.
- *Regular expression*. This field sets the regular expression used to match the DN components, such as `\(.*\)`. This field corresponds to the `nsSaslMapRegexString` value in the SASL mapping LDIF entry.
- *Search base DN*. This field gives the base DN to search to map entries, such as `ou=People,dc=example,dc=com`. This field corresponds to the `nsSaslMapBaseDNTemplate` value in the SASL mapping LDIF entry.
- *Search filter*. This field gives the search filter for the components to replace, such as `(objectclass=*)`. This field corresponds to the `nsSaslMapFilterTemplate` value in the SASL mapping LDIF entry.

To edit a SASL identity mapping, highlight that identity in the **SASL Mapping** tab, and click **Modify**. Change any values, and save.

To delete a SASL identity mapping, highlight it and hit **Delete**. A dialog box comes up to confirm the deletion.

7.11.3.2. Configuring SASL Identity Mapping from the Command Line

To configure SASL identity mapping from the command line, use the **ldapmodify** utility to add the identity mapping scheme. For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=example map,cn=mapping,cn=sasl,cn=config
changetype: add
objectclass: top
objectclass: nsSaslMapping
cn: example map
nsSaslMapRegexString: \(.*\)
nsSaslMapBaseDNTemplate: ou=People,dc=example,dc=com
nsSaslMapFilterTemplate: (cn=\1)
```

This matches any user's common name and maps it to the result of the subtree search with base **ou=People,dc=example,dc=com**, based on the filter **cn=userId**.



NOTE

When SASL maps are added over LDAP, they are not used by the server until it is restarted. Adding the SASL map with **ldapmodify** adds the mapping to the end of the list, regardless of its ASCII order.

7.12. USING KERBEROS GSS-API WITH SASL

Kerberos v5 must be deployed on the host for Directory Server to utilize the GSS-API mechanism for SASL authentication. GSS-API and Kerberos client libraries must be installed on the Directory Server host to take advantage of Kerberos services.

7.12.1. Authentication Mechanisms for SASL in Directory Server

Directory Server support the following SASL encryption mechanisms:

- **PLAIN**. PLAIN sends cleartext passwords for simple password-based authentication. This is not a preferred mechanism for most applications because of its relative lack of strength, but it can be used in some situations where anonymous access is disabled and an arbitrary UID (not a DN) is used to authenticate to the server because SASL can map the UID to a directory entry.
- **EXTERNAL**. EXTERNAL, as with TLS/SSL, performs certificate-based authentication. This method uses public keys for strong authentication.
- **CRAM-MD5**. **CRAM-MD5** is a simple challenge-response authentication method. It does not establish any security layer, unlike GSS-API. Both DIGEST-MD5 and GSS-API are much more secure, so both of those methods are recommended over CRAM-MD5.
- **DIGEST-MD5**. **DIGEST-MD5** is a mandatory authentication method for LDAPv3 servers. While it is not as strong as public key systems or Kerberos authentication methods, it is preferred over plain text passwords and does protect against plain text attacks.

- *Generic Security Services (GSS-API)*. Generic Security Services (GSS) is a security API that is the native way for UNIX-based operating systems to access and authenticate Kerberos services. GSS-API also supports session encryption, similar to TLS/SSL. This allows LDAP clients to authenticate with the server using Kerberos version 5 credentials (tickets) and to use network session encryption.

For Directory Server to use GSS-API, Kerberos must be configured on the host machine. See [Section 7.12, “Using Kerberos GSS-API with SASL”](#).



NOTE

GSS-API and, thus, Kerberos are only supported on platforms that have GSS-API support. To use GSS-API, it may be necessary to install the Kerberos client libraries; any required Kerberos libraries will be available through the operating system vendor.

CRAM-MD5, DIGEST-MD5, and GSS-API are all *shared secret* mechanisms. The server challenges the client attempting to bind with a *secret*, such as a password, that depends on the mechanism. The user sends back the response required by the mechanism.



NOTE

DIGEST-MD5 requires clear text passwords. The Directory Server requires the clear text password in order to generate the shared secret. Passwords already stored as a hashed value, such as **SHA1**, *cannot* be used with **DIGEST-MD5**.

7.12.2. About Kerberos in Directory Server

On Red Hat Enterprise Linux, the supported Kerberos libraries are MIT Kerberos version 5.

The concepts of Kerberos, as well as using and configuring Kerberos, are covered at the MIT Kerberos website, <http://web.mit.edu/Kerberos/>.

7.12.2.1. About Principals and Realms

A *principal* is a user in the Kerberos environment. A *realm* is a set of users and the authentication methods for those users to access the realm. A realm resembles a fully-qualified domain name and can be distributed across either a single server or a single domain across multiple machines. A single server instance can also support multiple realms.



NOTE

Kerberos realms are only supported for GSS-API authentication and encryption, not for DIGEST-MD5.

Realms are used by the server to associate the DN of the client in the following form, which looks like an LDAP DN:

```
uid=user_name/[server_instance],cn=realm,cn=mechanism,cn=auth
```

For example, Mike Connors in the **engineering** realm of the European division of **example.com** uses the following association to access a server in the US realm:

–

```
uid=mconnors/cn=Europe.example.com,cn=engineering,cn=gssapi,cn=auth
```

Babara Jensen, from the **accounting** realm of **US.example.com**, does not have to specify a realm when to access a local server:

```
uid=bjensen,cn=accounting,cn=gssapi,cn=auth
```

If realms are supported by the mechanism and the default realm is not used to authenticate to the server, then the *realm* must be specified in the Kerberos principal. Otherwise, the realm can be omitted.



NOTE

Kerberos systems treat the Kerberos realm as the default realm; other systems default to the server.

7.12.2.2. About the KDC Server and Keytabs

Kerberos is a protocol which allows users or servers to authenticate to a server securely over an insecure connection. As with protocols like TLS and SSL, Kerberos generates and issues session keys which encrypt information. A Kerberos server, then, has two functions: as an authenticating server to validate clients and as a ticket granting server.

Because of this, the Kerberos server is called a *key distribution center* or KDC.

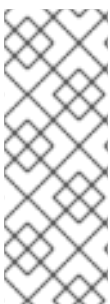
See the operating system documentation for information on installing and configuring a Kerberos server.

When a client authenticates to the Directory Server using GSS-API, the KDC sends a session key, followed by a ticket granting ticket (TGT). The TGT contains the client's ID and network address, a validity period, and the session key. The ticket and the ticket's lifetime are parameters in the Kerberos client and server configuration. In many systems, this TGT is issued to the client when the user first logs into the operating system.

Command-line utilities provided with the operating system — including **kinit**, **klist**, and **kdestroy** — can acquire, list, and destroy the TGT. These tools usually use a file called a *keytab* to the issued keys. This file is created by the Kerberos administrator by exporting the key from the KDC.

In order to respond to Kerberos operations, the Directory Server requires access to its own cryptographic key. The Kerberos key is written in the keytab file. The keytab gives the credentials that the Directory Server uses to authenticate to other servers. Directory Server assigns a keytab through the **KRB5_KTNAME** environment variable in its startup script (**/etc/sysconfig/dirsrv**):

```
KRB5_KTNAME=/etc/dirsrv/ds.keytab ; export KRB5_KTNAME
```



NOTE

The Directory Server must be able to access the host keytab and the **krb5.conf** file for GSS-API authentication in SASL. For this host keytab file to be properly labeled in SELinux in the **dirsrv_config_t** context, the file must be in the **/etc** directory.

Only the host keytab and **krb5.conf** must be in **/etc**. The user key tabs can still be in any directory.

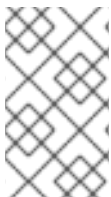
When the Directory Server authenticates to the server, it is as if it is running a **kinit** command to initiate the connection:

```
kinit -k -t /etc/dirsrv/ds.keytab ldap/FQDN@REALM
```

The Directory Server uses the service name **ldap**. Its Kerberos principal is **ldap/host-fqdn@realm**, such as **ldap/ldap.corp.example.com/EXAMPLE.COM**. The *host-fqdn* must be the fully-qualified host and domain name, which can be resolved by all LDAP and Kerberos clients through both DNS and reverse DNS lookups. A key with this identity must be stored in the server's keytab in order for Kerberos to work.

Whatever server the Directory Server is authenticating to must have a SASL mapping that maps the Directory Server's principal (its user entry, usually something like **ldap/server.example.com@EXAMPLE.COM**) to a real entry DN in the receiving server.

On Red Hat Enterprise Linux, the client-side Kerberos configuration is in the **/etc/krb5.conf**.



NOTE

The Directory Server must be able to access the **krb5.conf** file for GSS-API authentication in SASL. For these files to be properly labeled in SELinux in the **dirsrv_config_t** context, the file must be in the **/etc** directory.

For information on setting up the service key, see the Kerberos documentation.

[Example 7.4, “KDC Server Configuration”](#) shows a KDC server configured with the **company.example.com** realm.

Example 7.4. KDC Server Configuration

```
[libdefaults]
    ticket_lifetime = 24000
    default_realm = COMPANY.EXAMPLE.COM
    dns_lookup_realm = false
    dns_lookup_kdc = false
    ccache_type = 1
    forwardable = true
    proxiable = true
    default_tgs_enctypes = des3-hmac-sha1 des-cbc-crc
    default_tkt_enctypes = des3-hmac-sha1 des-cbc-crc
    permitted_enctypes = des3-hmac-sha1 des-cbc-crc
[realms]
    COMPANY.EXAMPLE.COM = {
        kdc = kdcserver.company.example.com:88
        admin_server = adminserver.company.example.com:749
        default_domain = company.example.com
    }
[appdefaults]
    pam = {
        debug = true
        ticket_lifetime = 36000
        renew_lifetime = 36000
        forwardable = true
        krb4_convert = false
```

```

    }
[logging]
    default = FILE:/var/krb5/kdc.log
    kdc = FILE:/var/krb5/kdc.log
    admin_server = FILE:/var/log/kadmind.log

```

7.12.3. Configuring SASL Authentication at Directory Server Startup

SASL GSS-API authentication has to be activated in Directory Server so that Kerberos tickets can be used for authentication. This is done by supplying a system configuration file for the init scripts to use which identifies the variable to set the keytab file location. When the init script runs at Directory Server startup, SASL authentication is then immediately active.

The default SASL configuration file is in **/etc/sysconfig/dirsrv**.

If there are multiple Directory Server instances and not all of them will use SASL authentication, then there can be instance-specific configuration files created in that directory named **dirsrv-instance**. For example, **dirsrv-example**. The default **dirsrv** file can be used if there is a single instance on a host.

To enable SASL authentication, uncomment the **KRB5_KTNAME** line in the **/etc/sysconfig/dirsrv** (or instance-specific) file, and set the keytab location for the **KRB5_KTNAME** variable. For example:

```

# In order to use SASL/GSSAPI the directory
# server needs to know where to find its keytab
# file - uncomment the following line and set
# the path and filename appropriately
KRB5_KTNAME=/etc/dirsrv/krb5.keytab ; export KRB5_KTNAME

```

7.12.4. Using an External Keytab

A default keytab file is specified in the Directory Server start script and is used by the Directory Server automatically. However, it is possible to specify a different keytab file, referencing a different principal, by manually running **kinit** and then specifying the cached credentials.

To specify the cached **kinit** credentials, add the principal as the **KRB5CCNAME** line in **/etc/sysconfig/dirsrv**:

```

KRB5CCNAME=/tmp/krb_ccache ; export KRB5CCNAME
kinit principalname
# how to provide the password here is left as an exercise
# or kinit -k -t /path/to/file.keytab principalname
chown serveruid:serveruid $KRB5CCNAME
# so the server process can read it
# start a cred renewal "daemon"
( while XXX ; do sleep NNN ; kinit ..... ; done ) &
# the exit condition XXX and sleep interval NNN are left as an exercise
...

```

The server has no way to renew these cached credentials. The **kinit** process must be run manually, external to Directory Server processes, or the server could begin receiving SASL bind failures when the server attempts to use expired credentials.

7.13. DISABLING SASL MECHANISMS

The root dse attribute ***supportedSASLMechanisms*** lists the SASL mechanisms that are *currently supported* by the Directory Server instance. However, editing that attribute does not change which mechanisms are supported. Directory Server uses the installed Cyrus SASL libraries to generate the list of supported SASL mechanisms. These libraries are located in ***/usr/lib[64]/sas12/***.

To change the list of SASL mechanisms supported by Directory Server:

1. Create a private SASL directory for the Directory Server instance to use. For example:

```
mkdir /etc/dirsrv/slapd-instance_name/sas12
```

2. Open that directory.

3. Create symlinks from the Cyrus SASL directory plug-ins to the instance directory. For example:

```
[root@server ~]# cd /etc/dirsrv/slapd-instance_name/sas12 ; for file
in /usr/lib64/sas12/*.so* ; do
    ln -s $file
done
```

4. Remove the symlinks for the mechanisms that should *not* be supported in the Directory Server instance. For example:

```
rm *cram*
```

5. Edit the Directory Server start shell script so that it uses the Directory Server instance's SASL directory.

```
vim /usr/lib[64]/dirsrv/slapd-example/start-slapd

SASL_PATH=/etc/dirsrv/slapd-instance_name/sas12 ; export SASL_PATH
```

6. Restart the Directory Server.

```
service dirsrv restart
```



NOTE

Cyrus SASL can set a specific list of mechanisms to use for different applications within its own configuration, in a ***/usr/lib/sas1/appName.conf*** file.

For more information, see the Cyrus SASL administrator's documentation at <http://cyrusimap.web.cmu.edu/docs/cyrus-sasl/1.5.28/sysadmin.php>.

7.14. USING SASL WITH LDAP CLIENTS

Directory Server uses SASL as an alternative TLS/SSL, particularly for environments which are using Kerberos to implement single sign-on. Directory Server allows user to use SASL to authenticate and bind to the server. This includes LDAP tools like ***ldapsearch*** and ***ldapmodify***. For example:

```
ldapsearch -O noplain,minssf=1,maxbufsize=512 -Y GSSAPI -U  
"dn:uid=jsmith,ou=people,dc=example,dc=com" -R EXAMPLE.COM ...
```

Using SASL with the LDAP client tools is described in [Section A.3, “Using SASL with LDAP Client Tools”](#).



NOTE

SASL proxy authorization is not supported in Directory Server; therefore, Directory Server ignores any SASL **authzid** value supplied by the client.

[2] While SSLv2 *can* be enabled, SSLv2 is disabled by default in Directory Server and generally does not need to be enabled.

CHAPTER 8. MANAGING THE DIRECTORY SCHEMA

Red Hat Directory Server comes with a standard schema that includes hundreds of object classes and attributes. While the standard object classes and attributes should meet most deployments' requirements, it can be necessary to extend the schema for specific directory data. Extending the schema is done by creating new object classes and attributes.

The [Red Hat Directory Server 9 Configuration, Command, and File Reference](#) is a reference for most the standard Directory Server attributes and object classes, with information on allowed and required attributes, which object classes take which attribute, and OID and value information. This is a good resource for identifying useful schema elements for a directory and determining what custom schema needs to be created.

8.1. OVERVIEW OF SCHEMA

The directory schema is a set of rules that defines how data can be stored in the directory. Directory information is stored discrete entries, and each entry is comprised of a set of attributes and their values. The kind of identity being described in the entry is defined in the entry's object classes. An object class specifies the kind of object the entry describes through the defined set of attributes for the object class.

In LDAP, an object class defines the set of attributes that can be used to define an entry. The LDAP standard provides object classes for many common types of entries, including people, groups, locations, organizations and divisions, and equipment. The identity is described in a directory entries with attributes and their values, pairs are called *attribute-value assertions* or AVAs. Any piece of information in the directory is associated with a descriptive attribute. Other aspects of the Directory Server configuration, including matching rules and LDAP controls, are also defined in the schema. All of these together are *schema elements*.

Every schema element is identified by a unique, dot-separated number. This is called the *object identifier* or *OID*.

8.1.1. Default Schema Files

The schema for Directory Server is defined in several different schema files (LDIF files which define schema elements). The Directory Server schema files are instance-specific and are located in the `/etc/dirsrv/slapd-instance_name/schema` directory. There is also a common `/etc/dirsrv/schema` directory; the files in this directory are used as templates for new Directory Server instances. Putting custom schema in the `/etc/dirsrv/schema` directory means that it is automatically included with any new instances.

The default schema files are listed and described in the [Red Hat Directory Server 9 Configuration, Command, and File Reference](#), which also describes the common standard attributes and object classes as well as the attributes used by the Directory Server to perform operations and manage entries.

8.1.2. Object Classes

In LDAP, an object class defines the set of attributes that can be used to define an entry. The LDAP standard provides object classes for many common types of entries, such as people (**person** and **inetOrgPerson**), groups (**groupOfUniqueNames**), locations (**locality**), organizations and divisions (**organization** and **organizationalUnit**), and equipment (**device**).

In a schema file, an object class is identified by the **objectclasses** line, then followed by its OID, name, a description, its direct superior object class (an object class which is required to be used in conjunction with the object class and which shares its attributes with this object class), and the list of

required (**MUST**) and allowed (**MAY**) attributes.

This is shown in [Example 8.1, “person Object Class Schema Entry”](#).

Example 8.1. person Object Class Schema Entry

```
objectClasses: ( 2.5.6.6 NAME 'person' DESC 'Standard LDAP objectclass'
SUP top MUST ( sn $ cn ) MAY ( description $ seeAlso $ telephoneNumber $
userPassword ) X-ORIGIN 'RFC 2256' )
```

Every object class defines a number of required attributes and of allowed attributes. Required attributes must be present in entries using the specified object class, while allowed attributes are permissible and available for the entry to use, but are not required for the entry to be valid.

As in [Example 8.1, “person Object Class Schema Entry”](#), the **person** object class requires the **cn**, **sn**, and **objectClass** attributes and allows the **description**, **seeAlso**, **telephoneNumber**, and **userPassword** attributes.

An object class can inherit attributes from another class, in addition to its own required and allowed attributes. The second object class is the *superior* or *parent* object class of the first.

For example, a user's entry has to have the **inetOrgPerson** object class. In that case, the entry must also include the superior object class for **inetOrgPerson**, **organizationalPerson**, and the superior object class for **organizationalPerson**, which is **person**:

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

8.1.3. Attributes

Directory entries are composed of attributes and their values. These pairs are called *attribute-value assertions* or AVAs. Any piece of information in the directory is associated with a descriptive attribute. For instance, the **cn** attribute is used to store a person's full name, such as **cn: John Smith**.

Additional attributes can supply additional information about John Smith:

```
givenname: John
surname: Smith
mail: jsmith@example.com
```

In a schema file, an attribute is identified by the **attributetypes** line, then followed by its OID, name, a description, syntax (allowed format for its value), optionally whether the attribute is single- or multi-valued, and where the attribute is defined.

This is shown in [Example 8.2, “description Attribute Schema Entry”](#).

Example 8.2. description Attribute Schema Entry

```

attributetypes: ( 2.5.4.13 NAME 'description' DESC 'Standard LDAP
attribute type' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORIGIN 'RFC 2256'
)

```

8.1.4. Extending the Schema

New, custom attributes and object classes can be added to a Directory Server instance to extend the schema, and there are several ways to add schema elements. Using the Directory Server Console or LDAP tools adds schema elements to the default custom schema file for an instance, `99user.ldif`. It is also possible to create a new, separate schema file and include it with the default schema files.

Adding new schema elements requires three things:

1. Planning and defining OIDs for the new schema. Schema elements are recognized by the server by their OID, so it is important for the OIDs to be unique and organized. Directory Server itself does not manage OIDs, but there are some best practices described in [Section 8.2, “Managing Object Identifiers”](#).
2. Create the new attributes. Attribute definitions require a name, a syntax (the allowed format of the values), an OID, and a description of whether the attribute can only be used once per entry or multiple times.
3. Create an object class to contain the new attributes. An object class lists the required attributes for that entry type and the allowed (permissible) attributes. Because the default schema should never be altered, if any new attributes are created, then they should be added to a custom object class.

The schema elements should be planned in advance; do not use multiple attributes for the same information. Whenever possible, use the standard Directory Server schema. Directory Server has hundreds of attributes and dozens of object classes defined in the default schema files. The [Red Hat Directory Server 9 Configuration, Command, and File Reference](#) lists and describes the standard attributes and object classes; all of the schema can be viewed in the Directory Server Console or read in the schema files in `/etc/dirsrv/slapd-instance_name/schema`. Become familiar with the available schema; then plan what information attributes are missing and how best to fill those gaps with custom attributes. Planning the schema is covered in the *Deployment Guide*.



WARNING

The default object classes and attributes in Directory Server are based on LDAP and X.500 standards and RFCs. Using standard schema makes the Directory Server more easily integrated with other applications and servers and allows interoperability with LDAP clients, legacy Directory Server instances, and future release. It is inadvisable for you to edit the standard attributes or change the object classes.

Keep the following rules in mind when customizing the Directory Server schema:

- Keep the schema as simple as possible.
- Reuse existing schema elements whenever possible.

- Minimize the number of mandatory attributes defined for each object class.
- Do not define more than one object class or attribute for the same purpose.
- Do not modify any existing definitions of attributes or object classes.



NOTE

Never delete or replace the standard schema. Doing so can lead to compatibility problems with other directories or other LDAP client applications.

The schema is loaded into the Directory Server instance when the instance is started; any new schema files are not loaded until the Directory Server is restarted or unless a reload task is initiated. Always, any custom schema is loaded before the standard schema.

8.1.5. Schema Replication

When the directory schema is updated in the **cn=schema** sub-tree, Directory Server stores the changes in the local **/etc/dirsrv/slapd-*instance_name*/schema/99user.ldif** file, including a change state number (CSN). The updated schema is not automatically replicated to other replicas. The schema replication starts when directory content is updated in the replicated tree. For example, if you update a user or group entry after modifying the schema, the supplier compares the CSN stored in the **nsSchemaCSN** attribute with the one on the consumer. If the remote CSN is lower than the one on the supplier, the schema is replicated to the consumer. For a successful replication, all object classes and attribute types on the supplier must be a superset of the consumer's definition.

Example 8.3. Schema subsets and supersets

- On **server1**, the **demo** object class allows the **a1**, **a2**, and **a3** attributes.
- On **server2**, the **demo** object class allows the **a1** and **a3** attributes.

In [Example 8.3, “Schema subsets and supersets”](#), the schema definition of the **demo** object class on **server1** is a superset of the object class on **server2**. During the validation phase, when the schema is being replicated or accepted, Directory Server retrieves the superset definitions. For example, if a consumer detects that an object class in the local schema allows less attributes than the object class in the supplier schema, the local schema is updated.

If the schema definitions are successfully replicated, the **nsSchemaCSN** attributes are identical on both servers and no longer compared at the beginning of a replication session.

In the following scenarios, the schema is not replicated:

- The schema on one host is a subset of the schema of another host.

For example, in [Example 8.3, “Schema subsets and supersets”](#), the schema definition of the **demo** object class on **server2** is a subset of the object class on **server1**. Subsets can also occur for attributes (a single-value attribute is a subset of a multi-value attribute) and attribute syntaxes (**IA5** is a subset of **Octet_string**).

- When definitions in supplier schema and consumer schema need to be merged.

Directory Server does not support merging schemas. For example, if an object class on one server allows the **a1**, **a2**, and **a3** attributes and **a1**, **a3**, and **a4** on the other, the schemas are not subsets and cannot be merged.

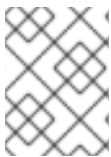
- Schema files other than `/etc/dirsrv/slapd-instance_name/schema/99user.ldif` are used.

Directory Server enables you to add additional schema files in the `/etc/dirsrv/slapd-instance_name/schema/` directory. However, only the CSN in the **99user.ldif** file is updated. For this reasons, other schema file are only used locally and are not automatically transferred to replication partners. Copy the updated schema file manually to the consumers and reload the schema. For details, see [Section 8.7, “Dynamically Reloading Schema”](#).

To avoid duplicate schema definitions and to enable automatic replication, store all custom schema in the `/etc/dirsrv/slapd-instance_name/schema/99user.ldif` file. For further information about creating custom schema files, see [Section 8.6, “Creating Custom Schema Files”](#).

8.2. MANAGING OBJECT IDENTIFIERS

Each LDAP object class or attribute must be assigned a unique name and *object identifier* (OID). An OID is a dot-separated number which identifies the schema element to the server. OIDs can be hierarchical, with a base OID that can be expanded to accommodate different branches. For example, the base OID could be **1**, and there can be a branch for attributes at **1.1** and for object classes at **1.2**.



NOTE

It is not required to have a numeric OID for creating custom schema, but Red Hat strongly recommends it for better forward compatibility and performance.

OIDs are assigned to an organization through the Internet Assigned Numbers Authority (IANA), and Directory Server does not provide a mechanism to obtain OIDs. To get information about obtaining OIDs, visit the IANA website at <http://www.iana.org/cgi-bin/enterprise.pl>.

After obtaining a base OID from IANA, plan how the OIDs are going to be assigned to custom schema elements. Define a branch for both attributes and object classes; there can also be branches for matching rules and LDAP controls.

Once the OID branches are defined, create an OID registry to track OID assignments. An OID registry is a list that gives the OIDs and descriptions of the OIDs used in the directory schema. This ensures that no OID is ever used for more than one purpose. Publish the OID registry with the custom schema.

8.3. DIRECTORY SERVER ATTRIBUTE SYNTAXES

The attribute's syntax defines the format of the values which the attribute allows; as with other schema elements, the syntax is defined for an attribute using the syntax's OID in the schema file entry. In the Directory Server Console, the syntax is referenced by its friendly name.

The Directory Server uses the attribute's syntax to perform sorting and pattern matching on entries.

For more information about LDAP attribute syntaxes, see [RFC 4517](#).

Table 8.1. Supported LDAP Attribute Syntaxes

Name	OID	Definition
Binary	1.3.6.1.4.1.1466.115.121.1.5	<i>Deprecated. Use Octet string instead.</i>
Bit String	1.3.6.1.4.1.1466.115.121.1.6	For values which are bitstrings, such as '0101111101'B .
Boolean	1.3.6.1.4.1.1466.115.121.1.7	For attributes with only two allowed values, TRUE or FALSE.
Country String	1.3.6.1.4.1.1466.115.121.1.11	For values which are limited to exactly two printable string characters; for example, US for the United States.
DN	1.3.6.1.4.1.1466.115.121.1.12	For values which are distinguished names (DNs).
Delivery Method	1.3.6.1.4.1.1466.115.121.1.14	For values which are contained a preferred method of delivering information or contacting an entity. The different values are separated by a dollar sign (\$). For example: telephone \$ physical
Directory String	1.3.6.1.4.1.1466.115.121.1.15	For values which are valid UTF-8 strings. These values are not necessarily case-insensitive. Both case-sensitive and case-insensitive matching rules are available for Directory String and related syntaxes.
Enhanced Guide	1.3.6.1.4.1.1466.115.121.1.21	For values which contain complex search parameters based on attributes and filters.
Facsimile	1.3.6.1.4.1.1466.115.121.1.22	For values which contain fax numbers.
Fax	1.3.6.1.4.1.1466.115.121.1.23	For values which contain the images of transmitted faxes.
Generalized Time	1.3.6.1.4.1.1466.115.121.1.24	For values which are encoded as printable strings. The time zone must be specified. It is strongly recommended to use GMT time.

Name	OID	Definition
Guide	1.3.6.1.4.1.1466.115.121.1.25	<i>Obsolete.</i> For values which contain complex search parameters based on attributes and filters.
IA5 String	1.3.6.1.4.1.1466.115.121.1.26	For values which are valid strings. These values are not necessarily case-insensitive. Both case-sensitive and case-insensitive matching rules are available for IA5 String and related syntaxes.
Integer	1.3.6.1.4.1.1466.115.121.1.27	For values which are whole numbers.
JPEG	1.3.6.1.4.1.1466.115.121.1.28	For values which contain image data.
Name and Optional UID	1.3.6.1.4.1.1466.115.121.1.34	For values which contain a combination value of a DN and (optional) unique ID.
Numeric String	1.3.6.1.4.1.1466.115.121.1.36	For values which contain a string of both numerals and spaces.
OctetString	1.3.6.1.4.1.1466.115.121.1.40	For values which are binary; this replaces the binary syntax.
OID	1.3.6.1.4.1.1466.115.121.1.37	For values which contain an object identifier (OID).
Postal Address	1.3.6.1.4.1.1466.115.121.1.41	<p>For values which are encoded in the format postal-address = <i>dstring</i> * ("\$" <i>dstring</i>). For example:</p> <pre>1234 Main St.\$Raleigh, NC 12345\$USA</pre> <p>Each <i>dstring</i> component is encoded as a DirectoryString value. Backslashes and dollar characters, if they occur, are quoted, so that they will not be mistaken for line delimiters. Many servers limit the postal address to 6 lines of up to thirty characters.</p>

Name	OID	Definition
PrintableString	1.3.6.1.4.1.1466.115.121.1.58	For values which contain strings containing alphabetic, numeral, and select punctuation characters (as defined in RFC 4517).
Space-Insensitive String	2.16.840.1.113730.3.7.1	For values which contain space-insensitive strings.
TelephoneNumber	1.3.6.1.4.1.1466.115.121.1.50	For values which are in the form of telephone numbers. It is recommended to use telephone numbers in international form.
Teletex Terminal Identifier	1.3.6.1.4.1.1466.115.121.1.51	For values which contain an international telephone number.
Telex Number	1.3.6.1.4.1.1466.115.121.1.52	For values which contain a telex number, country code, and answerback code of a telex terminal.
URI		For values in the form of a URL, introduced by a string such as http:// , https:// , ftp:// , ldap:// , and ldaps:// . The URI has the same behavior as IA5 String. See RFC 4517 for more information on this syntax.

8.4. MANAGING CUSTOM SCHEMA IN THE CONSOLE

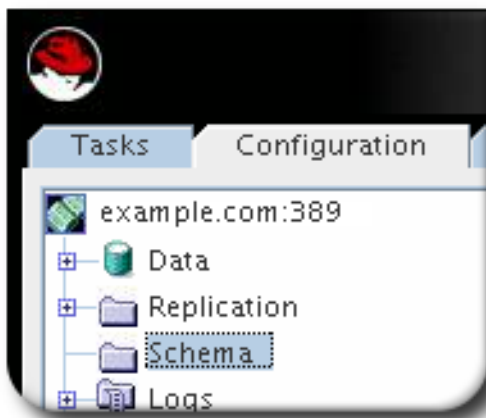
The Directory Server Console shows all attributes in the schema, and custom attributes can be created, edited, and deleted from the schema.

- [Section 8.4.1, “Viewing Attributes and Object Classes”](#)
- [Section 8.4.2, “Creating Attributes”](#)
- [Section 8.4.3, “Creating Object Classes”](#)
- [Section 8.4.4, “Editing Custom Schema Elements”](#)
- [Section 8.4.5, “Deleting Schema”](#)

8.4.1. Viewing Attributes and Object Classes

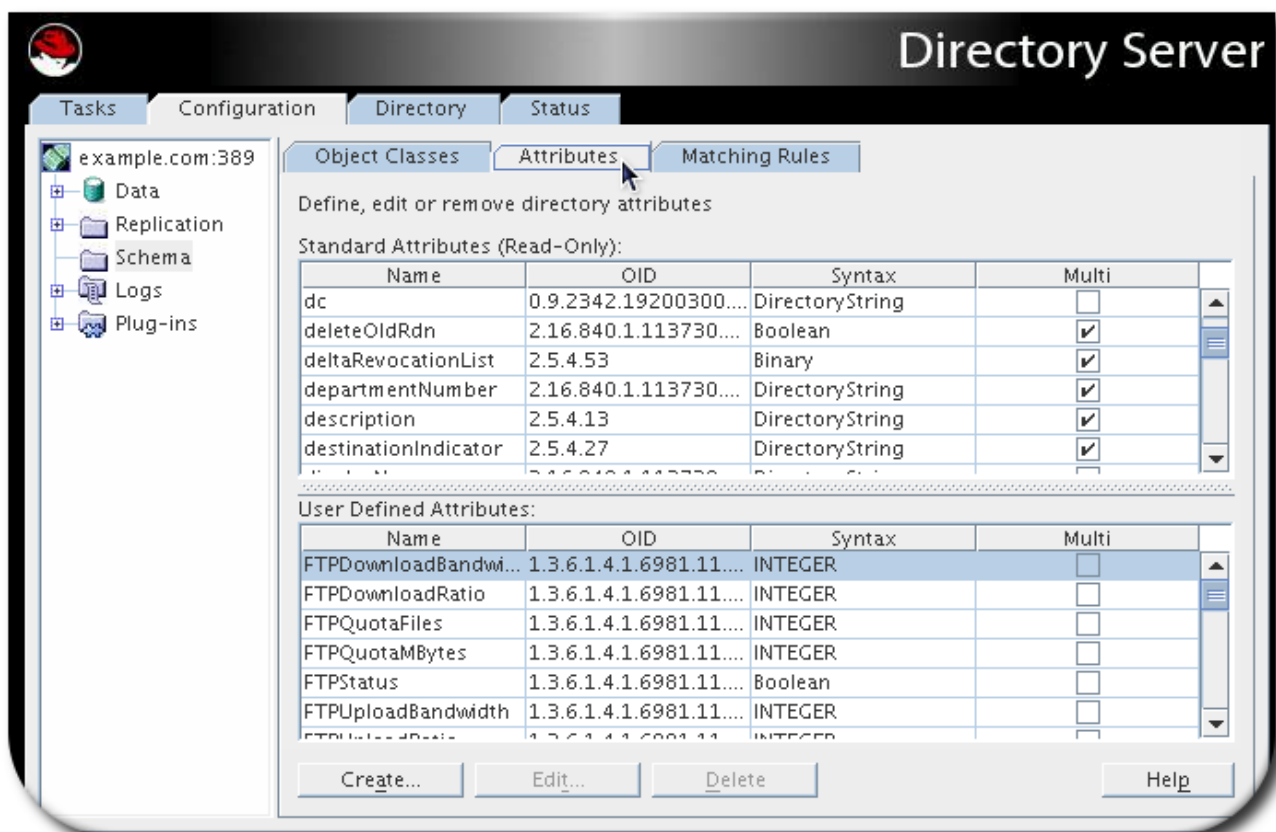
All of the information about the attributes and object classes which are currently loaded in the server instance are visible with the other server configuration.

1. In the Directory Server Console, select the **Configuration** tab.
2. In the left navigation tree, select the **Schema** folder.

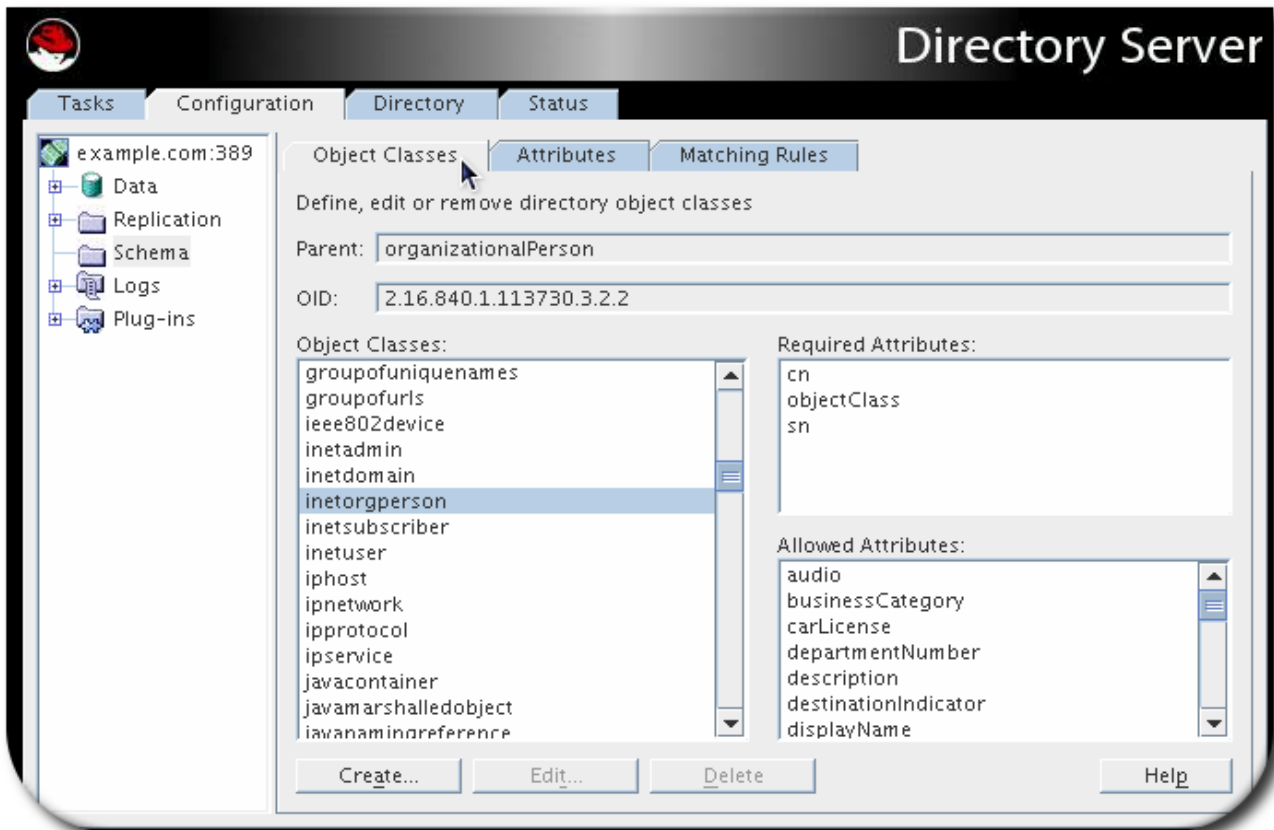


3. There are three tabs which display the schema elements loaded in Directory Server: **Object Class**, **Attributes**, and **Matching Rules**.

The **Attributes** tab is broken into two sections for default and custom attributes. Both sections show the attribute name, OID, syntax, and whether the attribute is multi-valued.



The **Object Classes** tab shows the list of object classes on the left. When an object class is highlighted, its OID and superior object class are listed in the fields at the top and its required and allowed attributes are listed in the boxes on the right.



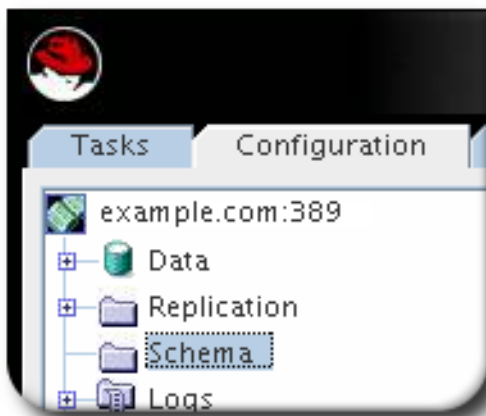
8.4.2. Creating Attributes



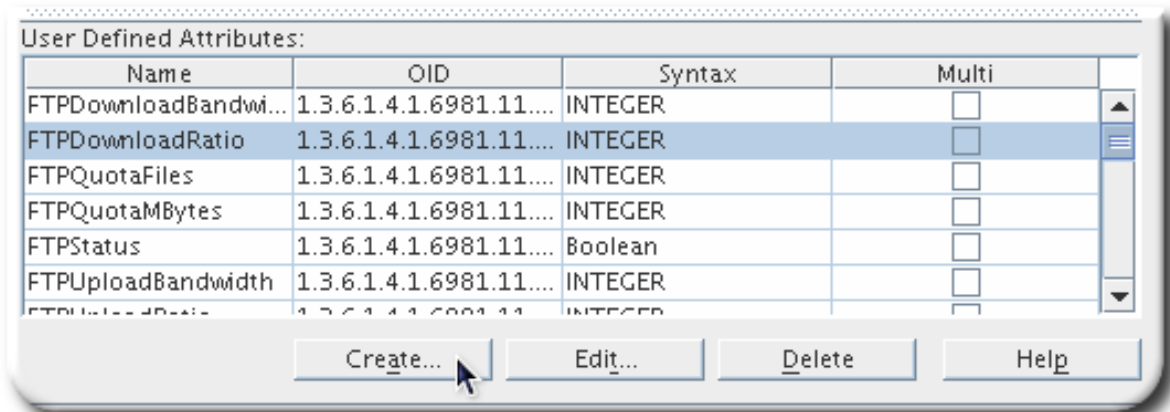
NOTE

After adding new attributes to the schema, create a new object class to contain them, as described in [Section 8.4.3, “Creating Object Classes”](#).

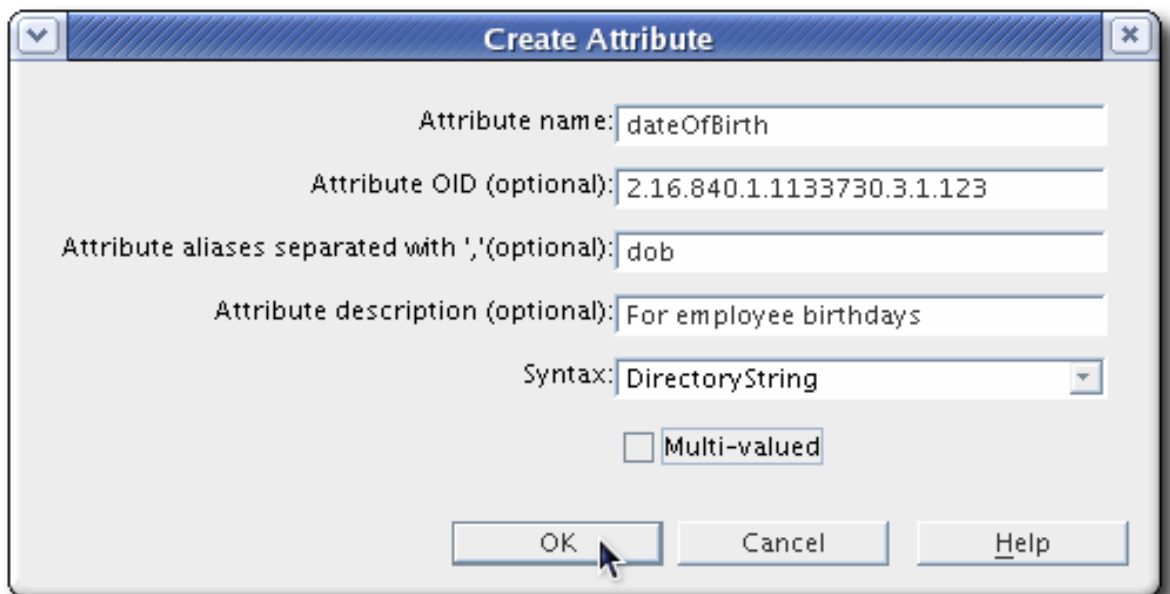
1. Select the **Configuration** tab.
2. In the left navigation tree, select the **Schema** folder, and then select the **Attributes** tab in the right pane.



3. Click **Create**.



4. Fill in the information for the new attribute.

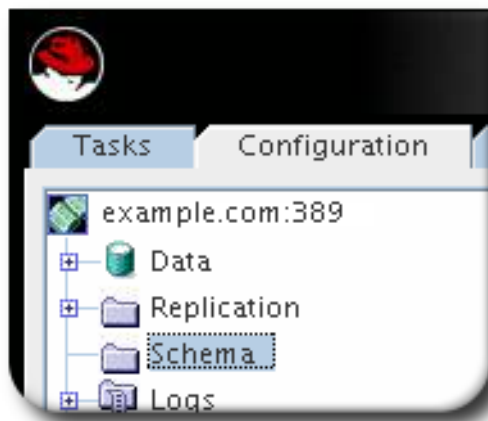


- The attribute name; this must be unique.
 - The OID; this is not required, but for compatibility and server performance, assigning a unique numeric OID is strongly recommended.
 - The syntax; this is the allowed format for the attributes values.
 - Whether the attribute is multi-valued; by default, all attributes can be used more than once in an entry, but deselecting the check box means the attribute can be used only once.
5. Click **OK**.

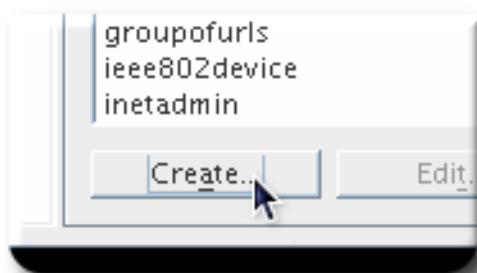
8.4.3. Creating Object Classes

A new object class must be created with a unique name, a parent object, and required and optional attributes. To create an object class:

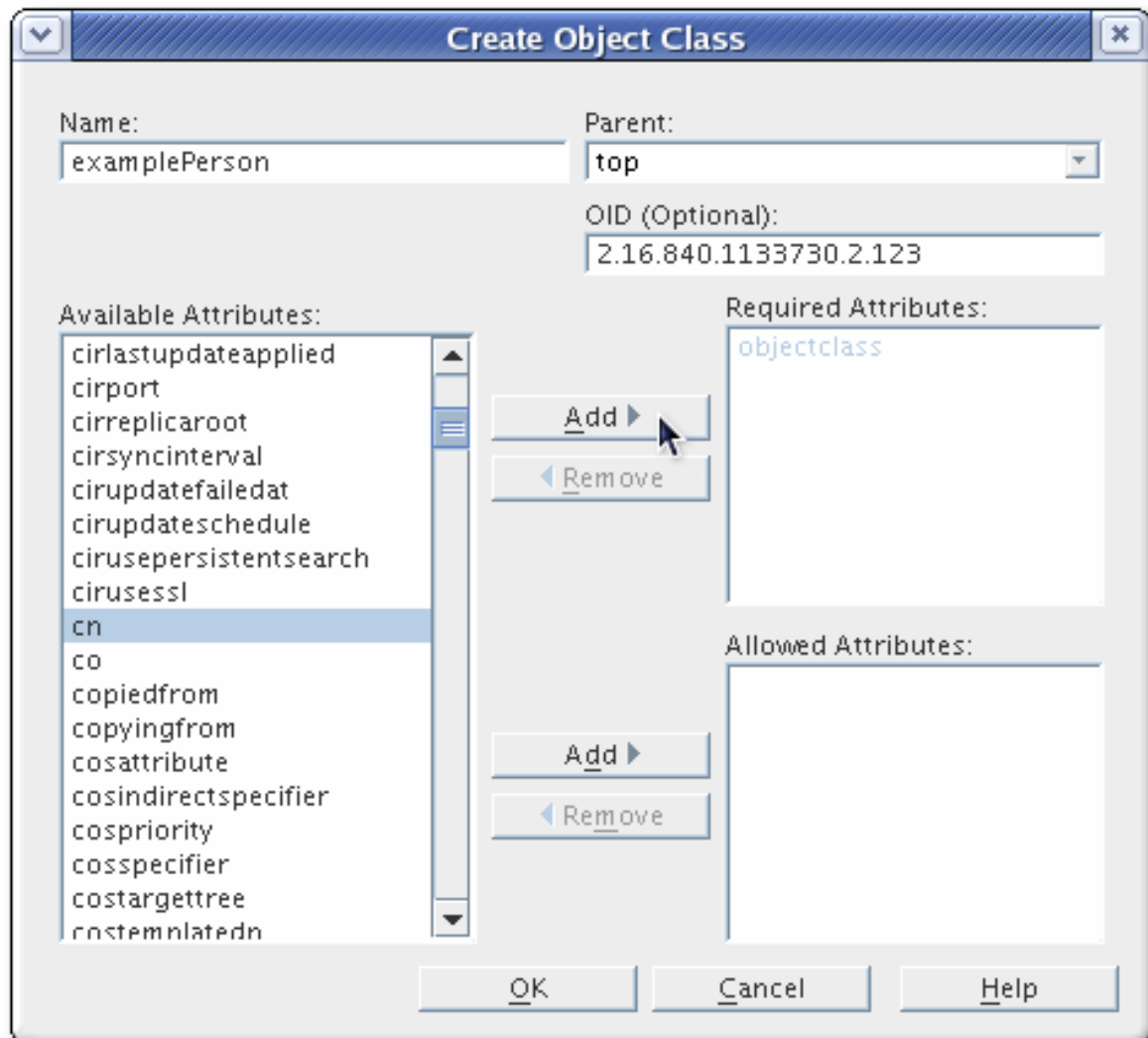
1. In the Directory Server Console, select the **Configuration** tab.
2. In the navigation tree, select the **Schema** folder, and then select the **Object Classes** tab in the right pane.



3. Click the **Create** button in the **Object Classes** tab.



4. Fill in the information about the new object class.



- The name; this must be unique.
- The OID; this is not required, but for compatibility and server performance, assigning a unique numeric OID is strongly recommended.
- The superior object class for the entry. The default is **top**; selecting another object class means that the new object class inherits all of the required and allowed attributes from the parent, in addition to its own defined attributes.
- Required and allowed attributes. Select the attributes on the left and used the **Add** buttons by the **Available Attributes** and **Required Attributes** boxes to add the attributes as appropriate.



NOTE

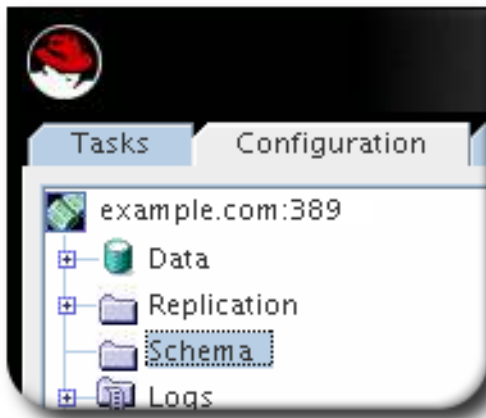
Attributes that are inherited from the parent object classes cannot be removed, regardless of whether they are allowed or required.

5. Click **OK** to save the new object class.

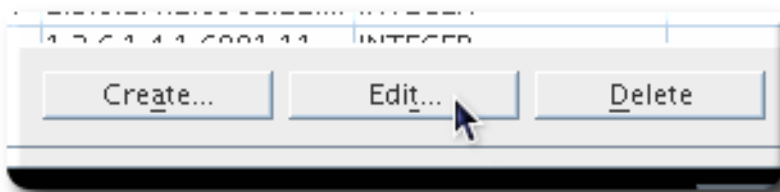
8.4.4. Editing Custom Schema Elements

Only user-created attributes or object classes can be edited; standard schema elements cannot be edited.

1. In the Directory Server Console, select the **Configuration** tab.
2. In the left navigation tree, select the **Schema** folder.



3. Open the **Object Classes** or **Attributes** tab.
4. Select the schema element to edit from the list. Only custom (user-defined) schema can be edited in the Directory Server Console.
5. Click the **Edit** button at the bottom of the window.

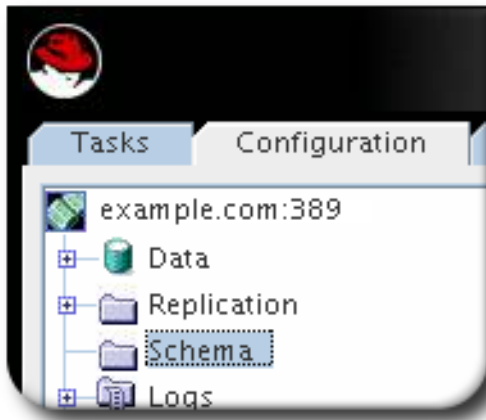


6. Edit any of the schema information.

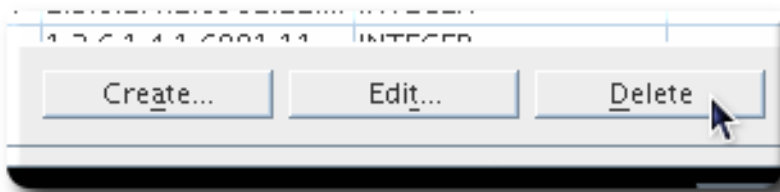
8.4.5. Deleting Schema

Only user-created attributes or object classes can be deleted; standard schema elements cannot be deleted.

1. In the Directory Server Console, select the **Configuration** tab.
2. In the left navigation tree, select the **Schema** folder.



3. Open the **Object Classes** or **Attributes** tab.
4. Select the schema element to delete from the list. Only custom (user-defined) schema can be deleted in the Directory Server Console.
5. Click the **Delete** button at the bottom of the window.



6. Confirm the deletion.



WARNING

The server immediately deletes the schema element. There is no undo.

8.5. MANAGING SCHEMA USING LDAPMODIFY

As with the Directory Server Console, **ldapmodify** can be used to add, edit, and delete custom schema elements. **ldapmodify** also modifies the default custom schema file for a Directory Server instance, **99user.ldif**.

8.5.1. Creating Attributes

A custom attribute entry is itself an **attributetypes** entry for the **cn=schema** entry. The **attributetypes** attribute has the format:

```
attributetypes: ( definition )
```

The definition contains five components:

- An OID, usually a dot-separated number
- A unique name, in the form **NAME** *name*
- A description, in the form **DESC** *description*
- The OID for the syntax of the attribute values, listed in [Table 8.1, “Supported LDAP Attribute Syntaxes”](#), in the form **SYNTAX** *OID*
- Optionally, the source where the attribute is defined

The attribute definition is added to the custom schema file, **99user.ldif**, by running an LDAP command and modifying the **cn=schema** entry. For example:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -v  
  
dn: cn=schema  
changetype: modify  
add: attributetypes  
attributetypes: ( 1.2.3.4.5.6.1 NAME 'dateofbirth' DESC 'For employee  
birthdays' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUED X-ORIGIN  
'Example defined')
```

8.5.2. Creating Object Classes

An object class definition is an **objectclasses** attribute for the **cn=schema** entry. The **objectclasses** attribute has the format:

```
objectclasses: ( definition )
```

The object class definition contains several components:

- An OID, usually a dot-separated number
- A unique name, in the form **NAME** *name*
- A description, in the form **DESC** *description*
- The superior, or parent, object class for this object class, in the form **SUP** *object_class*; if there is no related parent, use **SUP top**
- The word **AUXILIARY**, which gives the type of entry to which the object class applies; **AUXILIARY** means it can apply to any entry
- A list of required attributes, preceded by the word **MUST**; to include multiple attributes, enclose the group in parentheses and separate with attributes with dollar signs (\$)
- A list of allowed attributes, preceded by the word **MAY**; to include multiple attributes, enclose the group in parentheses and separate with attributes with dollar signs (\$)

The object class definition is added to the custom schema file, **99user.ldif**, by running an LDAP command and modifying the **cn=schema** entry. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x -v
```

```
dn: cn=schema
changetype: modify
add: objectclasses
objectclasses: ( 2.16.840.1133730.2.123 NAME 'examplePerson' DESC 'Example
Person Object Class' SUP inetOrgPerson AUXILIARY MUST cn MAY
(exampleDateOfBirth $ examplePreferredOS) )
```

8.5.3. Deleting Schema



WARNING

Never delete default schema elements. Those are required by the Directory Server to run.

1. Remove the unwanted attributes from any entries which use them, then from any object classes in the schema file which accept that attribute. Likewise, to remove an object class, remove it from any entries.
2. Run **ldapmodify** to remove the attribute. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=schema
changetype: modify
delete: objectclasses
objectclasses: ( 2.16.840.1133730.2.123 NAME 'examplePerson' DESC
'Example Person Object Class' SUP inetOrgPerson AUXILIARY MUST cn
MAY (exampleDateOfBirth $ examplePreferredOS) )
```



WARNING

Be sure to specify the exact object class or attribute to remove; using only the **attributetypes** or **objectclasses** attribute without the value will delete every user-defined attribute or object class in the file.

If the custom attribute or object class is in a custom schema file other than **99user.ldif**, edit the file directly. Neither the Directory Server Console nor LDAP tools can edit a schema file other than **99user.ldif**.

8.6. CREATING CUSTOM SCHEMA FILES

Schema files are simple LDIF files which define the **cn=schema** entry. Each attribute and object class is added as an attribute to that entry. Here are the requirements for creating a schema file:

- The first line must be **dn: cn=schema**.
- The schema file can include both attributes and object classes, but it can also include only one or the other.
- If both attributes and object classes are defined in the style, all of the attributes must be listed in the file first, then all of the attributes must be listed first, then the object classes.
- The object classes can use attributes defined in other schema files.
- The file must be named in the format **[1-9][0-9]text.ldif**.

The file must always begin with two numbers. Numerically, the schema file cannot be loaded before the core configuration schema (which begin with **00** and **01**).

Also, the Directory Server always writes its custom schema to the numerically and alphabetically highest named schema file in the schema directory. It expects this file to be **99user.ldif**. If this file is not **99user.ldif**, the server can experience problems. So, always make sure custom schema files are at least alphabetically lower than **99user.ldif**. The name **99alpha.ldif** is okay; the name **99zzz.ldif** is not.

Practices for creating schema files are described in more detail in the *Deployment Guide*.

Attributes are defined in the schema file as **attributetypes** attributes to the schema, with five components:

- An OID, usually a dot-separated number
- A unique name, in the form **NAME name**
- A description, in the form **DESC description**
- The OID for the syntax of the attribute values, listed in [Table 8.1, “Supported LDAP Attribute Syntaxes”](#), in the form **SYNTAX OID**
- Optionally, the source where the attribute is defined

For example:

```
attributetypes: ( 1.2.3.4.5.6.1 NAME 'dateofbirth' DESC 'For employee  
birthdays' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUED X-ORIGIN  
'Example defined')
```

Likewise, object classes are defined as **objectclasses** attributes, although there is slightly more flexibility in how the object class is defined. The only required configurations are the name and OID for the object class; all other configuration depends on the needs for the object class:

- An OID, usually a dot-separated number
- A unique name, in the form **NAME name**
- A description, in the form **DESC description**

- The superior, or parent, object class for this object class, in the form **SUP** *object_class*; if there is no related parent, use **SUP top**
- The word **AUXILIARY**, which gives the type of entry to which the object class applies; **AUXILIARY** means it can apply to any entry
- A list of required attributes, preceded by the word **MUST**; to include multiple attributes, enclose the group in parentheses and separate with attributes with dollar signs (\$)
- A list of allowed attributes, preceded by the word **MAY**; to include multiple attributes, enclose the group in parentheses and separate with attributes with dollar signs (\$)

For example:

```
objectclasses: ( 2.16.840.1133730.2.123 NAME 'examplePerson' DESC 'Example
Person Object Class' SUP inetOrgPerson AUXILIARY MUST cn MAY
(exampleDateOfBirth $ examplePreferredOS) )
```

Example 8.4, “Example Schema File” shows a simplified schema file.

Example 8.4. Example Schema File

```
dn: cn=schema
attributetypes: ( 2.16.840.1133730.1.123 NAME 'dateofbirth' DESC 'For
employee birthdays' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORIGIN
'Example defined')
objectclasses: ( 2.16.840.1133730.2.123 NAME 'examplePerson' DESC
'Example Person Object Class' SUP inetOrgPerson AUXILIARY MAY
(dateofbirth) )
```

Custom schema files should be added to the Directory Server instance's schema directory, **/etc/dirsrv/slapd-*instance_name*/schema**. The schema in these files are not loaded and available to the server unless the server is restarted or a dynamic reload task is run.

8.7. DYNAMICALLY RELOADING SCHEMA

By default, the schema files used by the Directory Server instance are loaded into the directory when it is started. This means that any new schema files added to the schema directory are not available for use unless the server is restarted. The Directory Server has a task which manually reloads the full schema for the Directory Server instance, including custom files, without requiring a server restart.

The schema reload task can be initiated in two ways:

- Using the **schema-reload.pl** script
- Adding a **cn=schema reload** task entry using **ldapmodify**

8.7.1. Reloading Schema Using **schema-reload.pl**

The **schema-reload.pl** script launches a special task to reload all of the schema files used by a specific Directory Server instance. This allows custom schema files to be loaded dynamically without having to add schema elements to **99user.ldif**.

1. Open the tool directory for the Directory Server instance, `/usr/lib/dirsrv/slapd-instance_name/`.
2. Run the script, binding as the Directory Manager.

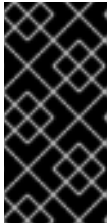
```
./schema-reload.pl -D "cn=Directory Manager" -w secret
```

The Directory Server responds that it has added the new reload task entry.

```
adding new entry cn=schema_reload_2009_1_6_17_52_4,cn=schema reload task,cn=tasks,cn=config
```

This reloads the schema from the default schema directory, `/etc/dirsrv/slapd-instance_name/schema`, which is recommended. It is also possible to specify a different directory using the `-d` option.

```
./schema-reload.pl -D "cn=Directory Manager" -w password -d /export/custom-schema
```



IMPORTANT

All of the schema is reloaded with the schema reload task, not just the newest schema or modified schema. This means that whatever directory is specified should contain the full schema for the Directory Server, or the Directory Server instance may have serious performance problems.

The **schema-reload.pl** is described in more detail in the [Configuration and Command-Line Tool Reference](#).

8.7.2. Reloading Schema Using ldapmodify

The **schema-reload.pl** script creates a special task entry in a Directory Server instance which reloads schema files; it is also possible to reload schema by creating the task entry directly. Task entries occur under the **cn=tasks** configuration entry in the **dse.ldif** file, so it is also possible to initiate a task by adding the entry using **ldapmodify**. As soon as the task is complete, the entry is removed from the directory.

To initiate a schema reload task, add an entry under the **cn=schema reload task,cn=tasks,cn=config** entry. The only required attribute is the **cn** for the specific task.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=example schema reload,cn=schema reload task,cn=tasks,cn=config
changetype: add
objectclass: extensibleObject
cn:example schema reload
```

The default schema directory from which the Directory Server instance reloads the schema is in `/etc/dirsrv/slapd-instance_name/schema`; it is possible to specify a different schema directory using the **schemadir** attribute, which is analogous to the `-d` option with **schema-reload.pl**.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
```



```
dn: cn=example schema reload,cn=schema reload task,cn=tasks,cn=config
changetype: add
objectclass: extensibleObject
cn:example schema reload
schemadir: /export/schema
```



IMPORTANT

All of the schema is reloaded with the schema reload task, not just the newest schema or modified schema. This means that whatever directory is specified should contain the full schema for the Directory Server, or the Directory Server instance may have serious performance problems.

As soon as the task is completed, the entry is deleted from the **dse.ldif** configuration, so it is possible to reuse the same task entry continually.

The **cn=schema reload task** configuration is described in more detail in the [Configuration and Command-Line Tool Reference](#).

8.7.3. Reloading Schema with Replication

The schema reload task is a local operation, so schema changes are not replicated in a multi-master environment if the schema is added to one supplier but not to the others. To load the new schema files on all of the supplier servers:

1. Stop replication.
2. Copy the new schema file over and run the schema reload task for every supplier and replica server.
3. Restart replication.

8.7.4. Schema Reload Errors

When the schema reload task runs, the command prompt only shows that the task is initiated.

```
adding new entry cn=schema reload task 1,cn=schema reload
task,cn=tasks,cn=config
```

However, the task does not return whether it completed successfully. To verify the schema reload operation was successful, check the error logs. The schema reload has two tasks, first validating the schema file and then loading it.

A success message shows that the validation passed and the task finished.

```
[06/Jan/2009:17:52:04 -0500] schemareload - Schema reload task starts
(schema dir: default) ...
[06/Jan/2009:17:52:04 -0500] schemareload - Schema validation passed.
[06/Jan/2009:17:52:04 -0500] schemareload - Schema reload task finished.
```

If there is a failure, then the logs show which step failed and why.

```
[..] schemareload - Schema reload task starts (schema dir: /bogus) ...  
[..] schema - No schema files were found in the directory /bogus  
[..] schema_reload - schema file validation failed  
[..] schemareload - Schema validation failed.
```

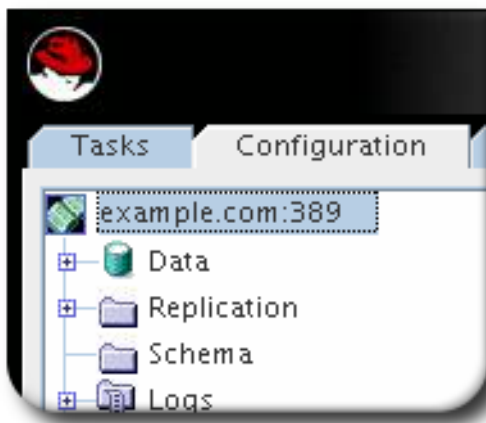
8.8. TURNING SCHEMA CHECKING ON AND OFF

When schema checking is on, the Directory Server ensures three things:

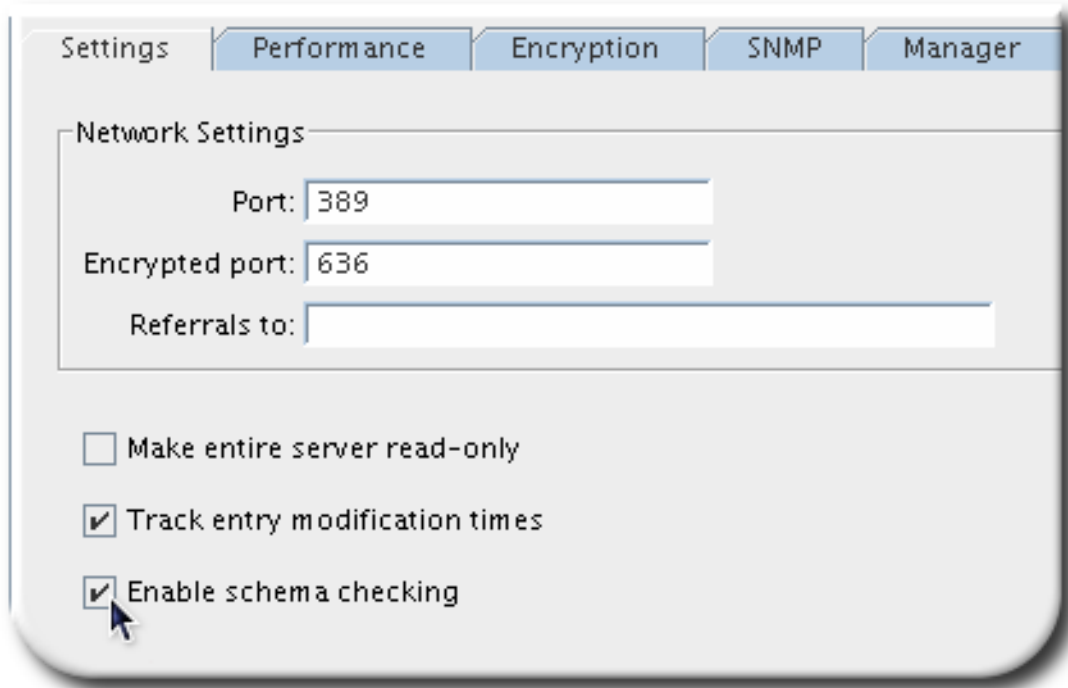
- The object classes and attributes using are defined in the directory schema.
- The attributes required for an object class are contained in the entry.
- Only attributes allowed by the object class are contained in the entry.

Schema checking is turned on by default in the Directory Server, and the Directory Server should always run with schema checking turned on. The only situation where it may be beneficial to turn schema checking off is to accelerate LDAP import operations. However, there is a risk of importing entries that do not conform to the schema. Consequently, it is impossible to search for these entries.

1. In the Directory Server Console, select the **Configuration** tab.



2. Highlight the server icon at the top of the navigation tree, then select the **Settings** tab in the right pane.
3. To enable schema checking, check the **Enable Schema Checking** check box; clear it to turn off schema checking.



4. Click **Save**.

To turn schema checking on and off using LDAP commands, edit the value of the **nsslapd-schemacheck** attribute. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=config
changetype: modify
replace: nsslapd-schemacheck: on
nsslapd-schemacheck: off
```

For information, see the *Directory Server Configuration and Command-Line Tool Reference*.

8.9. USING SYNTAX VALIDATION

With syntax validation, the Directory Server checks that the value of an attribute follows the rules of the syntax given in the definition for that attribute. For example, syntax validation will confirm that a new **telephoneNumber** attribute actually has a valid telephone number for its value.

8.9.1. About Syntax Validation

As with schema checking, validation reviews any directory modification and rejects changes that violate the syntax. Additional settings can be optionally configured so that syntax validation can log warning messages about syntax violations and then either reject the modification or allow the modification process to succeed.

This feature validates all attribute syntaxes, with the exception of binary syntaxes (which cannot be verified) and non-standard syntaxes, which do not have a defined required format. The syntaxes are validated against [RFC 4514](#).

8.9.2. Syntax Validation and Other Directory Server Operations

Syntax validation is mainly relevant for standard LDAP operations like creating entries (add) or editing attributes (modify). Validating attribute syntax can impact other Directory Server operations, however.

Database Encryption

For normal LDAP operations, an attribute is encrypted just before the value is written to the database. This means That encryption occurs *after* the attribute syntax is validated.

Encrypted databases (as described in [Section 2.2.3, “Configuring Attribute Encryption”](#)) can be exported and imported. Normally, it is strongly recommended that these export and import operations are done with the **-E** flag with **db2ldif** and **ldif2db**, which allows syntax validation to occur just fine for the import operation. However, if the encrypted database is exported without using the **-E** flag (which is not supported), then an LDIF with encrypted values is created. When this LDIF is then imported, the encrypted attributes cannot be validated, a warning is logged, and attribute validation is skipped in the imported entry.

Synchronization

There may be differences in the allowed or enforced syntaxes for attributes in Windows Active Directory entries and Red Hat Directory Server entries. In that case, the Active Directory values could not be properly synced over because syntax validation enforces the RFC standards in the Directory Server entries.

Replication

If the Directory Server 9.0 instance is a supplier which replicates its changes to a consumer, then there is no issue with using syntax validation. However, if the supplier in replication is an older version of Directory Server or has syntax validation disabled, then syntax validation should not be used on the 9.0 consumer because the Directory Server 9.0 consumer may reject attribute values that the master allows.

8.9.3. Enabling or Disabling Syntax Validation

Syntax validation is configured by the **nsslapd-syntaxcheck** attribute. The value of this attribute is either **on** or **off** (by default, this is on). To change the syntax validation, modify this attribute using **ldapmodify** or by editing the **dse.ldif** file directly.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
```

```
dn: cn=config
changetype: modify
replace: nsslapd-syntaxcheck
nsslapd-syntaxcheck: off
```



NOTE

If syntax validation is disabled, then run the **syntax-validate.pl** script to audit existing attribute values before re-enabling syntax validation. See [Section 8.9.6, “Validating the Syntax of Existing Attribute Values”](#).

8.9.4. Enabling Strict Syntax Validation for DNs

When syntax validation is enabled, DNs are validated against [RFC 4514](#), as are other attribute syntaxes. However, DN syntax validation is enabled separately because the strictness of later standards can invalidate old-style DNs, and therefore directory trees.

Syntax validation checks DNs against section 3 in [RFC 4514](#).

The value of this attribute is either **on** or **off** (by default, this is off). To change the syntax validation, modify this attribute using **ldapmodify** or by editing the **dse.ldif** file directly.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=config
changetype: modify
replace: nsslapd-dn-validate-strict
nsslapd-dn-validate-strict: on
```



NOTE

If strict DN validation is enabled and a DN value does not conform to the required syntax, then the operation fails with LDAP result code 34, **INVALID_DN_SYNTAX**.

8.9.5. Enabling Syntax Validation Warnings (Logging)

By default, syntax validation rejects any add or modify operations where an attribute value violates the required syntax. However, the violation itself is not recorded to the errors log by default. The **nsslapd-syntaxlogging** attribute enables error logging for any syntax violations.



NOTE

Syntax violations discovered by the syntax validation script and task are logged in the Directory Server error log.

If **nsslapd-syntaxlogging** and **nsslapd-syntaxcheck** are both enabled, then any invalid attribute modification is rejected and the message written to the log. If **nsslapd-syntaxlogging** is enabled but **nsslapd-syntaxcheck** is disabled, then the operation is allowed to succeed, but the warning message is still written to the error log.

The value of this attribute is either **on** or **off** (by default, this is off). To enable syntax validation logging, edit the attribute using **ldapmodify** or by editing the **dse.ldif** file directly.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=config
changetype: modify
replace: nsslapd-syntaxlogging
nsslapd-syntaxlogging: on
```

8.9.6. Validating the Syntax of Existing Attribute Values

Syntax validation checks every modification to attributes to make sure that the new value has the required syntax for that attribute type. However, syntax validation only audits *changes* to attribute values, such as when an attribute is added or modified. It does not validate the syntax of *existing* attribute values.

Validation of existing attribute values can be done with the syntax validation script. This script checks entries under a specified subtree (in the **-b** option) and, optionally, only entries which match a specified filter (in the **-f** option). For example:

```
/usr/lib64/dirsrv/instance_name/syntax-validate.pl -D "cn=directory
manager" -w secret -b "ou=people,dc=example,dc=com" -f "
(objectclass=inetorgperson)"
```

The script identifies syntax violations, however, you must fix them manually.



NOTE

If syntax validation is disabled or if a server is migrated, then there may be data in the server which does not conform to attribute syntax requirements. The syntax validation script can be run to evaluate those existing attribute values before enabling syntax validation.

Alternately, a task can be launched to initiate syntax validation, specifying the required base DN and, optionally, LDAP search filter.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x

dn: cn=example,cn=syntax validation,cn=tasks,cn=config
changetype: add
objectclass: extensibleObject
cn:example
basedn: ou=people,dc=example,dc=com
filter: "(objectclass=inetorgperson)"
```

CHAPTER 9. MANAGING INDEXES

Indexing makes searching for and retrieving information easier by classifying and organizing attributes or values. This chapter describes the searching algorithm itself, placing indexing mechanisms in context, and then describes how to create, delete, and manage indexes.

9.1. ABOUT INDEXES

This section provides an overview of indexing in Directory Server. It contains the following topics:

- [Section 9.1.1, “About Index Types”](#)
- [Section 9.1.2, “About Default, System, and Standard Indexes”](#)
- [Section 9.1.3, “Overview of the Searching Algorithm”](#)
- [Section 9.1.5, “Indexing Performance”](#)
- [Section 9.1.6, “Balancing the Benefits of Indexing”](#)

9.1.1. About Index Types

Indexes are stored in files in the directory's databases. The names of the files are based on the indexed attribute, not the type of index contained in the file. Each index file may contain multiple types of indexes if multiple indexes are maintained for the specific attribute. For example, all indexes maintained for the common name attribute are contained in the **cn.db4** file.

Directory Server supports the following types of index:

- *Presence index (pres)* contains a list of the entries that contain a particular attribute, which is very useful for searches. For example, it makes it easy to examine any entries that contain access control information. Generating an **aci.db4** file that includes a presence index efficiently performs the search for **ACI=*** to generate the access control list for the server.

The presence index is not used for base object searches.

- *Equality index (eq)* improves searches for entries containing a specific attribute value. For example, an equality index on the **cn** attribute allows a user to perform the search for **cn=Babs Jensen** far more efficiently.
- *Approximate index (approx)* is used for efficient approximate or *sounds-like* searches. For example, an entry may include the attribute value **cn=Robert E Lee**. An approximate search would return this value for searches against **cn~=Robert Lee**, **cn~=Robert**, or **cn~=Lee**. Similarly, a search against **l~=San Francisco** (note the misspelling) would return entries including **l=San Francisco**.
- *Substring index (sub)* is a costly index to maintain, but it allows efficient searching against substrings within entries. Substring indexes are limited to a minimum of three characters for each entry.

For example, searches of the form **cn=*derson** , match the common names containing strings such as **Bill Anderson**, **Jill Henderson**, or **Steve Sanderson**. Similarly, the search for **telephoneNumber= *555*** returns all the entries in the directory with telephone numbers that contain **555**.

- *International index* speeds up searches for information in international directories. The process for creating an international index is similar to the process for creating regular indexes, except that it applies a *matching rule* by associating an *object identifier* (OID) with the attributes to be indexed.

The supported locales and their associated OIDs are listed in [Appendix D, Internationalization](#). If there is a need to configure the Directory Server to accept additional matching rules, contact Red Hat Professional Services.

- *Browsing index*, or *virtual list view (VLV) index*, speeds up the display of entries in the Directory Server Console. This index is particularly useful if a branch of your directory contains hundreds of entries; for example, the **ou=people** branch. You can create a browsing index on any branch point in the directory tree to improve display performance through the Directory Server Console or by using the **vlvindex** command-line tool, which is explained in the *Directory Server Configuration and Command-Line Tool Reference*.

9.1.2. About Default, System, and Standard Indexes




When you install Directory Server, a set of default and system indexes is created per database instance. To maintain these indexes, the directory uses standard indexes.

9.1.2.1. Overview of Default Indexes


The default indexes can be modified depending on the directory indexing needs. Always ensure that no server plug-ins or other servers depend on a default index before removing it.

[Table 9.1, “Default Indexes”](#) lists the default indexes installed with the directory.

Table 9.1. Default Indexes

Attribute	Eq	Pres	Sub	Purpose
cn				Improves the performance of the most common types of user directory searches.
givenname				Improves the performance of the most common types of user directory searches.
mail				Improves the performance of the most common types of user directory searches.
mailHost				Used by a messaging server.







Attribute	Eq	Pres	Sub	Purpose
member	●			Improves Directory Server performance. This index is also used by the Referential Integrity Plug-in. See Section 3.6, “Maintaining Referential Integrity” for more information.
owner	●			Improves Directory Server performance. This index is also used by the Referential Integrity Plug-in. See Section 3.6, “Maintaining Referential Integrity” for more information.
see Also	●			Improves Directory Server performance. This index is also used by the Referential Integrity Plug-in. See Section 3.6, “Maintaining Referential Integrity” for more information.
sn	●	●	●	Improves the performance of the most common types of user directory searches.
telephoneNumber	●	●	●	Improves the performance of the most common types of user directory searches.
uid	●			Improves Directory Server performance.

Attribute	Eq	Pres	Sub	Purpose
unique member				Improves Directory Server performance. This index is also used by the Referential Integrity Plug-in. See Section 3.6, “Maintaining Referential Integrity” for more information.

9.1.2.2. Overview of System Indexes

System indexes cannot be deleted or modified. They are required by the directory to function properly. [Table 9.2, “System Indexes”](#) lists the system indexes included with the directory.

Table 9.2. System Indexes

Attribute	Eq	Pres	Purpose
aci			Allows the Directory Server to quickly obtain the access control information maintained in the database.
objectClass			Used to help accelerate subtree searches in the directory.
entryDN			Speeds up entry retrieval based on DN searches.
parentID			Enhances directory performance during one-level searches.
numSubordinates			Used by the Directory Server Console to enhance display performance on the Directory tab.
nsUniqueID			Used to search for specific entries.

9.1.2.3. Overview of Standard Indexes

Because of the need to maintain default indexes and other internal indexing mechanisms, the Directory Server also maintains certain standard index files. The standard index, **id2entry.db4**, exists by default in Directory Server; you do not need to generate it.

The **id2entry.db4** contains the actual directory database entries. All other database files can be recreated from this one.

9.1.3. Overview of the Searching Algorithm

Indexes are used to speed up searches. To understand how the directory uses indexes, it helps to understand the searching algorithm. Each index contains a list of attributes (such as the **cn**, common name, attribute) and a pointer to the entries corresponding to each value. Directory Server processes a search request as follows:

1. An LDAP client application sends a search request to the directory.
2. The directory examines the incoming request to make sure that the specified base DN matches a suffix contained by one or more of its databases or database links.
 - If they do match, the directory processes the request.
 - If they do not match, the directory returns an error to the client indicating that the suffix does not match. If a referral has been specified in the **nsslapd-referral** attribute under **cn=config**, the directory also returns the LDAP URL where the client can attempt to pursue the request.
 - The Directory Server examines the search filter to see what indexes apply, and it attempts to load the list of entry IDs from each index that satisfies the filter. The ID lists are combined based on whether the filter used AND or OR joins.
 - If the list of entry IDs is larger than the configured ID list scan limit or if there is no index, then the Directory Server searches every entry in the database. This is an *unindexed* search.
3. The Directory Server reads every entry from the **id2entry.db4** database or the entry cache for every entry ID in the ID list (or from the entire database for an unindexed search). The server then checks the entries to see if they match the search filter. Each match is returned as it is found.

The server continues through the list of IDs until it has searched all candidate entries or until it hits one of the configured resource limits. (Resource limits are listed in [Table 10.1, “Resource Limit Attributes”](#).)



NOTE

It's possible to set separate resource limits for searches using the simple paged results control. For example, administrators can set high or unlimited size and look-through limits with paged searches, but use the lower default limits for non-paged searches.

9.1.4. Approximate Searches

In addition, the directory uses a variation of the metaphone phonetic algorithm to perform searches on an approximate index. Each value is treated as a sequence of words, and a phonetic code is generated for each word.

**NOTE**

The metaphone phonetic algorithm in Directory Server supports only US-ASCII letters. Therefore, use approximate indexing only with English values.

Values entered on an approximate search are similarly translated into a sequence of phonetic codes. An entry is considered to match a query if both of the following are true:

- All of the query string codes match the codes generated in the entry string.
- All of the query string codes are in the same order as the entry string codes.

Name in the Directory (Phonetic Code)	Query String (Phonetic code)	Match Comments
Alice B Sarette (ALS B SRT)	Alice Sarette (ALS SRT)	Matches. Codes are specified in the correct order.
	Alice Sarrette (ALS SRT)	Matches. Codes are specified in the correct order, despite the misspelling of Sarette.
	Surette (SRT)	Matches. The generated code exists in the original name, despite the misspelling of Sarette.
	Bertha Sarette (BR0 SRT)	No match. The code BR0 does not exist in the original name.
	Sarette, Alice (SRT ALS)	No match. The codes are not specified in the correct order.

9.1.5. Indexing Performance

Each index that the directory uses is composed of a table of index keys and matching entry ID lists. This entry ID list is used by the directory to build a list of candidate entries that may match a client application's search request; [Section 9.1, “About Indexes”](#) describes each kind of Directory Server index. The Directory Server secondary index structure greatly improves write and search operations.

While achieving extremely high read performance, in previous versions of Directory Server, write performance was limited by the number of bytes per second that could be written into the storage manager's transaction log file. Large log files were generated for each LDAP write operation; in fact, *log file verbosity* could easily be 100 times the corresponding number of bytes changed in the Directory Server. The majority of the contents in the log files are related to index changes (ID insert and delete operations).

The secondary index structure was separated into two levels in the old design:

- The ID list structures, which were the province of the Directory Server back end and opaque to the storage manager.
- The storage manager structures (Btrees), which were opaque to the Directory Server back end code.

Because it had no insight into the internal structure of the ID lists, the storage manager had to treat ID lists as opaque byte arrays. From the storage manager's perspective, when the content of an ID list changed, the **entire list** had changed. For a single ID that was inserted or deleted from an ID list, the corresponding number of bytes written to the transaction log was the maximum configured size for that ID list, about 8 kilobytes. Also, every database page on which the list was stored was marked as dirty, since the *entire* list had changed.

In the redesigned index, the storage manager has visibility into the fine-grain index structure, which optimizes transaction logging so that only the number of bytes actually changed need to be logged for any given index modification. The Berkeley DB provides ID list semantics, which are implemented by the storage manager. The Berkeley API was enhanced to support the insertion and deletion of individual IDs stored against a common key, with support for duplicate keys, and an optimized mechanism for the retrieval of the complete ID list for a given key.

The storage manager has direct knowledge of the application's intent when changes are made to ID lists, resulting in several improvements to ID list handling:

- For long ID lists, the number of bytes written to the transaction log for any update to the list is significantly reduced, from the maximum ID list size (8 kilobytes) to twice the size of one ID (4 bytes).
- For short ID lists, storage efficiency, and in most cases performance, is improved because only the storage manager metadata need to be stored, not the ID list metadata.
- The average number of database pages marked as dirty per ID insert or delete operation is very small because a large number of duplicate keys will fit into each database page.

9.1.6. Balancing the Benefits of Indexing

Before creating new indexes, balance the benefits of maintaining indexes against the costs.

- Approximate indexes are not efficient for attributes commonly containing numbers, such as telephone numbers.
- Substring indexes do not work for binary attributes.
- Equality indexes should be avoided if the value is big (such as attributes intended to contain photographs or passwords containing encrypted data).
- Maintaining indexes for attributes not commonly used in a search increases overhead without improving global searching performance.
- Attributes that are not indexed can still be specified in search requests, although the search performance may be degraded significantly, depending on the type of search.
- The more indexes you maintain, the more disk space you require.

Indexes can become very time-consuming. For example:

1. The Directory Server receives an add or modify operation.
2. The Directory Server examines the indexing attributes to determine whether an index is maintained for the attribute values.
3. If the created attribute values are indexed, then the Directory Server generates the new index entries.

4. Once the server completes the indexing, the actual attribute values are created according to the client request.

For example, the Directory Server adds the entry:

```
dn: cn=John Doe,ou=People,dc=example,dc=com
objectclass: top
objectClass: person
objectClass: orgperson
objectClass: inetorgperson
cn: John Doe
cn: John
sn: Doe
ou: Manufacturing
ou: people
telephoneNumber: 408 555 8834
description: Manufacturing lead for the Z238 line of widgets.
```

The Directory Server maintains the following indexes:

- Equality, approximate, and substring indexes for **cn** (common name) and **sn** (surname) attributes.
- Equality and substring indexes for the telephone number attribute.
- Substring indexes for the description attribute.

When adding that entry to the directory, the Directory Server must perform these steps:

1. Create the **cn** equality index entry for **John** and **John Doe**.
2. Create the appropriate **cn** approximate index entries for **John** and **John Doe**.
3. Create the appropriate **cn** substring index entries for **John** and **John Doe**.
4. Create the **sn** equality index entry for **Doe**.
5. Create the appropriate **sn** approximate index entry for **Doe**.
6. Create the appropriate **sn** substring index entries for **Doe**.
7. Create the telephone number equality index entry for **408 555 8834**.
8. Create the appropriate telephone number substring index entries for **408 555 8834**.
9. Create the appropriate description substring index entries for **Manufacturing lead for the Z238 line of widgets**. A large number of substring entries are generated for this string.

As this example shows, the number of actions required to create and maintain databases for a large directory can be resource-intensive.

9.2. CREATING STANDARD INDEXES

This section describes how to create presence, equality, approximate, substring, and international indexes for specific attributes using the Directory Server Console and the command line.

When a new index type is created, that index is used as a template for any additional databases as they are added. The Directory Server uses the current set of default indexes defined for the instance as the basis for additional databases.

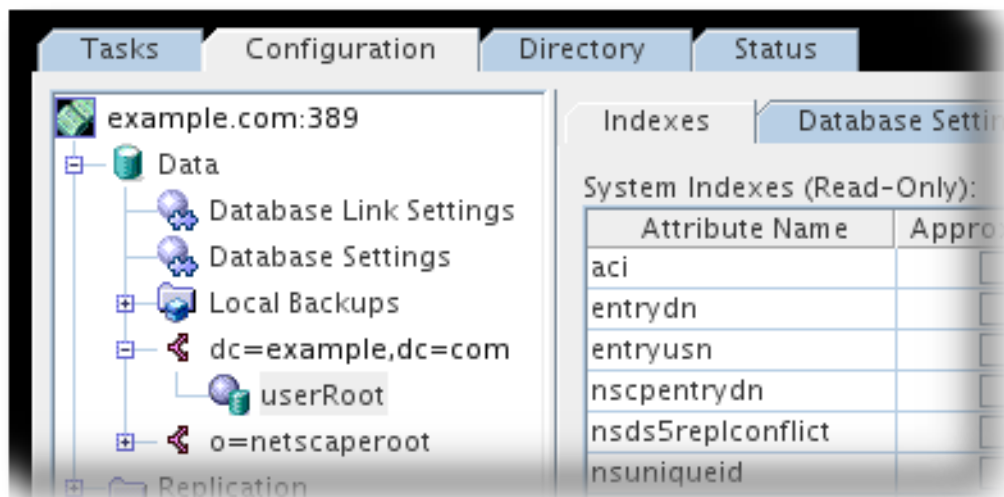
However, new indexes are not added to existing databases automatically, though they can be added manually. This means that if you add a default index to your second database instance, it will not be maintained in your first database instance but will be maintained in any subsequent instances. To apply a new index to an existing database, run the `db2index.pl` script or run a `cn=index,cn=tasks` task, as described in [Section 9.3, “Applying New Indexes to Existing Databases”](#).

- [Section 9.2.1, “Creating Indexes from the Server Console”](#)
- [Section 9.2.2, “Creating Indexes from the Command Line”](#)

9.2.1. Creating Indexes from the Server Console

To create presence, equality, approximate, substring, or international indexes:

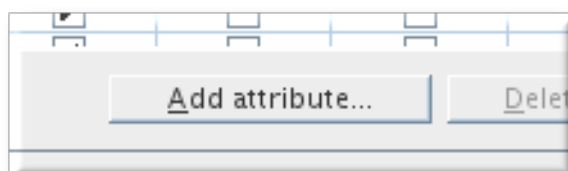
1. Select the **Configuration** tab.
2. Expand the **Data** node, expand the suffix of the database to index, and select the database.
3. Select the **Indexes** tab in the right pane.



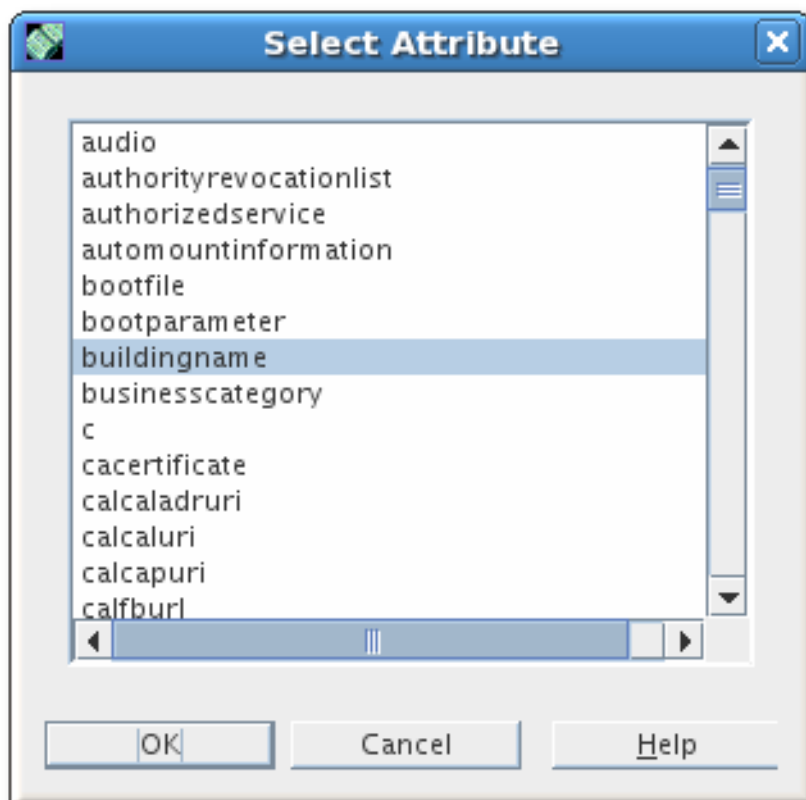
NOTE

Do not click the **Database Settings** node because this opens the **Default Index Settings** window, not the window for configuring indexes per database.

4. If the attribute to be indexed is listed in the **Additional Indexes** table, go to step 6. Otherwise, click **Add Attribute** to open a dialog box with a list of all of the available attributes in the server schema.

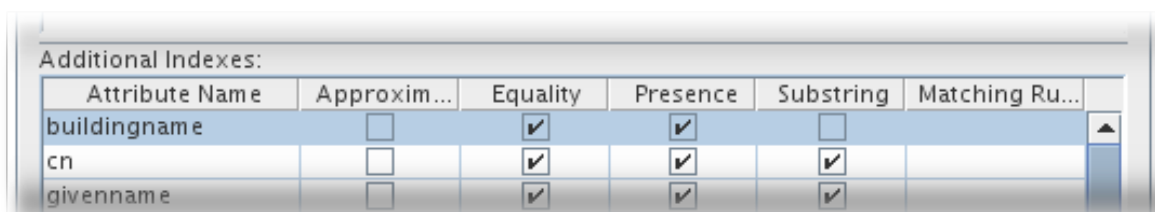


5. Select the attribute to index, and click **OK**.



The server adds the attribute to the **Additional Indexes** table.

6. Select the check box for each type of index to maintain for each attribute.



7. To create an index for a language other than English, enter the OID of the *collation order* to use in the **Matching Rules** field.

To index the attribute using multiple languages, list multiple OIDs separated by commas, but no whitespace. For a list of languages, their associated OIDs, and further information regarding collation orders, see [Appendix D, Internationalization](#).

8. Click **Save**.

The new index is immediately active for any new data that you add and any existing data in your directory. You do not have to restart your server.

9.2.2. Creating Indexes from the Command Line

**NOTE**

You cannot create new system indexes because system indexes are hard-coded in Directory Server.

Use **ldapmodify** to add the new index attributes to your directory.

- To create a new index that will become one of the default indexes, add the new index attributes to the **cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config** entry.
- To create a new index for a particular database, add it to the **cn=index,cn=database_name,cn=ldbm database,cn=plugins,cn=config** entry, where **cn=database_name** corresponds to the name of the database.

**NOTE**

Avoid creating entries under **cn=config** in the **dse.ldif** file. The **cn=config** entry in the simple, flat **dse.ldif** configuration file is not stored in the same highly scalable database as regular entries. As a result, if many entries, particularly entries that are likely to be updated frequently, are stored under **cn=config**, performance will probably suffer. Although we recommend you do not store simple user entries under **cn=config** for performance reasons, it can be useful to store special user entries such as the Directory Manager entry or replication manager (supplier bind DN) entry under **cn=config** since this centralizes configuration information.

For information on the LDIF update statements required to add entries, see [Section 3.3, “Using LDIF Update Statements to Create or Modify Entries”](#).

For example, to create presence, equality, and substring indexes for the **sn** (surname) attribute in the **Example1** database:

1. Run **ldapmodify** and add the LDIF entry for the new indexes:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=sn,cn=index,cn=Example1,cn=ldbm database,cn=plugins,cn=config
changetype: add
objectClass:top
objectClass:nsIndex
cn:sn
nsSystemIndex:false
nsIndexType:pres
nsIndexType:eq
nsIndexType:sub
nsMatchingRule:2.16.840.1.113730.3.3.2.3.1
```

The **cn** attribute contains the name of the attribute to index, in this example the **sn** attribute. The entry is a member of the **nsIndex** object class. The **nsSystemIndex** attribute is **false**, indicating that the index is not essential to Directory Server operations. The multi-valued **nsIndexType** attribute specifies the presence (**pres**), equality (**eq**) and substring (**sub**)

indexes. Each keyword has to be entered on a separate line. The **nsMatchingRule** attribute in the example specifies the OID of the Bulgarian collation order; the matching rule can indicate any possible value match, such as languages or other formats like date or integer.

You can use the keyword **none** in the **nsIndexType** attribute to specify that no indexes are to be maintained for the attribute. This example temporarily disables the **sn** indexes on the **Example1** database by changing the **nsIndexType** to **none**:

```
dn: cn=sn,cn=index,cn=Example1,cn=ldbm database,cn=plugins,cn=config
objectClass:top
objectClass:nsIndex
cn:sn
nsSystemIndex:false
nsIndexType:none
```

For a complete list of matching rules and their OIDs, see [Section 10.4.4, “Using Matching Rules”](#), and for the index configuration attributes, see the *Directory Server Configuration and Command-Line Tool Reference*.



NOTE

Always use the attribute's primary name (not the attribute's alias) when creating indexes. The primary name of the attribute is the first name listed for the attribute in the schema; for example, **uid** for the user ID attribute.

9.3. APPLYING NEW INDEXES TO EXISTING DATABASES

New indexes are not added to existing databases automatically. They must be added manually, and Directory Server has two methods for applying new indexes to an existing database: running the **db2index.pl** script or running a **cn=index,cn=tasks** task.

9.3.1. Running the db2index.pl Script

After creating an indexing entry or adding additional index types to an existing indexing entry, run the **db2index.pl** script to generate the new set of indexes to be maintained by the Directory Server. After the script is run, the new set of indexes is active for any new data added to the directory and any existing data in the directory.

Run the **db2index.pl** Perl script.

```
[root@server ~]# /usr/lib[64]/dirsrv/slapd-example/db2index.pl -D
"cn=Directory Manager" -w secret -n ExampleServer -t sn
```

For more information about using this Perl script, see the *Directory Server Configuration and Command-Line Tool Reference*.

[Table 9.3, “db2index.pl Options”](#) describes the **db2index.pl** options.

Table 9.3. db2index.pl Options

Option	Description
-D	Specifies the DN of the administrative user.
-w	Specifies the password of the administrative user.
-n	Specifies the name of the database being indexed.
-t	Specifies the name of the attribute contained by the index.

9.3.2. Using a `cn=tasks` Entry to Create an Index

The `cn=tasks,cn=config` entry in the Directory Server configuration is a container entry for temporary entries that the server uses to manage tasks. Several common directory tasks have container entries under `cn=tasks,cn=config`. Temporary task entries can be created under `cn=index,cn=tasks,cn=config` to initiate an indexing operation.

This task entry requires a unique name (`cn`) and a definition for the attribute and type of index, set in `nsIndexAttribute` in the format `attribute:index_type`.

For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=example presence index,cn=index,cn=tasks,cn=config
changetype: add
objectclass: top
objectclass: extensibleObject
cn: example presence index
nsInstance: userRoot
nsIndexAttribute: "cn:pres"
```

There are three possible `index_types`:

- **pres** for presence indexes
- **eq** for equality indexes
- **sub** for substring indexes

As soon as the task is completed, the entry is removed from the directory configuration.

The *Directory Server Configuration and Command-Line Tool Reference* has more information on running Directory Server tasks under the `cn=tasks` entries.

9.4. CREATING BROWSING (VLV) INDEXES

A *virtual list view (VLV) index* is a way of creating a truncated list for faster searching while enhancing server performance. The VLV index itself can be resource-intensive to maintain, but it can be beneficial in large directories (over 1000 entries).

A browsing index is a type of VLV index that organizes the entries listed into alphabetical order, making it easier to find entries.

VLV indexes are not applied to attributes, like standard indexes are, but they are dynamically generated based on attributes set in entries and the location of those entries in the directory tree. VLV indexes, unlike standard indexes, are special entries in the database rather than configuration settings for the database.

NOTE

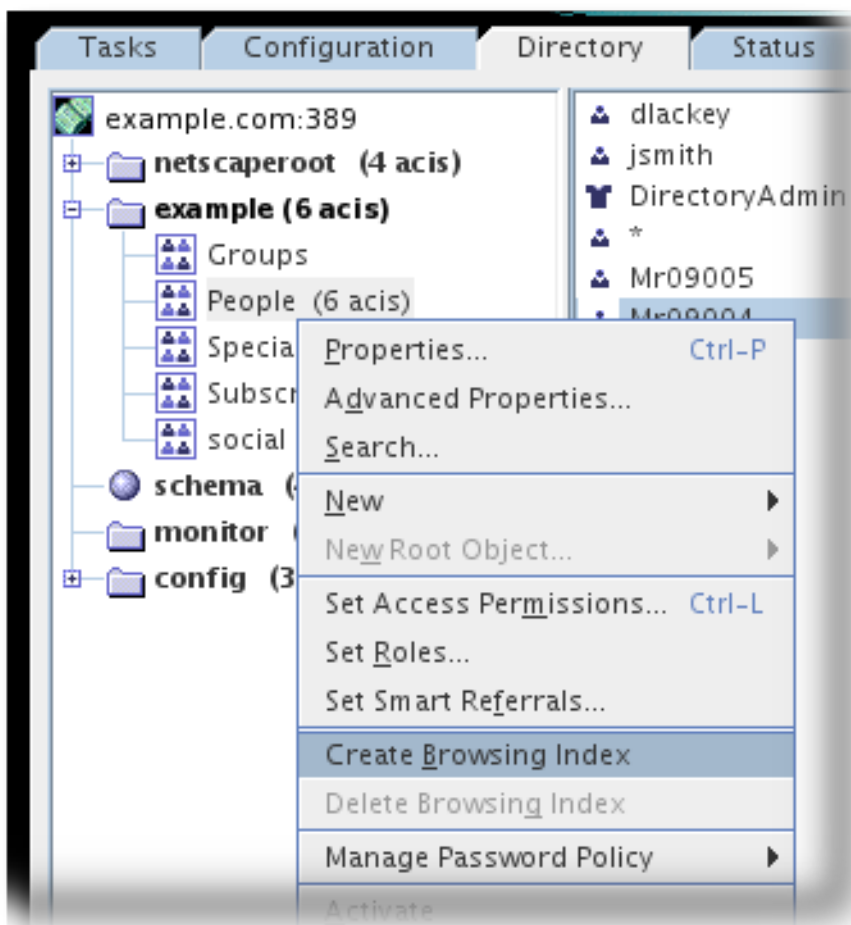
VLV indexes are similar to simple paged results, which can be returned with some external LDAP clients. Simple paged results are calculated per search, while VLV indexes are a permanent list, so VLV indexes are overall faster for searches, but do require some overhead for the server to maintain.

Simple paged results and VLV indexes *cannot* be used on the same search.

For more information, see [Section 10.7.4, “Using Simple Paged Results”](#).

9.4.1. Creating Browsing Indexes from the Server Console

1. Select the **Directory** tab.
2. In the left navigation tree, select the entry, such as **People**, for which to create the index.
3. From the **Object** menu, select **Create Browsing Index**.



The **Create Browsing Index** dialog box appears displaying the status of the index creation. Click the **Status Logs** box to view the status of the indexes created.



4. Click **Close**.

The new index is immediately active for any new data that is added to the directory. You do not have to restart your server.

For more information on how to change the VLV search information or the access control rules that are set by default for VLV searches, see [Section 9.4.2.1, “Adding a Browsing Index Entry”](#) and [Section 9.4.3, “Setting Access Control for VLV Information”](#).

9.4.2. Creating Browsing Indexes from the Command Line

Creating a browsing index or virtual list view (VLV) index from the command line has these steps:

1. Using **ldapmodify** to add new browsing index entries or edit existing browsing index entries. See [Section 9.4.2.1, “Adding a Browsing Index Entry”](#).
2. Running the **vlvindex** script to generate the new set of browsing indexes to be maintained by the server. See [Section 9.4.2.2, “Running the vlvindex Script”](#). Alternatively, launch an appropriate task under **cn=tasks, cn=config** ([Section 9.4.2.3, “Using a cn=tasks Entry to Create a Browsing Index”](#)).
3. Ensuring that access control on VLV index information is set appropriately. See [Section 9.4.3, “Setting Access Control for VLV Information”](#).

9.4.2.1. Adding a Browsing Index Entry

The type of browsing index entry to create depends on the type of **ldapsearch** attribute sorting to accelerate. It is important to take the following into account:

- The scope of the search (base, one, sub)
- The base of the search (the entry to use as a starting point for the search)
- The attributes to sort
- The filter of the search

For more information on specifying filters for searches, see [Chapter 10, Finding Directory Entries](#).

- The LDBM database to which the entry that forms the base of the search belongs. You can only create browsing indexes in LDBM databases.

For example, create a browsing index to accelerate an **ldapsearch** on the entry **ou=People,dc=example,dc=com** held in the **Example1** database with the following attributes:

- The search base is **ou=People,dc=example,dc=com**
- The search filter is **(|(objectclass=*)(objectclass=ldapsubentry))**
- The scope is **one**
- The sorting order for the returned attributes is **cn, givenname, o, ou, and sn**

1. Run **ldapmodify** and add an entry which specifies the base, scope, and filter of the browsing index:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=MCC ou=People dc=example dc=com,cn=userRoot,cn=ldbm
database,cn=plugins,cn=config
changetype: add
objectClass: top
objectClass: vlvSearch
cn: MCC ou=People dc=example dc=com
vlvBase: ou=People,dc=example,dc=com
vlvScope: 1
vlvFilter: (|(objectclass=*)(objectclass=ldapsubentry))
```

- The **cn** contains the browsing index identifier, which specifies the entry on which to create the browsing index; in this example, the **ou=People,dc=example,dc=com** entry. Red Hat recommends using the **dn** of the entry for the browsing index identifier, which is the approach adopted by the Directory Server Console, to prevent identical browsing indexes from being created. The entry is a member of the **vlvSearch** object class.
 - The **vlvbase** attribute value specifies the entry on which you want to create the browsing index; in this example, the **ou=People,dc=example,dc=com** entry (the browsing index identifier).
 - The **vlvScope** attribute is **1**, indicating that the scope for the search you want to accelerate is **1**. A search scope of **1** means that only the immediate children of the entry specified in the **cn** attribute, and not the entry itself, will be searched.
 - The **vlvFilter** specifies the filter to be used for the search; in this example, **(|(objectclass=*)(objectclass=ldapsubentry))**.
2. Add the second entry, to specify the sorting order for the returned attributes:

```
dn: cn=by MCC ou=People dc=example dc=com,cn=MCC ou=People
dc=example dc=com,cn=userRoot,cn=ldbm database,cn=plugins,
cn= config
objectClass: top
```

```
objectClass: vlvIndex
cn: by MCC ou=People dc=example dc=com
vlvSort: cn givenName o ou sn
```

- o The **cn** contains the browsing index sort identifier. The above **cn** is the type created by the Console by default, which has the sorting order as being set *by* the browsing index base. The entry is a member of the **vlvIndex** object class.
- o The **vlvSort** attribute value specifies the order in which you want your attributes to be sorted; in this example, **cn**, **givenName**, **o**, **ou**, and then **sn**.



NOTE

This first browsing index entry must be added to the **cn=database_name, cn=ldbm database, cn=plugins, cn=config** directory tree node, and the second entry must be a child of the first entry.

9.4.2.2. Running the vlvindex Script

After creating the two browsing indexing entries or added additional attribute types to an existing indexing browsing entries, run the **vlvindex** script to generate the new set of browsing indexes to be maintained by the Directory Server. After running the script, the new set of browsing indexes is active for any new data added to the directory and any existing data in the directory.

To run the **vlvindex** script:

1. Stop the server.

```
[root@server ~]# service dirsrv stop instance
```

2. Run the **vlvindex** script.

```
[root@server ~]# /etc/dirsrv/slapd-instance_name/vlvindex -n
Example1 -T "by MCC ou=people dc=example dc=com"
```

For more information about using this script, see the *Directory Server Configuration and Command-Line Tool Reference*.

3. Restart the server.

```
[root@server ~]# service dirsrv start instance
```

[Table 9.4, “vlvindex Options”](#) describes the **vlvindex** options used in the examples:

Table 9.4. vlvindex Options

Option	Description
-n	Name of the database containing the entries to index.
-T	Browsing index identifier to use to create browsing indexes.

9.4.2.3. Using a `cn=tasks` Entry to Create a Browsing Index

As an alternative to running the `vlvindex` script, it is possible to initiate an indexing task directly.



NOTE

Running the indexing task is the same as running the `vlvindex` script.

The `cn=tasks, cn=config` entry in the Directory Server configuration is a container entry for temporary entries that the server uses to manage tasks. Several common directory tasks have container entries under `cn=tasks, cn=config`. Temporary task entries can be created under `cn=index, cn=tasks, cn=config` to initiate an indexing operation.

This task entry requires a unique name (`cn`) and one other attribute, `nsIndexVLVAttribute`, which gives the name of the browsing index definition entry to use to generate the VLV index.

For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=example VLV index,cn=index,cn=tasks,cn=config
changetype: add
objectclass: extensibleObject
cn: example VLV index
nsIndexVLVAttribute: "by MCC ou=people,dc=example,dc=com"
```

As soon as the task is completed, the entry is removed from the directory configuration.

The *Directory Server Configuration and Command-Line Tool Reference* has more information on running Directory Server tasks under the `cn=tasks` entries.

9.4.3. Setting Access Control for VLV Information

The default access control instruction (ACI) allows only authenticated users to use the VLV index information. If you additionally require to allow non-authenticated users to use the VLV index information, update the `aci` attribute to set the `userdn` parameter to `ldap://anyone`:

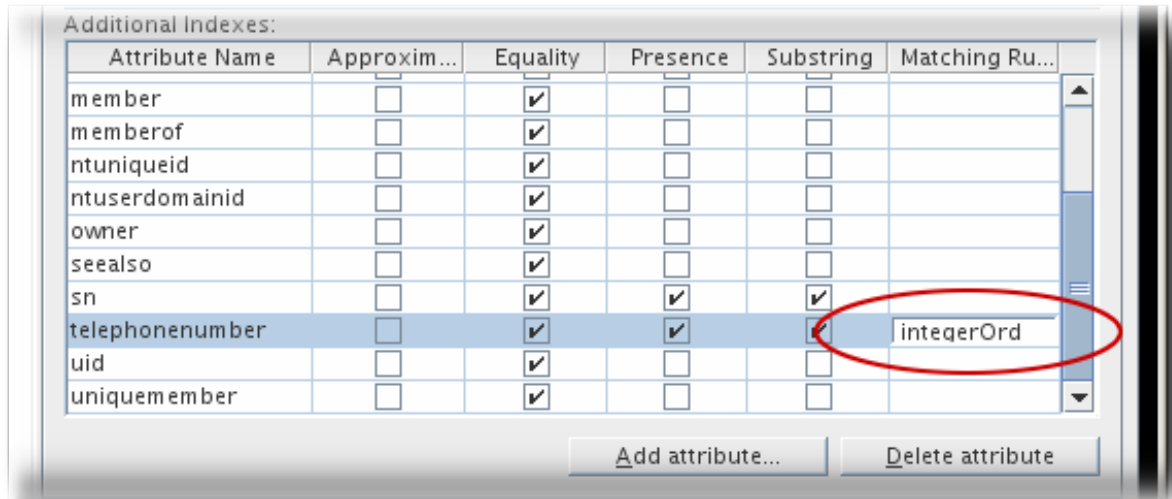
```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: oid=2.16.840.1.113730.3.4.9,cn=features,cn=config
changetype: modify
replace: aci
aci: aci: (targetattr != "aci")(version 3.0; acl "VLV Request Control";
    allow( read, search, compare, proxy ) userdn = "ldap://anyone" ;)
```

9.5. CHANGING THE INDEX SORT ORDER

By default, indexes are sorted alphabetically, in descending ASCII order. This is true for every attribute, even attributes which may have numeric attribute values like `Integer` or `TelephoneNumber`. It is possible to change the sort method by changing the matching rule set for the attribute.

9.5.1. Changing the Sort Order in the Console

1. Select the **Configuration** tab.
2. Expand the **Data** node, expand the suffix of the database to index, and select the database.
3. Select the **Indexes** tab in the right pane.
4. Select the index, and, in the **Matching Rules** field, enter the new sort order to use. For example, to sort by numbers, rather than alphabetically, enter **integerOrderingMatch**.



5. Click **Save**.

9.5.2. Changing the Sort Order in the Command Line

To change the sort order using the command line, change the **nsMatchingRule** for the attribute index. For example:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=sn,cn=index,cn=Example1,cn=ldb database,cn=plugins,cn=config
changetype: modify
replace: nsMatchingRule
nsMatchingRule: integerOrderingMatch
```

9.6. CHANGING THE WIDTH FOR INDEXED SUBSTRING SEARCHES

By default, for a search to be indexed, the search string must be at least three characters long, without counting any wildcard characters. For example, the string **abc** would be an indexed search while **ab*** would not be. Indexed searches are significantly faster than unindexed searches, so changing the minimum length of the search key is helpful to increase the number of indexed searches.

To improve search performance, particularly for sites with many wildcard searches, the search string length for indexed searches can be changed. Directory Server has three attributes which allow you to change the minimum number of characters required for an indexed search:

- The **nsSubStrBegin** attribute sets the required number of characters for an indexed search for the beginning of a search string, before the wildcard.

```
abc*
```

- The ***nsSubStrMiddle*** attribute sets the required number of characters for an indexed search where a wildcard is used in the middle of a search string. For example:

```
ab*z
```

- The ***nsSubStrEnd*** attribute sets the required number of characters for an indexed search for the end of a search string, after the wildcard. For example:

```
*xyz
```

The default substring search length for the string triplet (before, middle, and end) is 3, 3, and 3, meaning every search requires a minimum of three characters, in every wildcard position.

For any attribute index to have alternate string lengths, add the ***extensibleObject*** object class to the entry and then set the substring search lengths.

1. Set the new key length for the specific attribute index. This requires adding the ***extensibleObject*** object class and then adding the ***nsSubStrBegin***, ***nsSubStrEnd***, or ***nsSubStrMiddle*** attributes as appropriate. For example:

```
# ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: attribute_name,cn=index,cn=database_name,cn=ldbm
database,cn=plugins,cn=config
changetype: modify
add: objectclass
objectclass: extensibleObject
-
add: nsSubStrBegin
nsSubStrBegin: 2
-
add: nsSubStrMiddle
nsSubStrMiddle: 2
-
add: nsSubStrEnd
nsSubStrEnd: 2
```

2. Stop the server.

```
service dirsrv stop
```

3. Recreate the attribute index. If even one of the substring search width options is changed, then the entire index must be recreated.

```
db2index -t attribute_name
```

4. Start the server again.

```
service dirsrv start
```

9.7. DELETING INDEXES

This section describes how to remove attributes and index types from the index.

9.7.1. Deleting an Attribute from the Default Index Entry

When using the default settings of Directory Server, several attributes listed in the default index entry, such as **sn**, are indexed. The following attributes are part of the default index:

Table 9.5. Default Index Attributes

aci	cn	entryusn
givenName	mail	mailAlternateAddress
mailHost	member	memberOf
nsUniqueld	ntUniqueld	ntUserDomainId
numsubordinates	objectclass	owner
parentid	seeAlso	sn
telephoneNumber	uid	uniquemember



WARNING

Removing system indexes can significantly affect the Directory Server performance.

For example, to remove the **sn** attribute from the default index:

1. Remove the attribute from the **cn=default indexes, cn=config, cn=ldbm database, cn=plugins, cn=config** entry:

```
# ldapdelete -D "cn=Directory Manager" -W -p 389 -h
server.example.com -x
cn=sn,cn=default indexes,cn=config,cn=ldbm
database,cn=plugins,cn=config
```

If you do not remove the attribute from this entry, the index for the **sn** attribute is automatically recreated and corrupted after the server is restarted.

2. Remove the **cn=attribute_name, cn=index, cn=userRoot, cn=ldbm database, cn=plugins, cn=config** entry. For details, see:
 - [Section 9.7.2, “Removing an Attribute from the Index Using the Server Console”](#)
 - [Section 9.7.3, “Removing an Attribute from the Index Using the Command Line”](#)
3. Run the **db2index.pl** Perl script to recreate the index:

```
# db2index.pl -Z instance_name -D "cn=Directory Manager" -w secret -
n database_name
```

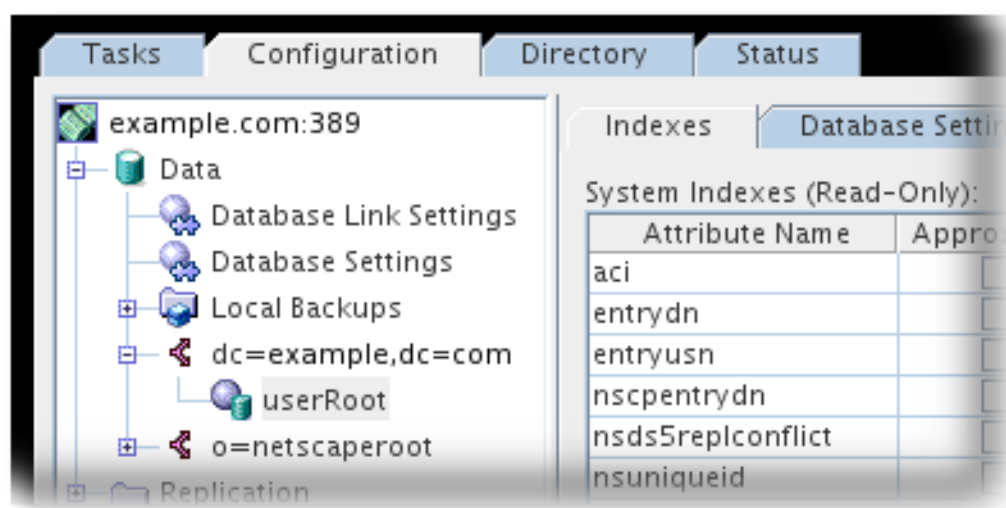
For further information about using the **db2index.pl** Perl script, see the db2index.pl(8) man page.

9.7.2. Removing an Attribute from the Index Using the Server Console

The Directory Server Console can delete any custom indexes, indexes used by other server applications such as a messaging or web server, and default indexes.

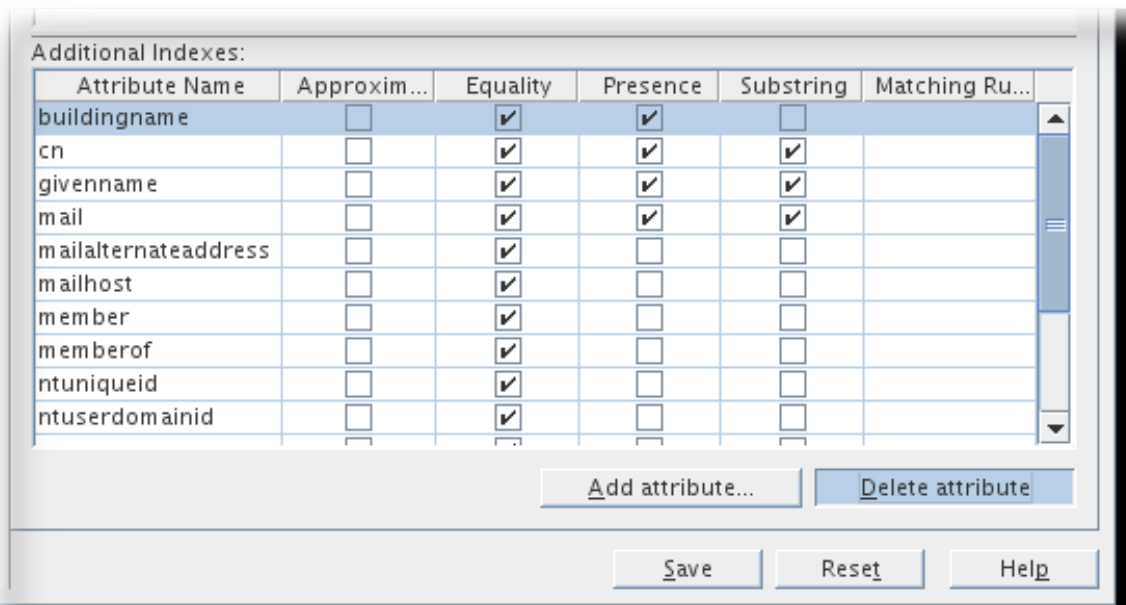
To remove an attribute from the index:

1. If the attribute to remove is listed in the ***cn=default indexes, cn=config, cn=ldbm database, cn=plugins, cn=config*** default index entry, remove it from this entry first. For details, see [Section 9.7.1, “Deleting an Attribute from the Default Index Entry”](#).
2. Select the **Configuration** tab.
3. Expand the **Data** node and expand the suffix associated with the database containing the index.
4. Select the database from which to delete the index.



5. Locate the attribute containing the index to delete. Clear the check box under the index.

To delete all indexes maintained for a particular attribute, select the attribute's cell under **Attribute Name**, and click **Delete Attribute**.



6. Click **Save**.

A **Delete Index** warning dialog box opens, requiring a confirmation to delete the index.

7. Click **Yes** to delete the index.

9.7.3. Removing an Attribute from the Index Using the Command Line

In certain situations you want to remove an attribute from the index. For example, to remove the **sn** attribute:

1. If the attribute to remove is listed in the **cn=default indexes, cn=config, cn=ldbm database, cn=plugins, cn=config** default index entry, you must remove it from this entry first. For details, see [Section 9.7.1, “Deleting an Attribute from the Default Index Entry”](#).
2. Remove the attribute from the index:

```
# ldapdelete -D "cn=Directory Manager" -W -p 389 -h
server.example.com -x
cn=sn,cn=index,cn=database_name,cn=ldbm
database,cn=plugins,cn=config
```

After deleting the entry, the index for the **sn** attribute is no longer maintained.

3. Run the **db2index.pl** Perl script to recreate the index.

```
# db2index.pl -Z instance_name -D "cn=Directory Manager" -w secret -
n database_name
```

For further information about using the **db2index.pl** Perl script, see the **db2index.pl(8)** man page.

9.7.4. Deleting Index Types from the Command Line

For example, to remove the **sub** index type of the **sn** attribute from the index:

1. Remove the index type:

```
# ldapmodify -D "cn=directory manager" -W -x
dn: cn=sn,cn=index,cn=database_name,cn=ldbm
database,cn=plugins,cn=config

changetype: modify
delete: nsIndexType
nsIndexType:sub
```

After deleting the index entry, the substring index for the **sn** attribute is no longer maintained.

2. Run the **db2index.pl** Perl script to recreate the index. For example:

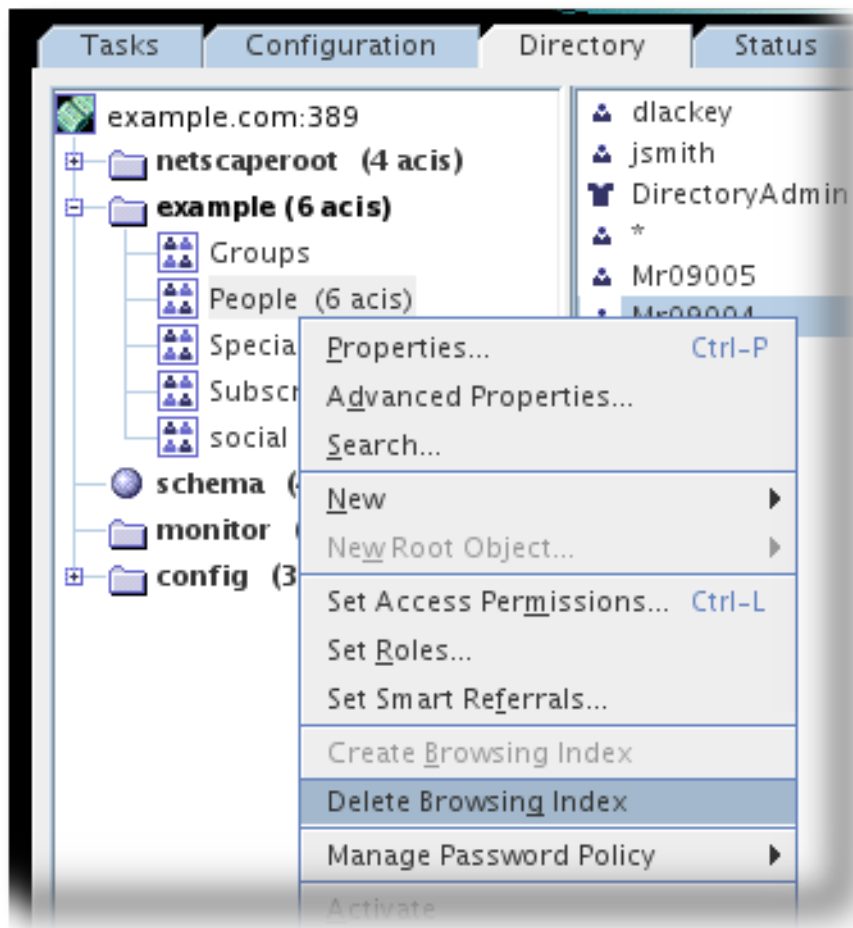
```
# db2index.pl -Z instance_name -D "cn=Directory Manager" -w secret -
n database_name
```

For further information about using the **db2index.pl** Perl script, see the `db2index.pl(8)` man page.

9.7.5. Deleting Browsing Indexes from the Server Console

1. Select the **Directory** tab.
2. Select the entry from which to delete the index in the navigation tree, and select **Delete Browsing Index** from the **Object** menu.

Alternatively, select and right-click the entry of the index to delete in the navigation tree, and then choose **Delete Browsing Index** from the pop-up menu.



3. A **Delete Browsing Index** dialog box appears asking you to confirm the deletion of the index. Click **Yes**.
4. The **Delete Browsing Index** dialog box appears displaying the status of the index deletion.

9.7.6. Deleting Browsing Indexes from the Command Line

Deleting a browsing index or virtual list view (VLV) index from the command line involves two steps:

1. Using the **ldapdelete** to delete browsing index entries or edit existing browsing index entries (Section 9.7.6.1, “Deleting a Browsing Index Entry”).
2. Running the **vlvindex** script to generate the new set of browsing indexes to be maintained by the server (Section 9.7.6.2, “Running the vlvindex Script”). Alternatively, launch an appropriate task under **cn=tasks,cn=config** (Section 9.4.2.3, “Using a cn=tasks Entry to Create a Browsing Index”).

The actual entries for an alphabetical browsing index and virtual list view are the same. The following sections describe the steps involved in deleting browsing indexes.

9.7.6.1. Deleting a Browsing Index Entry

Use the **ldapdelete** command-line utility to either delete browsing indexing entries or edit existing browsing index entries. To delete browsing indexes for a particular database, remove the browsing index entries from the **cn=index,cn=database_name,cn=ldb database,cn=plugins,cn=config** entry, where **cn=database_name** corresponds to the name of the database.

For example, there is a browsing index for accelerating **ldapsearch** operations on the entry **ou=People,dc=example,dc=com**. It held in the **Example1** database where the search base is **ou=People,dc=example,dc=com**, the search filter is **(|(objectclass=*)(objectclass=ldapsubentry))**, the scope is **1**, and the sorting order for the returned attributes is **cn, givenname, o, ou**, and **sn**.

To delete this browsing index, delete the two corresponding browsing index entries:

```
dn: cn=MCC ou=People dc=example dc=com,cn=userRoot,cn=ldbm
database,cn=plugins,cn=config
objectClass: top
objectClass: vlvSearch
cn: MCC ou=People dc=example dc=com
vlvBase: ou=People,dc=example,dc=com
vlvScope: 1 vlvFilter: (|(objectclass=*)(objectclass=ldapsubentry))

dn: cn=by MCC ou=People dc=example dc=com,cn=MCC ou=People
dc=example dc=com,cn=userRoot,cn=ldbm database,cn=plugins,cn=config
objectClass: top
objectClass: vlvIndex
cn: by MCC ou=People dc=example dc=com
vlvSort: cn givenname o ou sn
```

Run **ldapdelete**, specifying both entries.

```
ldapdelete -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x "cn=MCC ou=People dc=example
dc=com,cn=userRoot,cn=ldbm database,cn=plugins,cn=config" "cn=by MCC
ou=People dc=example dc=com,cn=MCC ou=People dc=example
dc=com,cn=userRoot,cn=ldbm database,cn=plugins,cn=config"
```

After deleting the two browsing index entries, the browsing index will no longer be maintained by the **Example1** database.

9.7.6.2. Running the vlvindex Script

After deleting browsing indexing entries or unwanted attribute types from existing browsing indexing entries, run the **vlvindex** script to generate the new set of browsing indexes to be maintained by the Directory Server. After the script is run, the new set of browsing indexes is active for any new data added to the directory and any existing data in the directory.

1. Stop the server.

```
[root@server ~]# service dirsrv stop instance
```

2. Run the **vlvindex** script.

```
[root@server ~]# /etc/dirsrv/slapd-instance_name/vlvindex -n
Example1 -T "by MCC ou=people dc=example dc=com"
```

For more information about using the **vlvindex** script, see the *Directory Server Configuration and Command-Line Tool Reference*.

3. Restart the server.


```
[root@server ~]# service dirsrv start instance
```

Table 9.4, “[vlvindex Options](#)” describes the **vlvindex** options.

Alternatively, create a new task entry under **cn=index, cn=tasks, cn=config** to initiate an indexing operation. This task entry requires a unique name (**cn**) and one other attribute, **nsIndexVLVAttribute**, which gives the name of the browsing index definition entry to use to generate the VLV index. This task is the same as running **vlvindex**.

For example:

```
ldapmodify -a -D "cn=directory manager" -w -p 389 -h server.example.com -x
dn: cn=example VLV index,cn=index,cn=tasks,cn=config
changetype: add
objectclass: extensibleObject
cn: example VLV index
nsIndexVLVAttribute: "by MCC ou=people,dc=example,dc=com"
```

As soon as the task is completed, the entry is removed from the directory configuration.

The *Directory Server Configuration and Command-Line Tool Reference* has more information on running Directory Server tasks under the **cn=tasks** entries.

CHAPTER 10. FINDING DIRECTORY ENTRIES

Entries in the directory can be searched for and found using any LDAP client. Most clients provide some form of search interface so that the directory can be searched easily and entry information can be easily retrieved.



NOTE

Users cannot search the directory unless the appropriate access control has been set in the directory. For information on setting access control in the directory, see [Chapter 13, Managing Access Control](#).

10.1. IMPROVING SEARCH PERFORMANCE THROUGH RESOURCE LIMITS

With large directories, searching through every entry in the database for a search can seriously degrade overall server performance. While this can be alleviated somewhat through effective indexing, indexing itself introduces a performance overhead, and, in large databases, may still not reduce the search scope enough to improve performance.

Reasonable limits can be set on user and client accounts to reduce the total number of entries or the total amount of time spent in an individual search, which both makes searches more responsive and improves overall server performance.

Server limits for search operations are controlled using special operational attribute values on the client application binding to the directory. You can set the following search operation limits:

- *Look through limit.* Specifies how many entries can be examined for a search operation.
- *Size limit.* Specifies the maximum number of entries the server returns to a client application in response to a search operation.
- *Time limit.* Specifies the maximum time the server spends processing a search operation.
- *Idle timeout.* Specifies the time a connection to the server can be idle before the connection is dropped.
- *Range timeout.* Specifies a separate look-through limit specifically for searches using a range.

The resource limits set for the client application take precedence over the default resource limits set for in the global server configuration.



NOTE

The Directory Manager receives unlimited resources by default, with the exception of range searches.

10.1.1. Search Performance and Resource Limits

For details, see the corresponding section in the [Red Hat Directory Server Performance Tuning Guide](#).

10.1.2. Fine Grained ID List Size

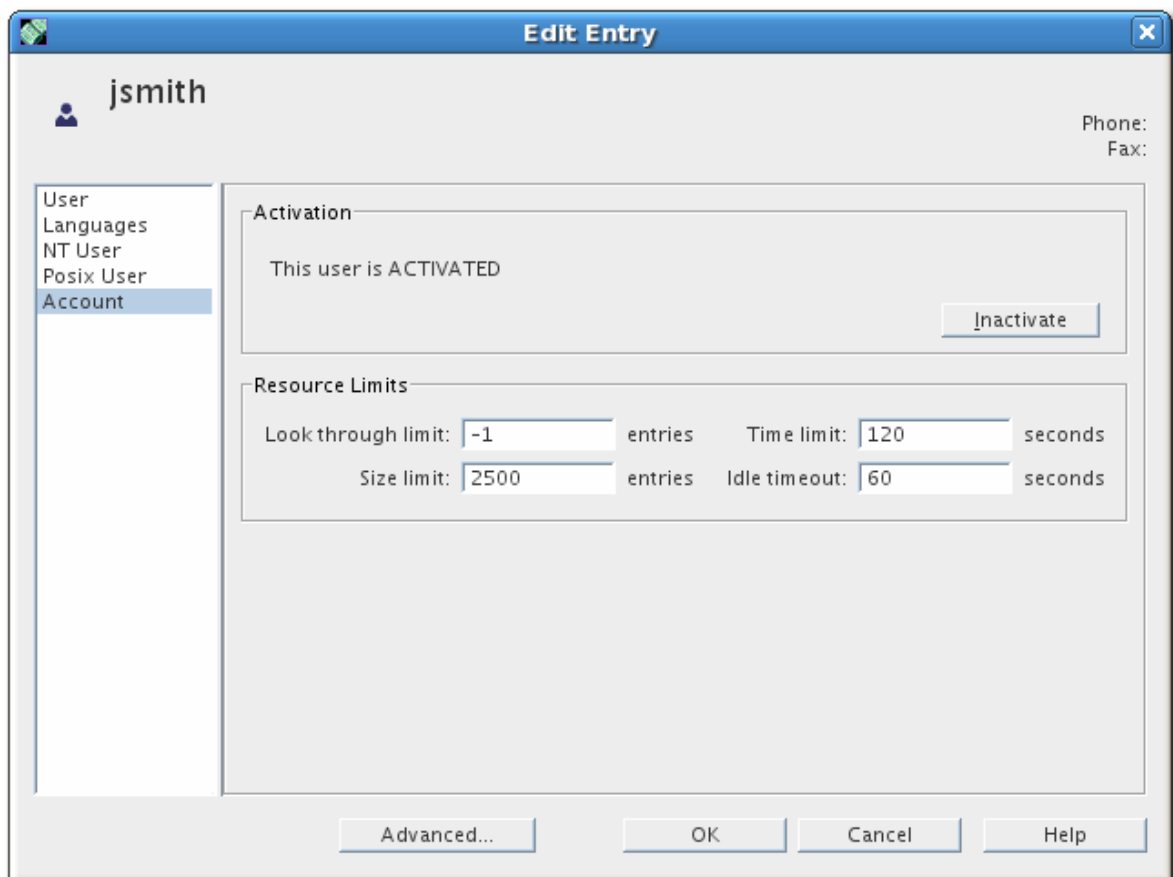
For details, see the corresponding section in the [Red Hat Directory Server Performance Tuning Guide](#).

10.1.3. Setting Resource Limits on a Single User

1. Select the **Directory** tab.
2. Browse the navigation tree in the left navigation pane, and double-click the user or role for which to set resource limits.

The **Edit Entry** dialog box appears.

3. Click **Account** in the left pane.
4. Set the resource limits. There are four different limits that can be set:
 - *Look through limit.* The maximum number of entries are examined for a search operation.
 - *Size limit.* The maximum number of entries the server returns to a client application in response to a search operation.
 - *Time limit.* The maximum time the server spends processing a search operation.
 - *Idle timeout.* The time a connection to the server can be idle before the connection is dropped.



Entering a value of **-1** indicates no limit.

5. Click **OK**.

10.1.4. Setting User and Global Resource Limits Using the Command Line

More options are available when setting resource limits in the command line than through the Directory

Server Console. The Directory Server Console sets user-level resource limits. Through the command line, administrators can set user-level resource limits, global resource limits, and limits for specific kinds of searches, such as simple paged and range searches. [Section 9.1.3, “Overview of the Searching Algorithm”](#) has more information on how these resource limits affect Directory Server search performance.

[Table 10.1, “Resource Limit Attributes”](#) lists operational attributes which can be set for each entry using the command line. Use **ldapmodify** to add the attributes to the entry.

User-level attributes are set on the individual entries, while global configuration attributes are set in the appropriate server configuration area.

Table 10.1. Resource Limit Attributes

User-Level Attribute	Global Configuration Attribute	Global Configuration Entry	Description
nsLookThroughLimit	nsslapd-lookthroughlimit	cn=config,cn=ldbm database,cn=plugins,cn=config	Specifies how many entries are examined for a search operation. Giving this attribute a value of -1 indicates that there is no limit.
nsPagedLookThroughLimit	nsslapd-pagedlookthroughlimit	cn=config,cn=ldbm database,cn=plugins,cn=config	As with the look-through limit, specifies how many entries are examined, but specifically for simple paged search operations. Giving this attribute a value of -1 indicates that there is no limit.
nsSizeLimit	nsslapd-sizelimit	cn=config	Specifies the maximum number of entries the server returns to a client application in response to a search operation. Giving this attribute a value of -1 indicates that there is no limit.
nsPagedSizeLimit	nsslapd-pagedsizelimit	cn=config	As with the size limit, specifies the maximum number of entries the server returns to a client application but only for simple paged search operations. Giving this attribute a value of -1 indicates that there is no limit.

User-Level Attribute	Global Configuration Attribute	Global Configuration Entry	Description
nsTimeLimit	nsslapd-timelimit	cn=config	Specifies the maximum time the server spends processing a search operation. Giving this attribute a value of -1 indicates that there is no time limit.
nsidletimeout	nsslapd-idletimeout	cn=config	Specifies the time a connection to the server can be idle before the connection is dropped. The value is given in seconds. Giving this attribute a value of -1 indicates that there is no limit.
nsIDListScanLimit	nsslapd-idlistscanlimit	cn=config,cn=ldbm database,cn=plugins,cn=config	Specifies the maximum number of entry IDs loaded from an index file for search results. If the ID list size is greater than this value, the search will not use the index list but will treat the search as an unindexed search and look through the entire database.
nsPagedIDListScanLimit	nsslapd-pagedidlistscanlimit	cn=config,cn=ldbm database,cn=plugins,cn=config	As with the ID list scan limit, specifies the maximum number of entry IDs loaded from an index file for search results, but specifically for paged search operations.

User-Level Attribute	Global Configuration Attribute	Global Configuration Entry	Description
	nsslapd-rangelookthroughlimit	cn=config,cn=ldbm database,cn=plugins,cn=config	Specifies how many entries are examined for a range search operation (a search using greater-than, equal-to-or-greater-than, less-than, or equal-to-less-than operators). Giving this attribute a value of -1 indicates that there is no limit.

For example, this sets the size limit for Barbara Jensen by using **ldapmodify** to modify her entry:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: uid=bjensen,ou=people,dc=example,dc=com
changetype: modify
add: nsSizeLimit
nsSizeLimit: 500
```

The **ldapmodify** statement adds the **nsSizeLimit** attribute to Babs Jensen's entry and gives it a search return size limit of 500 entries.

10.1.5. Setting Resource Limits on Anonymous Binds

Resource limits are set on a user entry. An anonymous bind, obviously, does not have a user entry associated with it. This means that the global resource limits usually apply to anonymous operations. However, it is possible to configure resource limits specifically for anonymous binds by creating a template user entry that has resource limits, and then applying that template to anonymous binds.

1. Create a template entry and set whatever resource limits you want to apply to anonymous binds.



NOTE

For performance reasons, the template should be in the normal back end, not in the **cn=config** suffix, which does not use an entry cache.

For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=anon template,ou=people,dc=example,dc=com
changetype: add
objectclass: person
objectclass: top
cn: anon template
sn: template
```

```
nsSizeLimit: 250
nsLookThroughLimit: 1000
nsTimeLimit: 60
```

2. Add the ***nsslapd-anonlimitsdn*** to the server configuration, pointing to the DN of the template entry. Any of the resource limits in [Section 10.1.4, “Setting User and Global Resource Limits Using the Command Line”](#) can be set. For example:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=config
changetype: modify
add: nsslapd-anonlimitsdn
nsslapd-anonlimitsdn: cn=anon template,ou=people,dc=example,dc=com
```

10.1.6. Improving Performance for Range Searches

Range searches use operators ([Section 10.4.2, “Using Operators in Search Filters”](#)) to set a bracket to search for and return an entire subset of entries within the directory. For example, this searches for every entry modified at or after midnight on January 1:

```
(modifyTimestamp>=20170101010101Z)
```

The nature of a range search is that it must evaluate every single entry within the directory to see if it is within the range given. Essentially, a range search is always an all IDs search. (Performance problems all-IDs searches are covered in detail in [Section 9.1.5, “Indexing Performance”](#).)

For most users, the look-through limit kicks in and prevents range searches from turning into an all IDs search. This improves overall performance and speeds up range search results. However, some clients or administrative users like Directory Manager may not have a look-through limit set. In that case, a range search can take several minutes to complete or even continue indefinitely.

It is possible to set a separate range look-through limit. This allows clients and administrative users to have high look-through limits while still allowing a reasonable limit to be set on potentially performance-impaired range searches.

This is configured in the ***nsslapd-rangelookthroughlimit*** attribute. The default value is **5000**, the same as the default ***nsslapd-lookthroughlimit*** attribute value.

For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=config,cn=ldbm database,cn=plugins,cn=config
changetype: add
add: nsslapd-rangelookthroughlimit
nsslapd-rangelookthroughlimit: 7500
```

10.2. FINDING ENTRIES USING THE DIRECTORY SERVER CONSOLE

Users can browse the **Directory** tab of the Directory Server Console to see the contents of the directory tree and search for specific entries in the directory.

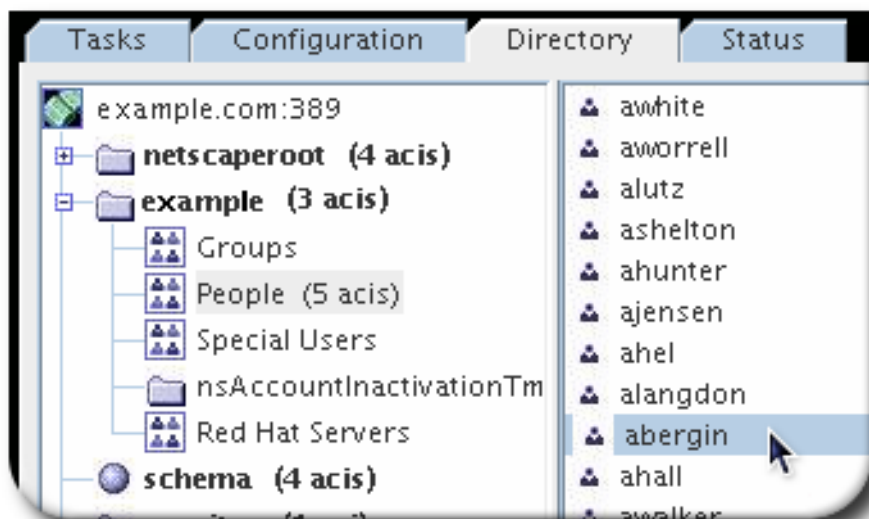


Figure 10.1. Browsing Entries in the Directory Tab

Depending on the DN used to authenticate to the directory, this tab displays the contents of the directory that the user account has access permissions to view. Browse through the contents of the tree, or right-click an entry, and select **Search** from the pop-up menu.

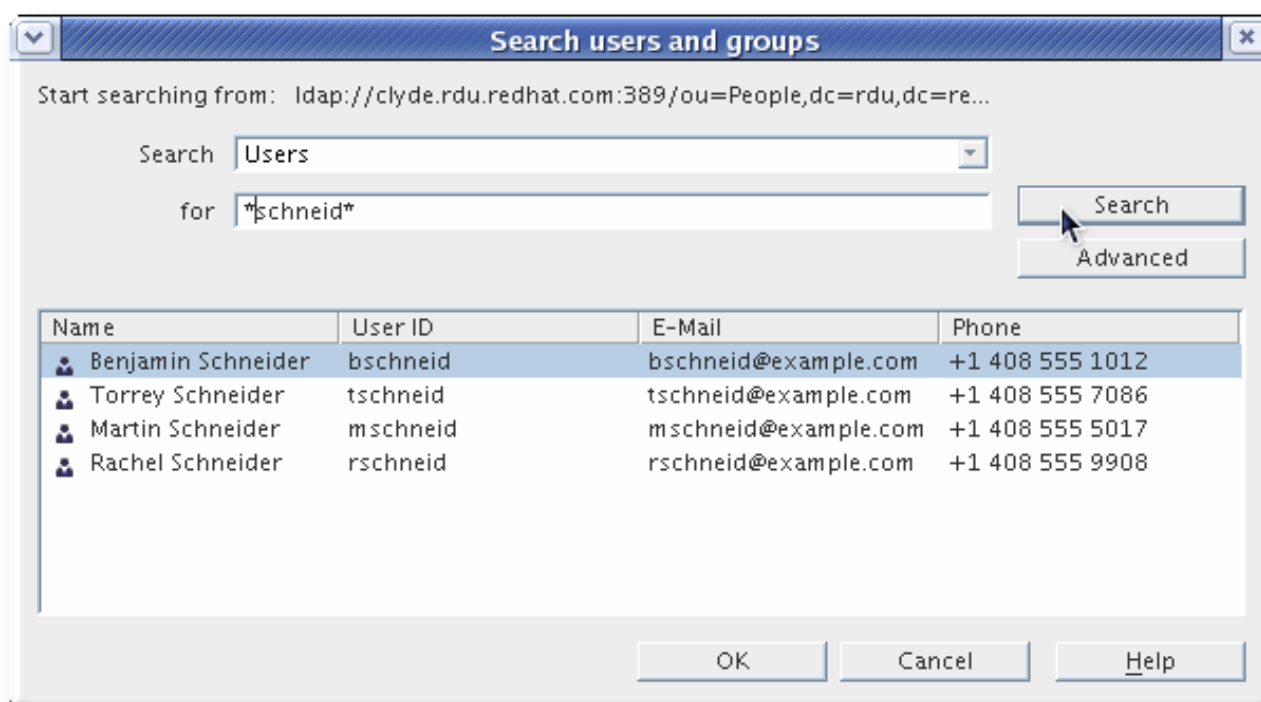


Figure 10.2. Searching for Entries



WARNING

Do not modify the contents of the **o=NetSCAPERoot** suffix using the **Directory** tab unless instructed to do so by Red Hat technical support.

10.3. USING LDAPSEARCH

The **ldapsearch** command-line utility can locate and retrieve directory entries. This utility opens a connection to the specified server using the specified identity and credentials and locates entries based on a specified search filter. The search scope can include a single entry (**-s base**), an entry's immediate subentries (**-s one**), or an entire tree or subtree (**-s sub**).



NOTE

A common mistake is to assume that the directory is searched based on the attributes used in the distinguished name. The distinguished name is only a unique identifier for the directory entry and cannot be used as a search key. Instead, search for entries based on the attribute-data pairs stored on the entry itself. Thus, if the distinguished name of an entry is **uid=bjensen,ou=People,dc=example,dc=com**, then a search for **dc=example** does not match that entry unless **dc:example** has explicitly been added as an attribute in that entry.

Search results are returned in LDIF format. LDIF is defined in [RFC 2849](#) and is described in detail in [Appendix B, LDAP Data Interchange Format](#).

This section contains information about the following topics:

- [Section 10.3.1, “ldapsearch Command-Line Format”](#)
- [Section 10.3.2, “Commonly Used ldapsearch Options”](#)
- [Section 10.3.3, “Using Special Characters”](#)

10.3.1. ldapsearch Command-Line Format

The **ldapsearch** command must use the following format:

```
ldapsearch [-x | -Y mechanism] [optional_options] [optional_search_filter]
[optional_list_of_attributes]
```

- Either **-x** (to disable SASL) or **-Y** (to set the SASL mechanism) must be used to configure the type of connection.
- *optional_options* is a series of command-line options. These must be specified before the search filter, if any are used.
- *optional_search_filter* is an LDAP search filter as described in [Section 10.4, “LDAP Search Filters”](#). Do not specify a separate search filter if search filters are specified in a file using the **-f** option.
- *optional_list_of_attributes* is a list of attributes separated by a space. Specifying a list of attributes reduces the number of attributes returned in the search results. This list of attributes must appear after the search filter. For an example, see [Section 10.5.6, “Displaying Subsets of Attributes”](#). If a list of attributes is not specified, the search returns values for all attributes permitted by the access control set in the directory (with the exception of operational attributes).



NOTE

For operational attributes to be returned as a result of a search operation, they must be explicitly specified in the search command. To retrieve regular attributes in addition to explicitly specified operational attributes, use an asterisk (*) in the list of attributes in the **ldapsearch** command. To retrieve no attributes, just a list of the matching DNs, use the special attribute **1.1**. This is useful, for example, to get a list of DNs to pass to the **ldapdelete** command.

10.3.2. Commonly Used ldapsearch Options


The following table lists the most commonly used **ldapsearch** command-line options. If a specified value contains a space (), the value should be surrounded by single or double quotation marks, such as **-b "cn=My Special Group,ou=groups,dc=example,dc=com"**.



IMPORTANT

The **ldapsearch** utility from OpenLDAP uses SASL connections by default. To perform a simple bind or to use TLS/SSL, use the **-x** argument to disable SASL and allow other connection methods.

Option	Description
-b	<p>Specifies the starting point for the search. The value specified here must be a distinguished name that currently exists in the database. This is optional if the LDAP_BASEDN environment variable has been set to a base DN. The value specified in this option should be provided in single or double quotation marks. For example:</p> <div><pre>-b "cn=Barbara Jensen,ou=Product Development,dc=example,dc=com"</pre></div> <p>To search the root DSE entry, specify an empty string here, such as -b "".</p>
-D	<p>Specifies the distinguished name with which to authenticate to the server. This is optional if anonymous access is supported by the server. If specified, this value must be a DN recognized by the Directory Server, and it must also have the authority to search for the entries. For example, -D "uid=bjensen,dc=example,dc=com".</p>

Option	Description
-H	<p>Specifies an LDAP URL to use to connect to the server. For a traditional LDAP URL, this has the following format:</p> <pre>ldap[s]://hostname[:port]</pre> <p>The <i>port</i> is optional; it will use the default LDAP port of 389 or LDAPS port of 636 if the port is not given.</p> <p>This can also use an LDAPI URL, with each element separated by the HTML hex code %2F, rather than a forward slash (/):</p> <pre>ldapi://%2Ffull%2Fpath%2Fto%2Fslapd-example.socket</pre> <p>For LDAPI, specify the full path and filename of the LDAPI socket the server is listening to. Since this value is interpreted as an LDAP URL, the forward slash characters (/) in the path and filename must be escaped encoded as the URL escape value %2F.</p> <p>The -H option is used instead of -h and -p.</p>
-h	<p>Specifies the host name or IP address of the machine on which the Directory Server is installed. For example, -h server.example.com. If a host is not specified, ldapsearch uses the localhost.</p> <div data-bbox="815 1234 922 1373">  </div> <p>NOTE</p> <p>Directory Server supports both IPv4 and IPv6 addresses.</p>
-l	<p>Specifies the maximum number of seconds to wait for a search request to complete. For example, -l 300. The default value for the nsslapd-timelimit attribute is 3600 seconds. Regardless of the value specified, ldapsearch will never wait longer than is allowed by the server's nsslapd-timelimit attribute.</p>
-p	<p>Specifies the TCP port number that the Directory Server uses. For example, -p 1049. The default is 389.</p> <p>If -h is specified, -p must also be specified, even if it gives the default value.</p>

Option	Description
<i>-s scope</i>	<p>Specifies the scope of the search. The scope can be one of the following:</p> <div> <p>base searches only the entry specified in the -b option or defined by the LDAP_BASEDN environment variable.</p> <p>one searches only the immediate children of the entry specified in the -b option. Only the children are searched; the actual entry specified in the -b option is not searched.</p> <p>sub searches the entry specified in the -b option and all of its descendants; that is, perform a subtree search starting at the point identified in the -b option. This is the default.</p> </div>
<i>-w</i>	<p>Gives the password associated with the distinguished name that is specified in the -D option. If this option is not specified, anonymous access is used. For example, -w diner892.</p> <p>If there are metacharacters in the password that may be interpreted by the shell (such as exclamation points, !) then use single quotes to enclose the password. For example, -w 'secret!'.</p> <p>Alternatively, use the -W for the utility to prompt for the password instead of entering it in plaintext on the command line.</p>
<i>-x</i>	Disables the default SASL connection to allow simple binds.
<i>-Y SASL_mechanism</i>	Defines the SASL mechanism to use for connections. For example, -Y GSSAPI . If -x is not used, then the -Y option must be used.
<i>-z</i>	Sets the maximum number of entries to return in response to a search request. For example, -z 1000 . Normally, regardless of the value specified here, ldapsearch never returns more entries than the number allowed by the server's nsslapd-sizelimit attribute. However, this limitation can be overridden by binding as the root DN when using this command-line argument. When binding as the root DN, this option defaults to zero (0). The default value for the nsslapd-sizelimit attribute is 2000 entries.

10.3.3. Using Special Characters

When using the **ldapsearch** command-line utility, it may be necessary to specify values that contain characters that have special meaning to the command-line interpreter, such as space (), asterisk (*), or backslash (\). Enclose the value which has the special character in quotation marks ("). For example:

```
-D "cn=Barbara Jensen,ou=Product Development,dc=example,dc=com"
```

Depending on the command-line interpreter, use either single or double quotation marks. In general, use single quotation marks (') to enclose values. Use double quotation marks (") to allow variable interpolation if there are shell variables. Refer to the operating system documentation for more information.

10.4. LDAP SEARCH FILTERS

Search filters select the entries to be returned for a search operation. They are most commonly used with the **ldapsearch** command-line utility. When using **ldapsearch**, there can be multiple search filters in a file, with each filter on a separate line in the file, or a search filter can be specified directly on the command line.

The basic syntax of a search filter is:

```
attribute operator value
```

For example:

```
buildingname>=alpha
```

In this example, **buildingname** is the attribute, **>=** is the operator, and **alpha** is the value. Filters can also be defined that use different attributes combined together with Boolean operators.

NOTE

When performing a substring search using a matching rule filter, use the asterisk (*) character as a wildcard to represent zero or more characters.

For example, to search for an attribute value that starts with the letter **l** and ends with the letter **n**, enter a **l*n** in the value portion of the search filter. Similarly, to search for all attribute values beginning with the letter **u**, enter a value of **u*** in the value portion of the search filter.

To search for a value that contains the asterisk (*) character, the asterisk must be escaped with the designated escape sequence, **\5c2a**. For example, to search for all employees with **businessCategory** attribute values of **Example*Net product line**, enter the following value in the search filter:

```
Example\5c2a*Net product line
```



NOTE

A common mistake is to assume that the directory is searched based on the attributes used in the distinguished name. The distinguished name is only a unique identifier for the directory entry and cannot be used as a search key. Instead, search for entries based on the attribute-data pairs stored on the entry itself. Thus, if the distinguished name of an entry is **uid=bjensen,ou=People,dc=example,dc=com**, then a search for **dc=example** does not match that entry unless **dc:example** has explicitly been added as an attribute in that entry.

10.4.1. Using Attributes in Search Filters

The most basic sort of search looks for the presence of attributes or specific values in entries. There are many variations on *how* to look for attributes in entries. It is possible to check that the attribute merely exists, to match an exact value, or to list matches against a partial value.

A *presence* search uses a wild card (an asterisk) to return every entry which has that attribute set, regardless of value. For example, this returns every entry which has a ***manager*** attribute:

```
"(manager=*)"
```

It is also possible to search for an attribute with a specific value; this is called an *equality* search. For example:

```
"(cn=babs jensen)"
```

This search filter returns all entries that contain the common name Babs Jensen. Most of the time, equality searches are not case sensitive.

When an attribute has values associated with a language tag, all of the values are returned. Thus, the following two attribute values both match the **"(cn=babs jensen)"** filter:

```
cn: babs jensen
cn;lang-fr: babs jensen
```

It is also possible to search for a partial match on an attribute value, a *substring* index. For example:

```
"(description=*X.500*)"
"(sn=*nderson)"
"(givenname=car*)"
```

The length of the substring searches is configured in the substring index itself, as described in [Section 9.6, "Changing the Width for Indexed Substring Searches"](#).

10.4.2. Using Operators in Search Filters

Operators in search filters set the relationship between the attribute and the given search value. For people searches, operators can be used to set a range, to return a last names within a subset of letters in the alphabet or employee numbers that come after a certain number.

```
"(employeeNumber>=500)"
"(sn~=suret)"
"(salary<=150000)"
```

Operators also enable phonetic and approximate searches, which allow more effective searches with imperfect information and are particularly useful in internationalized directories.

The operators that can be used in search filters are listed in [Table 10.2, “Search Filter Operators”](#). In addition to these search filters, special filters can be specified to work with a preferred language collation order. For information on how to search a directory with international charactersets, see [Section D.4, “Searching an Internationalized Directory”](#).

Table 10.2. Search Filter Operators

Search Type	Operator	Description
Equality	=	Returns entries containing attribute values that exactly match the specified value. For example, cn=Bob Johnson
Substring	=string* string	Returns entries containing attributes containing the specified substring. For example, cn=Bob* cn=*Johnson cn=*John* cn=B*John . The asterisk (*) indicates zero (0) or more characters.
Greater than or equal to	>=	Returns entries containing attributes that are greater than or equal to the specified value. For example, buildingname >= alpha .
Less than or equal to	<=	Returns entries containing attributes that are less than or equal to the specified value. For example, buildingname <= alpha .
Presence	=*	Returns entries containing one or more values for the specified attribute. For example, cn=* telephoneNumber=* manager=* .
Approximate	~=	Returns entries containing the specified attribute with a value that is approximately equal to the value specified in the search filter. For example, cn~=suret l~=san francisco could return cn=sarette l=san francisco .

10.4.3. Using Compound Search Filters

Multiple search filter components can be combined using Boolean operators expressed in prefix notation as follows:

```
(Boolean-operator(filter)(filter)(filter)...) 
```

Boolean-operator is any one of the Boolean operators listed in [Table 10.3, “Search Filter Boolean Operators”](#).

For example, this filter returns all entries that do not contain the specified value:

```
(!(cn=Ray Kultgen))
(!(objectClass=person))
```

Obviously, compound search filters are most useful when they are nested together into completed expressions:

```
(Boolean-operator(filter)((Boolean-operator(filter)(filter)))
```

These compound filters can be combined with other types of searches (approximate, substring, other operators) to get very detailed results. For example, this filter returns all entries whose organizational unit is **Marketing** and whose description field does not contain the substring **X.500**:

```
(&(ou=Marketing)(!(description=*X.500*)))
```

That filter can be expanded to return entries whose organizational unit is **Marketing**, that do not have the substring **X.500**, and that have Julie Fulmer or Cindy Zwaska as a manager:

```
(&(ou=Marketing)(!(description=*X.500*))(|(manager=cn=Julie
Fulmer,ou=Marketing,dc=example,dc=com)(manager=cn=Cindy
Zwaska,ou=Marketing,dc=example,dc=com)))
```

This filter returns all entries that do not represent a person and whose common name is similar to **printer3b**:

```
(&(!(objectClass=person))(cn~=printer3b))
```

Table 10.3. Search Filter Boolean Operators

Operator	Symbol	Description
AND	&	All specified filters must be true for the statement to be true. For example, <i>(&(filter)(filter)(filter)...) </i> .
OR		At least one specified filter must be true for the statement to be true. For example, <i>((filter)(filter)(filter)...) </i> .

Operator	Symbol	Description
NOT	!	The specified statement must not be true for the statement to be true. Only one filter is affected by the NOT operator. For example, <i>(!(filter))</i> .

Boolean expressions are evaluated in the following order:

- Innermost to outermost parenthetical expressions first.
- All expressions from left to right.

10.4.4. Using Matching Rules

A *matching rule* tells the Directory Server how to compare two values (the value stored in the attribute and the value in the search filter). A matching rule also defines how to generate index keys. Matching rules are somewhat related to attribute syntaxes. Syntaxes define *the format* of an attribute value; matching rules define how that format is compared and indexed.

There are three different types of matching rules:

- **EQUALITY** specifies how to compare two values for an equal match. For example, how to handle strings like “Fred” and “FRED”. Search filters that test for equality (e.g. *attribute=value*) use the EQUALITY rule. Equality (eq) indexes use the EQUALITY rule to generate the index keys. Update operations use the EQUALITY rule to compare values to be updated with values already in an entry.
- **ORDERING** specifies how to compare two values to see if one value is greater or less than another value. Search filters that set a range (e.g. *attribute<=value* or *attribute>=value*) use the ORDERING rule. An index for an attribute with an ORDERING rule orders the equality values.
- **SUBSTR** specifies how to do substring matching. Substring search filters (e.g. *attribute=*partial_string** or *attribute=*end_string*) use the SUBSTR rule. Substring (sub) indexes use the SUBSTR rule to generate the index keys.

IMPORTANT

A matching rule is required in order to support searching or indexing for the corresponding search filter or index type. For example, an attribute must have an EQUALITY matching rule in order to support equality search filters and eq indexes for that attribute. An attribute must have both an ORDERING matching rule and an EQUALITY matching rule in order to support range search filters and indexed range searches.

A search operation will be rejected with `PROTOCOL_ERROR` or `UNWILLING_TO_PERFORM` if an attempt is made to use a search filter for an attribute that has no corresponding matching rule.

Example 10.1. Matching Rules and Custom Attributes

Example Corp. administrators create a custom attribute type called **MyFirstName** with IA5 String (7-bit ASCII) syntax and an EQUALITY matching rule of `caseExactIA5Match`. An entry with a

MyFirstName value of **Fred** is returned in a search with a filter of **(MyFirstName=Fred)**, but it is not returned for filters like **(MyFirstName=FRED)** and **(MyFirstName=fred)**. **Fred**, **FRED**, and **fred** are all valid IA5 String values, but they do not match using the **caseExactIA5Match** rule.

For all three variants of Fred to be returned in a search, then the **MyFirstName** should be defined to use the **caseIgnoreIA5Match** matching rule.

An extensible matching rule search filter can be used to search for an attribute value with a different matching rule than the one defined for the attribute. The matching rule must be compatible with the syntax of the attribute being searched. For example, to run a case insensitive search for an attribute that has a case-sensitive matching rule defined for it, specify a case insensitive matching rule in the search filter.

```
(MyFirstName:caseIgnoreIA5Match:=fred)
```



NOTE

Matching rules are used for searches in internationalized directories, to specify the language types to use for the results. This is covered in [Section D.4, “Searching an Internationalized Directory”](#).



NOTE

An index for an attributes uses whatever matching rules are defined for that attribute in its schema definition. Additional matching rules to use for an index can be configured using the **nsMatchingRule** attribute, as in [Section 9.2.2, “Creating Indexes from the Command Line”](#).

The syntax of the matching rule filter inserts a matching rule name or OID into the search filter:

```
attr:matchingRule:=value
```

- *attr* is an attribute belonging to entries being searched, such as **cn** or **mail**.
- *matchingRule* is a string that contains the name or OID of the rule to use to match attribute values according to the required syntax.
- *value* is either the attribute value to search for or a relational operator plus the attribute value to search for. The syntax of the value of the filter depends on the matching rule format used.

A matching rule is actually a schema element, and, as with other schema elements is uniquely identified by an object identifier (OID).

Many of the matching rules defined for Red Hat Directory Server relate to language codes and set internationalized collation orders supported by the Directory Server. For example, the OID **2.16.840.1.113730.3.3.2.17.1** identifies the Finnish collation order.



NOTE

Unlike other schema elements, additional matching rules cannot be added to the Directory Server configuration.

Most of the matching rules list in [Table 10.4, “General Syntax Matching Rules”](#) are used for equality indexes. Matching rules with *ordering* in their name are used for ordering indexes, and those with *substring* in their name are used for substring (SUBSTR) indexes. (The matching rules used for international matching and collation orders use a different naming scheme.)

Table 10.4. General Syntax Matching Rules

Matching Rule	Object Identifiers (OIDs)	Definitions	Compatible Syntaxes
Bitwise AND Match	1.2.840.113556.1.4.803	Performs bitwise AND matches.	Typically used with:[a] Integer Numeric String
Bitwise OR Match	1.2.840.113556.1.4.804	Performs bitwise OR matches.	Typically used with:[a] Integer Numeric String
booleanMatch	2.5.13.13	Evaluates whether the values to match are TRUE or FALSE.	Boolean
caseExactIA5Match	1.3.6.1.4.1.1466.109.114.1	Makes a case-sensitive comparison of values.	IA5 Syntax URI
caseExactMatch	2.5.13.5	Makes a case-sensitive comparison of values.	Directory String Printable String OID
caseExactOrderingMatch	2.5.13.6	Allows case-sensitive ranged searches (less than and greater than).	Directory String Printable String OID

Matching Rule	Object Identifiers (OIDs)	Definitions	Compatible Syntaxes
caseExactSubstringsMatch	2.5.13.7	Performs case-sensitive substring and index searches.	<div>Directory String</div> <div>Printable String</div> <div>OID</div>
caseIgnoreIA5Match	1.3.6.1.4.1.1466.109.114.2	Performs case-insensitive comparisons of values.	<div>IA5 Syntax</div> <div>URI</div>
caseIgnoreIA5SubstringsMatch	1.3.6.1.4.1.1466.109.114.3	Performs case-insensitive searches on substrings and indexes.	<div>IA5 Syntax</div> <div>URI</div>
caseIgnoreListMatch	2.5.13.11	Performs case-insensitive comparisons of values.	Postal Address
caseIgnoreListSubstringsMatch	2.5.13.12	Performs case-insensitive searches on substrings and indexes.	Postal Address
caseIgnoreMatch	2.5.13.2	Performs case-insensitive comparisons of values.	<div>Directory String</div> <div>Printable String</div> <div>OID</div>
caseIgnoreOrderingMatch	2.5.13.3	Allows case-insensitive ranged searches (less than and greater than).	<div>Directory String</div> <div>Printable String</div> <div>OID</div>

Matching Rule	Object Identifiers (OIDs)	Definitions	Compatible Syntaxes
caseIgnoreSubstringsMatch	2.5.13.4	Performs case-insensitive searches on substrings and indexes.	<div>Directory String</div> <div>Printable String</div> <div>OID</div>
distinguishedNameMatch	2.5.13.1	Compares distinguished name values.	Distinguished name (DN)
generalizedTimeMatch	2.5.13.27	Compares values that are in a Generalized Time format.	Generalized Time
generalizedTimeOrderingMatch	2.5.13.28	Allows ranged searches (less than and greater than) on values that are in a Generalized Time format.	Generalized Time
integerMatch	2.5.13.14	Evaluates integer values.	Integer
integerOrderingMatch	2.5.13.15	Allows ranged searches (less than and greater than) on integer values.	Integer
keywordMatch	2.5.13.33	Compares the given search value to a string in an attribute value.	Directory String
numericStringMatch	2.5.13.8	Compares more general numeric values.	Numeric String
numericStringOrderingMatch	2.5.13.9	Allows ranged searches (less than and greater than) on more general numeric values.	Numeric String
numericStringSubstringMatch	2.5.13.10	Compares more general numeric values.	Numeric String
objectIdentifierMatch	2.5.13.0	Compares object identifier (OID) values.	OID
octetStringMatch	2.5.13.17	Evaluates octet string values.	Octet String

Matching Rule	Object Identifiers (OIDs)	Definitions	Compatible Syntaxes
octetStringOrderingMatch	2.5.13.18	Supports ranged searches (less than and greater than) on a series of octet string values.	Octet String
telephoneNumberMatch	2.5.13.20	Evaluates telephone number values.	Telephone Number
telephoneNumberSubstringsMatch	2.5.13.21	Performs substring and index searches on telephone number values.	Telephone Number
uniqueMemberMatch	2.5.13.23	Compares both name and UID values.	Name and Optional UID
wordMatch	2.5.13.32	Compares the given search value to a string in an attribute value. This matching rule is case-insensitive.	Directory String
[a] This has a special format; the value is converted to integer before being used by Directory Server.			

Table 10.5. Language Ordering Matching Rules

Matching Rule	Object Identifiers (OIDs)
English (Case Exact Ordering Match)	2.16.840.1.113730.3.3.2.11.3
Albanian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.44.1
Arabic (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.1.1
Belorussian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.2.1
Bulgarian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.3.1
Catalan (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.4.1
Chinese - Simplified (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.49.1
Chinese - Traditional (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.50.1

Matching Rule	Object Identifiers (OIDs)
Croatian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.22.1
Czechoslovakian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.5.1
Danish (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.6.1
Dutch (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.33.1
Dutch - Belgian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.34.1
English - US (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.11.1
English - Canadian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.12.1
English - Irish (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.14.1
Estonian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.16.1
Finnish (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.17.1
French (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.18.1
French - Belgian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.19.1
French - Canadian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.20.1
French - Swiss (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.21.1
German (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.7.1
German - Austrian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.8.1
German - Swiss (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.9.1
Greek (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.10.1
Hebrew (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.27.1
Hungarian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.23.1
Icelandic (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.24.1
Italian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.25.1

Matching Rule	Object Identifiers (OIDs)
Italian - Swiss (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.26.1
Japanese (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.28.1
Korean (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.29.1
Latvian, Lettish (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.31.1
Lithuanian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.30.1
Macedonian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.32.1
Norwegian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.35.1
Norwegian - Bokmul (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.36.1
Norwegian - Nynorsk (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.37.1
Polish (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.38.1
Romanian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.39.1
Russian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.40.1
Serbian - Cyrillic (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.45.1
Serbian - Latin (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.41.1
Slovakian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.42.1
Slovenian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.43.1
Spanish (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.15.1
Swedish (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.46.1
Turkish (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.47.1
Ukrainian (Case Insensitive Ordering Match)	2.16.840.1.113730.3.3.2.48.1

Table 10.6. Language Substring Matching Rules

Matching Rule	Object Identifiers (OIDs)
English (Case Exact Substring Match)	2.16.840.1.113730.3.3.2.11.3.6
Albanian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.44.1.6
Arabic (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.1.1.6
Belorussian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.2.1.6
Bulgarian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.3.1.6
Catalan (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.4.1.6
Chinese - Simplified (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.49.1.6
Chinese - Traditional (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.50.1.6
Croatian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.22.1.6
Czechoslovakian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.5.1.6
Danish (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.6.1.6
Dutch (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.33.1.6
Dutch - Belgian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.34.1.6
English - US (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.11.1.6
English - Canadian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.12.1.6
English - Irish (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.14.1.6
Estonian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.16.1.6
Finnish (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.17.1.6
French (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.18.1.6
French - Belgian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.19.1.6
French - Canadian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.20.1.6

Matching Rule	Object Identifiers (OIDs)
French - Swiss (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.21.1.6
German (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.7.1.6
German - Austrian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.8.1.6
German - Swiss (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.9.1.6
Greek (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.10.1.6
Hebrew (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.27.1.6
Hungarian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.23.1.6
Icelandic (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.24.1.6
Italian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.25.1.6
Italian - Swiss (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.26.1.6
Japanese (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.28.1.6
Korean (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.29.1.6
Latvian, Lettish (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.31.1.6
Lithuanian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.30.1.6
Macedonian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.32.1.6
Norwegian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.35.1.6
Norwegian - Bokmul (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.36.1.6
Norwegian - Nynorsk (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.37.1.6
Polish (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.38.1.6
Romanian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.39.1.6
Russian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.40.1.6

Matching Rule	Object Identifiers (OIDs)
Serbian - Cyrillic (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.45.1.6
Serbian - Latin (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.41.1.6
Slovakian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.42.1.6
Slovenian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.43.1.6
Spanish (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.15.1.6
Swedish (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.46.1.6
Turkish (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.47.1.6
Ukrainian (Case Insensitive Substring Match)	2.16.840.1.113730.3.3.2.48.1.6

10.5. EXAMPLES OF COMMON LDAPSEARCHES

The next set of examples assumes the following:

- The search is for all entries in the directory.
- The directory is configured to support anonymous access for search and read. This means that no bind information has to be supplied in order to perform the search. For more information on anonymous access, see [Section 13.4.2, “Defining User Access - userdn Keyword”](#).
- The server is located on a host named **server.example.com**.
- The server uses port number **389**. Since this is the default port, the port number does not have to be sent in the search request.
- SSL is enabled for the server on port **636** (the default SSL port number).
- The suffix under which all data are stored is **dc=example,dc=com**.

10.5.1. Returning All Entries

Given the previous information, the following call will return all entries in the directory (subject to the configured size and time resource limits):

```
ldapsearch -D "cn=directory manager" -W -p 389 -h server.example.com -b
"dc=example,dc=com" -s sub -x "(objectclass=*)"
```

"objectclass=*" is a search filter that matches any entry in the directory. Since every entry must have an object class, and the *objectclass* attribute is always indexed, this is a useful search filter to return every entry.

10.5.2. Specifying Search Filters on the Command Line

A search filter can be specified directly on the command line as long as the filter is enclosed in quotation marks ("filter"). If the filter is supplied with the command, do not specify the **-f** option. For example:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -b "dc=example,dc=com" -s sub -x "cn=babs jensen"
```

10.5.3. Searching the Root DSE Entry

The root DSE is a special entry that contains information about the directory server instance, including all of the suffixes supported by the local Directory Server. This entry can be searched by supplying a search base of "", a search scope of **base**, and a filter of **"objectclass=*"**. For example:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -b "" -s base "objectclass=*"
```

10.5.4. Searching the Schema Entry

The **cn=schema** entry is a special entry that contains information about the directory schema, such as object classes and attribute types.

The following command lists the content of the **cn=schema** entry:

```
# ldapsearch -o ldif-wrap=no -D "cn=directory manager" -W -b "cn=schema" \
'(objectClass=subSchema)' -s sub objectClasses attributeTypes
matchingRules \
    matchingRuleUse    dITStructureRules nameForms ITContentRules
ldapSyntaxes
```

10.5.5. Using LDAP_BASEDN

To make searching easier, it is possible to set the search base using the **LDAP_BASEDN** environment variable. Doing this means that the search base does not have to be set with the **-b** option. For information on how to set environment variables, see the documentation for the operating system.

Typically, set **LDAP_BASEDN** to the directory's suffix value. Since the directory suffix is equal to the root, or topmost, entry in the directory, this causes all searches to begin from the directory's root entry.

For example, set **LDAP_BASEDN** to **dc=example,dc=com** and search for **cn=babs jensen** in the directory, use the following command-line call:

```
export LDAP_BASEDN="dc=example,dc=com"

ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x "cn=babs jensen"
```

In this example, the default scope of **sub** is used because the **-s** option was not used to specify the scope.

10.5.6. Displaying Subsets of Attributes

The **ldapsearch** command returns all search results in LDIF format. By default, **ldapsearch** returns the entry's distinguished name and all of the attributes that a user is allowed to read. The directory

access control can be set such that users are allowed to read only a subset of the attributes on any given directory entry. Only operational attributes are not returned. For operational attributes to be returned as a result of a search operation, explicitly specify them in the search command.

It may not be necessary to have all of the attributes for an entry returned in the search results. The returned attributes can be limited to just a few specific attributes by specifying the desired ones on the command line immediately after the search filter. For example, to show the **cn** and **sn** attributes for every entry in the directory, use the following command-line call:

```
ldapsearch -D "cn=directory manager" -W -p 389 -h server.example.com -b
"dc=example,dc=com" -s sub -x "(objectclass=*)" sn cn
```

10.5.7. Searching for Operational Attributes

Operational attributes are special attributes set by the Directory Server itself that are used by the server to perform maintenance tasks, like processing access control instructions. They also show specific information about the entry, like the time it was initially created and the name of the user who created it. Operational attributes are available for use on every entry in the directory, regardless of whether the attribute is specifically defined for the object class of the entry.

Operational attributes are not returned in regular **ldapsearches**, so to return operational attributes, they have to be explicitly specified in the **ldapsearch** request.

```
ldapsearch -D "cn=directory manager" -W -p 389 -h server.example.com -b
"dc=example,dc=com" -s sub -x "(objectclass=*)" creatorsName
createTimestamp modifiersName modifyTimestamp
```

The complete list of operational attributes is in the "Operational Attributes and Object Classes" chapter in the [Red Hat Directory Server 9 Configuration, Command, and File Reference](#).

NOTE

To return all of the regular entry attributes along with the specified operational attributes, use the special search attribute, **"*"**, in addition to the operational attributes that are listed.

```
ldapsearch -D "cn=directory manager" -W -p 389 -h
server.example.com -b "dc=example,dc=com" -s sub -x "
(objectclass=*)" "*" aci
```

The asterisk must be enclosed in quotation marks to prevent it from being interpreted by the shell.

10.5.8. Specifying Search Filters Using a File

Search filters can be entered into a file instead of entering them on the command line. In this case, specify each search filter on a separate line in the file. The **ldapsearch** command runs each search in the order in which it appears in the file.

For example:

```
sn=Francis
givenname=Richard
```

ldapsearch first finds all the entries with the surname **Francis**, then all the entries with the givenname **Richard**. If an entry is found that matches both search criteria, then the entry is returned twice.

For example, in this search, the filters are specified in a file named **searchdb**:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -f searchdb
```

The set of attributes returned here can be limited by specifying the attribute names at the end of the search line. For example, the following **ldapsearch** command performs both searches but returns only the DN and the **givenname** and **sn** attributes of each entry:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -f searchdb sn givenname
```

10.5.9. Specifying DN's That Contain Commas in Search Filters

When a DN within a search filter contains a comma as part of its value, the comma must be escaped with a backslash (\). For example, to find everyone in the **example.com Bolivia, S.A.** subtree, use the following command:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -s base -b "l=Bolivia\,S.A.,dc=example,dc=com"
"objectclass=*"
```

10.5.10. Using Client Authentication When Searching

Client authentication uses a stored certificate to bind to the directory rather than simple user name and password combination. The SSL parameters are set separately as an environment variable or by editing **ldap.conf**. Then, the **ldapsearch** can be run by disabling SASL and specifying the SSL port. For example:

```
export LDAPTLS_CACERTDIR=/etc/dirsrv/slapd-instance_name
export LDAPTLS_CERT=Server-Cert
export LDAPTLS_KEY=internal:secret

ldapsearch -h server.example.com -p 636 -b "dc=example,dc=com" -x
"givenname=Richard"
```

The possible environment variables are described more in [Section A.1, "Environment Variables Used with LDAP Client Tools"](#).

10.5.11. Searching with Language Matching Rules

To explicitly submit a matching rule in a search filter, insert the matching rule after the attribute:

```
attr:matchingRule:=value
```

Matching rules are frequently used for searching internationalized directories. For example, this searches for the department numbers after N4709 in the Swedish (2.16.840.1.113730.3.3.2.46.1) matching rule.

```
departmentNumber:2.16.840.1.113730.3.3.2.46.1:=>= N4709
```

More examples of performing internationalized searches are given in [Section D.4, “Searching an Internationalized Directory”](#).

10.5.12. Searching for Attributes with Bit Field Values

Bitwise searches use the bitwise AND or bitwise OR matching rules to perform bitwise search operations on attributes with values that are bit fields.



NOTE

Attributes with values for bit fields are not common in LDAP. (No default Directory Server schema use bit fields as attribute syntax.) However, several LDAP syntaxes support integer-style values. Custom attributes can be defined which use bit field values, and applications can use those custom attributes to perform bitwise operations against bit field values.

The bitwise AND matching rule (**1.2.840.113556.1.4.803**) checks that the bit given in the assertion value is set in the bit field attribute value. (This is somewhat analogous to an equality search.) In this example, the `userAccountControl` value must be set to the bit representing 2.

```
"(UserAccountControl:1.2.840.113556.1.4.803:=2)"
```

In this example, the `userAccountControl` value must have all of the bits set that are set in the value 6 (bits 2 and 4).

```
"(UserAccountControl:1.2.840.113556.1.4.803:=6)"
```

The bitwise OR matching rule (**1.2.840.113556.1.4.804**) checks to see if *any* of the bits in the assertion string are represented in the attribute value. (This is somewhat analogous to a substring search.) In this example, the `userAccountControl` value must have any of the bits which are set in the bit field of 6, meaning that the attribute value can be 2, 4, or 6.

```
"(UserAccountControl:1.2.840.113556.1.4.804:=6)"
```

Bitwise searches can be used with Windows-Red Hat Enterprise Linux integration, such as using Samba file servers.



NOTE

Microsoft has good documentation on bitwise operators at <http://msdn.microsoft.com/en-us/library/aa746475>.

10.6. USING PERSISTENT SEARCH

A persistent search is an `ldapssearch` which remains open even after the initial search results are returned.



IMPORTANT

The OpenLDAP client tools with Red Hat Enterprise Linux do not support persistent searches. The server itself, however, does. Other LDAP clients must be used to perform persistent searches.

The purpose of a persistent search is to provide a continuous list of changes to the directory entries as well as the complete entries themselves, something like a hybrid search and changelog. Therefore, the search command must specify what entries to return (the search parameters) and what changes cause an entry to be returned (entry change parameters).

Persistent searches are especially useful for applications or clients which access the Directory Server and provide two important benefits:

- Keep a consistent and current local cache.

Any client will query local cache before trying to connect to and query the directory. Persistent searches provide the local cache necessary to improve performance for these clients.

- Automatically initiate directory actions.

The persistent cache can be automatically updated as entries are modified, and the persistent search results can display what kind of modification was performed on the entry. Another application can use that output to update entries automatically, such as automatically creating an email account on a mail server for new users or generating a unique user ID number.

There are some performance considerations when running persistent searches, as well:

- The **ldapsearch** does not send a notification when the client disconnects, and the change notifications are not sent for any changes made while the search is disconnected. This means that the client's cache will not be updated if it is ever disconnected and there is no good way to update the cache with any new, modified, or deleted entries that were changed while it was disconnected.
- An attacker could open a large number of persistent searches to launch a denial of service attack.
- A persistent search requires leaving open a TCP connection between the Directory Server and client. This should only be done if the server is configured to allow a lot of client connections and has a way to close idle connections.
- Each persistent search runs on a separate thread in Directory Server.

In the access logs, a persistent search is identified with the tag **options=persistent**.

```
[12/Jan/2009:12:51:54 -0500] conn=19636710736396323 op=0 SRCH  
base="dc=example,dc=com" scope=2 filter="(objectClass=person)" attrs=ALL  
options=persistent
```

10.7. SEARCHING WITH SPECIFIED CONTROLS

The Directory Server has defined controls in its **supportedControls** attribute in its DSE. Some of these define server operations like replication; other are allowed extended operations like get effective rights or dereferencing controls which clients can pass through LDAP operations to the server.

These controls can be specified using the **-E** option by giving the control OID, its criticality for the **ldapsearch**, and any information required for the control operation.

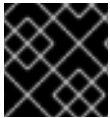
```
-E '[!]control_OID:control_information'
```

Some controls, like server-side sorting and simple paged results, have an alias that can be used to pass the control to the search operation. When the control alias is used, then the results are formatted, since the control is recognized by the client.

10.7.1. Retrieving Effective User Rights

A get effective-rights search control is passed using the control OID. For example:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -b "dc=example,dc=com" -s sub -x -E
'!1.3.6.1.4.1.42.2.27.9.5.2:dn:uid=jsmith,ou=people,dc=example,dc=com' "(objectclass=*)"
```



IMPORTANT

When a control is passed with its OID, the results from the search are unformatted.

Get effective rights searches are covered in much more detail in the access control chapter, [Section 13.7, “Checking Access Rights on Entries \(Get Effective Rights\)”](#).

10.7.2. Using Server-Side Sorting

Server-side sorting is performed as other control operations, using the **-E** flag and the **sss** control alias. The structure of the operation sets the attribute by which to sort the results and, optionally, the sort order and ordering rule.

```
-E sss=[-]attribute_name:[ordering_rule_OID]
```

The dash (-) is an optional flag that reverses the sort order, which naturally runs descending. The matching rule tables in [Section 10.4.4, “Using Matching Rules”](#) contain the ordering rules supported by the Directory Server.

For example:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -b "dc=example,dc=com" -s sub -x -E sss=-
uidNumber:2.5.13.15 "(objectclass=*)"
```

10.7.3. Performing Dereferencing Searches

A *dereferencing* search is a quick way to track back over cross-references in an entry and return information about the referenced entry. For example, a group entry contains references to its member's user entries. A regular search first searches for the group, then lists its members, and then requires a separate search for each member. A dereferencing search for the group entry returns information about the members — such as their locations, email addresses, or managers — *along with* the information for the group, all in a single search request.

Dereferencing simplifies many client operations and reduces the number of search operations that are performed. Cross-links show relationships between entries. Some operations may require getting a list of cross-links from one entry and then performing a series of subsequent searches to get information from each entry on the list. Dereferencing allows those sequences of searches to be consolidated into a single search.



IMPORTANT

Dereferencing operations must be done using OpenLDAP command-line tools version 2.4.18 or later or other clients which support dereferencing searches.

The format of the dereference arguments is:

```
-E 'deref=deref_attribute:list_of_attributes'
```

The *deref_attribute* is the attribute in the search target that contains the reference. This can be any attribute which has a DN for its value, such as **member** or **manager**.



NOTE

Not only must the value of the *deref_attribute* be a DN, but the actual defined syntax for the attribute must be DN syntax (**1.3.6.1.4.1.1466.115.121.1.12**).

The *list_of_attributes* is one or more attributes in the referenced entry which will be returned along with the primary search results. Multiple attributes can be separated by commas, like **1,mail,cn**.

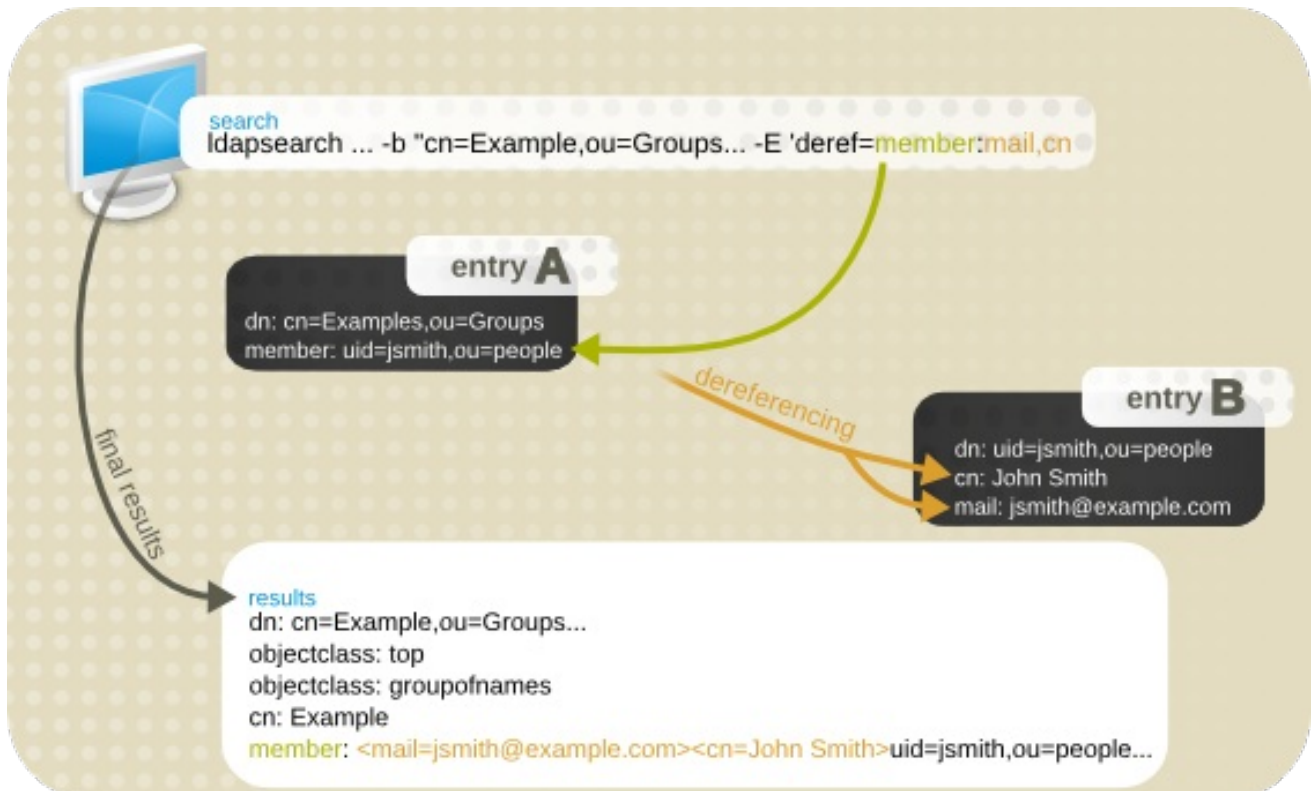


Figure 10.3. Simple Dereferencing Search Command

The requested dereferenced information requested in the search argument is returned with the rest of the search results. For example, this dereferencing search tells the server to use the **member** attribute in

the search target entry (the Engineers group) as the *deref_attribute*. It then returns the locality attribute for each member.

```
ldapsearch -x -D "cn=Directory Manager" -W -b
"cn=Example,ou=Groups,dc=example,dc=com" -E 'deref=member:mail,cn' "
(objectclass=*)"

# Engineers, Groups, example.com
dn: cn=Engineers,ou=Groups,dc=example,dc=com
control: 1.3.6.1.4.1.4203.666.5.16 false
MIQAAADNMIQAAAA1BAZtZW1iZXIEK2NuPURld
mVsb3BlcnMsIG91PUdyb3VwcywgZGM9ZXhhbXBsZSxkYz1jb20whAAAADIEBm1lbWJlcmQoY2
49VG
VzdGVycywgY3U9R3JvdXBzLCBkYz1leGFtcGx1LGRjPWNvbTCEAAAVAQGbWVtYmVyBCp1awQ
9ZW5
nLCBvdT1lbmdpbmVlcmluZywgZGM9ZXhhbXBsZSxkYz1jb22ghAAAABowhAAAABQEAWwxhAAA
AASE
CUNhbWJyawRnZQ==
# member: <mail=jsmith@example.com><cn=John
Smith>;uid=jsmith,ou=people,dc=example,dc=com
objectClass: top
objectClass: inetuser
objectClass: groupofnames
cn: Engineers
member: uid=jsmith,ou=people,dc=example,dc=com
```

10.7.4. Using Simple Paged Results

Search results can be very large, and a part of processing the results is organizing the results. One method of doing this is using *simple paged results*, a control that breaks the results into pages of a certain length.

The simple paged results control sets the number of entries to display at a time. The results can be scrolled through one page at a time which makes the results easier to digest. The full behavior of the control is described in [RFC 2696](#).

Simple paged results are implemented as an LDAP control extension for the Directory Server. Its OID is **1.2.840.113556.1.4.319**.

How Simple Paged Results Work

When you start a simple paged results search:

1. The client sends the search to the server, together with the paged results control and with how many records to return in the first page.
2. Before Directory Server starts returning data, the server generates an estimate how many records can be returned in total.

The estimate of records is not an exact number. The total number of records returned can be lower than the estimate. The reasons for such a scenario include

- attributes used in the search filter do not exist in the index. For an optimal result, all queried attributes must be indexed.

- before an entry is sent to the client, access control lists (ACL) are validated. Insufficient permissions can prevent the entry from being returned.

After generating the estimate, the server sends the first set of results, a cookie, and the estimated number of records.

3. The returned records are displayed in the client. The user can now enter how many records should be returned in the next request. The requested number is now sent, together with the cookie, to the server.
4. The server retrieves the requested number of records from the database and sends them together with a cookie to the client.
5. The previous two steps are repeated until all records are sent or the search is cancelled.

Simple Paged Results and OpenLDAP Tools

The format of the simple paged result search option with **ldapsearch** is:

```
-E pg=size
```

The *size* value is the page size, or the number of entries to include per page. For example:

```
ldapsearch -x -D "cn=Directory Manager" -W -b
"ou=Engineers,ou=People,dc=example,dc=com" -E pg=3 "(objectclass=*)" cn

dn: uid=jsmith,ou=Engineers,ou=People,dc=example,dc=com
   cn: John Smith

dn: uid=bjensen,ou=Engineers,ou=People,dc=example,dc=com
   cn: Barbara Jensen

dn: uid=hmartin,ou=Engineers,ou=People,dc=example,dc=com
   cn: Henry Martin

Results are sorted.
next page size (3): 5
```

The tag at the end shows the configured page size (the number in parentheses) from the search. After the colon, one enters the page size for the next page, so entering **5** as shown would open the next page of results with five entries.



IMPORTANT

Simple paged results operations must be done using OpenLDAP command-line tools version 2.4.18 or later or other clients which support simple paged results, such as Perl Net::LDAP.

Simple Paged Results and Server-Side Sorting

Simple paged results can be used together with server-side sorting. Server-side sorting is a control which performs the sort process on the server rather than in a client; this is usually done for a search which uses a particular matching rule. (This behavior is defined in [RFC 2891](#).) The OpenLDAP client tools do not support server-side sort with the simple paged results control, but other LDAP utilities such as Perl Net::LDAP do support both.

Multiple Simple Paged Results Requests on a Single Connection

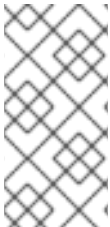
Some clients may open a single connection to the Directory Server, but send multiple operation requests, including multiple search requests using the simple paged results extension.

Directory Server can manage and interpret multiple simple paged searches. Each search is added as an entry in an array. When the paged search request is first sent, there is a cookie created and associated with the search results. Each page of results is returned with that cookie, and that cookie is used to request the next page of results. On the last page, the cookie is empty, signalling the end of the results. This keeps each set of search results separate.

When there are multiple simple paged results on a single connection, the timeout limits are still observed, but *all* open search requests must reach their configured time limit before *any* paged search is disconnected.

Simple Paged Results, Contrasted with VLV Indexes

VLV indexes are similar to simple paged results in that they also return a usable browsing list of results. The main difference is in how that list is generated. Simple paged results are calculated per search, while VLV indexes are a permanent list. Overall, VLV indexes are faster for searches, but do require some server-side configuration and overhead for the server to maintain.



NOTE

Simple paged results and VLV indexes *cannot* be used on the same search. Simple paged results would attempt to manipulate the VLV index, which is already a browsing index. If the control is passed for a search using a VLV index, then the server returns an **UNWILLING_TO_PERFORM** error.

For more information on VLV indexes, see [Section 9.4, “Creating Browsing \(VLV\) Indexes”](#).

10.7.5. Pre- and Post-read Entry Response Controls

Red Hat Directory Server supports pre- and post-read entry response controls according to [RFC 4527](#). If a client requests one or both response controls, an LDAP search entry is returned, that contains the attribute's value before and after the update.

When the pre-read control is used, an LDAP search query is returned containing the specified attribute's value before modification. When the post-read control is used, the query contains the attribute's value after modification. Both controls can be used at the same time. For example, to update the **description** attribute and display the value before and after the modification:

```
# ldapmodify -D "cn=directory manager" -W -x \
    -e \!preread=description -e \!postread=description
dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
replace: description
description: new description

modifying entry "uid=user,ou=People,dc=example,dc=com"
control: 1.3.6.1.1.13.1 false
ZCKEJXVpZD1qdXNlcixvdT1QZW9wbGUsZGM9ZXhhbXBsZSxk
Yz1jb20wAA==
# ==> preread
dn: uid=user,ou=People,dc=example,dc=com
description: old description
```

```
# <== prered  
control: 1.3.6.1.1.13.2 false  
ZEsEJXVpZD1qdXNlcixvdT1QZW9wbGUsZGM9ZXhhbXBsZSxk  
Yz1jb20wIjAgBAtkZXNjcmlwdGlvbjerBA9uZXcgZGVzY3JpcHRpb24=  
# ==> postread  
dn: uid=user,ou=People,dc=example,dc=com  
description: new description  
# <== postread
```

CHAPTER 11. MANAGING REPLICATION

Replication is the mechanism by which directory data is automatically copied from one Red Hat Directory Server instance to another; it is an important mechanism for extending the directory service beyond a single server configuration. This chapter describes the tasks to be performed on the master and consumer servers to set up single-master replication, multi-master replication, and cascading replication.

11.1. REPLICATION OVERVIEW

Replication is the mechanism by which directory data is automatically copied from one Directory Server to another. Updates of any kind — entry additions, modifications, or even deletions — are automatically mirrored to other Directory Servers using replication.

- [Section 11.1.1, “What Directory Units Are Replicated”](#)
- [Section 11.1.2, “Read-Write and Read-Only Replicas”](#)
- [Section 11.1.3, “Suppliers and Consumers”](#)
- [Section 11.1.4, “Changelog”](#)
- [Section 11.1.5, “Replication Identity”](#)
- [Section 11.1.6, “Replication Agreement”](#)
- [Section 11.1.8, “Replication with 4.x Versions of Directory Server”](#)

11.1.1. What Directory Units Are Replicated

The smallest unit of of the directory which can be replicated is a database. This means that one can replicate an entire database but not a subtree within a database. Therefore, when creating the directory tree, consider any replication plans as part of determining how to distribute information.

Replication also requires that one database correspond to one suffix. This means that a suffix (or namespace) that is distributed over two or more databases using custom distribution logic cannot be replicated. For more information on this topic, see [Section 2.2, “Creating and Maintaining Databases”](#).

11.1.2. Read-Write and Read-Only Replicas

A database that participates in replication is called a *replica*. There are two kinds of replicas: read-write or read-only. A *read-write replica* contains master copies of directory information and can be updated. A *read-only replica* services read, search, and compare requests, but refers all update operations to read-write replicas. A server can hold any number of read-only or read-write replicas.

11.1.3. Suppliers and Consumers

A server that holds a replica that is copied to a replica on a different server is called a *supplier* for that replica. A server that holds a replica that is copied from a different server is called a *consumer* for that replica. Generally, the replica on the supplier server is a read-write replica, and the one on the consumer server is a read-only replica, with two exceptions:

- In the case of cascading replication, the hub server holds a read-only replica that it supplies to consumers. [Section 11.2.3, “Cascading Replication”](#) has more information.

- In the case of multi-master replication, the *masters* are both suppliers and consumers for the same information. For more information, see [Section 11.2.2, “Multi-Master Replication”](#).

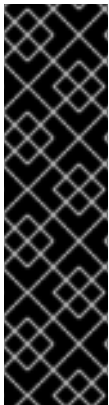
Replication is always initiated by the supplier server, never by the consumer (*supplier-initiated replication*). Supplier-initiated replication allows a supplier server to be configured to push data to multiple consumer servers.

11.1.4. Changelog

Every supplier server maintains a *changelog*, a record of all changes that a supplier or hub needs to send to its consumers. A changelog is a special kind of database that describes the modifications that have occurred on a replica. The supplier server then replays these modifications to the replicas stored on consumer servers or to other suppliers, in the case of multi-master replication.

When an entry is modified, a change record describing the LDAP operation that was performed is recorded in the changelog.

The changelog uses the same database environment as the main database. Implementing the changelog as part of the main database ensures the database and changelog are always synchronized, reduces the required database cache size (10MB by default), and simplifies backup and restore operations.



IMPORTANT

When the database of a master server is backed up, then it should be backed up using the **db2bak.p1** Perl script or using the Directory Server Console if the server is kept running. The changelog only writes its RUV entries to the database when the server is shut down; while the server is running, the changelog keeps its changes in memory. For the Perl script and the Console, these changelog RUVs are written to the database before the backup process runs. However, that step is not performed by the command-line script.

The **db2bak** should not be run on a running master server. Either use the Perl script or stop the server before performing the backup.

In Directory Server, the changelog is only intended for internal use by the server. For other applications to read the changelog, use the Retro Changelog Plug-in, as described in [Section 11.21, “Using the Retro Changelog Plug-in”](#).

11.1.5. Replication Identity

When replication occurs between two servers, the replication process uses a special entry, called the *replication manager* entry, to identify replication protocol exchanges and to control access to the directory data. The replication manager entry, or any entry used during replication, must meet the following criteria:

- It is created on the consumer server (or hub) and *not* on the supplier server.
- Create this entry on *every* server that receives updates from another server, meaning on every hub or dedicated consumer.
- When a replica is configured as a consumer or hub (a replica which receives updates from another server), this entry must be specified as the one authorized to perform replication updates.

- The replication agreement is created on the supplier server, the DN of this entry must be specified in the replication agreement.
- The supplier bind DN entry must not be part of the replicated database for security reasons.
- This entry, with its special user profile, bypasses all access control rules defined on the consumer server for the database involved in that replication agreement.



NOTE

In the Directory Server Console, this replication manager entry is referred to as the *supplier bind DN*, which may be misleading because the entry does not actually exist on the supplier server. It is called the supplier bind DN because it is the entry which the supplier uses to bind to the consumer. This entry actually exists, then, on the consumer.

For more information on creating the replication manager entry, see [Section 11.3, “Creating the Supplier Bind DN Entry”](#).

11.1.6. Replication Agreement

Directory Servers use replication agreements to define their replication configuration. A replication agreement describes replication between *one* supplier and *one* consumer only. The agreement is configured on the supplier server and must specify all required replication information:

- The database to be replicated.
- The consumer server to which the data is pushed.
- The days and times during which replication can occur.
- The DN and credentials that the supplier server must use to bind (the replication manager entry or supplier bind DN).
- How the connection is secured (SSL, client authentication).
- Any attributes that will not be replicated (fractional replication).

11.1.7. Replicating a Subset of Attributes with Fractional Replication

Fractional replication sets a specific subset of attributes that will not be transmitted from a supplier to the consumer (or another supplier). Administrators can therefore replicate a database without replicating all the information that it contains or all of the information in every entry.

Fractional replication is enabled and configured per replication agreement, not per entry. Excluding attributes from replication is applied equally to all entries within the replication agreement's scope.

As far as the consumer server is concerned, the excluded attributes always have no value. Therefore, a client performing a search against the consumer server will never see the excluded attributes. Similarly, should it perform a search that specifies those attributes in its filter, no entries will match.

It is possible to set different attributes to be replicated for an incremental update and a total update. The incremental update list (*nsDS5ReplicatedAttributeList*) must always be set to enable fractional replication; if that is the only attribute set, then it applies to both incremental and total updates. The

optional ***nsDS5ReplicatedAttributeListTotal*** attribute sets an additional fractional replication list for total updates. This is described in [Section 11.9.1, “Setting Different Fractional Replication Attributes for Total and Incremental Updates”](#).



NOTE

An update to an excluded attribute still triggers a modify event and generates an empty replication update. The ***nsds5ReplicaStripAttrs*** attribute adds a list of attributes which cannot be sent in an empty replication event and are stripped from the update sequence. This logically includes operational attributes like ***modifiersName***.

If a replication event is *not* empty, the stripped attributes *are* replicated. These attributes are removed from updates only if the event would otherwise be empty.

11.1.7.1. The Replication Keep-alive Entry

When you update an attribute on a master, the change sequence number (CSN) is increased on the master. In a replication topology, this server now connects to the first consumer and compares the local CSN with the CSN on the consumer. If it is lower, the update is retrieved from the local changelog and replicated to the consumer. In a replication topology with fractional replication enabled, this can cause problems: For example, if only attributes are updated on the master that are excluded from replication, no update to replicate is found, and therefore the CSN is not updated on the consumer. In certain scenarios, such as when only attributes are updated on a master that are excluded from replication, unnecessary searching for updates on the supplier can cause other servers to receive the data later than needed. To work around this problem, Directory Server uses keep-alive entries.

If all updated attributes on the master are excluded from replication and the number of skipped updates exceeds 100, the ***keepalivetimestamp*** attribute is updated on the supplier and replicated to the consumer. Because the ***keepalivetimestamp*** attribute is not excluded from replication, the update of the keep-alive entry is replicated, the CSN on the consumer is updated, and then equal to the one on the supplier. The next time the supplier connects to the consumer, only updates that are newer than the CSN on the consumer are searched. This reduces the amount of time spent by a supplier to search for new updates to send.

The replication keep-alive entry is created on demand on a master and contains the replica ID of the master in the distinguished name (DN). Each keep-alive entry is specific to a given master. For example:

```
dn: cn=repl keep alive 14,dc=example,dc=com
objectclass: top
objectclass: ldapsubentry
objectclass: extensibleObject
cn: repl keep alive 14
keepalivetimestamp: 20170227190346Z
```

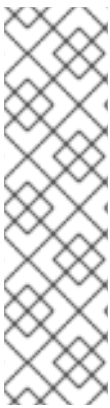
The keep-alive entry is updated in the following situations (if it does not exist before the update, it is created first):

- When a fractional replication agreement skips more than 100 updates and does not send any updates before ending the replication session.
- When a master initializes a consumer, initially it creates its own keep-alive entry. A consumer that is also a master does not create its own keep-alive entry unless it also initializes another consumer.

11.1.8. Replication with 4.x Versions of Directory Server

The replication mechanism in Directory Server 9.0 is different from the mechanism used in 4.x of Directory Server. Compatibility with Netscape 4.x and iPlanet versions of Directory Server is provided through two Directory Server plug-ins:

- *Legacy Replication Plug-in.* The Legacy Replication Plug-in makes a Directory Server 9.0 instance behave as a 4.x Directory Server in a consumer role. For information on how to implement legacy replication using this plug-in, see [Section 11.20, “Replication with Earlier Releases”](#).
- *Retro Changelog Plug-in.* The Retro Changelog Plug-in can be used for a Directory Server supplier to maintain a 4.x-style changelog. This is sometimes necessary for legacy applications that have a dependency on the Directory Server 4.x changelog format because they read information from the changelog. For more information on the Retro Changelog Plug-in, see [Section 11.21, “Using the Retro Changelog Plug-in”](#).



NOTE

Replication with 8.x versions of Red Hat Directory Server, including multi-master replication and using 8.x replicas with 9.0 masters, is fully supported. The replication mechanisms used for Directory Server 7.x and 8.x is similar to the mechanism used in Directory Server 9.0. No additional plug-ins or configuration are required for replication to work with versions 6.x, 7.x, 8.x, or 9.x of Directory Server.

Any incompatibilities are related to enhanced features, additional schema, and other features that are available in Directory Server 9.0 which may not be available in 7.x or 8.x servers.

11.2. REPLICATION SCENARIOS

- [Section 11.2.1, “Single-Master Replication”](#)
- [Section 11.2.2, “Multi-Master Replication”](#)
- [Section 11.2.3, “Cascading Replication”](#)

These basic strategies can be combined in a variety of ways to create the best replication environment.



NOTE

Whatever replication scenario is implemented, consider schema replication. To avoid conflict resolution loops, the Referential Integrity Plug-in should only be enabled on one supplier replica in a multi-master replication environment. The plug-in is off by default.

11.2.1. Single-Master Replication

In the simplest replication scenario, the master copy of directory data is held in a single read-write replica on one server called the *supplier server*. The supplier server also maintains changelog for this replica. On another server, called the *consumer server*, there can be multiple read-only replicas. Such scenarios are called *single-master configurations*. [Figure 11.1, “Single-Master Replication”](#) shows an example of single-master replication.

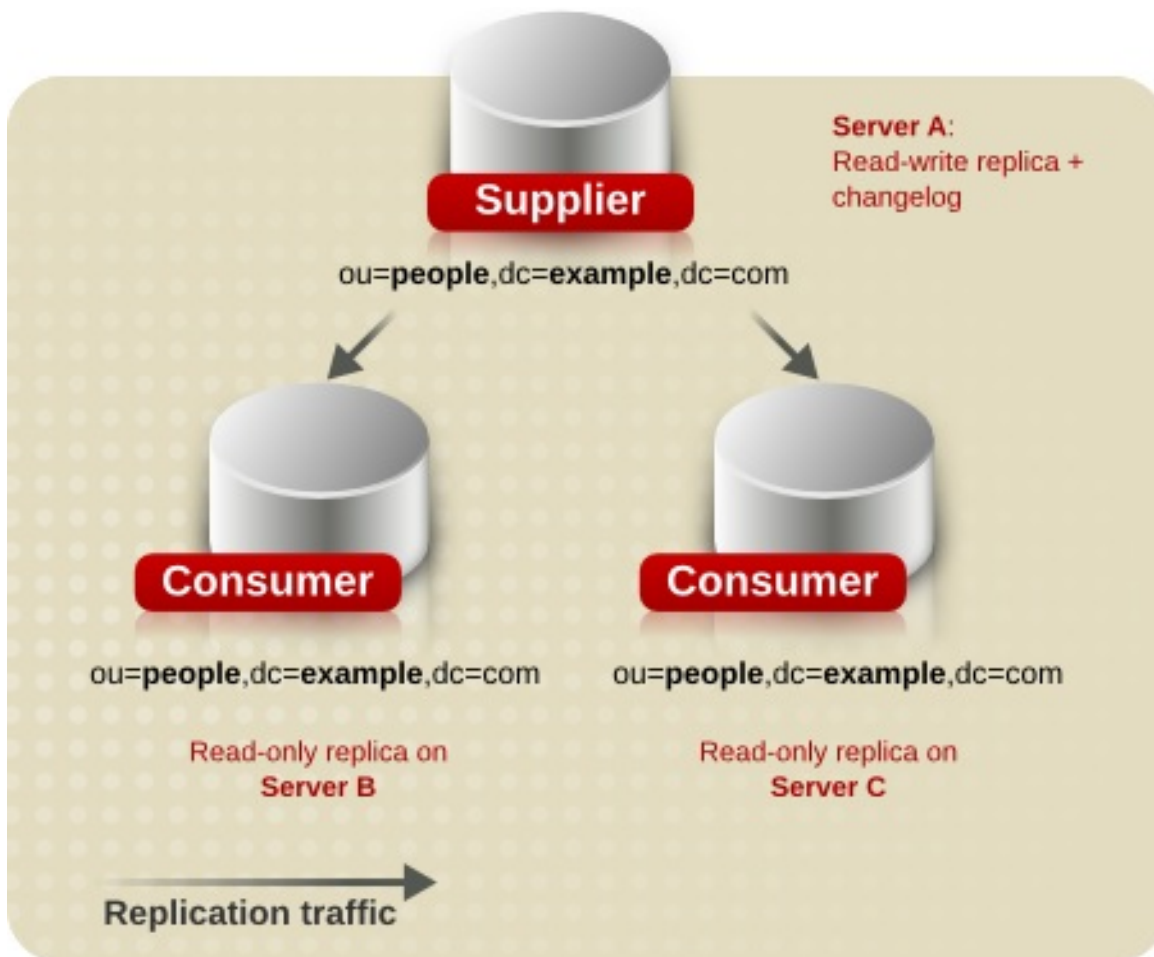


Figure 11.1. Single-Master Replication

In this particular configuration, the `ou=people,dc=example,dc=com` suffix receives a large number of search requests. Therefore, to distribute the load, this tree, which is mastered on Server A, is replicated to two read-only replicas located on Server B and Server C.

For information on setting up a single-master replication environment, see [Section 11.4, “Configuring Single-Master Replication”](#).

11.2.2. Multi-Master Replication

Directory Server also supports complex replication scenarios in which the same suffix (database) can be mastered on many servers. This suffix is held in a read-write replica on each server. This means that each server maintains a changelog for the read-write replica.

Multi-master replication in Directory Server supports as many as 20 masters, an unlimited number of hub suppliers, and an unlimited number of consumer servers. Each consumer server holds a read-only replica. The consumers can receive updates from any or all the suppliers. The consumers also have referrals defined for all the suppliers to forward any update requests that the consumers receive. Such scenarios are called *multi-master configurations*.

[Figure 11.2, “Multi-Master Replication \(Two Masters\)”](#) shows an example of multi-master replication scenario with two supplier servers and two consumer servers.

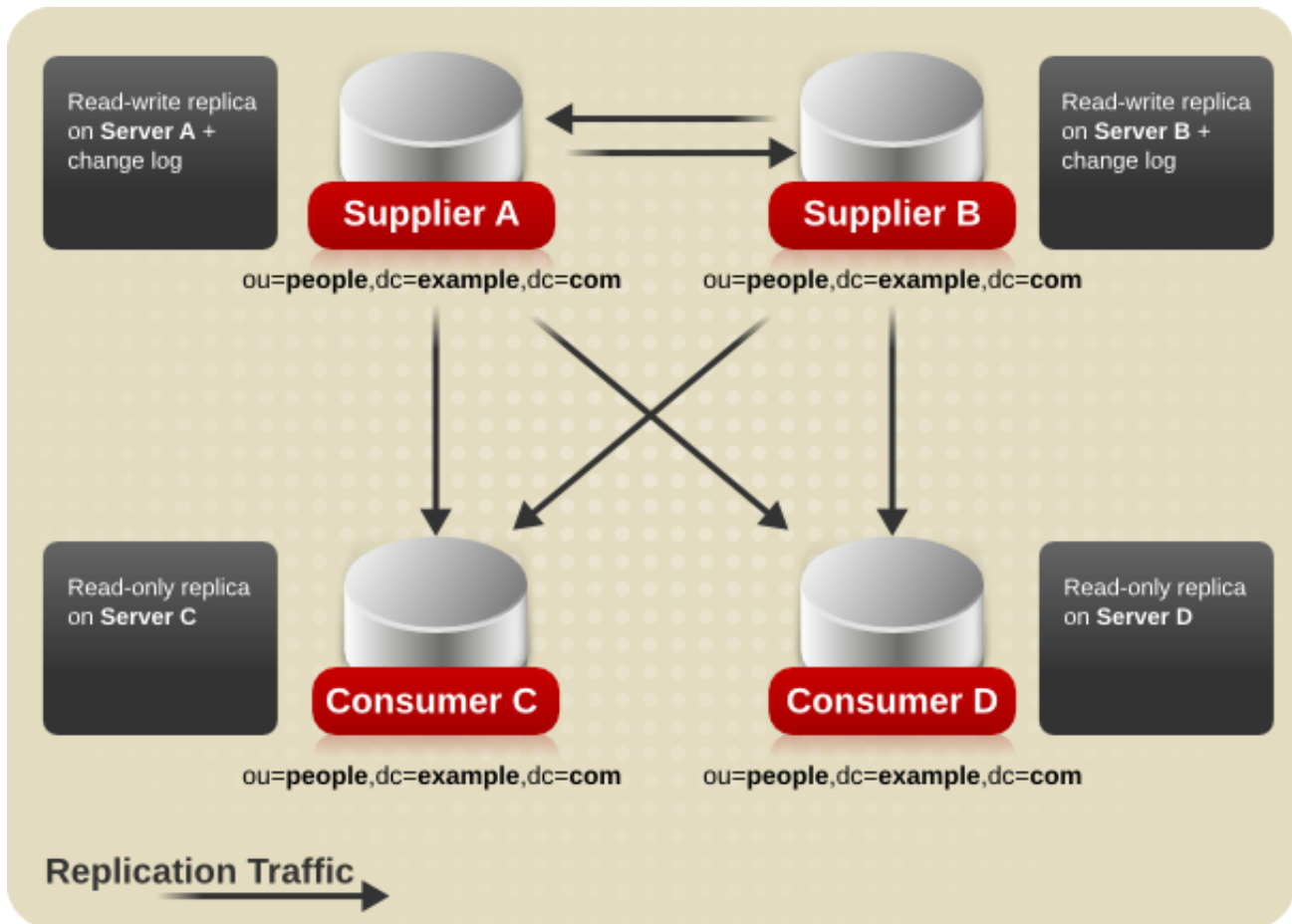


Figure 11.2. Multi-Master Replication (Two Masters)

Figure 11.3, “Multi-Master Replication (Four Masters)” shows a sample of multi-master replication scenario with four supplier servers and eight consumer servers. In this sample setup, each supplier server is configured with ten replication agreements to feed data to two other supplier servers and all eight consumer servers. (The Directory Server can have as many as 20 masters in a multi-master setup.)

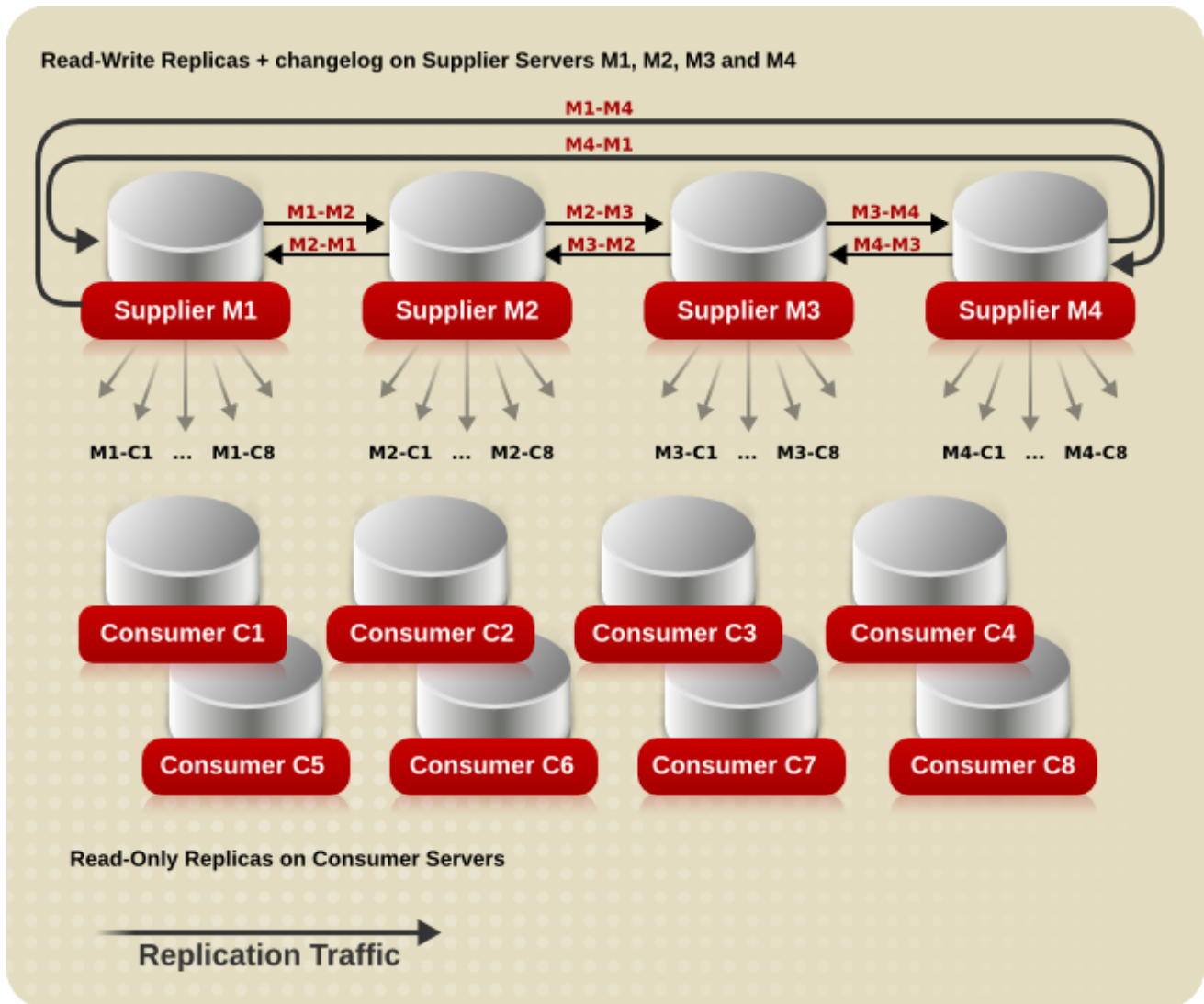


Figure 11.3. Multi-Master Replication (Four Masters)

Multi-master configurations have the following advantages:

- Automatic write failover when one supplier is inaccessible.
- Updates are made on a local supplier in a geographically distributed environment.



NOTE

The speed that replication proceeds depends on the speed of the network. Plan changes and directory configuration accordingly, and realize that changes to one directory may not be quickly replicated to other directories over slow links, such as wide-area networks, in geographically-distributed environments.

For the procedure to set up multi-master replication, see [Section 11.5, “Configuring Multi-Master Replication”](#).

11.2.3. Cascading Replication

In a cascading replication scenario, one server, a *hub*, acts both as a consumer and a supplier. It holds a read-only replica and maintains a changelog, so it receives updates from the supplier server that holds the master copy of the data and, in turn, supplies those updates to the consumer. Cascading replication

is very useful for balancing heavy traffic loads or to keep master servers based locally in geographically-distributed environments.

Figure 11.4, “Cascading Replication” shows an example of a simple cascading replication scenario, though it is possible to create more complex scenarios with several hub servers.

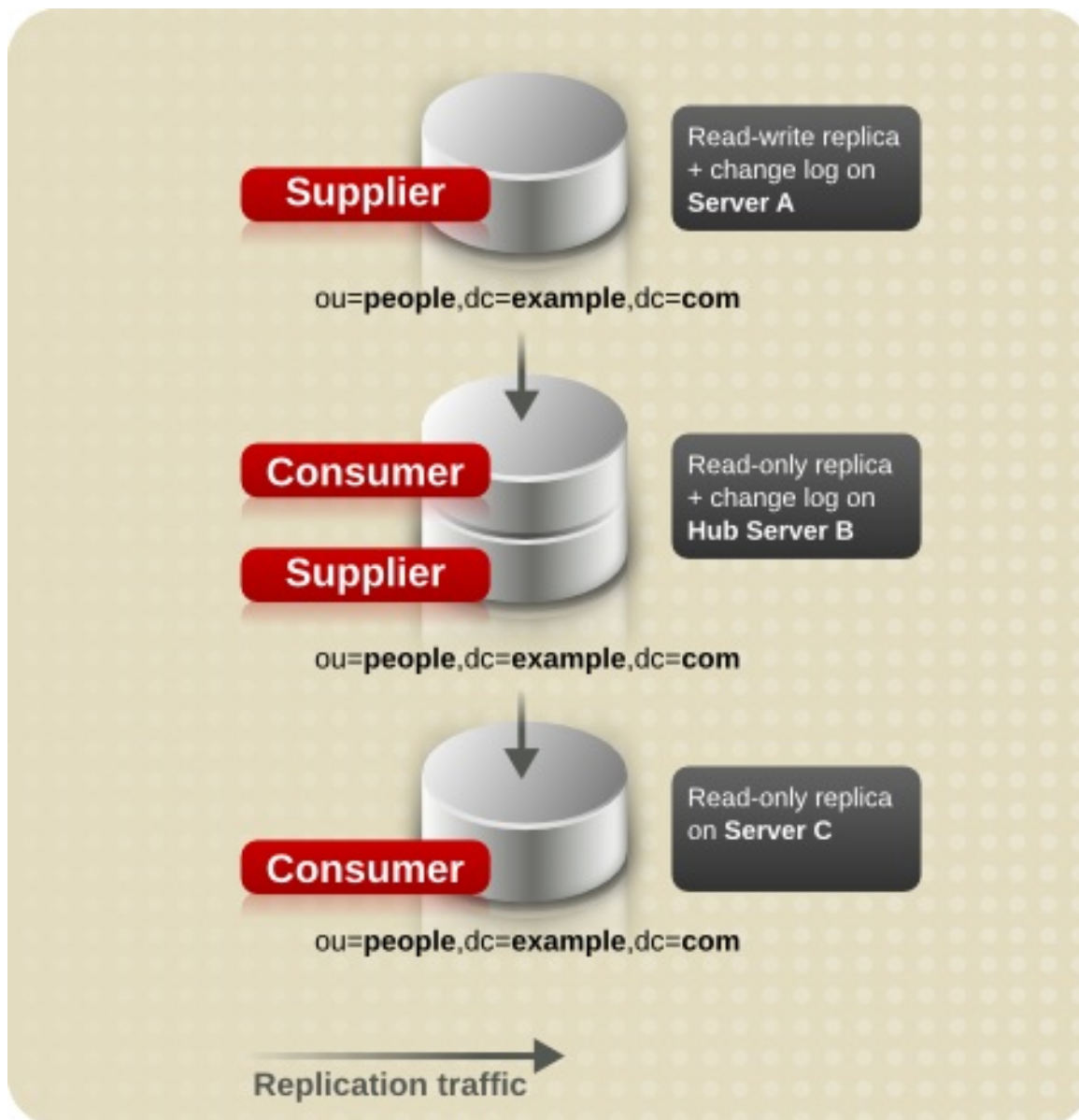
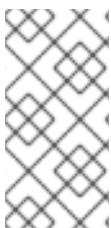


Figure 11.4. Cascading Replication

For information on setting up cascading replication, see [Section 11.6, “Configuring Cascading Replication”](#).



NOTE

Multi-master and cascading replication can be combined. For example, in the multi-master scenario illustrated in [Figure 11.2, “Multi-Master Replication \(Two Masters\)”](#), Server C and Server D could be hub servers that would replicate to any number of consumer servers.

11.3. CREATING THE SUPPLIER BIND DN ENTRY

A critical part of setting up replication is to create the entry, called the replication manager or supplier bind DN entry, that the suppliers use to bind to the consumer servers to perform replication updates.

The supplier bind DN must meet the following criteria:

- It must be unique.
- It must be created on the consumer server (or hub) and *not* on the supplier server.
- It must correspond to an actual entry on the consumer server.
- It must be created on *every* server that receives updates from another server.
- It must not be part of the replicated database for security reasons.
- It must be defined in the replication agreement on the supplier server.
- It must have an idle timeout period set to a high enough limit to allow the initialization process for large databases to complete. Using the ***nsIdleTimeout*** operational attribute allows the replication manager entry to override the global ***nsslapd-idletimeout*** setting.

For example, the entry **cn=Replication Manager,cn=config** can be created under the **cn=config** tree on the consumer server. This would be the supplier bind DN that all supplier servers would use to bind to the consumer to perform replication operations.



NOTE

Avoid creating simple entries under the **cn=config** entry in the **dse.ldif** file. The **cn=cn=config** entry in the simple, flat **dse.ldif** configuration file is not stored in the same highly scalable database as regular entries. As a result, if many entries, and particularly entries that are likely to be updated frequently, are stored under **cn=config**, performance will suffer. However, although Red Hat recommends not storing simple user entries under **cn=config** for performance reasons, it can be useful to store special user entries such as the Directory Manager entry or replication manager (supplier bind DN) entry under **cn=config** since this centralizes configuration information.

On each server that acts as a consumer in replication agreements, create a special entry that the supplier will use to bind to the consumers. Make sure to create the entry with the attributes required by the authentication method specified in the replication agreement.

1. Stop the Directory Server. If the server is not stopped, the changes to the **dse.ldif** file will not be saved. See [Section 1.3, “Starting and Stopping Servers”](#) for more information on stopping the server.
2. Create a new entry, such as **cn=replication manager,cn=config**, in the **dse.ldif** file.
3. Specify a **userPassword** attribute-value pair.
4. Set an ***nsIdleTimeout*** period that gives the replication user a long enough time limit to allow replication initialization on large databases to complete.
5. If password expiration policy is enabled or ever will be enabled, disable it on the replication manager entry to prevent replication from failing due to passwords expiring. To disable the password expiration policy on the **userPassword** attribute, add the

passwordExpirationTime attribute with a value of **20380119031407Z**, which means that the password will never expire.

6. Restart the Directory Server. See [Section 1.3, “Starting and Stopping Servers”](#) for more information on starting the server.

The final entry should resemble [Example 11.1, “Example Supplier Bind DN Entry”](#).

Example 11.1. Example Supplier Bind DN Entry

```
dn: cn=replication manager,cn=config
objectClass: inetorgperson
objectClass: person
objectClass: top
cn: replication manager
sn: RM
userPassword: password
passwordExpirationTime: 20380119031407Z
nsIdleTimeout: 0
```

When configuring a replica as a consumer, use the DN of this entry to define the supplier bind DN.

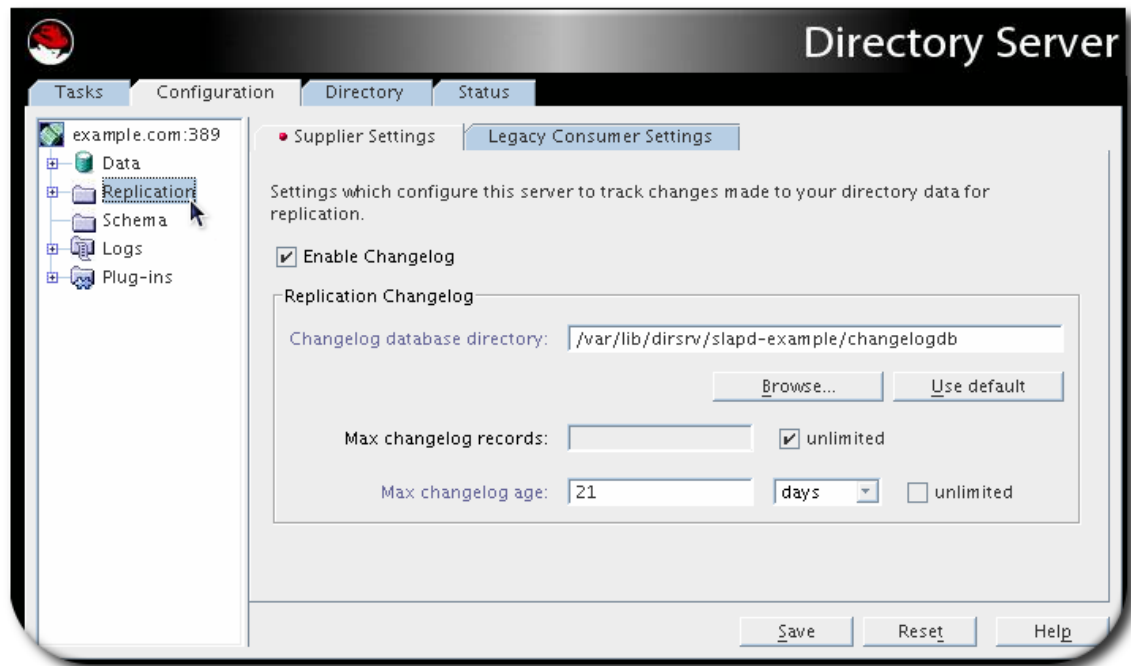
11.4. CONFIGURING SINGLE-MASTER REPLICATION

To set up single-master replication such as the configuration shown in [Figure 11.1, “Single-Master Replication”](#), between supplier Server A, which holds a read-write replica, and the two consumers Server B and Server C, which each hold a read-only replica, there are three major steps:

- [Section 11.4.1, “Configuring the Read-Write Replica on the Supplier Server”](#)
- [Section 11.4.2, “Configuring the Read-Only Replica on the Consumer”](#)
- [Section 11.4.3, “Creating the Replication Agreement”](#)

11.4.1. Configuring the Read-Write Replica on the Supplier Server

1. Specify the supplier settings for the server.
 1. In the Directory Server Console, select the **Configuration** tab.
 2. In the navigation tree, select the **Replication** folder.
 3. In the right-hand side of the window, select the **Supplier Settings** tab.



4. Check the **Enable Changelog** check box.

This activates all of the fields in the pane below that were previously grayed out.

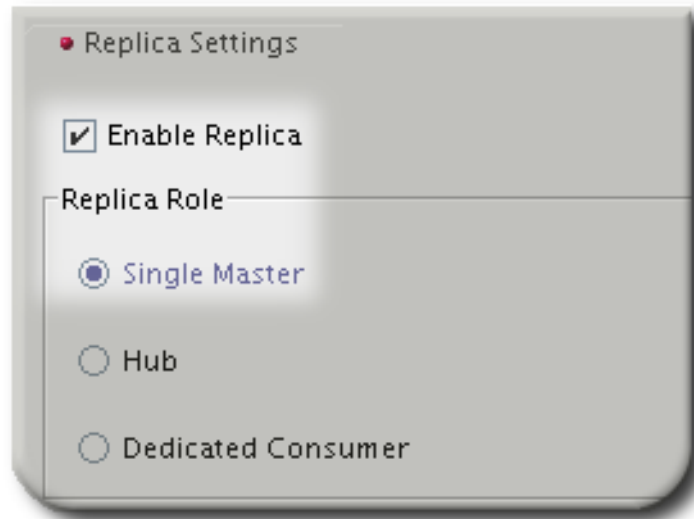
5. Specify a changelog by clicking the **Use default** button, or click the **Browse** button to display a file selector.
6. Set the changelog parameters for the number and age of the log files.
Clear the unlimited check boxes to specify different values.
7. Click **Save**.

2. Specify the replication settings required for a read-write replica.

1. In the navigation tree on the **Configuration** tab, expand the **Replication** node, and highlight the database to replicate.

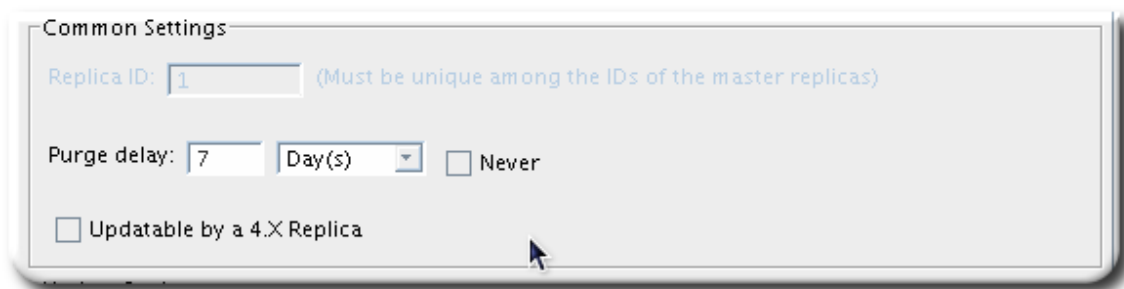
The **Replica Settings** tab opens in the right-hand side of the window.

2. Check the **Enable Replica** check box.
3. In the **Replica Role** section, select the **Single Master** radio button.



4. In the **Common Settings** section, specify a **Replica ID**, which is an integer between **1** and **65534**, inclusive.

The replica ID must be unique for a given suffix, different from any other ID used for read-write replicas on this server and on other servers.



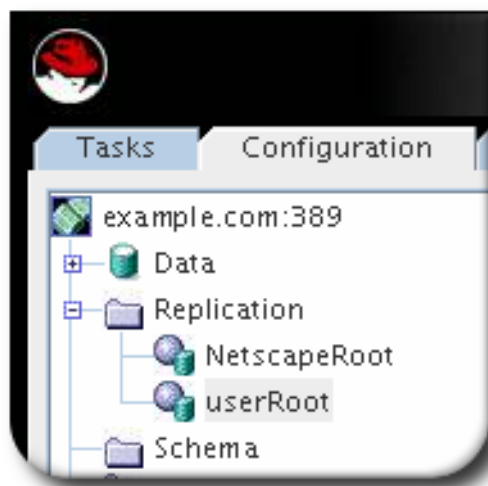
5. In the **Common Settings** section, specify a purge delay in the **Purge delay** field.

The purge delay is how often the state information stored for the replicated entries is deleted.

6. Click **Save**.

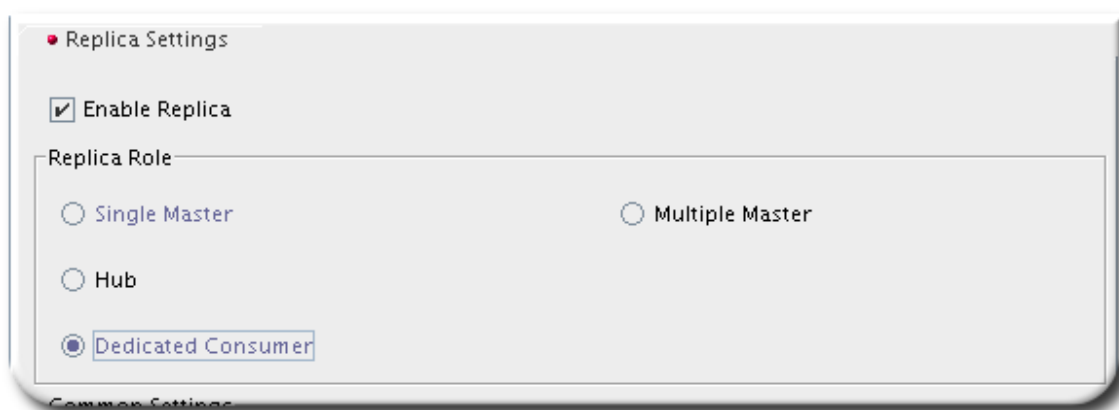
11.4.2. Configuring the Read-Only Replica on the Consumer

1. Create the database for the read-only replica if it does not exist. See [Section 2.1.1, “Creating Suffixes”](#) for instructions on creating suffixes.
2. Create the entry for the supplier bind DN on the consumer server if it does not exist. The supplier bind DN is the special entry that the supplier will use to bind to the consumer. This is described in [Section 11.3, “Creating the Supplier Bind DN Entry”](#).
3. Specify the replication settings required for a read-only replica.
 1. In the Directory Server Console, select the **Configuration** tab.
 2. In the navigation tree, expand the **Replication** folder, and highlight the replica database.



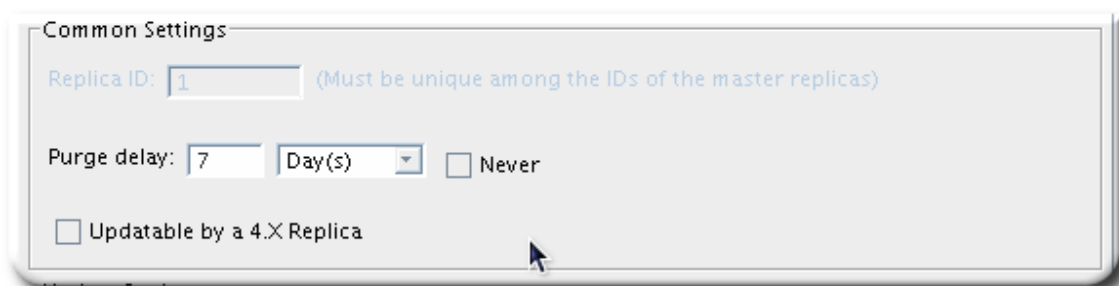
The **Replica Settings** tab for that database opens in the right-hand side of the window.

3. Check the **Enable Replica** check box.

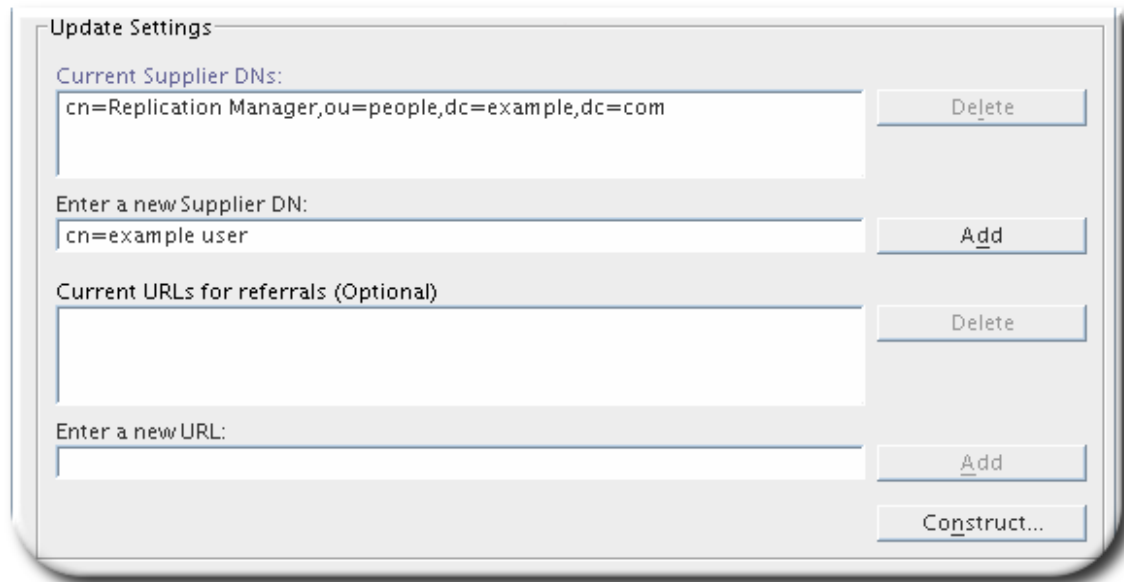


4. In the **Replica Role** section, select the **Dedicated Consumer** radio button.
5. In the **Common Settings** section, specify a purge delay in the **Purge delay** field.

This option indicates how often the state information stored for the replicated entries is purged.



6. In the **Update Settings** section, specify the bind DN that the supplier will use to bind to the replica. Enter the supplier bind DN in the **Enter a new Supplier DN** field, and click **Add**. The supplier bind DN appears in the **Current Supplier DNs** list.



The supplier bind DN should be the entry created in step 2. The supplier bind DN is a privileged user because it is not subject to access control.



NOTE

There can be multiple supplier bind DNs per consumer but only one supplier DN per replication agreement.

7. Specify the URL for any supplier servers to which to refer updates.

By default, all updates are first referred to the supplier servers that are specified here. If no suppliers are set here, updates are referred to the supplier servers that have a replication agreement that includes the current replica.

Automatic referrals assume that clients bind over a regular connection; this has a URL in the form **ldap://hostname:port**. For clients to bind to the supplier using SSL, use this field to specify a referral of the form **ldaps://hostname:port**, where the **s** in **ldaps** indicates a secure connection.



NOTE

It is also possible to use IPv4 or IPv6 addresses instead of the host name.

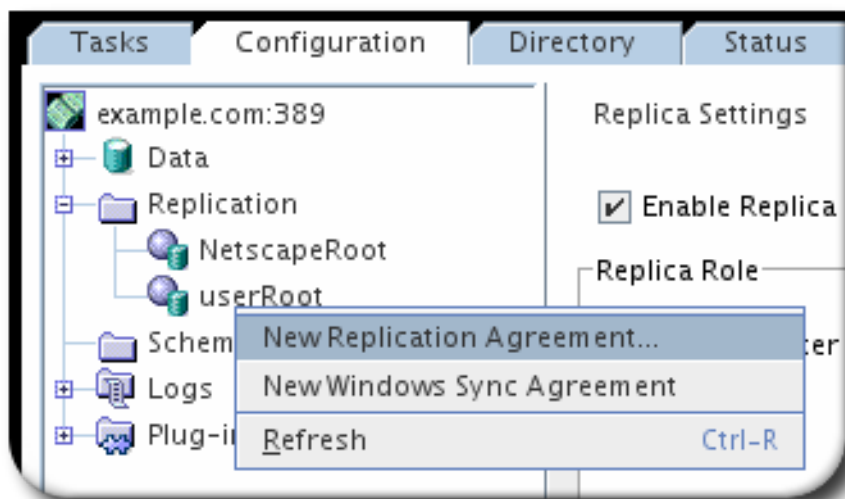
4. Click **Save**.

Repeat these steps for every consumer server in the replication configuration.

11.4.3. Creating the Replication Agreement

Create one replication agreement for each read-only replica. For example, in the scenario illustrated in [Figure 11.1, “Single-Master Replication”](#), Server A has two replication agreements, one for Server B and one for Server C.

1. In the navigation tree of the **Configuration** tab, right-click the database to replicate, and select **New Replication Agreement**.



Alternatively, highlight the database, and select **New Replication Agreement** from the **Object** menu to start the **Replication Agreement Wizard**.

2. In the first screen, fill in a name and description for the replication agreement, and hit **Next**.
3. In the **Source and Destination** screen, fill in the URL (*hostname:port* or *IP_address:port*, with IPv4 or IPv6 addresses) for the consumer and the supplier bind DN and password on that consumer. If the target server is not available, hit in other to fill in the information manually.

Source and Destination

Provide server and content information:

Supplier
example.com:389

Consumer
server2.example.com:389 Other...

Connection
☐ Use LDAP (no encryption)
☐ Use TLS/SSL (TLS/SSL encryption with LDAPS)
☒ Use StartTLS (TLS/SSL encryption with LDAP)

Authentication mechanism:
☐ Server TLS/SSL Certificate (requires TLS/SSL server set up)
☐ SASL/GSSAPI (requires server Kerberos keytab)
☐ SASL/DIGEST-MD5 (SASL user id and password)
☒ Simple (Bind DN/Password)

Bind as: cn=replication manager,cn=config

Password:

Subtree:
dc=example, dc=com

Back Next Cancel Help

- o Unless there is more than one instance of Directory Server configured, by default, there are no consumers available in the drop-down menu.
- o The port listed is the non-SSL port, even if the Directory Server instance is configured to run over SSL. This port number is used only for identification of the Directory Server instance in the Console; it does not specify the actual port number or protocol that is used for replication.
- o If SSL is enabled on the servers, it is possible to select the **Using encrypted SSL connection** radio button for SSL client authentication. Otherwise, fill in the supplier bind DN and password.

**NOTE**

If attribute encryption is enabled, a secure connection *must* be used for the encrypted attributes to be replicated.

4. Select the connection type. There are three options:

- *Use LDAP*. This sets a standard, unencrypted connection.
- *Use TLS/SSL*. This uses a secure connection over the server's secure LDAPS port, such as **636**. This setting is required to use TLS/SSL.
- *Use Start TLS*. This uses Start TLS to establish a secure connection over the server's standard port.

**NOTE**

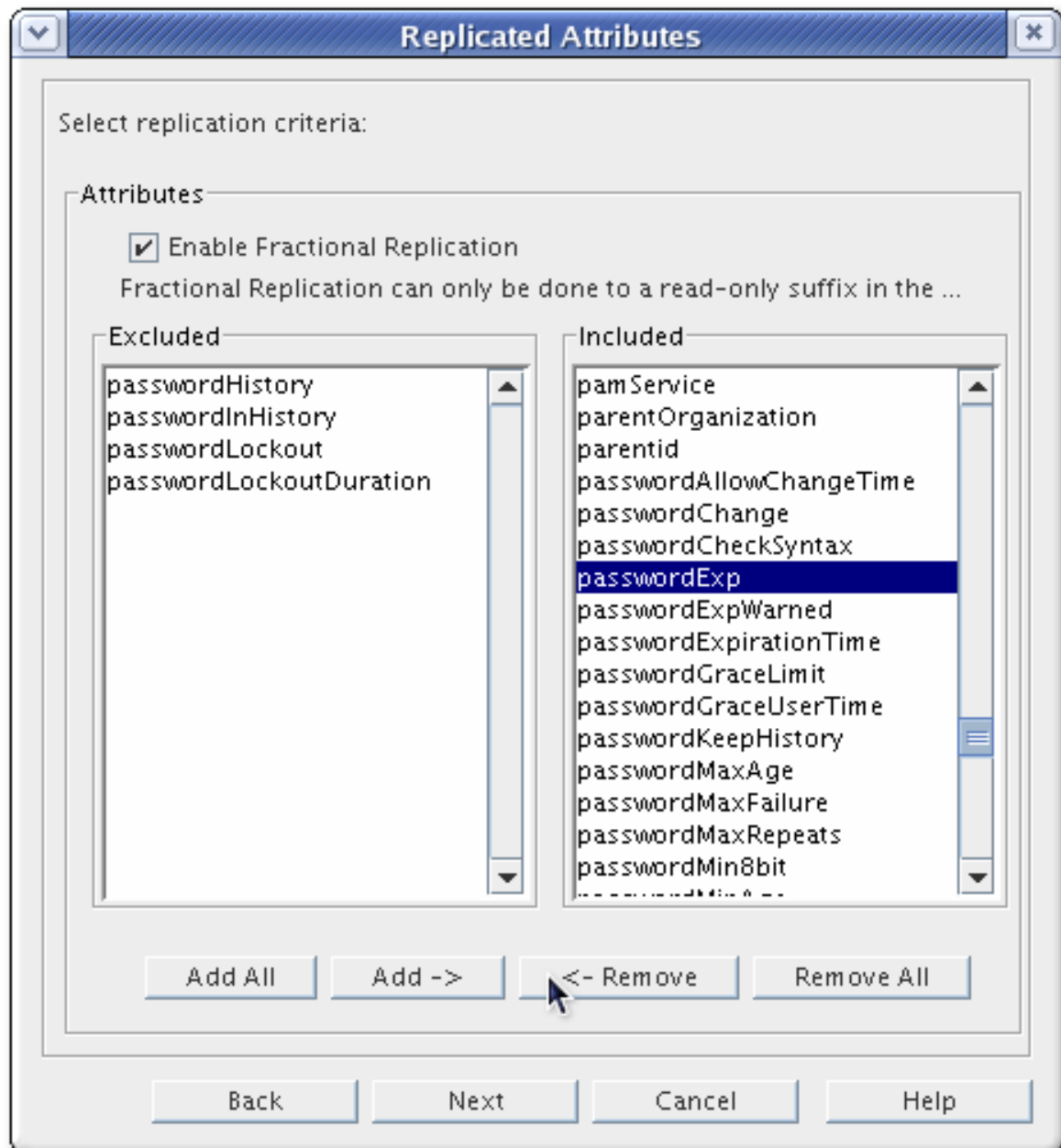
If secure binds are required for simple password authentication ([Section 14.8.1, “Requiring Secure Binds”](#)), then any replication operations will fail unless they occur over a secure connection. Using a secure connection (SSL/TLS and Start TLS connections or SASL authentication) is recommended, anyway.

5. Select the appropriate authentication method and supply the required information. This gives the information that the supplier uses to authenticate and bind to the consumer server to send updates.

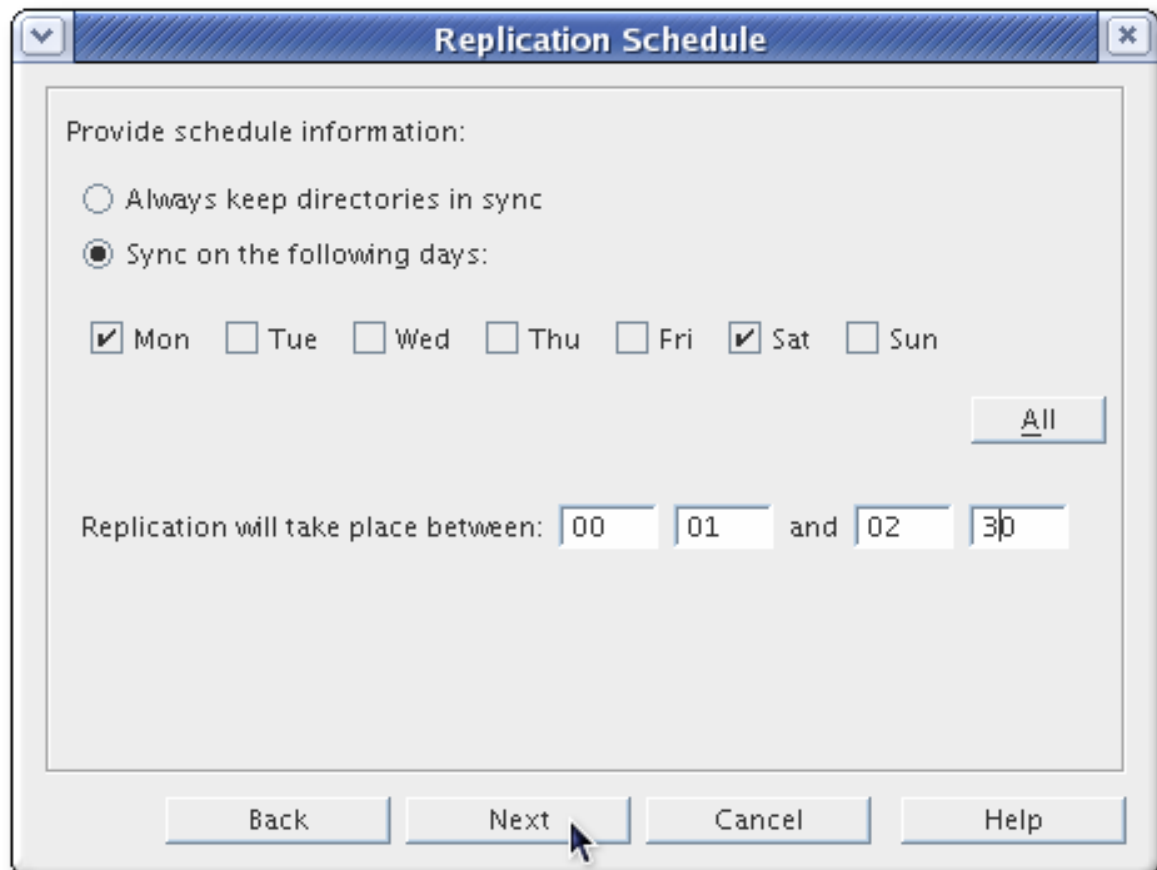
- *Simple* means that the server connects over the standard port with no encryption. The only required information is the bind DN and password for the Replication Manager (which must exist on the consumer server).
- *Server TLS/SSL Certificate* uses the supplier's SSL certificate to authenticate to the consumer server. A certificate must be installed on the supplier for certificate-based authentication, and the consumer server must have certificate mapping configured so that it can map the subject DN in the supplier's certificate to its Replication Manager entry.

Configuring SSL and certificate mapping is described in [Section 7.4, “Setting up TLS/SSL”](#).

- *SASL/DIGEST-MD5*, like simple authentication, requires only the bind DN and password to authenticate. This can run over a standard or SSL/TLS connection.
 - *SASL/GSSAPI* requires the supplier server to have a Kerberos keytab (as in [Section 7.12.2.2, “About the KDC Server and Keytabs”](#)), and the consumer server to have a SASL mapping to map the supplier's principal to the real replication manager entry (as in [Section 7.11.3.1, “Configuring SASL Identity Mapping from the Console”](#)).
6. Fractional replication controls which entry attributes are replicated between servers. By default, all attributes are replicated. To select attributes that will *not* be replicated to the consumer, check the **Enable Fractional Replication** check box. Then, highlight the attribute (or attributes) in the **Included** column on the right, and click **Remove**. All attributes that will not be replicated are listed in the **Excluded** column on the left, as well as in the summary the replication agreement is complete.



7. Set the schedule for when replication runs. By default, replication runs continually.



The image shows a Windows-style dialog box titled "Replication Schedule". It has a standard title bar with a minimize button, a maximize button, and a close button. The main area contains the text "Provide schedule information:" followed by two radio buttons. The first radio button is labeled "Always keep directories in sync" and is unselected. The second radio button is labeled "Sync on the following days:" and is selected. Below the second radio button are seven checkboxes for the days of the week: Mon, Tue, Wed, Thu, Fri, Sat, and Sun. The "Mon" and "Sat" checkboxes are checked, while the others are unchecked. To the right of these checkboxes is a button labeled "All". Below the checkboxes, the text "Replication will take place between:" is followed by four input fields. The first two fields contain "00" and "01", and the last two fields contain "02" and "30". The "and" text is placed between the second and third input fields. At the bottom of the dialog box are four buttons: "Back", "Next", "Cancel", and "Help". A mouse cursor is pointing at the "Next" button.

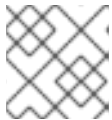
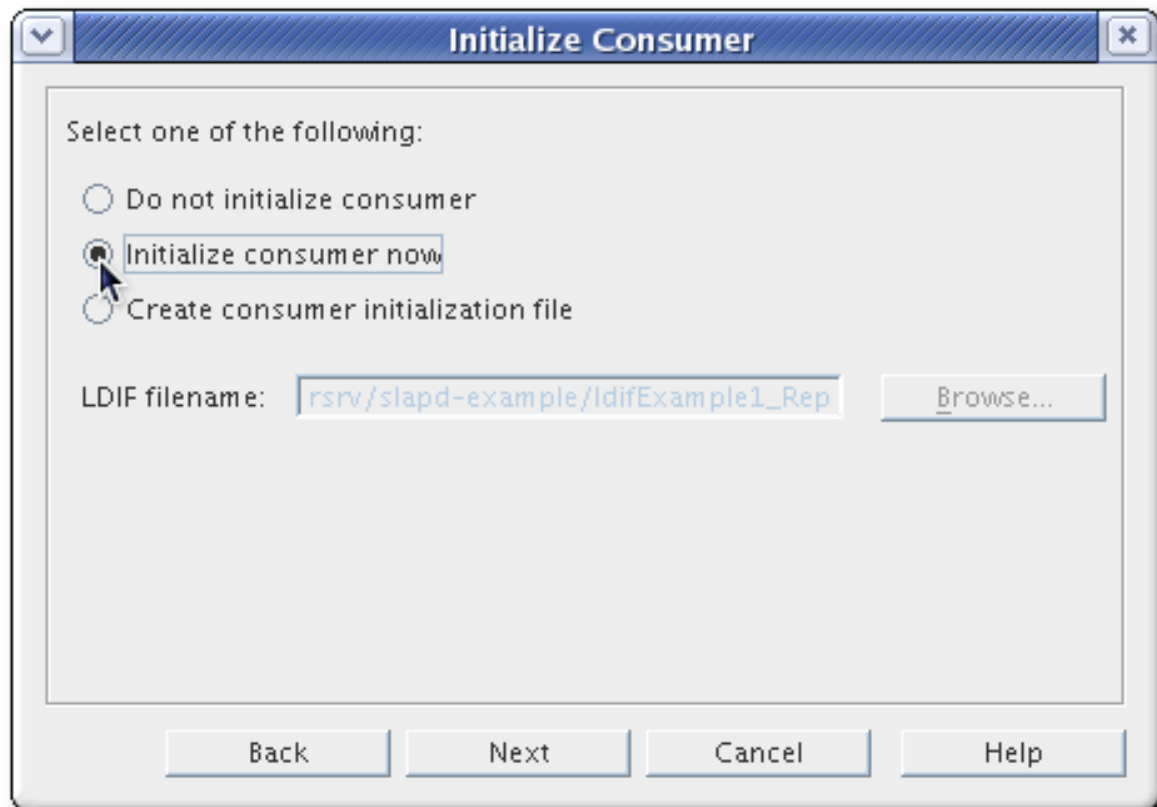


NOTE

The replication schedule cannot cross midnight (0000). So, it is possible to set a schedule that begins at 0001 and ends at 2359 on the same day, but it is not possible to set one that begins at 2359 on one day and ends at 0001 on the next.

Hit **Next**.

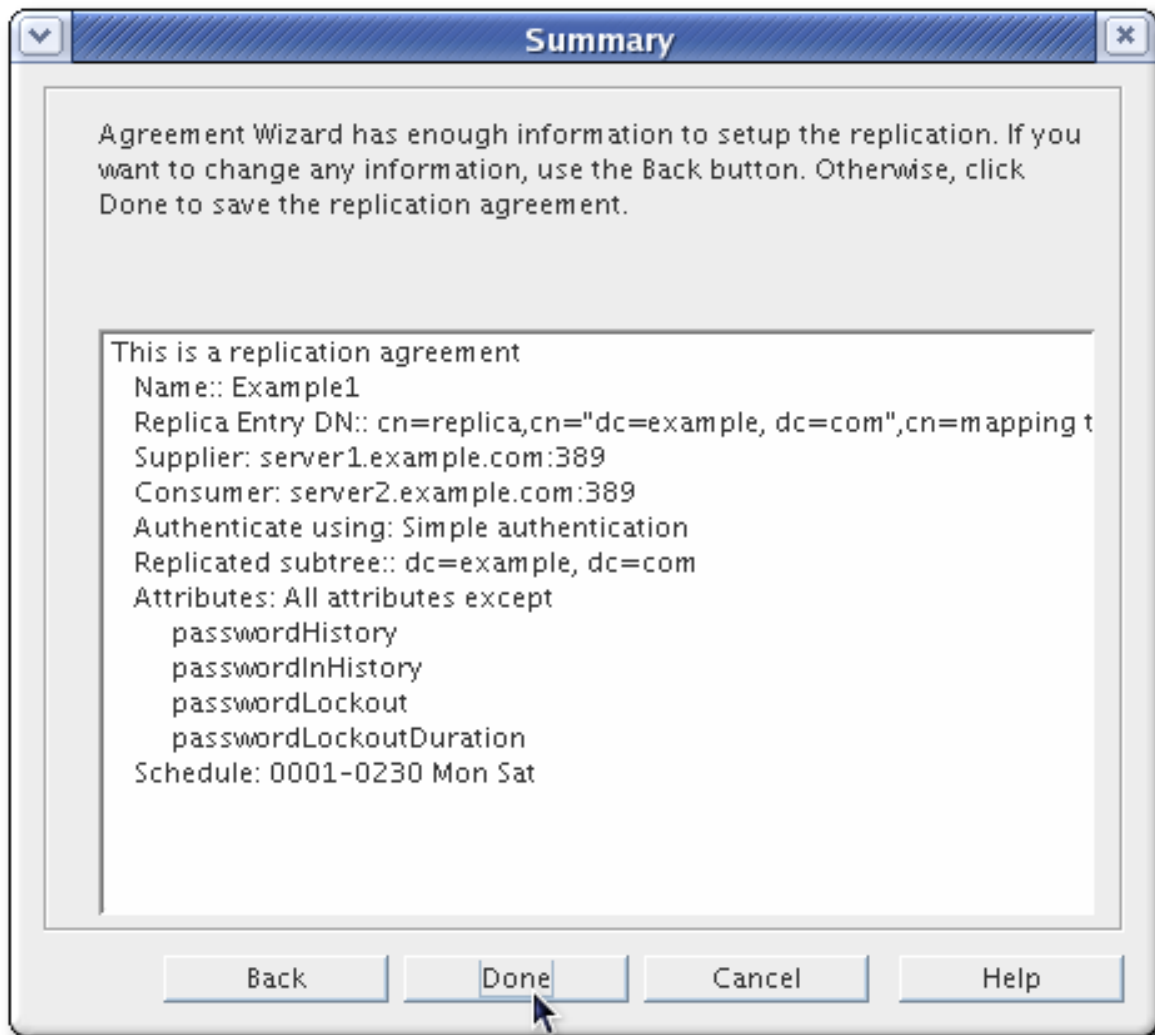
8. Set when the consumer is initialized. *Initializing* a consumer manually copies all data over from the supplier to the consumer. The default is to create an initialization file (an LDIF of all supplier data) so that the consumer can be initialized later. It is also possible to initialize the consumer as soon as the replication agreement is completed or not at all. For information on initializing consumers, see [Section 11.15, "Initializing Consumers"](#).

**NOTE**

Replication *will not* begin until the consumer is initialized.

Hit **Next**.

9. The final screen shows the settings for the replication agreement, as it will be included in the **dse.ldif** file. Hit **Done** to save the agreement.



The replication agreement is set up.



NOTE

After creating a replication agreement, the connection type (SSL or non-SSL) cannot be changed because LDAP and LDAPS connections use different ports. To change the connection type, re-create the replication agreement.

11.5. CONFIGURING MULTI-MASTER REPLICATION

In a multi-master configuration, many suppliers can accept updates, synchronize with each other, and update all consumers. The consumers can send referrals for updates to all masters.

Directory Server supports 20-way multi-master replication, meaning that there can be up to 20 masters (and an unlimited number of hub suppliers) in a single replication scenario. Directory Server allows an unlimited number of consumers.

To set up multi-master replication, set up all of the consumers first, then set up the suppliers, and last, initialize all of the databases.

- [Section 11.5.1, “Configuring the Read-Write Replicas on the Supplier Servers”](#)
- [Section 11.5.2, “Configuring the Read-Only Replicas on the Consumer Servers”](#)

- [Section 11.5.3, “Setting up the Replication Agreements”](#)
- [Section 11.5.4, “Preventing Monopolization of the Consumer in Multi-Master Replication”](#)



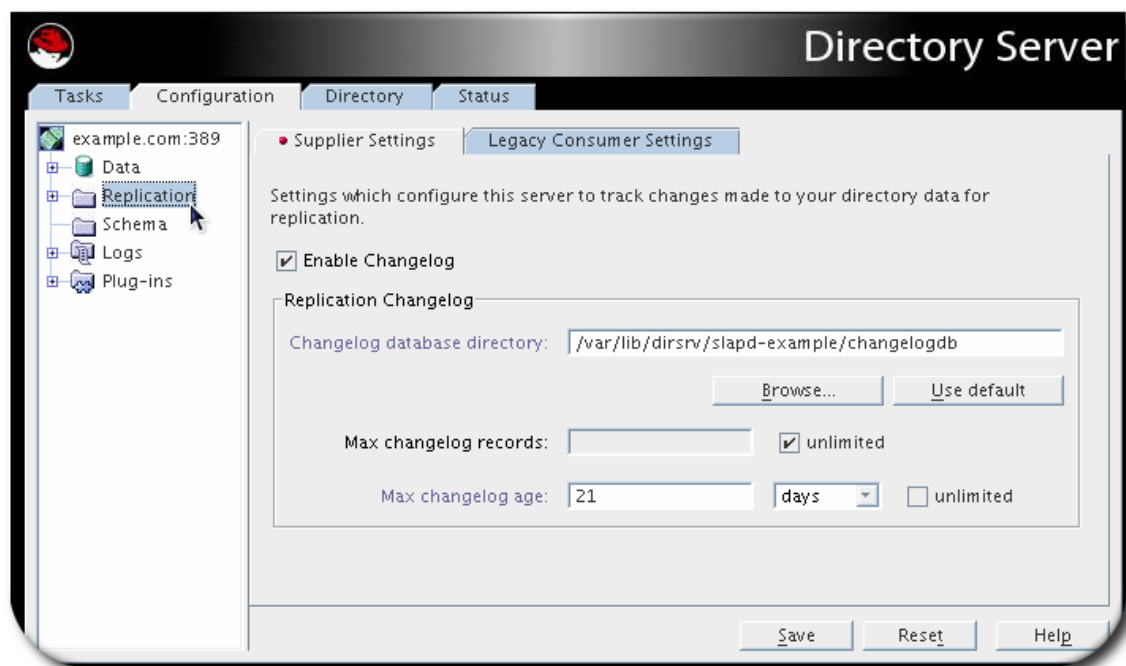
NOTE

More than 10 databases running with replication or more than on a supplier can cause performance degradation. To support that many consumers, introduce hub replicas between the suppliers and consumers. See [Section 11.6, “Configuring Cascading Replication”](#).

11.5.1. Configuring the Read-Write Replicas on the Supplier Servers

Set up each supplier server. The first supplier configured should be used to initialize the other suppliers in the multi-master replication environment.

1. Specify the supplier settings for the server.
 1. In the Directory Server Console, select the **Configuration** tab.
 2. In the navigation tree, select the **Replication** folder.
 3. In the right-hand side of the window, select the **Supplier Settings** tab.



4. Check the **Enable Changelog** check box.

This activates all of the fields in the pane below that were previously grayed out.

5. Specify a changelog by clicking the **Use default** button, or click the **Browse** button to display a file selector.
6. Set the changelog parameters for the number and age of the log files.

Clear the unlimited check boxes to specify different values.

7. Click **Save**.
2. Create the entry for the supplier bind DN on the consumer server if it does not exist. This is the special entry that the other suppliers will use to bind to this supplier, as in other supplier-consumer relationships. This is described in [Section 11.3, “Creating the Supplier Bind DN Entry”](#).



NOTE

For multi-master replication, it is necessary to create this supplier bind DN on the supplier servers as well as the consumers because the suppliers act as both consumer and supplier to the other supplier servers.

3. Specify the replication settings for the multi-mastered read-write replica.
 1. In the Directory Server Console, select the **Configuration** tab.
 2. In the navigation tree, expand the **Replication** folder, and highlight the replica database.

The **Replica Settings** tab for that database opens in the right-hand side of the window.

Replica Settings

☒ Enable Replica

Replica Role

☐ Single Master ☒ Multiple Master

☐ Hub

☐ Dedicated Consumer

3. Check the **Enable Replica** check box.
4. In the **Replica Role** section, select the **Multiple Master** radio button.
5. In the **Common Settings** section, specify a **Replica ID**, which is an integer between **1** and **65534**, inclusive.

Common Settings

Replica ID: (Must be unique among the IDs of the master replicas)

Purge delay: Day(s) ☐ Never

☐ Updatable by a 4.X Replica

The replica ID must be unique for a given suffix, different from any other ID used for read-write replicas on this server and on other servers.

6. In the **Common Settings** section, specify a purge delay in the **Purge delay** field.

The purge delay is how often the state information stored for the replicated entries is deleted.

7. In the **Update Settings** section, specify the bind DN that the supplier will use to bind to the replica. Enter the supplier bind DN in the **Enter a new Supplier DN** field, and click **Add**. The supplier bind DN appears in the **Current Supplier DNs** list.

The screenshot shows a window titled "Update Settings". It contains several sections:

- Current Supplier DNs:** A list box containing the entry "cn=Replication Manager,ou=people,dc=example,dc=com". To the right of the list is a "Delete" button.
- Enter a new Supplier DN:** A text input field containing "cn=example user". To the right is an "Add" button.
- Current URLs for referrals (Optional):** An empty list box. To the right is a "Delete" button.
- Enter a new URL:** An empty text input field. To the right are "Add" and "Construct..." buttons.

The supplier bind DN should be the entry created in step 2. The supplier bind DN is a privileged user because it is not subject to access control in the replicated database.



NOTE

There can be multiple supplier bind DNs per consumer but only one supplier DN per replication agreement.

8. Specify the LDAP URL (*ldap://hostname:port* or *ldap://IP_address:port*, with IPv4 or IPv6 addresses) for any supplier servers to which to refer updates, such as the other suppliers in the multi-master replication set. Only specify the URL for the supplier server.

For clients to bind using SSL, specify a URL beginning with **ldaps://**.

9. Click **Save**.

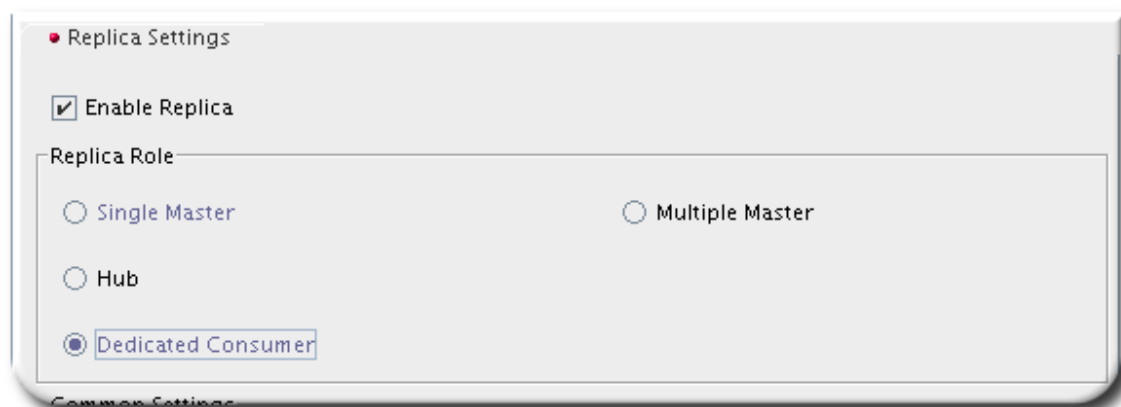
11.5.2. Configuring the Read-Only Replicas on the Consumer Servers

First, configure *every* consumer before creating any replication agreements.

1. Create the database for the read-only replica if it does not exist. See [Section 2.1.1, “Creating Suffixes”](#) for instructions on creating suffixes.
2. Create the entry for the supplier bind DN on the consumer server if it does not exist. The supplier bind DN is the special entry that the supplier will use to bind to the consumer. This is described in [Section 11.3, “Creating the Supplier Bind DN Entry”](#).
3. Specify the replication settings required for a read-only replica.

1. In the Directory Server Console, select the **Configuration** tab.
2. In the navigation tree, expand the **Replication** folder, and highlight the replica database.

The **Replica Settings** tab for that database opens in the right-hand side of the window.



Replica Settings

☒ Enable Replica

Replica Role

☐ Single Master ☐ Multiple Master

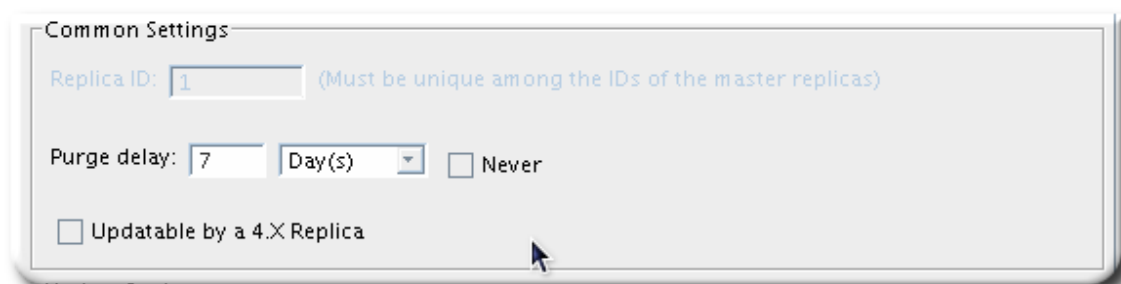
☐ Hub

☒ Dedicated Consumer

Common Settings

3. Check the **Enable Replica** check box.
4. In the **Replica Role** section, select the **Dedicated Consumer** radio button.
5. In the **Common Settings** section, specify a purge delay in the **Purge delay** field.

This option indicates how often the state information stored for the replicated entries is purged.



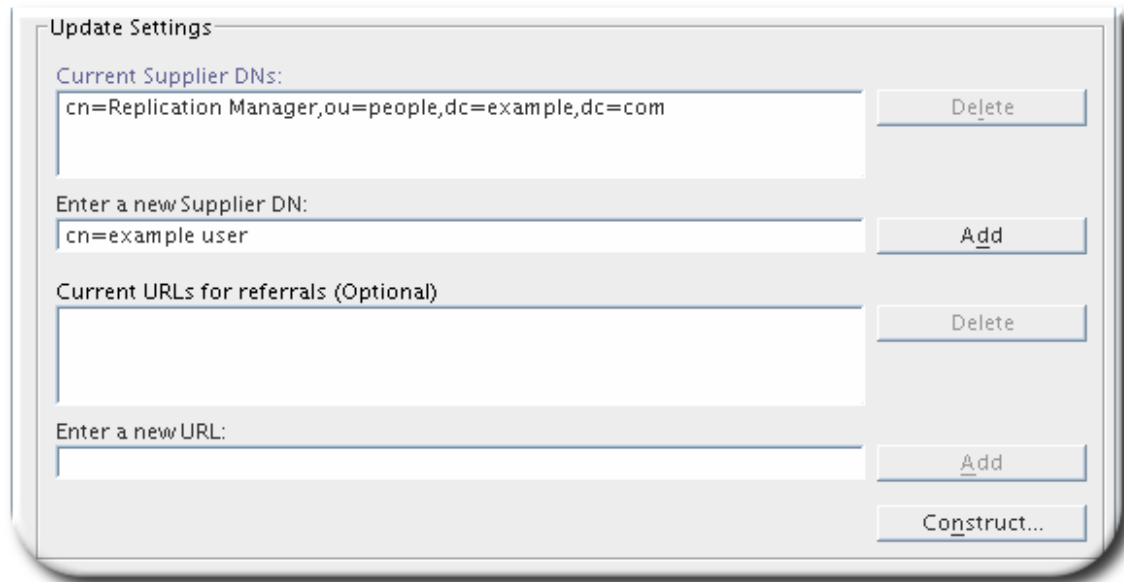
Common Settings

Replica ID: (Must be unique among the IDs of the master replicas)

Purge delay: Day(s) ☐ Never

☐ Updatable by a 4.X Replica

6. In the **Update Settings** section, specify the bind DN that the supplier will use to bind to the replica. Enter the supplier bind DN in the **Enter a new Supplier DN** field, and click **Add**. The supplier bind DN appears in the **Current Supplier DNs** list.



The supplier bind DN should be the entry created in step 2. The supplier bind DN is a privileged user because it is not subject to access control in the replicated database.



NOTE

There can be multiple supplier bind DNs per consumer but only one supplier DN per replication agreement.

7. Specify the URL for any supplier servers to which to refer updates.

By default, all updates are first referred to the supplier servers that are specified here. If no suppliers are set here, updates are referred to the supplier servers that have a replication agreement that includes the current replica.

Automatic referrals assume that clients bind over a regular connection; this has a URL in the form **ldap://hostname:port**. For clients to bind to the supplier using SSL, use this field to specify a referral of the form **ldaps://hostname:port**, where the **s** in **ldaps** indicates a secure connection.



NOTE

It is also possible to use IPv4 or IPv6 addresses instead of the host name.

4. Click **Save**.

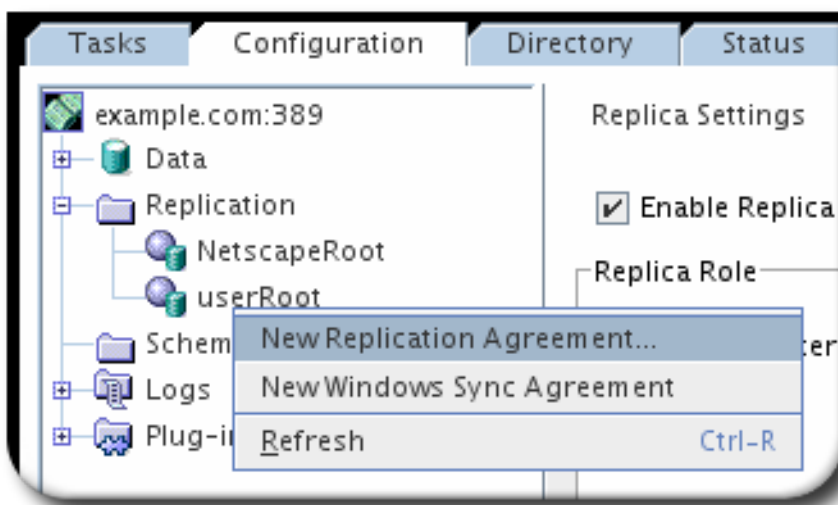
Repeat these steps for every consumer server in the replication configuration.

11.5.3. Setting up the Replication Agreements

NOTE

1. First set up replication agreements on a single supplier, the data master, between the other multi-master suppliers, and initialize all of the other suppliers.
2. Then create replication agreements for all other suppliers in the multi-master replication set, but *do not* reinitialize any of the suppliers.
3. Then create replication agreements for all of the consumers from the single data master, and initialize the consumers.
4. Then create replication agreements for all of the consumers from for all of the other suppliers, but *do not* reinitialize any of the consumers.

1. In the navigation tree of the **Configuration** tab, right-click the database to replicate, and select **New Replication Agreement**.



Alternatively, highlight the database, and select **New Replication Agreement** from the **Object** menu to start the **Replication Agreement Wizard**.

2. In the first screen, fill in a name and description for the replication agreement, and hit **Next**.
3. In the **Source and Destination** screen, fill in the URL (*hostname:port* or *IP_address:port*, with IPv4 or IPv6 addresses) for the consumer and the supplier bind DN and password on that consumer. If the target server is not available, hit in other to fill in the information manually.

Source and Destination

Provide server and content information:

Supplier
example.com:389

Consumer
server2.example.com:389 Other...

Connection
☐ Use LDAP (no encryption)
☐ Use TLS/SSL (TLS/SSL encryption with LDAPS)
☒ Use StartTLS (TLS/SSL encryption with LDAP)

Authentication mechanism:
☐ Server TLS/SSL Certificate (requires TLS/SSL server set up)
☐ SASL/GSSAPI (requires server Kerberos keytab)
☐ SASL/DIGEST-MD5 (SASL user id and password)
☒ Simple (Bind DN/Password)

Bind as: cn=replication manager,cn=config

Password:

Subtree:
dc=example, dc=com

Back Next Cancel Help

- o Unless there is more than one instance of Directory Server configured, by default, there are no consumers available in the drop-down menu. The server URL can be entered manually, in the format *hostname:port* or *IP_address:port*, with IPv4 or IPv6 addresses.
- o The port listed is the non-SSL port, even if the Directory Server instance is configured to run over SSL. This port number is used only for identification of the Directory Server instance in the Console; it does not specify the actual port number or protocol that is used for replication.
- o If SSL is enabled on the servers, it is possible to select the **Using encrypted SSL connection** radio button for SSL client authentication. Otherwise, fill in the supplier bind DN and password.

**NOTE**

If attribute encryption is enabled, a secure connection is required for the encrypted attributes to be replicated.

4. Select the connection type. There are three options:

- *Use LDAP.* This sets a standard, unencrypted connection.
- *Use TLS/SSL.* This uses a secure connection over the server's secure LDAPS port, such as **636**. This setting is required to use TLS/SSL.
- *Use Start TLS.* This uses Start TLS to establish a secure connection over the server's standard port.

**NOTE**

If secure binds are required for simple password authentication ([Section 14.8.1, “Requiring Secure Binds”](#)), then any replication operations will fail unless they occur over a secure connection. Using a secure connection (SSL/TLS and Start TLS connections or SASL authentication) is recommended, anyway.

5. Select the appropriate authentication method and supply the required information. This gives the information that the supplier uses to authenticate and bind to the consumer server to send updates.

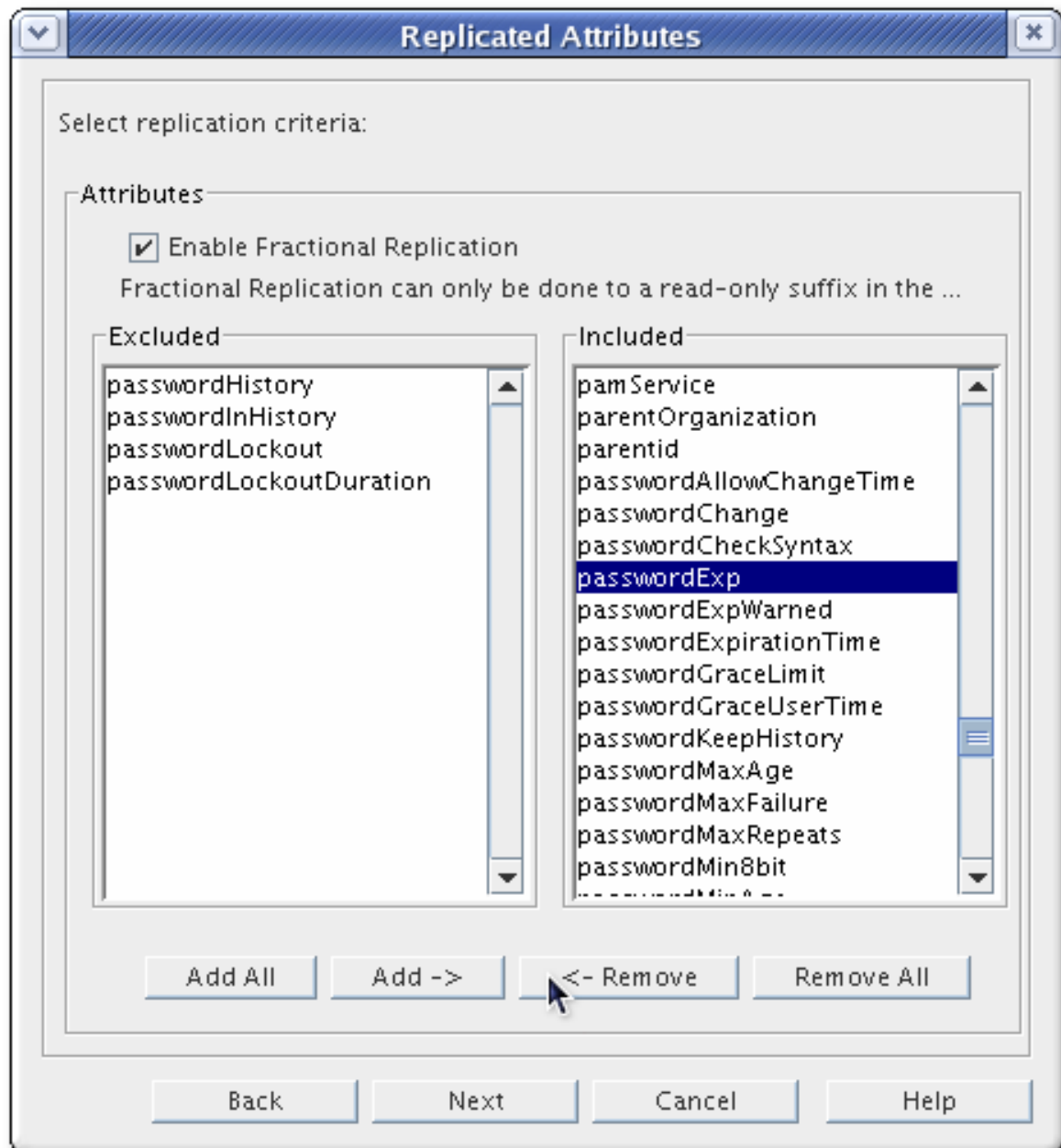
- *Simple* means that the server connects over the standard port with no encryption. The only required information is the bind DN and password for the Replication Manager (which must exist on the consumer server).
- *Server TLS/SSL Certificate* uses the supplier's SSL certificate to authenticate to the consumer server. A certificate must be installed on the supplier for certificate-based authentication, and the consumer server must have certificate mapping configured so that it can map the subject DN in the supplier's certificate to its Replication Manager entry.

Configuring SSL and certificate mapping is described in [Section 7.4, “Setting up TLS/SSL”](#).

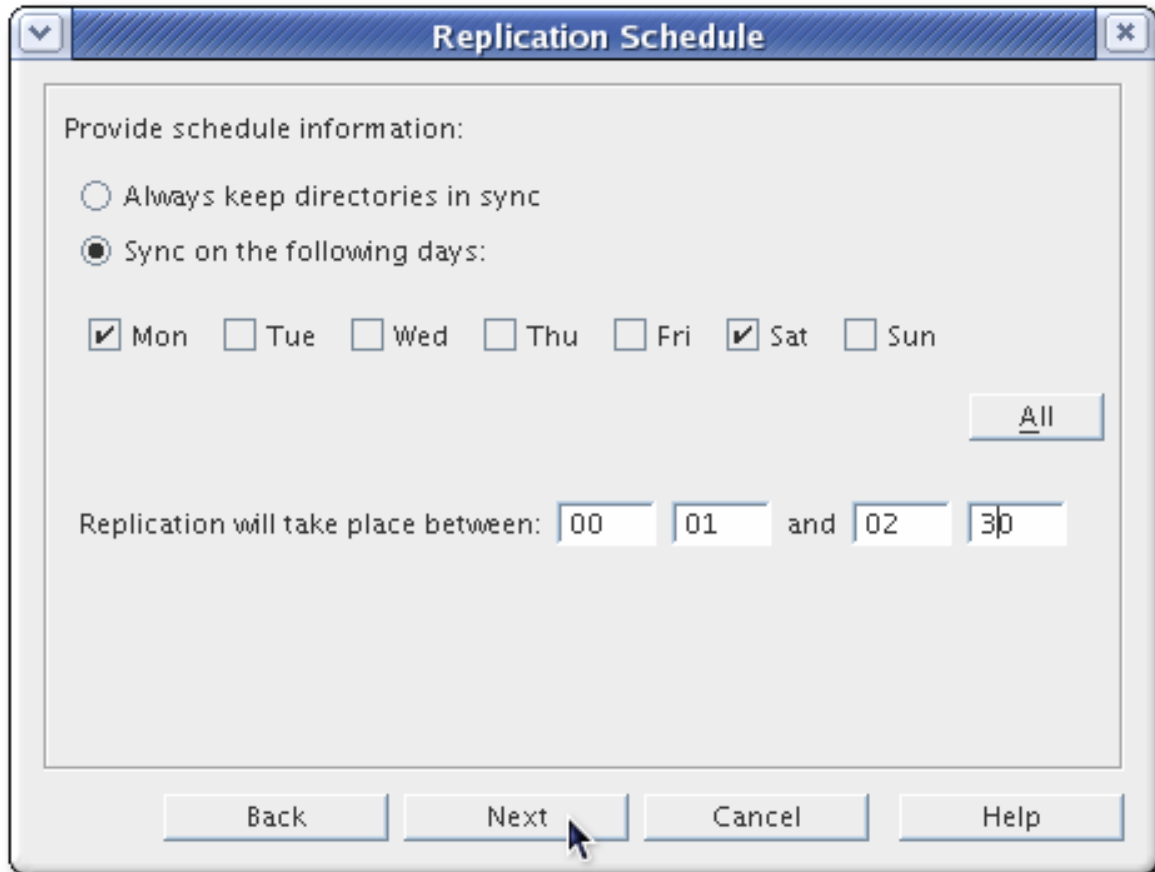
- *SASL/DIGEST-MD5*, like simple authentication, requires only the bind DN and password to authenticate. This can run over a standard or SSL/TLS connection.
- *SASL/GSSAPI* requires the supplier server to have a Kerberos keytab (as in [Section 7.12.2.2, “About the KDC Server and Keytabs”](#)), and the consumer server to have a SASL mapping to map the supplier's principal to the real replication manager entry (as in [Section 7.11.3.1, “Configuring SASL Identity Mapping from the Console”](#)).

6. Hit **Next**.

7. Fractional replication controls which entry attributes are replicated between servers. By default, all attributes are replicated. To select attributes that will *not* be replicated to the consumer, check the **Enable Fractional Replication** check box. Then, highlight the attribute (or attributes) in the **Included** column on the right, and click **Remove**. All attributes that will not be replicated are listed in the **Excluded** column on the left, as well as in the summary the replication agreement is complete.



8. Set the schedule for when replication runs. By default, replication runs continually.



The image shows a 'Replication Schedule' dialog box. It has a title bar with a close button. Inside, it says 'Provide schedule information:'. There are two radio buttons: 'Always keep directories in sync' (unselected) and 'Sync on the following days:' (selected). Below the second radio button are checkboxes for days of the week: Mon (checked), Tue (unchecked), Wed (unchecked), Thu (unchecked), Fri (unchecked), Sat (checked), and Sun (unchecked). To the right of these checkboxes is an 'All' button. Below this section, it says 'Replication will take place between:' followed by four input fields: '00', '01', 'and', '02', and '30'. At the bottom are four buttons: 'Back', 'Next' (with a mouse cursor over it), 'Cancel', and 'Help'.

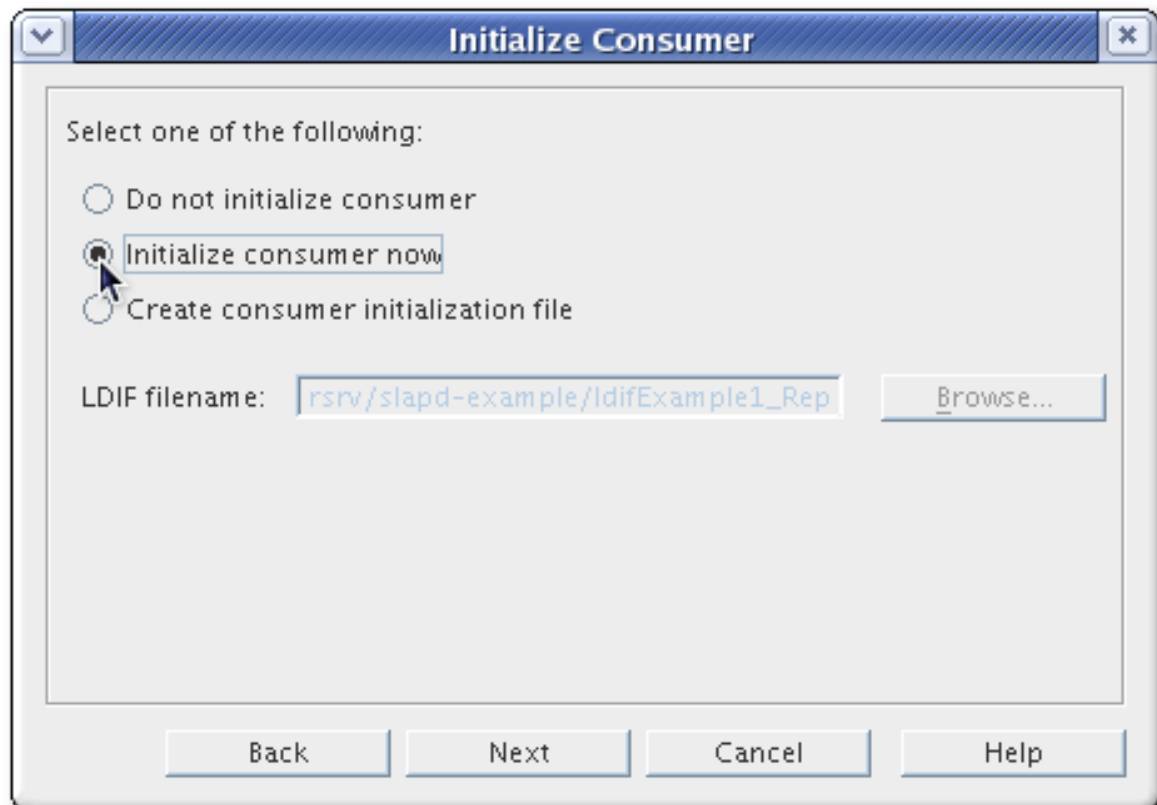


NOTE

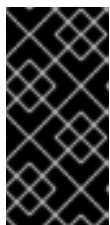
The replication schedule cannot cross midnight (**0000**). So, it is possible to set a schedule that begins at **0001** and ends at **2359** on the same day, but it is not possible to set one that begins at **2359** on one day and ends at **0001** on the next.

Hit **Next**.

9. Set when the consumer is initialized. *Initializing* a consumer manually copies all data over from the supplier to the consumer. The default is to create an initialization file (an LDIF of all supplier data) so that the consumer can be initialized later. It is also possible to initialize the consumer as soon as the replication agreement is completed or not at all. For information on initializing consumers, see [Section 11.15, “Initializing Consumers”](#). For multi-master replication, consider the following:
 - Ensure one supplier has the complete set of data to replicate to the other suppliers. Use this one supplier to initialize the replica on all other suppliers in the multi-master replication set.
 - Initialize the replicas on the consumer servers from any of the multi-master suppliers.
 - Do not try to *reinitialize* the servers when the replication agreements are set up. For example, do not initialize server1 from server2 if server2 has already been initialized from server1. In this case, select **Do not initialize consumer**.

**NOTE**

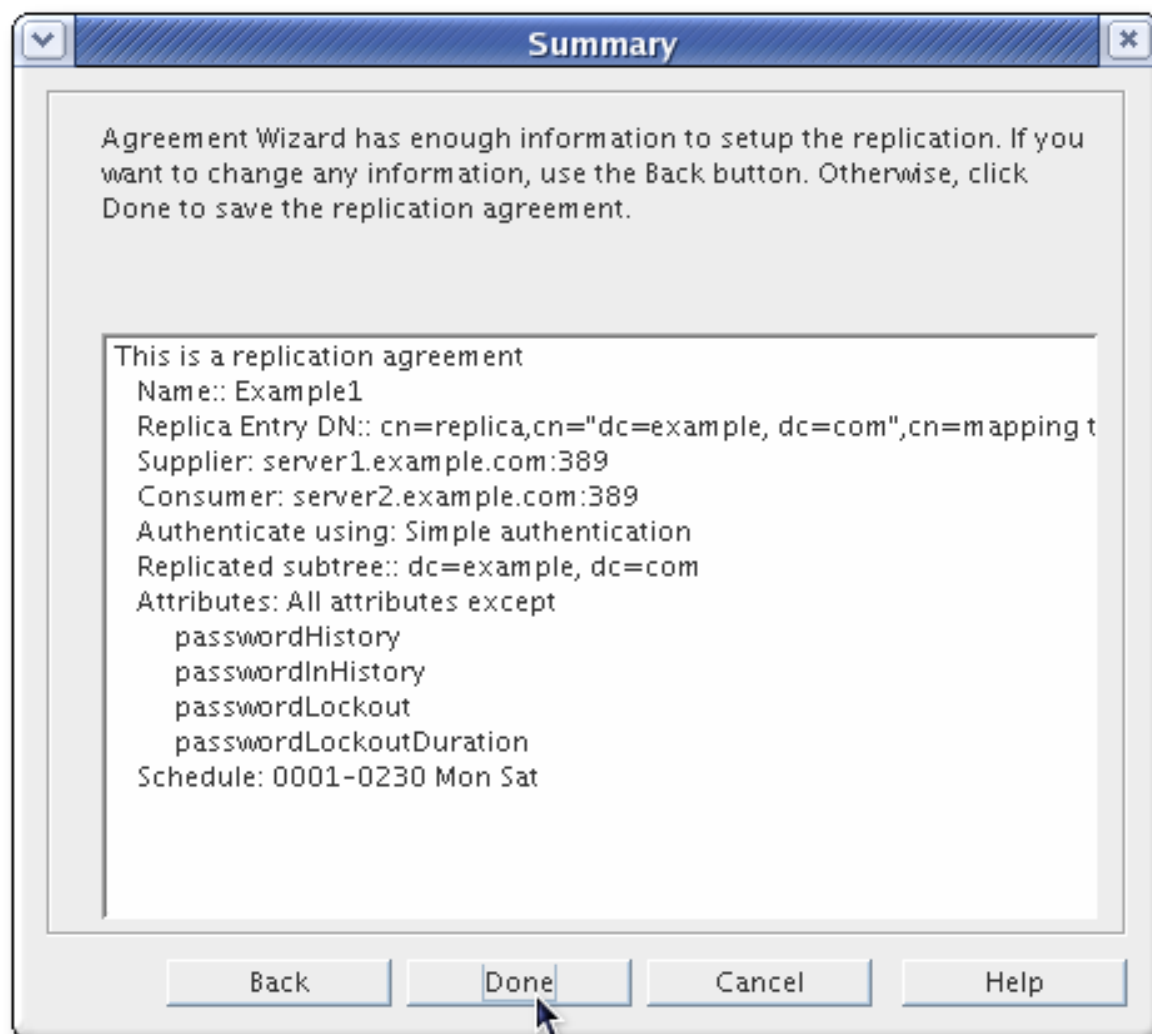
Replication *will not* begin until the consumer is initialized.

**IMPORTANT**

For multi-master replication, be sure that consumers are only initialized **once**, by one supplier. When checking the replication status, be sure to check the replication agreement entry, on the appropriate supplier, which was used to initialize the consumer.

Hit **Next**.

10. The final screen shows the settings for the replication agreement, as it will be included in the **dse.ldif** file. Hit **Done** to save the agreement.



The replication agreement is set up.



NOTE

At the end of this procedure, all supplier servers will have mutual replication agreements, which means that they can accept updates from each other.



NOTE

After creating a replication agreement, the connection type (SSL or non-SSL) cannot be changed because LDAP and LDAPS connections use different ports. To change the connection type, re-create the replication agreement.

11.5.4. Preventing Monopolization of the Consumer in Multi-Master Replication

One of the features of multi-master replication is that a supplier acquires exclusive access to the consumer for the replicated area. During this time, other suppliers are locked out of direct contact with the consumer. If a supplier attempts to acquire access while locked out, the consumer sends back a busy response, and the supplier sleeps for several seconds before making another attempt. Normally, this is all right; the supplier simply sends its update to another consumer while the first consumer is locked and then send updates when the first consumer is free again.

A problem can arise if the locking supplier is under a heavy update load or has a lot of pending updates

in the changelog. If the locking supplier finishes sending updates and then has more pending changes to send, it will immediately attempt to reacquire the consumer and will most likely succeed, since the other suppliers usually will be sleeping. This can cause a single supplier to monopolize a consumer for several hours or longer.

Two attributes address this issue, ***nsds5ReplicaBusyWaitTime*** and ***nsds5ReplicaSessionPauseTime***.

- ***nsds5ReplicaBusyWaitTime***. The ***nsds5ReplicaBusyWaitTime*** attribute sets the amount of time in seconds a supplier should wait after a consumer sends back a busy response before making another attempt to acquire access. The default is **3** seconds.
- ***nsds5ReplicaSessionPauseTime***. The ***nsds5ReplicaSessionPauseTime*** attribute sets the amount of time in seconds a supplier should wait between update sessions. Set this interval so that it is at least one second longer than the interval specified for ***nsds5ReplicaBusyWaitTime***. Increase the interval as needed until there is an acceptable distribution of consumer access among the suppliers. The default is **0**.

These two attributes may be present in the ***nsds5ReplicationAgreement*** object class, which is used to describe replication agreements. Set the attributes at any time by using **changetype:modify** with the **replace** operation. The change takes effect for the next update session if one is already in progress.



NOTE

If either attribute is set to a negative value, Directory Server sends the client a message and an **LDAP_UNWILLING_TO_PERFORM** error code.

The two attributes are designed so that the ***nsds5ReplicaSessionPauseTime*** interval will always be at least one second longer than the interval specified for ***nsds5ReplicaBusyWaitTime***. The longer interval gives waiting suppliers a better chance to gain consumer access before the previous supplier can re-access the consumer.

- If either attribute is specified but not both, ***nsds5ReplicaSessionPauseTime*** is set automatically to 1 second more than ***nsds5ReplicaBusyWaitTime***.
- If both attributes are specified, but ***nsds5ReplicaSessionPauseTime*** is less than or equal to ***nsds5ReplicaBusyWaitTime***, ***nsds5ReplicaSessionPauseTime*** is set automatically to 1 second more than ***nsds5ReplicaBusyWaitTime***.

If Directory Server has to automatically reset the value of ***nsds5ReplicaSessionPauseTime***, the value is changed internally only. The change is not visible to clients, and it not saved to the configuration file. From an external viewpoint, the attribute value appears as originally set.

Replica busy errors are not logged by default because they are usually benign. To see the errors, turn on the replication error logging, log level **8192**. The log levels are described in more detail in the *Directory Server Configuration and Command-Line Tool Reference*.

11.6. CONFIGURING CASCADING REPLICATION

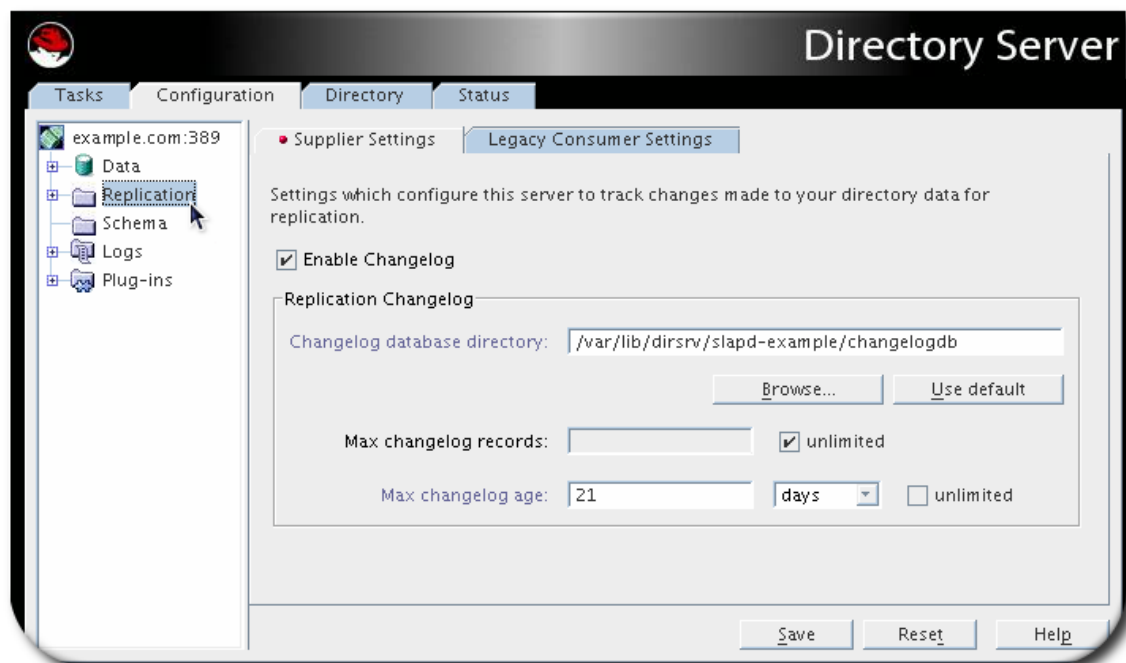
Setting up cascading replication, as shown in [Figure 11.4, “Cascading Replication”](#), has three major steps, for each server in the scenario, the supplier on Server A, which holds a read-write replica; the consumer/supplier on hub Server B, which holds a read-only replica; and the consumer on Server C, which holds a read-only replica:

- [Section 11.6.1, “Configuring the Read-Write Replica on the Supplier Server”](#)
- [Section 11.6.2, “Configuring the Read-Only Replica on the Consumer Server”](#)
- [Section 11.6.3, “Configuring the Read-Only Replica on the Hub”](#)
- [Section 11.6.4, “Setting up the Replication Agreements”](#)

11.6.1. Configuring the Read-Write Replica on the Supplier Server

Next, configure the supplier server, which holds the original copy of the database:

1. Specify the supplier settings for the server.
 1. In the Directory Server Console, select the **Configuration** tab.
 2. In the navigation tree, select the **Replication** folder.
 3. In the right-hand side of the window, select the **Supplier Settings** tab.



4. Check the **Enable Changelog** check box.

This activates all of the fields in the pane below that were previously grayed out.

5. Specify a changelog by clicking the **Use default** button, or click the **Browse** button to display a file selector.
6. Set the changelog parameters for the number and age of the log files.

Clear the unlimited check boxes to specify different values.

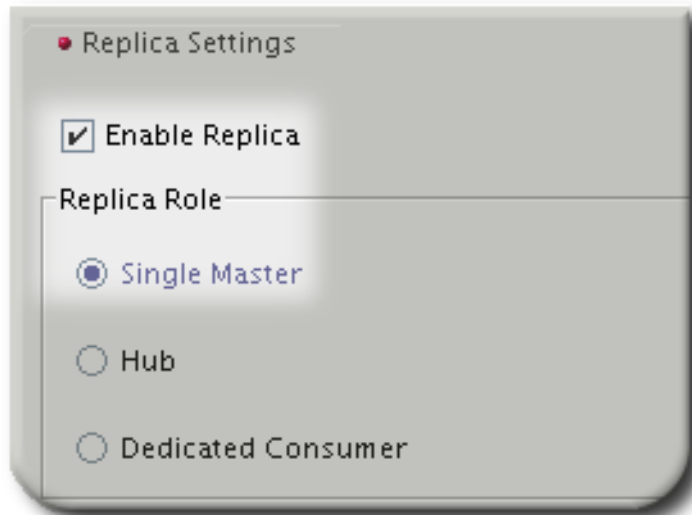
7. Click **Save**.

2. Specify the replication settings required for a read-write replica.

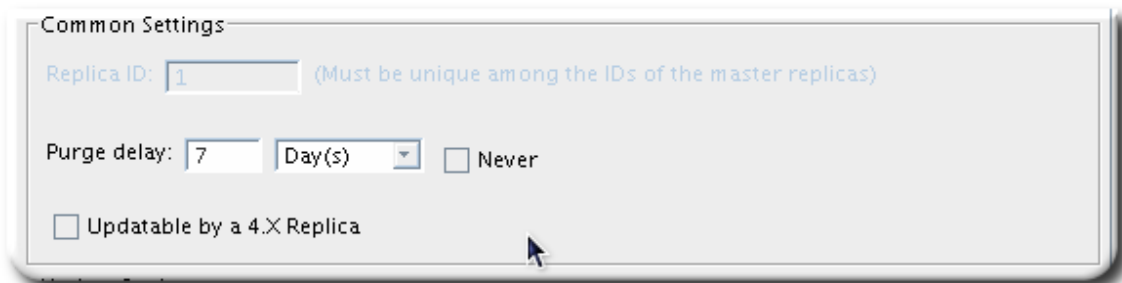
1. In the navigation tree on the **Configuration** tab, expand the **Replication** node, and highlight the database to replicate.

The **Replica Settings** tab opens in the right-hand side of the window.

2. Check the **Enable Replica** check box.
3. In the **Replica Role** section, select the **Single Master** radio button.



4. In the **Common Settings** section, specify a **Replica ID**, which is an integer between **1** and **65534**, inclusive.



The replica ID must be unique for a given suffix, different from any other ID used for read-write replicas on this server and on other servers.

5. In the **Common Settings** section, specify a purge delay in the **Purge delay** field.

The purge delay is how often the state information stored for the replicated entries is deleted.

6. Click **Save**.

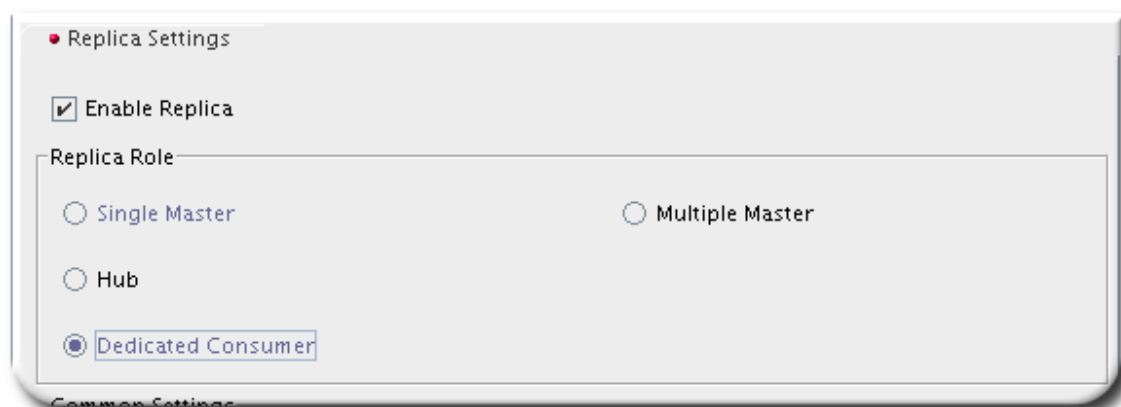
After setting up the supplier replica, begin configuring the replication agreements.

11.6.2. Configuring the Read-Only Replica on the Consumer Server

1. Create the database for the read-only replica if it does not exist. See [Section 2.1.1, “Creating Suffixes”](#) for instructions on creating suffixes.

2. Create the entry for the supplier bind DN on the consumer server if it does not exist. The supplier bind DN is the special entry that the supplier will use to bind to the consumer. This is described in [Section 11.3, “Creating the Supplier Bind DN Entry”](#).
3. Specify the replication settings required for a read-only replica.
 1. In the Directory Server Console, select the **Configuration** tab.
 2. In the navigation tree, expand the **Replication** folder, and highlight the replica database.

The **Replica Settings** tab for that database opens in the right-hand side of the window.



Replica Settings

☒ Enable Replica

Replica Role

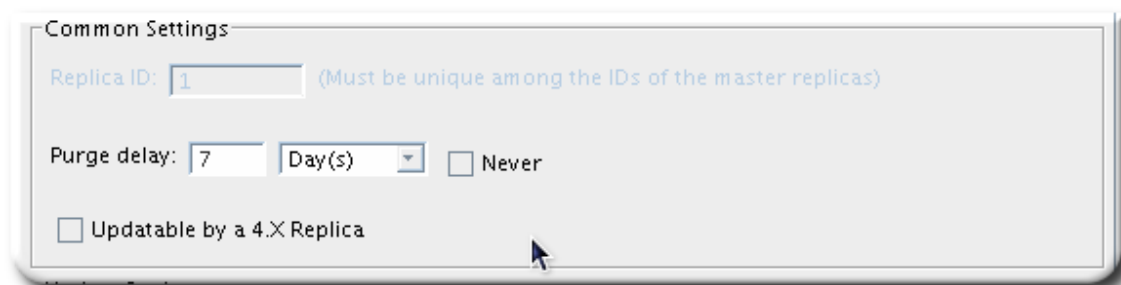
☐ Single Master ☐ Multiple Master

☐ Hub

☒ Dedicated Consumer

Common Settings

3. Check the **Enable Replica** check box.
4. In the **Replica Role** section, select the **Dedicated Consumer** radio button.
5. In the **Common Settings** section, specify a purge delay in the **Purge delay** field.



Common Settings

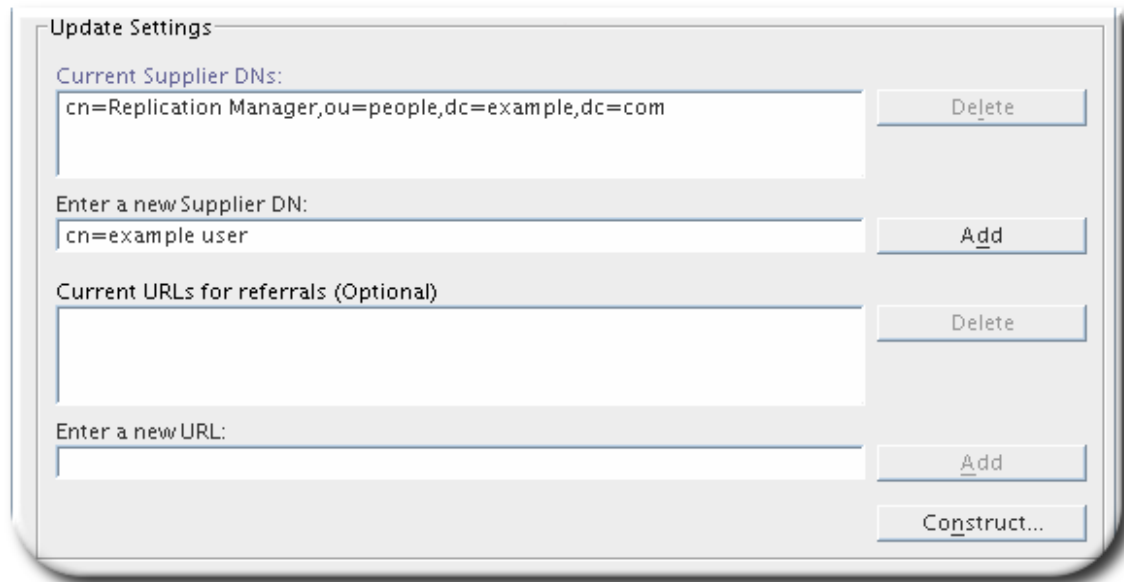
Replica ID: (Must be unique among the IDs of the master replicas)

Purge delay: ☐ Never

☐ Updatable by a 4.X Replica

This option indicates how often the state information stored for the replicated entries is purged.

6. In the **Update Settings** section, specify the bind DN that the supplier will use to bind to the replica. Enter the supplier bind DN in the **Enter a new Supplier DN** field, and click **Add**. The supplier bind DN appears in the **Current Supplier DNs** list.



The supplier bind DN should be the entry created in step 2. The supplier bind DN is a privileged user because it is not subject to access control in the replicated database.



NOTE

There can be multiple supplier bind DNs per consumer but only one supplier DN per replication agreement.

7. Specify the URL (*hostname:port* or *IP_address:port*, with IPv4 or IPv6 addresses) for any supplier servers to which to refer updates.

By default, all updates are first referred to the supplier servers that are specified here. If no suppliers are set here, updates are referred to the supplier servers that have a replication agreement that includes the current replica.

In cascading replication, referrals are automatically sent to the hub server, which in turn refers the request to the original supplier. Therefore, set a referral to the original supplier to replace the automatically generated referral.

4. Click **Save**.

Repeat these steps for every consumer server in the replication configuration, then configure the hub replica.

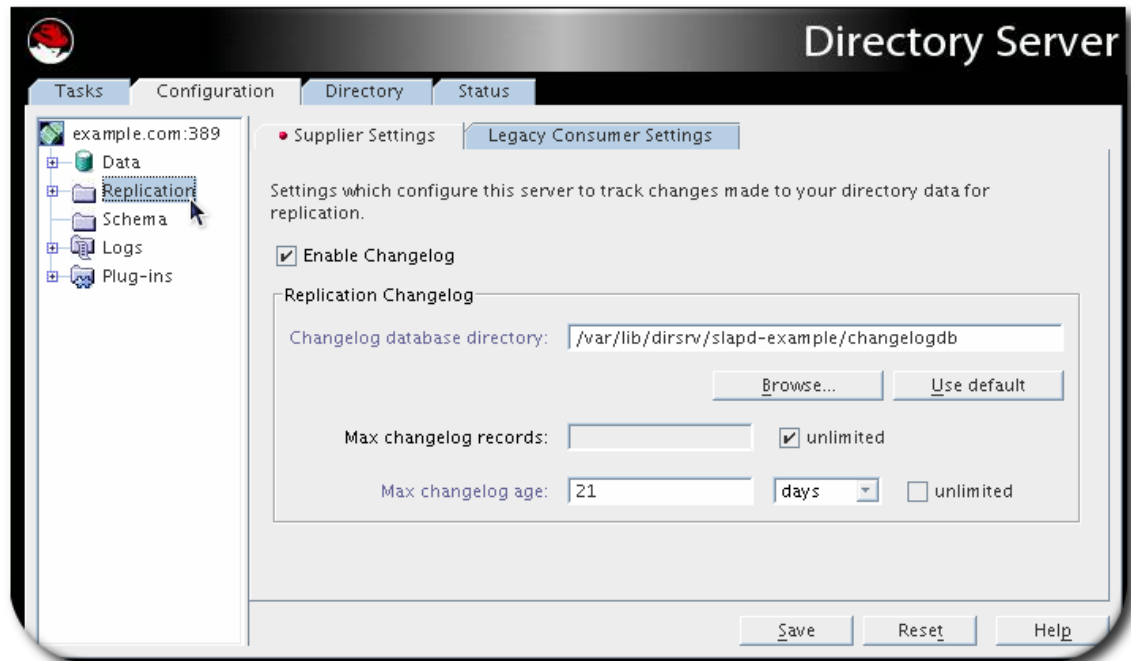
11.6.3. Configuring the Read-Only Replica on the Hub

Do this to set up a hub, which receives replication updates from the supplier and propagates them to consumers:

1. Create the database for the read-only replica if it does not exist. See [Section 2.1.1, “Creating Suffixes”](#) for instructions on creating suffixes.
2. Create the entry for the supplier bind DN on the consumer server if it does not exist. The supplier bind DN is the special entry that the supplier will use to bind to the consumer. This is described in [Section 11.3, “Creating the Supplier Bind DN Entry”](#).
3. Create the changelog for the hub server.

The hub must maintain a changelog even though it does not accept update operations because it records the changes sent from the supplier server.

1. In the Directory Server Console, select the **Configuration** tab.
2. In the navigation tree, select the **Replication** folder.
3. In the right-hand side of the window, select the **Supplier Settings** tab.



4. Check the **Enable Changelog** check box.

This activates all of the fields in the pane below that were previously grayed out.

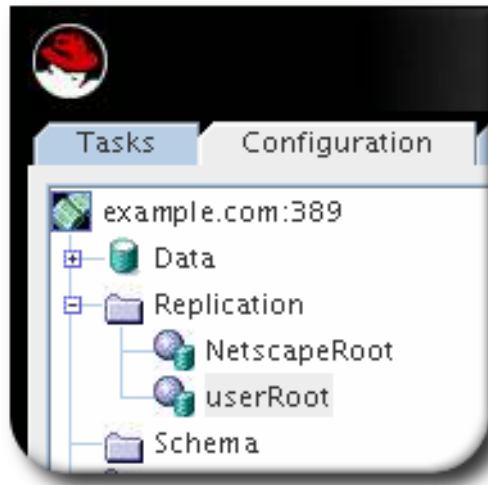
5. Specify a changelog by clicking the **Use default** button, or click the **Browse** button to display a file selector.
6. Set the changelog parameters for the number and age of the log files.

Clear the unlimited check boxes to specify different values.

7. Click **Save**.

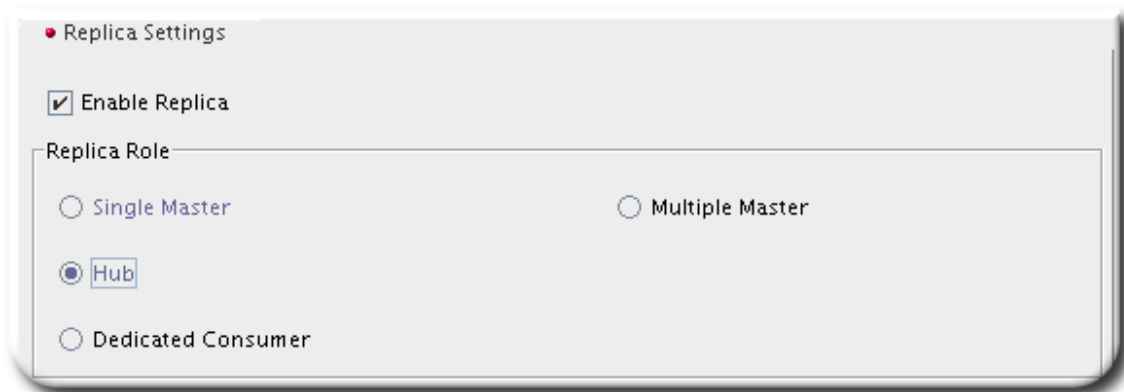
4. Specify the required hub replica settings.

1. In the Directory Server Console, select the **Configuration** tab.
2. In the navigation tree, expand the **Replication** folder, and highlight the replica database.

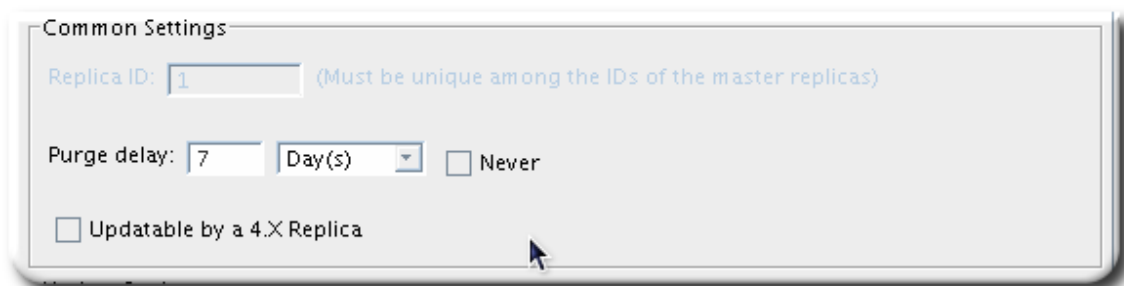


The **Replica Settings** tab for that database opens in the right-hand side of the window.

3. Check the **Enable Replica** check box.



4. In the **Replica Role** section, select the **Hub** radio button.
5. In the **Common Settings** section, specify a purge delay in the **Purge delay** field.



This option sets how often the state information stored for the replicated entries is purged.

6. In the **Update Settings** section, specify the bind DN that the supplier will use to bind to the replica. Enter the supplier bind DN in the **Enter a new Supplier DN** field, and click **Add**. The supplier bind DN appears in the **Current Supplier DNs** list.

The supplier bind DN should be the entry created in step 2. The supplier bind DN is a privileged user because it is not subject to access control in the replicated database.



NOTE

There can be multiple supplier bind DNs per consumer but only one supplier DN per replication agreement.

7. Specify the URL for any supplier servers to which to refer updates.

By default, all updates are first referred to the supplier servers that are specified here. If no suppliers are set here, updates are referred to the supplier servers that have a replication agreement that includes the current replica.

Automatic referrals assume that clients bind over a regular connection; this has a URL in the form **ldap://hostname:port**. For clients to bind to the supplier using SSL, use this field to specify a referral of the form **ldaps://hostname:port**, where the **s** in **ldaps** indicates a secure connection.



NOTE

It is also possible to use IPv4 or IPv6 addresses instead of the host name.

5. Click **Save**.

When all the hubs are configured, then configure the supplier replica.

11.6.4. Setting up the Replication Agreements

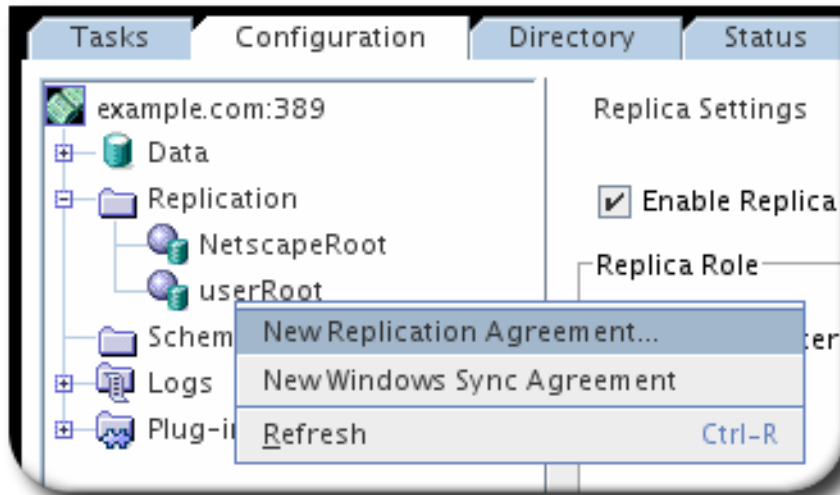
Cascading replication requires two sets of replication agreements, the first between the supplier and the hub and the second between the hub and the consumer. To set up the replication agreements:

1. Create the replication agreement on the supplier for the hub, then use the supplier server to initialize the replica on the hub server.

2. Then create the replication agreement on the hub for each consumer, and initialize the consumer replicas from the hub.

To set up a replication agreement:

1. In the navigation tree of the **Configuration** tab, right-click the database to replicate, and select **New Replication Agreement**.



Alternatively, highlight the database, and select **New Replication Agreement** from the **Object** menu to start the **Replication Agreement Wizard**.

2. In the first screen, fill in a name and description for the replication agreement, and hit **Next**.
3. In the **Source and Destination** screen, fill in the URL (*hostname:port* or *IP_address:port*, with IPv4 or IPv6 addresses) for the consumer and the supplier bind DN and password on that consumer. If the target server is not available, hit in other to fill in the information manually.

Source and Destination

Provide server and content information:

Supplier
example.com:389

Consumer
server2.example.com:389 Other...

Connection
☐ Use LDAP (no encryption)
☐ Use TLS/SSL (TLS/SSL encryption with LDAPS)
☒ Use StartTLS (TLS/SSL encryption with LDAP)

Authentication mechanism:
☐ Server TLS/SSL Certificate (requires TLS/SSL server set up)
☐ SASL/GSSAPI (requires server Kerberos keytab)
☐ SASL/DIGEST-MD5 (SASL user id and password)
☒ Simple (Bind DN/Password)

Bind as: cn=replication manager,cn=config

Password:

Subtree:
dc=example, dc=com

Back Next Cancel Help

- o Unless there is more than one instance of Directory Server configured, by default, there are no consumers available in the drop-down menu. The server URL can be entered manually as *.hostname:port* or *IP_address:port*, with IPv4 or IPv6 addresses.
- o The port listed is the non-SSL port, even if the Directory Server instance is configured to run over SSL. This port number is used only for identification of the Directory Server instance in the Console; it does not specify the actual port number or protocol that is used for replication.
- o If SSL is enabled on the servers, it is possible to select the **Using encrypted SSL connection** radio button for SSL client authentication. Otherwise, fill in the supplier bind DN and password.

**NOTE**

If attribute encryption is enabled, a secure connection *must* be used for the encrypted attributes to be replicated.

4. Select the connection type. There are three options:

- *Use LDAP*. This sets a standard, unencrypted connection.
- *Use TLS/SSL*. This uses a secure connection over the server's secure LDAPS port, such as **636**. This setting is required to use TLS/SSL.
- *Use Start TLS*. This uses Start TLS to establish a secure connection over the server's standard port.

**NOTE**

If secure binds are required for simple password authentication ([Section 14.8.1, “Requiring Secure Binds”](#)), then any replication operations will fail unless they occur over a secure connection. Using a secure connection (SSL/TLS and Start TLS connections or SASL authentication) is recommended, anyway.

5. Select the appropriate authentication method and supply the required information. This gives the information that the supplier uses to authenticate and bind to the consumer server to send updates.

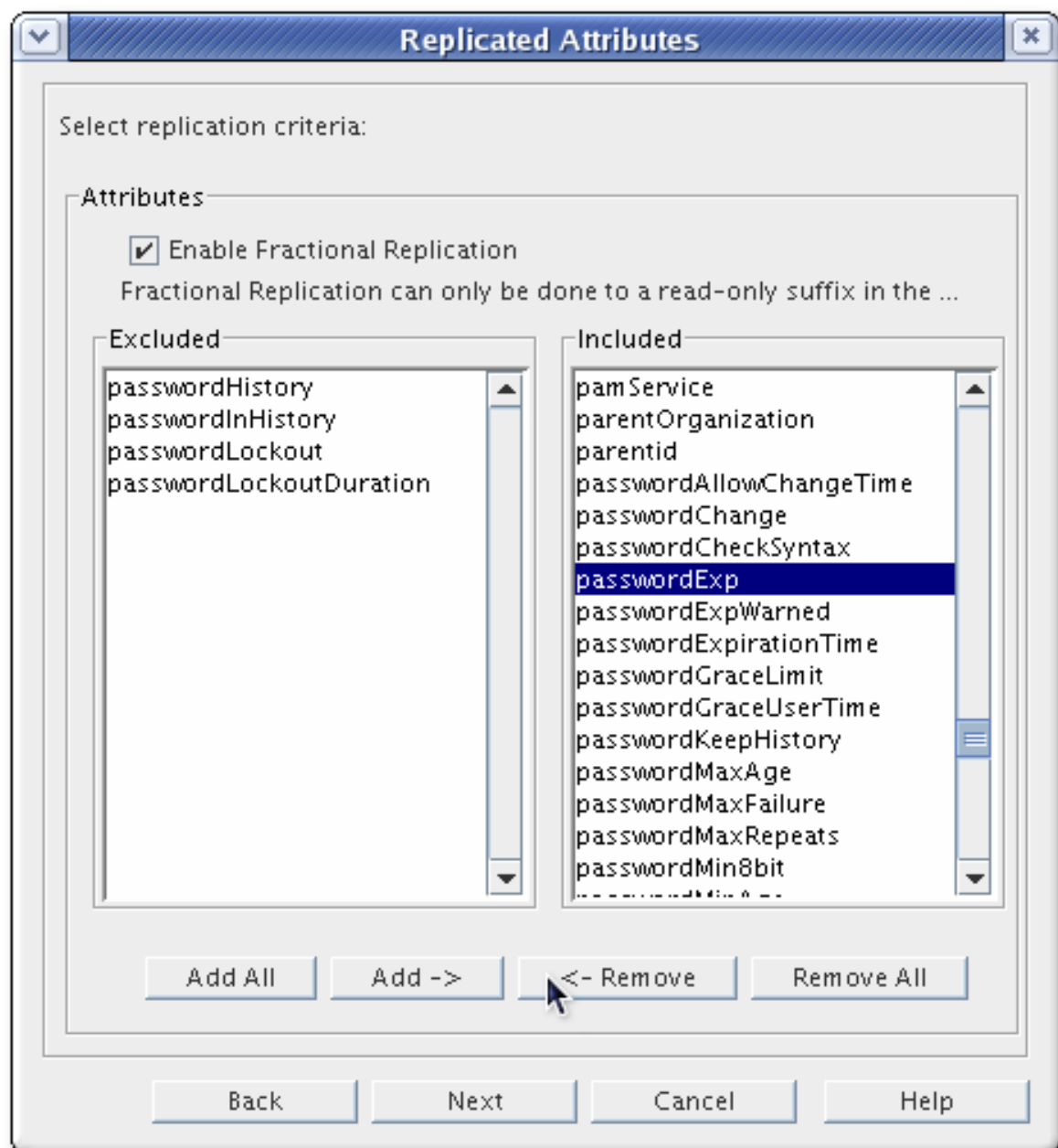
- *Simple* means that the server connects over the standard port with no encryption. The only required information is the bind DN and password for the Replication Manager (which must exist on the consumer server).
- *Server TLS/SSL Certificate* uses the supplier's SSL certificate to authenticate to the consumer server. A certificate must be installed on the supplier for certificate-based authentication, and the consumer server must have certificate mapping configured so that it can map the subject DN in the supplier's certificate to its Replication Manager entry.

Configuring SSL and certificate mapping is described in [Section 7.4, “Setting up TLS/SSL”](#).

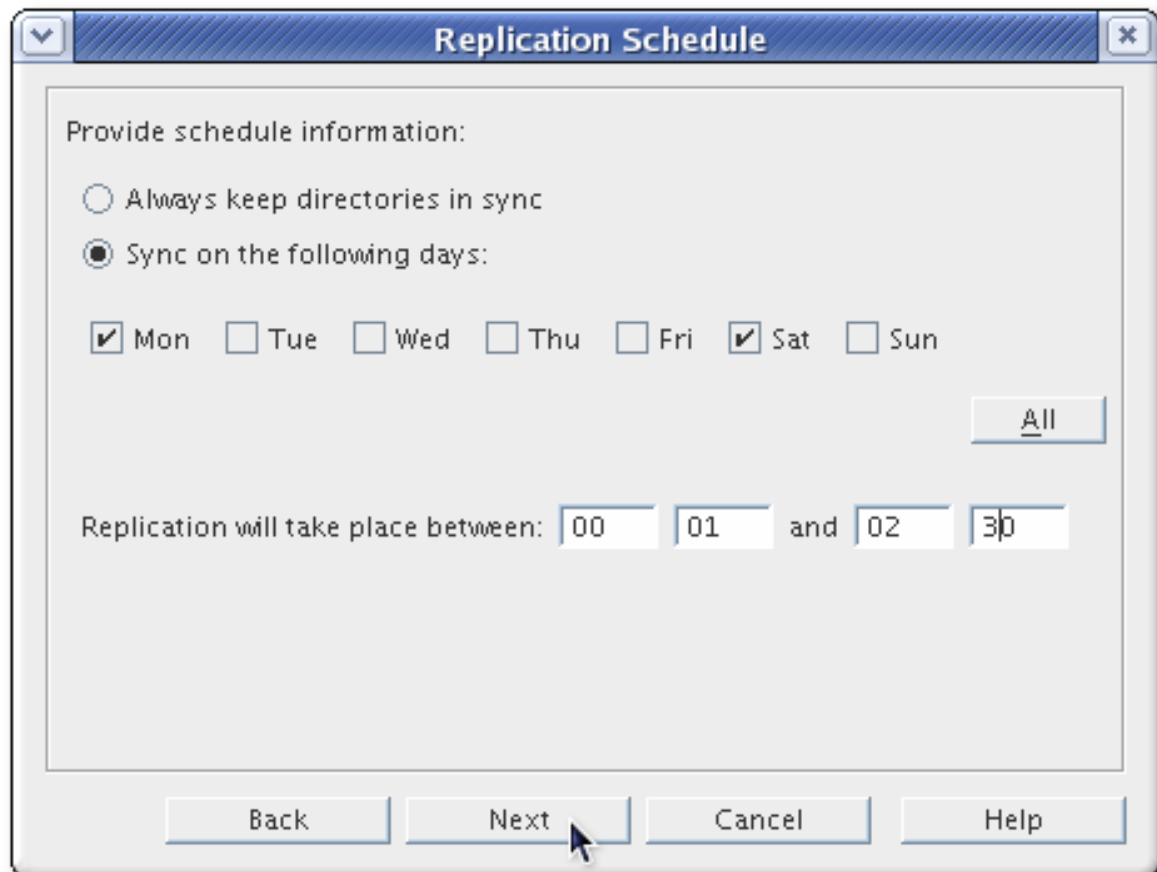
- *SASL/DIGEST-MD5*, like simple authentication, requires only the bind DN and password to authenticate. This can run over a standard or SSL/TLS connection.
- *SASL/GSSAPI* requires the supplier server to have a Kerberos keytab (as in [Section 7.12.2.2, “About the KDC Server and Keytabs”](#)), and the consumer server to have a SASL mapping to map the supplier's principal to the real replication manager entry (as in [Section 7.11.3.1, “Configuring SASL Identity Mapping from the Console”](#)).

6. Hit **Next**.

7. Fractional replication controls which entry attributes are replicated between servers. By default, all attributes are replicated. To select attributes that will *not* be replicated to the consumer, check the **Enable Fractional Replication** check box. Then, highlight the attribute (or attributes) in the **Included** column on the right, and click **Remove**. All attributes that will not be replicated are listed in the **Excluded** column on the left, as well as in the summary the replication agreement is complete.



8. Set the schedule for when replication runs. By default, replication runs continually.



The image shows a Windows-style dialog box titled "Replication Schedule". It has a standard title bar with a minimize button, a maximize button, and a close button. The main area contains the text "Provide schedule information:" followed by two radio buttons. The first radio button is labeled "Always keep directories in sync" and is unselected. The second radio button is labeled "Sync on the following days:" and is selected. Below the second radio button are seven checkboxes for the days of the week: Mon, Tue, Wed, Thu, Fri, Sat, and Sun. The "Mon" and "Sat" checkboxes are checked, while the others are unchecked. To the right of these checkboxes is a button labeled "All". Below the day checkboxes, the text "Replication will take place between:" is followed by four text input fields. The first two fields contain "00" and "01", and the last two fields contain "02" and "30". At the bottom of the dialog box are four buttons: "Back", "Next", "Cancel", and "Help". A mouse cursor is pointing at the "Next" button.

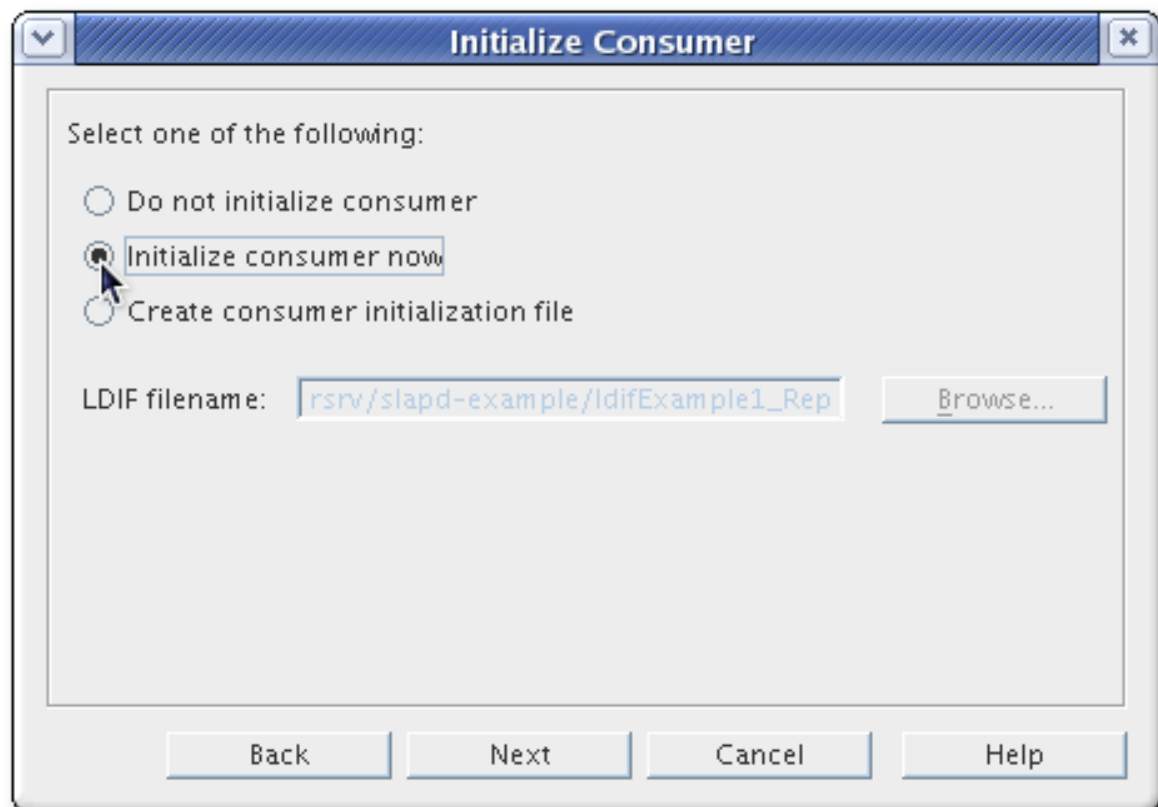


NOTE

The replication schedule cannot cross midnight (**0000**). So, it is possible to set a schedule that begins at **0001** and ends at **2359** on the same day, but it is not possible to set one that begins at **2359** on one day and ends at **0001** on the next.

Hit **Next**.

9. Set when the consumer is initialized. *Initializing* a consumer manually copies all data over from the supplier to the consumer. The default is to create an initialization file (an LDIF of all supplier data) so that the consumer can be initialized later. It is also possible to initialize the consumer as soon as the replication agreement is completed or not at all. For information on initializing consumers, see [Section 11.15, "Initializing Consumers"](#). For cascading replication, consider the following:
 - Create the supplier-hub replication agreement on the supplier first, and initialize the hub from the supplier.
 - Create the hub-consumer replication agreements on the hub, and initialize the consumers from the hub.

**NOTE**

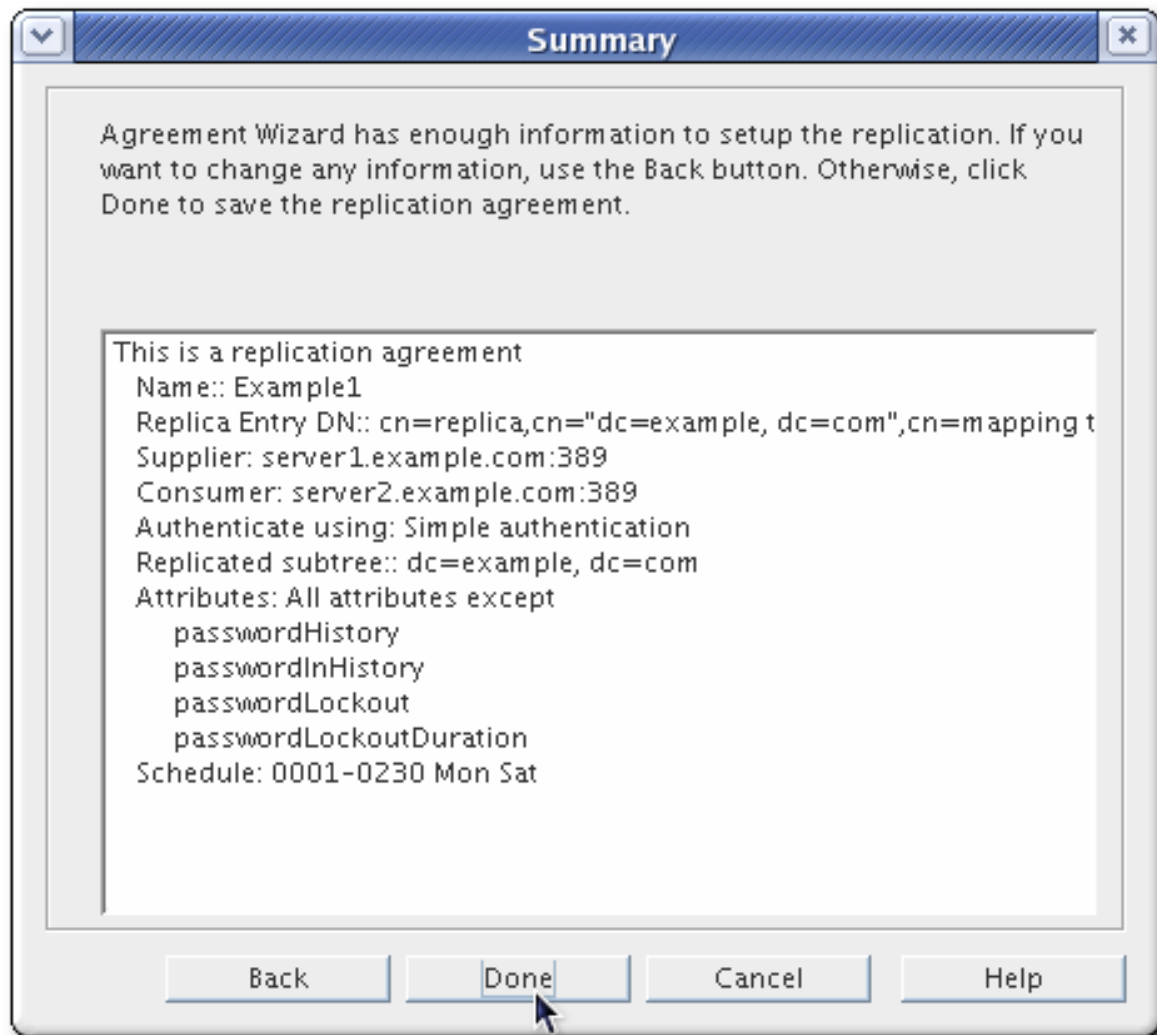
Replication *will not* begin until the consumer is initialized.

**IMPORTANT**

For multi-master replication, be sure that consumers are only initialized **once**, by one supplier. When checking the replication status, be sure to check the replication agreement entry, on the appropriate supplier, which was used to initialize the consumer.

Hit **Next**.

10. The final screen shows the settings for the replication agreement, as it will be included in the **dse.ldif** file. Hit **Done** to save the agreement.



NOTE

After creating a replication agreement, the connection type (SSL or non-SSL) cannot be change because LDAP and LDAPS connections use different ports. To change the connection type, re-create the replication agreement.

11.7. CONFIGURING REPLICATION FROM THE COMMAND LINE

Replication can be configured on the command line by creating the appropriate replica and agreement entries on the servers. The process follows the same order as setting up replication through the Directory Server Console:

1. Create the supplier bind DN on every consumer, hub, and multi-master supplier ([Section 11.3, “Creating the Supplier Bind DN Entry”](#)).
2. If the corresponding database and suffix do not exist on one of the replicas, create it ([Section 2.1.1, “Creating Suffixes”](#)).
3. Configure the supplier replicas ([Section 11.7.1, “Configuring Suppliers from the Command Line”](#)).
4. Configure consumers ([Section 11.7.2, “Configuring Consumers from the Command Line”](#)).

5. Configure hubs for cascading replication ([Section 11.7.3, “Configuring Hubs from the Command Line”](#)).
6. Create the replication agreements ([Section 11.7.4, “Configuring Replication Agreements from the Command Line”](#)). For cascading replication, create the agreement between the supplier and hub, then between the hub and consumers; for multi-master, create the agreements between all suppliers, then between the suppliers and consumers.
7. Lastly, initialize all of the consumers ([Section 11.7.5, “Initializing Consumers Online from the Command Line”](#)), if the consumers were not initialized when the replication agreement was created.

11.7.1. Configuring Suppliers from the Command Line

There are two steps to setting up the supplier replica. First, the changelog must be enabled, which allows the supplier to track changes to the Directory Server. Then, the supplier replica is created.

1. On the supplier server, use **ldapmodify** to create the changelog entry.

Example 11.2. Example Changelog Entry

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h
supplier1.example.com -v -a

dn: cn=changelog5,cn=config
changetype: add
objectclass: top
objectclass: extensibleObject
cn: changelog5
nsslapd-changelogdir:
/var/lib/dirsrv/slapd-instance_name/changelogdb
nsslapd-changelogmaxage: 10d
```

There are two important attributes with the changelog.

- **nsslapd-changelogdir** sets the directory where the changelog is kept.
- **nsslapd-changelogmaxage** sets how long the changelog is kept; since the changelog can get very large, this helps trim the changelog to prevent affecting server performance and using up disk space. If this parameter is not set, the default is for the changelog to be kept forever.

The changelog entry attributes are described in [Table 11.1, “Changelog Attributes”](#). These attributes are described in more detail in the *Directory Server Configuration and Command-Line Tool Reference*.

2. Create the supplier replica.

Example 11.3. Example Supplier Replica Entry

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h
supplier1.example.com -v -a

dn: cn=replica,cn=dc\=example\,dc\=com,cn=mapping tree,cn=config
```



```

changetype: add
objectclass: top
objectclass: nsds5replica
objectclass: extensibleObject
cn: replica
nsds5replicaroot: dc=example,dc=com
nsds5replicaid: 7
nsds5replicatype: 3
nsds5flags: 1
nsds5ReplicaPurgeDelay: 604800
nsds5ReplicaBindDN: cn=replication manager,cn=config

```

- *nsds5replicaroot* sets the subtree (suffix) which is being replicated.
- *nsds5replicatype* sets what kind of replica this database is. For either a single master or a multi-master supplier, this value must be **3**.
- *nsds5replicaid* sets the replica ID. The value must be unique among all suppliers and hubs; the valid range is **1** to **65534**.
- *nsds5ReplicaPurgeDelay* sets how long an entry holds its status information or how long a tombstone entry is kept before deleting the information. The default value is **604800** (one week).
- *nsds5flags* sets whether the replica writes to the changelog. For a supplier, this value must be **1**.

The replica entry attributes are described in [Table 11.2, “Replica Attributes”](#).

After creating every supplier which will take part in the replication setup, then begin creating the replication agreements.


Table 11.1. Changelog Attributes

Object Class or Attribute	Description	Values
objectclass: top	Required object class for every entry.	
objectclass: extensibleObject	An object class which allows any other object class or attribute to be added to an entry.	
cn: changelog5	The naming attribute for the changelog entry.	Any string; the default usage is to set the common name to changelog5 .
nsslapd-changelogdir: <i>directory</i>	Sets the file and directory changelog, to which the Directory Server writes changes.	Any directory; the default is /var/lib/dirsrv/slapped-<i>instance_name</i>/changelogdb .

Object Class or Attribute	Description	Values
nsslapd-changelogmaxage: <i>number unit</i>	Sets how long the changelog keeps an entry before purging it. This is not used by consumers, but is recommended for hubs and suppliers, which keep changelogs.	The <i>number</i> is any integer. The <i>unit</i> can be s for seconds, m for minutes, h for hours, d for days, or w for weeks.

Table 11.2. Replica Attributes

Object Class or Attribute	Description	Values
objectclass: top	Required object class for every entry.	
objectclass: extensibleObject	An object class which allows any other object class or attribute to be added to an entry.	
objectclass: nsds5replica	An object class which allows replication attributes to be added to an entry.	
cn: replica	The naming attribute for the replica.	Any string; the default usage is to set the common name to replica for every configured replica.
nsds5replicaroot: <i>suffix</i>	Sets which subtree is replicated.	A root suffix associated with a database, since the entire database is replicated. For example: dc=example, dc=com
nsds5replicaid: <i>number</i>	The ID of the replica. This <i>must</i> be set to 65535 for consumers or hubs. For suppliers, this value must be a unique value.	1 to 65534 , inclusive, for suppliers. 65535 for consumers and hubs.
nsds5replicatype: <i>number</i>	Sets the type of replica, either read-only or read-write.	2 for consumers and hubs (read-only replicas) 3 for both single and multi-master suppliers (read-write replicas)

Object Class or Attribute	Description	Values
<code>nsds5flags: number</code>	Sets whether the replica writes to the changelog.	<div> 0 means the replica does not write to the changelog; this is the default for consumers. </div> <div> 1 means the replica writes to the changelog; this is the default for hubs and suppliers. </div>
<code>nsds5ReplicaPurgeDelay: number</code>	Sets the period of time in seconds to wait before purging the state information from an entry or purging tombstone entries. This setting is required for all types of replicas — suppliers, hubs, and consumers.	0 (keep forever) to 2147483647 (the maximum 32-bit integer); the default value is 604800 , one week.
<code>nsds5ReplicaBindDN: DN</code>	The supplier bind DN used by the supplier to bind to the consumer. This is required for consumers, hubs, and multi-master suppliers, but not for single-master suppliers.	Any DN; the recommended DN is cn=Replication Manager, cn=config . <div>  NOTE For security, it is strongly recommended that you do <i>not</i> use the Directory Manager as the supplier bind DN. </div>
<code>nsds5replicareferral: URL</code>	<i>Optional.</i> An LDAP URL which a consumer or hub to which a consumer or hub can forward update requests. By default, update requests are sent to the masters for the consumer; use this parameter to override the default. The URL can use the format <code>ldap[s]://hostname:port</code> or <code>ldap[s]://IP_address:port</code> , with IPv4 or IPv6 addresses.	Any LDAP URL. For example: <pre>nsds5replicareferral: ldap://supplier1.example.com:389</pre>
<code>nsDS5ReplicaReleaseTimeout: seconds</code>	<i>Optional.</i> Sets the timeout in seconds after which a master releases a replica.	From 0 to the maximum 32-bit integer: 2147483647 . To disable the timeout, set the value to 0 . The default value is 60 seconds.

11.7.2. Configuring Consumers from the Command Line

On the consumer host, such as **consumer1.example.com**, create the replica entry. This entry identifies the database and suffix as participating in replication and sets what kind of replica the database is. There are four key attributes:

- *nsds5replicaroot* sets the subtree (suffix) which is being replicated.
- *nsds5replicatype* sets what kind of replica this database is. For a consumer, this value must be **2**.
- *nsds5ReplicaBindDN* give the DN as which the supplier will bind to the consumer to make changes.
- *nsds5flags* sets whether the replica writes to the changelog. For a consumer, this value must be **0**.

This **ldapmodify** creates a new consumer replica on the **consumer1.example.com** host for the **dc=example,dc=com** subtree.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h
consumer1.example.com -v -a

dn: cn=replica,cn=dc\=example\,dc\=com,cn=mapping tree,cn=config
changetype: add
objectclass: top
objectclass: nsds5replica
objectclass: extensibleObject
cn: replica
nsds5replicaroot: dc=example,dc=com
nsds5replicaid: 65535
nsds5replicatype: 2
nsds5ReplicaBindDN: cn=replication manager,cn=config
nsds5flags: 0
```

The replica entry attributes are described in [Table 11.2, “Replica Attributes”](#). These attributes are described in more detail in the *Directory Server Configuration and Command-Line Tool Reference*.

11.7.3. Configuring Hubs from the Command Line

Hubs are intermediate read-only replicas which receive updates from suppliers and pass them on to other consumers. These are part of the cascading replication scenario, described in [Section 11.2.3, “Cascading Replication”](#). Creating the hub has two steps: first, creating the changelog database since the hub keeps a record of changes sent by the supplier, and second, configuring the hub replica.

1. On the hub server, such as **hub1.example.com**, use **ldapmodify** to create the changelog entry.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h
hub1.example.com -v -a

dn: cn=changelog5,cn=config
changetype: add
objectclass: top
objectclass: extensibleObject
```

```
cn: changelog5
nsslapd-changelogdir:
/var/lib/dirsrv/slapd-instance_name/changelogdb
```

There is one important attribute with the changelog, ***nsslapd-changelogdir***, which sets the directory where the changelog is kept.

The changelog entry attributes are described in [Table 11.1, “Changelog Attributes”](#). These attributes are described in more detail in the *Directory Server Configuration and Command-Line Tool Reference*.

2. On the hub host, create the replica entry. This **ldapmodify** command creates a new hub replica on the **hub1.example.com** host for the **dc=example,dc=com** subtree.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h
hub1.example.com -v -a

dn: cn=replica,cn=dc\=example\,dc\=com,cn=mapping tree,cn=config
changetype: add
objectclass: top
objectclass: nsds5replica
objectclass: extensibleObject
cn: replica
nsds5replicaid: 65535
nsds5replicaroot: dc=example,dc=com
nsds5replicatype: 2
nsds5ReplicaPurgeDelay: 604800
nsds5ReplicaBindDN: cn=replication manager,cn=config
nsds5flags: 1
```

This entry identifies the database and suffix as participating in replication and sets what kind of replica the database is. There are five key attributes:

- ***nsds5replicaroot*** sets the subtree (suffix) which is being replicated.
- ***nsds5replicatype*** sets what kind of replica this database is. For a hub, this value must be **2**.
- ***nsds5replicaid*** sets the ID of the replica. This *must* be set to **65535** for consumers or hubs.
- ***nsds5ReplicaPurgeDelay*** sets how long an entry holds its status information or how long a tombstone entry is kept before deleting the information. The default value is **604800** (one week).
- ***nsds5ReplicaBindDN*** give the DN as which the supplier will bind to the hub to make changes.
- ***nsds5flags*** sets whether the replica writes to the changelog. For a hub, this value must be **1**.

The replica entry attributes are described in [Table 11.2, “Replica Attributes”](#). These attributes are described in more detail in the *Directory Server Configuration and Command-Line Tool Reference*.

11.7.4. Configuring Replication Agreements from the Command Line

When setting up replication agreements, first set them up between all suppliers, then between the suppliers and the hubs, and last between the hub and the consumers.

The replication agreement has to define eight separate attributes:

- The consumer host (***nsds5replicahost***) and port (***nsds5replicaport***).
- The DN for the supplier to use to bind with the consumer (***nsds5ReplicaBindDN***).
- The way that the supplier binds (***nsds5replicabindmethod***).
- Any credentials required (***nsDS5ReplicaCredentials***) for that bind method and specified DN.
- The subtree being replicated (***nsds5replicaroot***).
- The replication schedule (***nsds5replicaupdateschedule***).
- Any attributes which will *not* be replicated (***nsds5replicatedattributelist*** and ***nsDS5ReplicatedAttributeListTotal***).

Use **ldapmodify** to add a replication agreement to every supplier for every consumer which it will update. For example:

Example 11.4. Example Replication Agreement Entry

```
dn: cn=ExampleAgreement,cn=replica,cn=dc\=example\,dc\=com,cn=mapping
tree,cn=config
objectclass: top
objectclass: nsds5ReplicationAgreement
cn: ExampleAgreement
nsds5replicahost: consumer1
nsds5replicaport: 389
nsds5ReplicaBindDN: cn=replication manager,cn=config
nsds5replicabindmethod: SIMPLE
nsds5replicaroot: dc=example,dc=com
description: agreement between supplier1 and consumer1
nsds5replicaupdateschedule: 0000-0500 1
nsds5replicatedattributelist: (objectclass=*) $ EXCLUDE
authorityRevocationList accountUnlockTime memberof
nsDS5ReplicatedAttributeListTotal: (objectclass=*) $ EXCLUDE
accountUnlockTime
nsds5replicacredentials: secret
nsds5BeginReplicaRefresh: start
```

The replication agreement attributes are listed in [Table 11.3, “Replication Agreement Attributes”](#). These attributes are described in more detail in the *Directory Server Configuration and Command-Line Tool Reference*.


After creating every replication agreement, begin initializing consumers.

Table 11.3. Replication Agreement Attributes

Object Class or Attribute	Description	Values
---------------------------	-------------	--------

Object Class or Attribute	Description	Values
objectclass: top	Required object class for every entry.	
objectclass: nsds5ReplicationAgreement	An operational object class which contains the replication agreement attributes.	
cn: <i>agreement_name</i>	The naming attribute for the replication agreement.	Any string.
nsds5replicahost: <i>hostname</i>	Gives the host name of the consumer server; the host name can be the fully qualified host and domain name. If TLS/SSL is enabled, the fully-qualified domain name is required. It is also possible to use IPv4 or IPv6 addresses instead of the host name.	Any host name. For example: <code>nsds5replicahost: consumer1</code>
nsds5replicaport: <i>number</i>	Gives the LDAP port for the consumer server. To use TLS/SSL, give the secure port number (636 by default) and set the <i>nsds5ReplicaTransportInfo</i> attribute to SSL . To use Start TLS, which initiates a secure connection over a standard port, use the standard port, 389 , with the <i>nsds5ReplicaTransportInfo</i> attribute to TLS . To use simple authentication, use the standard port, 389 , with the <i>nsds5ReplicaTransportInfo</i> attribute to LDAP .	Any port number.
nsds5ReplicaTransportInfo: <i>method</i>	To use TLS/SSL, set this parameter to SSL . To use Start TLS, which initiates a secure connection over a standard port, set this parameter to TLS . To use simple authentication, set this parameter to LDAP .	<div>empty or LDAP (standard connection over port 389)</div> <div>SSL (secure connection over the secure port, such as 636)</div> <div>TLS (secure connection over the standard port, 389, using Start TLS)</div>

Object Class or Attribute	Description	Values
nsds5ReplicaBindDN: <i>DN</i>	The supplier bind DN used by the supplier to bind to the consumer. This is required for consumers, hubs, and multi-master suppliers, but not for single-master suppliers.	Any DN; the recommended DN is cn=Replication Manager, cn=config .
nsds5replicabindmethod: <i>type</i>	<p>The connection type for replication between the servers. The connection type defines how the supplier authenticates to the consumer.</p> <p>Leaving the bind method empty or setting it to SIMPLE means that the server uses basic password-based authentication. This requires the nsds5ReplicaBindDN and nsds5ReplicaCredentials attributes to give the bind information.</p> <p>The SSLCLIENTAUTH option uses a secure connection. This requires setting the nsds5ReplicaTransportInfo attribute be set to SSL or TLS. For certificate-based authentication, the consumer server must also have a certificate mapping to map the subject DN in the supplier's certificate to the replication manager entry.</p> <p>Using SASL/GSSAPI requires that the supplier server have a Kerberos keytab (as in Section 7.12.2.2, "About the KDC Server and Keytabs"), and the consumer server have a SASL mapping to map the supplier's principal to the real replication manager entry (as in Section 7.11.3.1, "Configuring SASL Identity Mapping from the Console").</p> <p>The SASL/DIGEST-MD5 setting, like SIMPLE, uses password-based authentication and requires the nsds5ReplicaBindDN and nsds5ReplicaCredentials attributes to give the bind information.</p>	<div>SIMPLE</div> <div>SSLCLIENTAUTH</div> <div>SASL/GSSAPI</div> <div>SASL/DIGEST-MD5</div>

Object Class or Attribute	Description	NOTE	Values
		If secure binds are required for simple password authentication (Section 14.8.1, “Requiring Secure Binds”), then any replication operations will fail unless they occur over a secure connection. Using a secure connection (SSL/TLS and Start TLS connections or SASL authentication) is recommended, anyway.	
nsDS5ReplicaCredentials: <i>hash</i>	<i>Only for simple authentication.</i> Stores the hashed password used with the bind DN given for simple authentication.		
nsds5replicaroot: <i>suffix</i>	Sets which subtree is replicated.		A root suffix associated with a database, since the entire database is replicated. For example: dc=example, dc=com
description: <i>text</i>	A text description of the replication agreement.		Any text string. It is advisable to make this a useful description, such as <i>agreement between supplier1 and consumer1</i> .

Object Class or Attribute	Description	Values
nsds5replicatedattributelist: '(objectclass=*)' \$ EXCLUDE <i>attributes</i>	<p><i>Optional.</i> Sets which attributes will <i>not</i> be replicated. The filter must be set to "(objectclass=*)", and the list of attributes are separated by a single space.</p> <p>If this is the only fractional replication attribute set, then it applies to both incremental and total updates. If <i>nsDS5ReplicatedAttributeListTotal</i> is set, then this attribute applies to incremental updates.</p>	<pre>'(objectclass=*)' \$ EXCLUDE authorityRevocationL ist memberof accountUnlockTime</pre>
nsds5replicatedattributelisttotal: '(objectclass=*)' \$ EXCLUDE <i>attributes</i>	<p><i>Optional.</i> Sets which attributes will <i>not</i> be replicated during a total update.</p> <p>The <i>nsDS5ReplicatedAttributeList</i> attribute must be set for incremental updates before these attributes can be set for total updates.</p>	<pre>'(objectclass=*)' \$ EXCLUDE accountUnlockTime</pre>

Object Class or Attribute	Description	Values
nsds5replicaupdateschedule: <i>start_time end_time days</i>	<p>Sets the start and end time for the replication updates and the days on which replication occurs. If the schedule is omitted, replication will take place all the time.</p> <p>The replication schedule cannot cross midnight (0000). So, it is possible to set a schedule that begins at 0001 and ends at 2359 on the same day, but it is not possible to set one that begins at 2359 on one day and ends at 0001 on the next.</p>	<p>Has the following value, with the start (<i>SSSS</i>) and end (<i>EEEE</i>) times set in the form HHMM</p> <p>The times are given in 24 hour clock format, so 0000 is midnight and 2359 is 11:59 PM. For example, the setting 1030 1630 schedules replication from 10:30 AM to 4:30 PM. The times cannot wrap around midnight, so the setting 2300 0100 is not valid.</p> <p>The days ranging from 0 (Sunday) to 6 (Saturday). Setting 06 schedules replication on Sunday and Saturday, while 135 schedules replication on Monday, Wednesday, and Friday.</p> <pre>nsds5replicaupdateschedule: SSSS EEEE DDDDDDD</pre> <p>For example, this schedules replication between midnight (0000) and 5am (0500) on Monday and Tuesday:</p> <pre>nsds5replicaupdateschedule: 0000 0500 12</pre>
nsds5BeginReplicaRefresh: start	<p><i>Optional.</i> Performs an online (immediate) initialization of the consumer. If this is set, the attribute is only present as long as the consumer is being initialized; when the initialization is complete, the attribute is deleted automatically.</p> <p>If this is not set, then consumer initialization must be performed manually.</p>	<div>start</div> <p>To initialize the consumer, this attribute must have a value of start; any other value is ignored.</p>

11.7.5. Initializing Consumers Online from the Command Line

An online initialization can be initiated from the command line by adding the ***nsds5replicarefresh*** attribute to the replication agreement entry. If the attribute is included when the replication agreement is created, initialization begins immediately. It can be added later to initialize the consumer at any time. This attribute is absent by default, and it will be automatically deleted once the consumer initialization is complete.

1. Find the DN of the replication agreement on the supplier server that is for the consumer to be initialized. For example:

```
ldapsearch -x -h supplier1.example.com -p 389 -D "cn=directory
manager" -W -s sub -b cn=config "
(objectclass=nsds5ReplicationAgreement)"
```

This command returns all of the replication agreements configured on the supplier in LDIF format. Get the DN of the replication agreement with the consumer to be initialized. This is the replication agreement which will be edited.

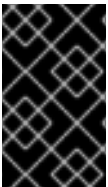
2. Edit the replication agreement, and add the ***nsds5BeginReplicaRefresh*** attribute:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h
supplier1.example.com

dn:
cn=ExampleAgreement,cn=replica,cn=dc\=example\,dc\=com,cn=mapping
tree,cn=config
changetype: modify
replace: nsds5BeginReplicaRefresh
nsds5BeginReplicaRefresh: start
```

ldapmodify does not prompt for input; simply type in the LDIF statement, and then hit enter twice when the LDIF statement is complete. Close the **ldapmodify** utility by hitting **Ctrl+C**.

When the initialization is complete, the ***nsds5BeginReplicaRefresh*** attribute is automatically deleted from the replication agreement entry.



IMPORTANT

For multi-master replication, be sure that consumers are only initialized **once**, by one supplier. When checking the replication status, be sure to check the replication agreement entry, on the appropriate supplier, which was used to initialize the consumer.

Initializing consumers from the command line is also explained in [Section 11.15.3, “Initializing Consumers Online Using the Command Line”](#). Manually initializing consumers is explained in [Section 11.15.4, “Manual Consumer Initialization Using the Command Line”](#). The replication monitoring attributes are described in more detail in the *Directory Server Configuration and Command-Line Tool Reference*.



NOTE

For large databases, the ***nsslapd-idletimeout*** setting must be set to a large enough time period (or even an unlimited period) to allow the entire database to be initialized before the operation times out. Alternatively, the ***nsIdleTimeout*** setting for the supplier bind DN entry can be set high enough to allow the online initialization operation to complete, without having to change the global setting.

To keep data integrity, initialize the consumer databases from the appropriate supplier. Determining the correct supplier can be more difficult in mixed replication environments, but, even when manually initializing consumers, consider four things:

- Use one supplier, a *data master*, as the source for initializing consumers.
- Do not *reinitialize* a data master when the replication agreements are created. For example, do not initialize server1 from server2 if server2 has already been initialized from server1.
- For a multi-master scenario, initialize all of the other master servers in the configuration from one master.
- For cascading replication, initialize all of the hubs from a supplier, then initialize the consumers from the hubs.

11.8. TEMPORARILY SUSPENDING REPLICATION

By default, replication is enabled and active as soon as the replication agreement is created, and replication proceeds on the given schedule. There can be times when it is necessary to suspend replication, such as when a server is taken down for maintenance. An attribute can be added to the replication agreement entry which disables replication.

11.8.1. Disabling Replication

The ***nsds5ReplicaEnabled*** attribute on the replication entry sets whether the replication agreement is active. This attribute is missing by default and has a presumed value of **on**, meaning that replication is enabled.

To stop replication from a supplier, set the ***nsds5ReplicaEnabled*** attribute to **off**.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h
supplier1.example.com

dn: cn=ExampleAgreement,cn=replica,cn=dc\=example\,dc\=com,cn=mapping
tree,cn=config
changetype: modify
add: nsds5ReplicaEnabled
nsds5ReplicaEnabled: off
```

11.8.2. Re-enabling Replication

1. Using **ldapmodify**, set the ***nsds5ReplicaEnabled*** attribute to **on**.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h
supplier1.example.com

dn:
cn=ExampleAgreement,cn=replica,cn=dc\=example\,dc\=com,cn=mapping
tree,cn=config
changetype: modify
add: nsds5ReplicaEnabled
nsds5ReplicaEnabled: on
```

Since the default value for this parameter is true, it is also possible to remove the attribute from the replication agreement entry.

```
changetype: modify
remove: nsds5ReplicaEnabled
```

- 2. Initiate an update, such as using the **replicate_now.sh** script as described in [Section 11.16.2, “Forcing Replication Updates from the Command Line”](#).

11.9. MANAGING ATTRIBUTES WITHIN FRACTIONAL REPLICATION

As [Section 11.1.7, “Replicating a Subset of Attributes with Fractional Replication”](#) describes, fractional replication allows administrators to set attributes that are *excluded* from replication updates. Administrators can do this for a variety of performance reasons — to limit the number of large attributes that are sent over a network or to reduce the number of times that fixup tasks (like **memberOf** calculations) are run.

The list of attributes to exclude from replication are defined in the **nsDS5ReplicatedAttributeList** attribute. This attribute is part of the replication agreement and it can be configured in the replication agreement wizard in the Directory Server Console (or through the command line) when the replication agreement is created.

```
nsDS5ReplicatedAttributeList: (objectclass=*) $ EXCLUDE memberof
authorityRevocationList accountUnlockTime
```

11.9.1. Setting Different Fractional Replication Attributes for Total and Incremental Updates

When fractional replication is first configured, the list of excluded attributes applies to every update operation. Meaning, this list of attributes is excluded for a total update as well as regular incremental updates. However, there can be times when attributes should be excluded from incremental updates for performance but should be included in a total update to ensure the directory data sets are complete. In this case, it is possible to add a second attribute that defines a separate list of attributes to exclude from total updates, **nsDS5ReplicatedAttributeListTotal**.



NOTE

nsDS5ReplicatedAttributeList is the primary fractional replication attribute. If only **nsDS5ReplicatedAttributeList** is set, then it applies to both incremental updates and total updates. If both **nsDS5ReplicatedAttributeList** and **nsDS5ReplicatedAttributeListTotal** are set, then **nsDS5ReplicatedAttributeList** only applies to incremental updates.

For example, every time a **memberOf** attribute is added to an entry, a **memberOf** fixup task is run to resolve the group membership. This can cause overhead on the server if that task is run every time replication occurs. Since a total update only occurs for a database which is newly-added to replication or that has been offline for a long time, running a **memberOf** fixup task after a total update is an acceptable option. In this case, the **nsDS5ReplicatedAttributeList** attribute lists **memberOf** so it is excluded from incremental updates, but **nsDS5ReplicatedAttributeListTotal** does not list **memberOf** so that it is included in total updates.

The exclusion list for incremental updates is set in the **nsDS5ReplicatedAttributeList** attribute for the replication agreement.

```
nsds5replicatedattributelist: (objectclass=*) $ EXCLUDE
authorityRevocationList accountUnlockTime memberof
```

If ***nsDS5ReplicatedAttributeList*** is the only attribute set, then that list applies to both incremental and total updates. To set a separate list for total updates, add the ***nsDS5ReplicatedAttributeListTotal*** attribute to the replication agreement.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -D
"cn=directory manager" -w secret -p 389 -h server.example.com -x

dn: cn=ExampleAgreement,cn=replica,cn=dc\=example\,dc\=com,cn=mapping
tree,cn=config
changetype: modify
add: nsDS5ReplicatedAttributeListTotal
nsDS5ReplicatedAttributeListTotal: (objectclass=*) $ EXCLUDE
accountUnlockTime
```



NOTE

The ***nsDS5ReplicatedAttributeList*** attribute must be set for incremental updates before ***nsDS5ReplicatedAttributeListTotal*** can be set for total updates.

11.9.2. Preventing "Empty" Updates from Fractional Replication

Fractional replication allows a list of attributes which are removed from replication updates (***nsDS5ReplicatedAttributeList***). However, a change to an excluded attribute still triggers a modify event and generates an empty replication update.

The ***nsds5ReplicaStripAttrs*** attribute adds a list of attributes which cannot be sent in an empty replication event and are stripped from the update sequence. This logically includes operational attributes like ***modifiersName***.

For example, let's say that the ***accountUnlockTime*** attribute is excluded. John Smith's user account is locked and then the time period expires and it is automatically unlocked. Only the ***accountUnlockTime*** attribute has changed, and that attribute is excluded from replication. However, the operational attribute ***internalmodifytimestamp*** also changed. A replication event is triggered because John Smith's user account was modified — but the only data to send is the new modify time stamp and the update is otherwise empty. If there are a large number of attributes related to login times or password expiration times (for example), this could create a flood of empty replication updates that negatively affect server performance or that interfere with associated applications.

To prevent this, add the ***nsds5ReplicaStripAttrs*** attribute to the replication agreement to help tune the fractional replication behavior:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -D
"cn=directory manager" -w secret -p 389 -h server.example.com -x

dn: cn=ExampleAgreement,cn=replica,cn=dc\=example\,dc\=com,cn=mapping
tree,cn=config
changetype: modify
add: nsds5ReplicaStripAttrs
nsds5ReplicaStripAttrs: modifiersname modifytimestamp
internalmodifiersname internalmodifytimestamp
```

If a replication event is *not* empty, the stripped attributes are still replicated with the other changes. These attributes are removed from updates only if the event would otherwise be empty.

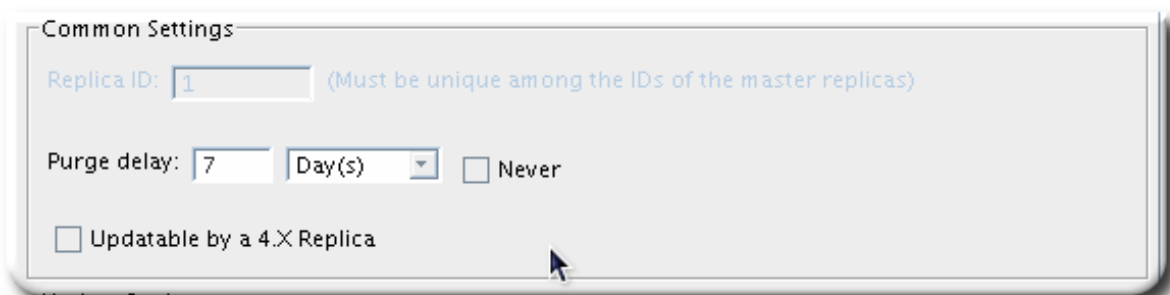
11.10. MAKING A READ-ONLY REPLICA UPDATABLE

Making a read-only server writable means changing the replica from a dedicated consumer or a hub to a supplier.

1. Make sure there are no updates in progress.
2. Stop the supplier server.
3. Open the Directory Server Console for the read-only replica.
4. In the **Configuration** tab, select **Replication**. In the right pane, select the **Enable changelog** check box.
5. Select the suffix, and, in the **Replica Settings** tab, change the replica role to a single master or multi-master.



6. Assign a unique replica ID.



7. Save the changes
 8. Stop the Directory Server instance:
- ```
systemctl stop dirsrv
```
9. Back up the `/etc/dirsrv/slapd-instance/dse.ldif` file:

```
cp /etc/dirsrv/slapd-instance/dse.ldif
 /etc/dirsrv/slapd-instance/dse.ldif-1
```



-

Do not name the backup file **dse.ldif.bak**. Directory Server uses this file name to keep a known working copy of the **dse.ldif** file.

10. Edit the **/etc/dirsrv/slapd-instance/dse.ldif** file.

a. Search for replication agreements. For example:

```
dn: cn=replica,cn=dc\5c3Dexample\5c2Cdc\5c3Dcom,cn=mapping
tree,cn=config
```

b. Remove the line containing the **nsState** attribute from every replication agreement.

11. Start the Directory Server instance:

```
systemctl start dirsrv.target
```

12. Monitor the error log file for error messages. For details, see [Section E.2.4, “Viewing Logs”](#).

If the replication fails:

a. Delete all replication agreements: [Section 11.11, “Removing a Supplier from the Replication Topology”](#).

b. Disable replication: [Section 11.8.1, “Disabling Replication”](#).

c. Remove the changelog configuration: [Section 11.13, “Removing the Changelog”](#).

d. Restart the Directory Server and Admin Console:

```
systemctl restart dirsrv.target
systemctl restart dirsrv-admin.service
```

e. Enable replication: [Section 11.8.2, “Re-enabling Replication”](#).

f. Create replication agreements: [Section 11.2, “Replication Scenarios”](#).

## 11.11. REMOVING A SUPPLIER FROM THE REPLICATION TOPOLOGY

Removing a supplier cleanly from the replication topology is more complex than simply removing the supplier entry. This is because every supplier in the topology stores information about other suppliers in the topology, and they retain that information even if a supplier is suddenly unavailable.

Information about the replication topology — all of the suppliers which are supplying updates to each other and other replicas within the same replication group — is contained in a set of metadata called the *replica update vector (RUV)*. The RUV contains information about the supplier like its ID and URL, its latest change state number for changes made on the local server, and the CSN of the first change. Both suppliers and consumers store RUV information, and they use it to control replication updates.

To remove a supplier cleanly, this metadata must be removed along with the configuration entries.

1. *On the replica to remove*, put the database into read-only mode to prevent any updates.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -p 389 -
```

```
h dead-replica.example.com
```

```
dn: cn=userRoot,cn=ldbm database,cn=plugins,cn=config
changetype: modify
replace: nsslapd-readonly
nsslapd-readonly: on
```

Allow a few minutes for the replica to flush all of its pending changes.

2. *On all other suppliers in the topology*, delete the replication agreement with the replica to be removed.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -p 389 -
h replica1.example.com
```

```
dn: cn=Agmt_with_dead-
replica,cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping
tree,cn=config
changetype: delete
```

3. *On the replica to remove*, get the replica ID for the replica to remove. This is in the **nsds5replicaid** attribute in the configuration entry.

```
ldapsearch -xLLL -D "cn=directory manager" -W -s sub -b cn=config
objectclass=nsds5replica nsds5replicaid
```

```
dn: cn=dead-replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping
tree,cn=config
nsds5replicaid: 55
...
```

4. *On the replica to remove*, remove all replication agreement entries and its own configuration entry.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -p 389 -
h dead-replica.example.com
```

```
dn: cn=to_replica1,cn=dead-
replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping tree,cn=config
changetype: delete
...
```

```
dn: cn=to_replica2,cn=dead-
replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping tree,cn=config
changetype: delete
```

```
dn: cn=dead-replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping
tree,cn=config
changetype: delete
```

5. *On one of the other servers in the topology*, run the **CLEANALLRUV** replication task. This operation will be propagated to all the servers in the replication environment. The task name has the format **CLEANALLRUV***replica#*. For example, for a supplier with the replica ID 55, the task name is **CLEANALLRUV55**.

```

ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=dead-replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping
tree,cn=config
changetype: modify
replace: nsds5task
nsds5task: CLEANALLRUV55

```

It is possible to monitor the progress of the task on the other replicas by searching on the tombstone entry on each replica:

```

ldapsearch -xLLL -D "cn=directory manager" -W -h remaining-
replica.example.com -b "dc=example,dc=com" '(&(nsuniqueid=ffffffff-
ffffffff-ffffffff-ffffffff)(objectclass=nstombstone))' nsds50ruv

```

## 11.12. MANAGING DELETED ENTRIES WITH REPLICATION

When an entry is deleted, it is not immediately removed from the database. Rather, it is converted into a *tombstone* entry, a kind of backup entry that is used by servers in replication to resolve conflicts. The tombstone entry preserves state information about the original entry.

If there is ever a replication conflict, then the supplier uses the replica ID (the server where the change was initiated) and the timestamp of the change in the change sequence number to resolve the conflict. The oldest change wins.

As with deleted entries, deleted attributes are also kept in tombstone entries.

Tombstones are not preserved indefinitely. A purge job is run periodically, at a specified interval (set in the ***nsDS5ReplicaTombstonePurgeInterval*** attribute); the purge removes old tombstone entries. Tombstone entries are saved for a given amount of time (set in the ***nsDS5ReplicaPurgeDelay*** attribute); once a tombstone entry is older than the delay period, it is reaped at the next purge job.

Both the purge delay and the purge interval are set on the replica entry for a supplier server in the ***cn=replica,cn=replicated suffix,cn=mapping tree,cn=config*** configuration entry.

There are two considerations when defining the purge settings for replication:

- The purge operation is time-consuming, especially if the server handles a lot of delete operations. Do not set the purge interval too low or it could consume too many server resources and affect performance.
- Suppliers use change information, including tombstone entries, to prime replication after initialization. There should be enough of a backlog of changes to effectively re-initialize consumers and to resolve replication conflicts. Do not set the purge delay (the age of tombstone entries) too low or you could lose information required to resolve replication conflicts.

Set the purge delay so that it is slightly longer than the longest replication schedule in the replication topology. For example, if the longest replication interval is 24 hours, keep tombstone entries around for 25 hours. This ensures that there is enough change history to initialize consumers and prevent the data stored in different suppliers from diverging.

To change the purge settings:

```

[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h

```

```
supplier1.example.com
```

```
dn: cn=replica,cn=dc\=example\,dc\=com,cn=mapping tree,cn=config
changetype: modify
replace: nsDS5ReplicaTombstonePurgeInterval
nsDS5ReplicaTombstonePurgeInterval: 43200 # in seconds, 12 hours
```

```
changetype: modify
replace: nsDS5ReplicaPurgeDelay
nsDS5ReplicaPurgeDelay: 90000 # in seconds, 25 hours
```



## NOTE

To clean up the tombstone entries and the state information immediately, set a very small value to the ***nsDS5ReplicaTombstonePurgeInterval*** and ***nsDS5ReplicaPurgeDelay*** attributes. Both attributes have values set in seconds, so the purge operations can be initiated almost immediately.



## WARNING

Always use the purge intervals to clean out tombstone entries from the changelog. **Never delete tombstone entries manually.**

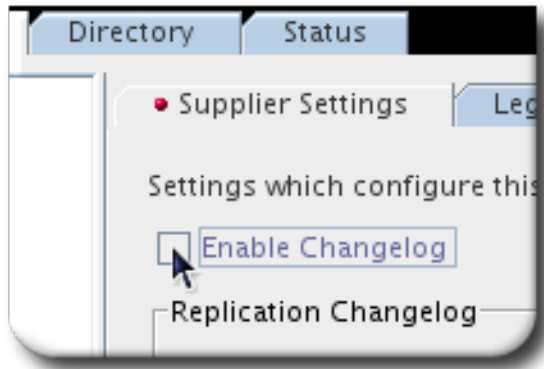
## 11.13. REMOVING THE CHANGELOG

The changelog is a record of all modifications on a given replica that the supplier uses to replay these modifications to replicas on consumer servers (or suppliers in the case of multi-master replication).

If a supplier server goes offline, it is important to be able to delete the changelog because it no longer holds a true record of all modifications and, as a result, should not be used as a basis for replication. A changelog can be effectively deleted by deleting the log file.

To remove the changelog from the supplier server:

1. In the Directory Server Console, select the **Configuration** tab.
2. Select the **Replication** folder in the left navigation tree and then the **Supplier Server Settings** tab in the right pane.
3. Clear the **Enable Changelog** check box.



4. Click **Save**.
5. Restart Directory Server. See [Section 1.3.1, “Starting and Stopping Directory Server from the Console”](#).
6. Reinitialize the consumers. See [Section 11.15, “Initializing Consumers”](#).

### 11.13.1. Moving the Changelog to a New Location

Sometimes, it may be necessary to delete the changelog while the supplier server is still running and continuing to log changes.

In this case, simply move the current changelog file to a new location. The server then automatically creates a new changelog in the specified directory.

After moving the changing, reinitialize the consumers; any changes to the changelog requires consumer reinitialization.

## 11.14. TRIMMING THE REPLICATION CHANGELOG

The Directory Server changelog manages a list of received and processed changes. It includes changes clients run on the server and additionally directory changes from other replication partners. Changelog entries are removed only when both of the following conditions meet:

- The entry was successfully transferred to all replicas.
- The entry exceeded the time set in the ***nsslapd-changelogmaxage*** parameter or the total number of records exceeded the value set in ***nsslapd-changelogmaxentries***. For further details, see the parameter descriptions in the [Red Hat Directory Server Configuration, Command, and File Reference](#).

A record, including all subsequently created records, remains in the changelog until it is successfully replicated to all servers in the topology. This situation occurs, for example, if a Directory Server master was removed from the topology, but the replica update vector (RUV) has not been removed.

The file size of the changelog is not automatically reduced if you set a lower value in the ***nsslapd-changelogmaxage*** or ***nsslapd-changelogmaxentries*** parameter. To recreate the changelog:

1. Stop the Directory Server instance:

```
service dirsrv stop instance_name
```

2. Remove the changelog files:

```
rm /var/lib/dirsrv/slaped-instance_name/changelogdb/*
```

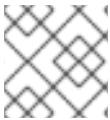
3. Start the instance:

```
service dirsrv start instance_name
```

4. Reinitialize the instance from an incoming replica. See [Section 11.15, “Initializing Consumers”](#).

## 11.15. INITIALIZING CONSUMERS

Once a replication agreement is created, the consumer must be *initialized*; that is, the data must be physically copied from the supplier server to the consumer servers.



### NOTE

Replication *will not* begin until the consumer is initialized.

- [Section 11.15.1, “When to Initialize a Consumer”](#)
- [Section 11.15.2, “Online Consumer Initialization Using the Console”](#)
- [Section 11.15.3, “Initializing Consumers Online Using the Command Line”](#)
- [Section 11.15.4, “Manual Consumer Initialization Using the Command Line”](#)
- [Section 11.15.5, “Filesystem Replica Initialization”](#)



### NOTE

For large databases, the ***nsslapd-idletimeout*** setting must be set to a large enough time period (or even an unlimited period) to allow the entire database to be initialized before the operation times out. Alternatively, the ***nsIdleTimeout*** setting for the supplier bind DN entry can be set high enough to allow the online initialization operation to complete, without having to change the global setting.

### 11.15.1. When to Initialize a Consumer

Consumer initialization involves copying data from the supplier server to the consumer server. Once the subtree has been physically placed on the consumer, the supplier server can begin replaying update operations to the consumer server.

Under normal operations, the consumer should not ever have to be reinitialized. However, any time there is a chance that there is a big discrepancy between the supplier's data and the consumer's, reinitialize the consumer. For example, if the data on the supplier server is restored from backup, then all consumers supplied by that server should be reinitialize. As another example, if the supplier has not been able to contact the consumer for a long time, like a week, the supplier may determine that the consumer is too far out of date to be updated, and must be reinitialized.

The consumer can either be initialized online using the Console or manually using the command line. Online consumer initialization using the Console is an effective method of initializing a small number of consumers. However, since each replica is initialized in sequence, this method is not suited to initializing a large number of replicas. Online consumer initialization is the method to use when the consumer is initialized as part of configuring the replication agreement on the supplier server.

Manual consumer initialization using the command line is a more effective method of initializing a large number of consumers from a single LDIF file.

### 11.15.2. Online Consumer Initialization Using the Console

Online consumer initialization using the Console is the easiest way to initialize or reinitialize a consumer. However, for replicating across a slow link, this process can be very time-consuming, and manual consumer initialization using the command line may be a more efficient approach. This is described in more detail [Section 11.15.4, “Manual Consumer Initialization Using the Command Line”](#).

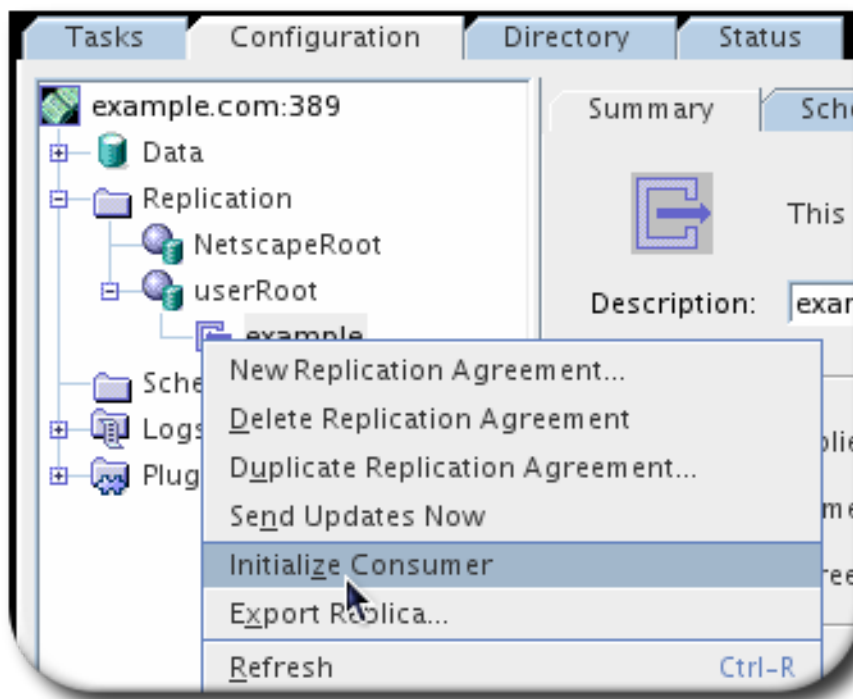


#### NOTE

When a consumer server is being initialized using the online consumer creation method, all operations (including searches) on the replica are referred to the supplier server until the initialization process is completed.

To initialize or reinitialize a consumer online:

1. Create a replication agreement.
2. On the supplier server, on the Directory Server Console, select the **Configuration** tab.
3. Expand the **Replication** folder, then expand the replicated database. Right-click the replication agreement, and choose **Initialize Consumer** from the pop-up menu.



A message opens warning that any information already stored in the replica on the consumer will be removed.

4. Click **Yes** in the confirmation box.

Online consumer initialization begins immediately. To check the status of the online consumer initialization, open the **Summary** tab in the **Status** box. If online consumer initialization is in progress, the status shows that a replica is being initialized.



## IMPORTANT

For multi-master replication, be sure that consumers are only initialized **once**, by one supplier. When checking the replication status, be sure to check the replication agreement entry, on the appropriate supplier, which was used to initialize the consumer.

To update this window, right-click the replicated database icon in the navigation tree, and choose **Refresh Replication Agreements**. When online consumer initialization finishes, the status changes to reflect this.

For more information about monitoring replication and initialization status, see [Section 11.22, “Monitoring Replication Status”](#).

### 11.15.3. Initializing Consumers Online Using the Command Line

Online consumer initialization can be performed through the command line by adding the ***nsds5BeginReplicaRefresh*** attribute to the replication agreement entry. This attribute is absent by default, and it will be automatically deleted once the consumer initialization is complete.

1. Find the DN of the replication agreement on the supplier server that is for the consumer to be initialized. For example:

```
ldapsearch -h supplier1.example.com -p 389 -D "cn=directory manager"
-W -s sub
-b cn=config "(objectclass=nsds5ReplicationAgreement)"
```

This command returns all of the replication agreements configured on the supplier in LDIF format. Get the DN of the replication agreement with the consumer to be initialized. This is the replication agreement which will be edited.

2. Edit the replication agreement, and add the ***nsds5BeginReplicaRefresh*** attribute:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h
supplier1.example.com

dn:
cn=ExampleAgreement,cn=replica,cn=dc\=example\,dc\=com,cn=mapping
tree,cn=config
changetype: modify
replace: nsds5BeginReplicaRefresh
nsds5BeginReplicaRefresh: start
```

**ldapmodify** does not prompt for input; simply type in the LDIF statement, and then hit enter twice when the LDIF statement is complete. Close the **ldapmodify** utility by hitting **Ctrl+C**.

To check the initialization status, do an **ldapsearch** for the replication agreement entry.

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -s base -b
'cn=ExampleAgreement,cn=dc\=example\,dc\=com,cn=mapping tree,cn=config'
'(objectclass=*)'
```

If the ***nsds5BeginReplicaRefresh*** attribute is present, the initialization is still in progress. If the initialization is complete, then the attribute ***nsds5ReplicaLastInitStatus*** shows the status. If the



initialization was successful, the value of `nsds5ReplicaLastInitStatus` is **Total update succeeded**. If the initialization was not successful, this attribute shows information about the error; check the error logs for both the supplier and consumer for additional information.



## IMPORTANT

For multi-master replication, be sure that consumers are only initialized **once**, by one supplier. When checking the replication status, be sure to check the replication agreement entry, on the appropriate supplier, which was used to initialize the consumer.

The replication monitoring attributes are described in more detail in the *Directory Server Configuration and Command-Line Tool Reference*.

### 11.15.4. Manual Consumer Initialization Using the Command Line

Manual consumer initialization using the command line is the fastest method of consumer initialization for sites that are replicating very large numbers of entries. However, the manual consumer initialization process is more complex than the online consumer initialization process. Red Hat suggests using the manual process whenever the online process is inappropriate due to performance concerns.

Initializing or reinitializing a server manually has three steps:

1. Create a replication agreement.
2. Export the replica on the supplier server to an LDIF file.  
See [Section 11.15.4.1, “Exporting a Replica to LDIF”](#).
3. Import the LDIF file with the supplier replica contents to the consumer server.  
See [Section 11.15.4.2, “Importing the LDIF File to the Consumer Server”](#).

#### 11.15.4.1. Exporting a Replica to LDIF

There are three ways to convert a replica database to LDIF:

- When creating a replication agreement, by selecting **Create consumer initialization file** in the **Initialize Consumer** dialog box of the **Replication Agreement Wizard**.
- From the Directory Server Console, by right-clicking the replication agreement under the **Replication** folder and choosing **Create LDIF File** from the pop-up menu.
- From the command line by using the export command, as described in [Section 4.2.3, “Exporting to LDIF from the Command Line”](#). Exporting to LDIF with any of the command-line tools requires using an option to export the database as a replica; this means that the exported LDIF contains the proper entries to initialize the consumer when the LDIF is imported.

For the `db2ldif` and `db2ldif.pl` scripts, this is the `-r` option. For example:

```
[root@server ~]# db2ldif -r -n database1 -a /export/output.ldif
```

For the `cn=export`, `cn=tasks`, `cn=config` entry, this is the `nsExportReplica` attribute.

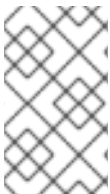
```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
```

```
server.example.com -x

dn: cn=example export,cn=export,cn=tasks,cn=config
changetype: add
objectclass: extensibleObject
cn: example export
nsInstance: userRoot
nsFilename: /home/files/example.ldif
nsExportReplica: true
```

#### 11.15.4.2. Importing the LDIF File to the Consumer Server

Import the LDIF file which contains the supplier replica contents to the consumer server by using the import features in the Directory Server Console or by using either the **ldif2db** script or **ldif2db.pl** script. Both import methods are described in [Section 4.1.6, “Importing from the Command Line”](#).



#### NOTE

With the **ldif2db.pl** script, the LDIF file import operation does not require a server restart. For more information on command-line scripts, see the *Directory Server Configuration and Command-Line Tool Reference*.

#### 11.15.5. Filesystem Replica Initialization

A very large database, such as one with several million entries, can take an hour or more to initialize a consumer from the Console or even with manual initialization. To save time, use *filesystem replica initialization*.

Directory Server has the capability to initialize a replica using the database files from the supplier server. This avoids the need to rebuild the consumer database and can be done at essentially the speed of the network between the two servers by transferring the files with FTP or NFS, for example. Instead of sending entries using LDAP to replica servers, filesystem replica initialization populates the new database on the destination server by *backing up* the supplier database on one server and *restoring* the database on the destination server.

This method of initializing consumers is especially useful in replication over wide-area networks or over networks with slow or unstable connections.

For smaller databases, Red Hat recommends using manual initialization or initialize consumers from the Console.



#### NOTE

The destination server must have the same architecture and the same bit size as the supplier server for the initialization to succeed. For example, Red Hat Enterprise Linux 6 (64-bit) to Red Hat Enterprise Linux 6 (64-bit).

### 11.16. FORCING REPLICATION UPDATES

When a Directory Server involved in replication is stopped for regular maintenance, it must be updated immediately when it comes back online. In the case of a supplier in a multi-master environment, the directory information needs to be updated by the other supplier in the multi-master set. In other cases, when a hub or a dedicated consumer is taken offline for maintenance, when they come back online, they need to be updated by the supplier server.

Even if the replication agreements are configured to keep the supplier and consumer servers always in sync, it is not sufficient to bring back up-to-date a server that has been offline for over five minutes. The **Always Keep in Sync** option means that the server generates a replication operation for every update operation it processes. However, if this replication operation cannot be performed because the consumer is offline, the operation times out after 10 minutes.



## NOTE

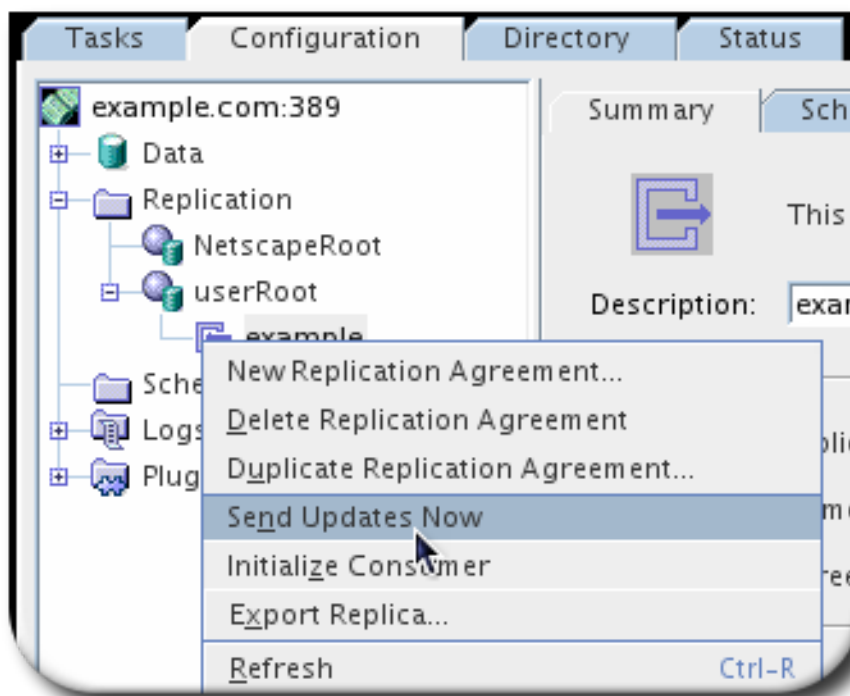
The procedures described in this section can only be used when replication is already set up and consumers have been initialized.

To ensure that directory information will be synchronized immediately when a server comes back online, use either the Directory Server Console on the supplier server that holds the reference copy of the directory information or a customizable script.

### 11.16.1. Forcing Replication Updates from the Console

To ensure that replication updates are sent immediately when a consumer or a supplier in a multi-master replication configuration comes back online after a period of time, do the following on the supplier server that holds the most recent version of the directory information:

1. In the Directory Server Console, click the **Configuration** tab, expand the **Replication** folder and database nodes, and select the replication agreement corresponding to the replica to update.
2. Right click the replication agreement, and choose **Send Updates Now** from the drop-down list.

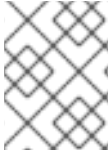


This initiates replication toward the server that holds the information that needs to be updated.

### 11.16.2. Forcing Replication Updates from the Command Line

From the consumer that requires updating, run a script that prompts the supplier to send replication updates immediately. This script is shown in [Example 11.5, “replicate\\_now.sh Script Example”](#).

Copy this example script and name it something like **replicate\_now.sh**. Substitute the actual values for the variables listed in [Example 11.5, “replicate\\_now.sh Script Example”](#).



## NOTE

This script must be run manually since it cannot be configured to run automatically as soon as the server, which was offline, comes back online again.

### Example 11.5. replicate\_now.sh Script Example

```
#!/bin/sh
SUP_HOST=supplier_hostname
SUP_PORT=supplier_portnumber
SUP_MGRDN=supplier_directoryManager
SUP_MGRPW=supplier_directoryManager_password
MY_HOST=consumer_hostname
MY_PORT=consumer_portnumber

ldapsearch -x -LLL -h ${SUP_HOST} -p ${SUP_PORT} -D "${SUP_MGRDN}" \
 -w ${SUP_MGRPW} -b "cn=mapping tree,cn=config" \
 "(&(objectclass=nsds5replicationagreement) \
 (nsDS5ReplicaHost=${MY_HOST}) \
 (nsDS5ReplicaPort=${MY_PORT}))" \
 -o ldif-wrap=no dn | grep "^dn: " > /tmp/$$dn

cp /tmp/$$dn /tmp/$$off
cp /tmp/$$dn /tmp/$$on

cat >> /tmp/$$off <<EOF
changetype: modify
replace: nsds5ReplicaEnabled
nsds5ReplicaEnabled: off
EOF

cat >> /tmp/$$on<<EOF
changetype: modify
replace: nsds5ReplicaEnabled
nsds5ReplicaEnabled: on
EOF

ldapmodify -x -h ${SUP_HOST} -p ${SUP_PORT} -D "${SUP_MGRDN}" \
 -w ${SUP_MGRPW} -f /tmp/$$off

sleep 1

ldapmodify -x -h ${SUP_HOST} -p ${SUP_PORT} -D "${SUP_MGRDN}" \
 -w ${SUP_MGRPW} -f /tmp/$$on

rm -f /tmp/$$.*
```

**Table 11.4. Replicate\_Now Variables**

| Variable                                  | Definition                                                                                               |
|-------------------------------------------|----------------------------------------------------------------------------------------------------------|
| <i>supplier_hostname</i>                  | Hostname of the supplier to contact for information on replication agreements with the current consumer. |
| <i>supplier_portnumber</i>                | LDAP port in use on the supplier.                                                                        |
| <i>supplier_directoryManager</i>          | DN of the privileged Directory Manager user on the supplier.                                             |
| <i>supplier_directoryManager_password</i> | Password of the privileged Directory Manager user on the supplier.                                       |
| <i>consumer_hostname</i>                  | Hostname of the current consumer.                                                                        |
| <i>consumer_portnumber</i>                | LDAP port in use on the consumer.                                                                        |

For the update operation to occur over an SSL connection, modify the **ldapmodify** command in the script with the appropriate parameters and values. For more information on the **ldapmodify** command, see [Section 3.2, “Managing Entries from the Command Line”](#).

## 11.17. REPLICATION OVER SSL

The Directory Servers involved in replication can be configured so that all replication operations occur over an SSL connection. To use replication over SSL:

- Configure both the supplier and consumer servers to use SSL.
- Configure the consumer server to recognize the supplier server's certificate as the supplier DN. Do this only to use SSL client authentication rather than simple authentication.

These procedures are described in [Section 7.4, “Setting up TLS/SSL”](#).

If attribute encryption is enabled, a secure connection is required for replication.



### NOTE

Replication configured over SSL with certificate-based authentication will fail if the supplier's certificate is only capable of behaving as a server certificate, and not also a client during an SSL handshake. Replication with certificate-based authentication uses the Directory Server's server certificate for authentication to the remote server.

When the servers are configured to use SSL, configure an SSL connection for replication in the **Replication Agreement Wizard**. The **Source** and **Destination** sets how to bind between the supplier and the consumer, and this is where SSL is set.

There are two ways to use SSL for replication:

- Select **SSL Client Authentication**.

With SSL client authentication, the supplier and consumer servers use certificates to authenticate to each other.

- Select **Simple Authentication**.

With simple authentication, the supplier and consumer servers use a bind DN and password to authenticate to each other, which are supplied in the **Replication Agreement Wizard** text fields provided. Simple authentication takes place over a secure channel but without certificates.



#### NOTE

If secure binds are required for simple password authentication ([Section 14.8.1, “Requiring Secure Binds”](#)), then any replication operations will fail unless they occur over a secure connection. Using a secure connection (SSL/TLS and Start TLS connections or SASL authentication) is recommended, anyway.

Once a replication agreement is created, the connection type (SSL or non SSL) cannot be changed in the agreement because LDAP and LDAPS connections use different ports. To change the connection type, re-create the replication agreement.

Also, the port listed for the consumer is the non-SSL port, even if the Directory Server instance is configured to run over SSL. This port number is used only for identification of the Directory Server instance in the Console; it does not specify the actual port number or protocol that is used for replication.

## 11.18. SETTING REPLICATION TIMEOUT PERIODS

Suppliers must have an exclusive connection to a consumer to send updates to the directory. As mentioned in [Section 11.5.4, “Preventing Monopolization of the Consumer in Multi-Master Replication”](#), it is possible to configure a wait time for suppliers attempting to connect to a consumer, so that the supplier does not hang while the consumer is tied up with another supplier.

It is also possible to set a timeout period for a supplier, so that it does not stay connected to a consumer interminably attempting to send updates over a slow or broken connection.

There are two attributes which set the timeout period:

- ***nsDS5ReplicaTimeout*** sets the number of seconds that the replication operation waits for a response from the consumer before timing out and failing. To set the optimum number, check the access logs to see the average amount of time that the replication process takes, and set the timeout period accordingly.
- ***nsDS5DebugReplicaTimeout*** sets the timeout period for the replication operation when debug logging is enabled. This setting may be appreciably higher than the ***nsDS5ReplicaTimeout*** setting because debug logging can slow down directory operations. This attribute can optionally set an error log level where this parameter is applied; the default is replication debugging (8192).



#### NOTE

The timeout period is limited to the maximum 32-bit integer in seconds, which translates to 24.8 days.

Both of these attributes are set in the configuration for the replicated suffix. For example, this sets timeout periods for the **ou=People** suffix:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x

dn: cn=replica,cn=ou=People\,dc=example\,dc=com,cn=mapping tree,cn=config
changetype: modify
add: nsDS5ReplicaTimeout
nsDS5ReplicaTimeout: 600
add: nsDS5DebugReplicaTimeout
nsDS5DebugReplicaTimeout: 6000
```

## 11.19. REPLICATING O=NETSCAPEROOT FOR ADMIN SERVER FAILOVER

Replication usually occurs between Directory Server user databases to distribute directory data, but it is also possible to use replication to provide failover support for the Admin Server database, **o=NetcapeRoot**.

1. Install and configure the first Directory Server instance.

The **setup-ds-admin.pl** script has an option, **-f**, which references an **inf**. The **inf** can be used to import LDIF files through the **ConfigFile** parameter, and the LDIF files can create databases, suffixes, and replication entries. (The **inf** file is described in more detail in the *Directory Server Installation Guide*.)

```
[root@server ~]# setup-ds-admin.pl -f /tmp/server1.inf
```

To configure the **o=NetcapeRoot** database on **server1** as a multi-master supplier replica, use the following statements in the **inf** file:

```
[slapd]
...
ConfigFile = repluser.ldif Example 11.1, "Example Supplier Bind DN Entry"
ConfigFile = changelog.ldif Example 11.2, "Example Changelog Entry"

ConfigFile = replica.ldif Example 11.3, "Example Supplier Replica Entry"
ConfigFile = replagreement.ldif Example 11.4, "Example Replication Agreement Entry"
...
```

2. Install and configure the second Directory Server instance. For the second server, **server2.example.com**, use the **setup-ds.pl** command, which installs a Directory Server instance without installing a local Admin Server.

```
[root@server ~]# setup-ds.pl -f /tmp/server2.inf
```

With **server2**, use the **inf** file to create and configure a **o=NetcapeRoot** database on **server2** as a multi-master supplier replica:

```
[slapd]
...
ConfigFile = netscaperootdb.ldif Example 2.1, "Example Root Suffix"
```



*Entry"*

ConfigFile = repluser.ldif *Example 11.1, "Example Supplier Bind DN Entry"*

ConfigFile = changelog.ldif *Example 11.2, "Example Changelog Entry"*

ConfigFile = replica.ldif *Example 11.3, "Example Supplier Replica Entry"*

ConfigFile = replagreement.ldif *Example 11.4, "Example Replication Agreement Entry"*

...

3. Initialize the **o=NetscapeRoot** database on **server2** from **server1**. Add the **nsds5replicarefresh** attribute to the replication agreement on **server1**.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h
supplier1.example.com
```

```
dn: cn=ExampleAgreement1,cn=replica,cn=o=NetscapeRoot,cn=mapping
tree,cn=config
changetype: modify
replace: nsds5beginreplicarefresh
nsds5beginreplicarefresh: start
```

4. Run the **register-ds-admin.pl** to create a local Admin Server on **server2** and switch the configuration directory for **server2** to its own **o=NetscapeRoot** database from **server1**.

```
[root@server ~]# register-ds-admin.pl
```

5. Add the following access control instructions (ACI) on **server2**, to enable members of the **Configuration Administrators Group**, the server instance entry **SIE group**, and the **admin** user, to run on suffixes belonging to **server2**. For example, to run on the **dc=example,dc=com** suffix, enter:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h
server2.example.com
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="*)(version 3.0; acl "Configuration Administrators
Group";
 allow (all) groupdn="ldap:///cn=Configuration
Administrators,ou=Groups,
 ou=TopologyManagement,o=NetscapeRoot";)
-
add: aci
aci: (targetattr="*)(version 3.0; acl "Configuration
Administrator";
 allow (all) userdn="ldap:///uid=admin,
 ou=Administrators,ou=TopologyManagement,o=NetscapeRoot";)
-
add: aci
aci: (targetattr = "*)(version 3.0; acl "SIE Group"; allow (all)
groupdn =
```



```
"ldap:///cn=slapd-instance,cn=Red Hat Directory Server,cn=Server
Group,
cn=machine_name,ou=example.com,o=NetscapeRoot";)
```

6. Disable the PTA Plug-in on **server2** so that it does not pass bind operations for the administrative users in its **o=NetscapeRoot** to **server1**.

See [Section 1.8.1, “Enabling Plug-ins in the Directory Server Console”](#).

## 11.20. REPLICATION WITH EARLIER RELEASES

This section provides information on how to optimize replication with earlier releases of Directory Server.

### 11.20.1. Using Legacy Replication

Directory Server 9.0 can be involved in replication with earlier releases of Directory Server, providing the following conditions are met:

- Directory Server 9.0 is a consumer.
- The legacy suppliers can be Directory Server 4.0, 4.1, and 4.1x.

The following restrictions apply:

- A legacy Directory Server and Directory Server 9.0 cannot update the same replica. However, this version of Directory Server can have different replicas, where one is supplied by a legacy Directory Server and the other is supplied by Directory Server 9.0.
- Directory Server 9.0 cannot be a supplier for other replicas.

The main advantage of using Directory Server 9.0 as a consumer of a legacy Directory Server is to ease the migration of a replicated environment, especially since migration is not supported from 4.x servers to 9.0. For more information on migration, see the *Directory Server Installation Guide*.

### 11.20.2. Legacy Replication and Parent Object Classes

There is one important difference between the way that Directory Server 4.x servers handle replicated entries and the way that Directory Server 9.0 handles replicated entries. In Directory Server 4.x, entries could be added without specifying parent object classes, and when those entries were modified or replicated, the server would not automatically insert those parent object classes. For example, a user could be added with the **inetorgperson** object class, but not the **top** or **person** object classes:

```
dn: uid=jsmith,ou=People,dc=example,dc=com
objectclass: inetorgperson
uid: jsmith
cn: John Smith
sn: Smith
```

However, in Directory Server 9.0, the parent object classes are automatically added to the entry when the entry is added, modified, or replicated. This means that the entries will be slightly different on the Directory Server 4.x supplier and the Directory Server 9.0 consumer, because the Directory Server 9.0 entry will have the parent object classes:

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

```

objectclass: top
objectclass: person
objectclass: inetorgperson
uid: jsmith
cn: John Smith
sn: Smith

```

### 11.20.3. Configuring Legacy Replication

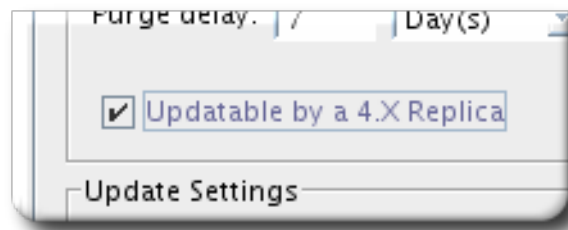
To set up legacy replication:

1. In the Directory Server Console, click the **Configuration** tab.
2. Select the **Replication** node, and click the **Legacy Consumer Settings** tab in the right pane.
3. Check the **Enable Legacy Consumer** check box.

The screenshot shows the 'Legacy Consumer Settings' tab in the Directory Server Console. The 'Enable Legacy Consumer' checkbox is checked. Below it, the 'Authentication' section contains three input fields: 'Supplier DN' with the value 'cn=legacy repl user', 'New supplier password (min 8 char):' with masked characters, and 'Confirm new supplier password:' with masked characters.

This activates the fields in the **Authentication** box.

4. Specify the supplier bind DN that the legacy supplier server will use to bind to the consumer.  
Optionally, specify a password at least 8 characters long.
5. Click **Save**.
6. Now configure legacy consumer settings for each replica that will receive updates from a legacy supplier.
  1. In the navigation tree, expand the **Replication** node, and select a replica that will receive updates from the legacy supplier.
  2. In the **Common Settings** area, select the **Enable Replica** and **Updatable by a 4.x Replica** check boxes.



These options are the only ones required for replication to work. Optionally, specify a replica ID. It is not necessary to specify a supplier DN because the one specified in step 4 will be used.

3. Click **Save**.
7. Repeat step 6 for each read-only replica that will receive updates from a legacy supplier.
8. To complete the legacy replication setup, configure the legacy supplier to replicate to the Directory Server 9.0 instance. For instructions on configuring a replication agreement on a 4.x Directory Server, see the documentation for the legacy Directory Server.



## NOTE

The Directory Server Console will not prevent you from configuring a database as a read-write replica and enabling legacy consumer settings. This makes migration easier because the Directory Server can be configured as it should be after the migration and legacy consumer settings only have to be active for the duration of the transition.

## 11.21. USING THE RETRO CHANGELOG PLUG-IN

The Retro Changelog plug-in configures Directory Server to maintain a changelog that is compatible with the changelog implemented in Directory Server 4.0, 4.1, and 4.1x. Maintaining a retro changelog is essential to maintain a changelog for directory clients that depend on a Directory Server 4.x-style changelog.

To use the retro changelog plug-in, the Directory Server 9.0 instance must be configured as a single-master replica.

When the Directory Server is configured to maintain a retro changelog, this changelog is stored in a separate database under a special suffix, **cn=changelog**.

The retro changelog consists of a single level of entries. Each entry in the changelog has the object class **changeLogEntry** and can include the attributes listed in [Table 11.5, “Attributes of a Retro Changelog Entry”](#).

**Table 11.5. Attributes of a Retro Changelog Entry**

| Attribute    | Definition                                                                                                                                                                                                                   |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| changeNumber | This single-valued attribute is always present. It contains an integer which uniquely identifies each change. This number is related to the order in which the change occurred. The higher the number, the later the change. |

| Attribute    | Definition                                                                                                                                                                                                                 |
|--------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| targetDN     | This attribute contains the DN of the entry that was affected by the LDAP operation. In the case of a <b>modrdn</b> operation, the <b>targetDN</b> attribute contains the DN of the entry before it was modified or moved. |
| changetype   | Specifies the type of LDAP operation. This attribute can have a value of add, delete, modify, or <b>modrdn</b> .                                                                                                           |
| changes      | For add and modify operations, contains the changes made to the entry in LDIF format.                                                                                                                                      |
| newrdn       | In the case of <b>modrdn</b> operations, specifies the new RDN of the entry.                                                                                                                                               |
| deleteoldrdn | In the case of <b>modrdn</b> operations, specifies whether the old RDN was deleted.                                                                                                                                        |
| newSuperior  | In the case of <b>modrdn</b> operations, specifies the <b>newSuperior</b> attribute of the entry.                                                                                                                          |

This section contains information on the following retro changelog items:

- [Section 11.21.1, “Enabling the Retro Changelog Plug-in”](#)
- [Section 11.21.2, “Trimming the Retro Changelog”](#)
- [Section 11.21.3, “Searching and Modifying the Retro Changelog”](#)
- [Section 11.21.4, “Retro Changelog and the Access Control Policy”](#)

### 11.21.1. Enabling the Retro Changelog Plug-in

The retro changelog plug-in configuration information is stored in the **cn=Retro Changelog Plugin,cn=plugins,cn=config** entry in **dse.ldif**. To enable the retro changelog plug-in from the command line:

1. Create an LDIF file that contains the following LDIF update statements:

```
dn: cn=Retro Changelog Plugin,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
```

2. Use the **ldapmodify** command to import the LDIF file into the directory.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x -f retro.ldif
```

### 3. Restart the server.

For information on restarting the server, see [Section 1.3, “Starting and Stopping Servers”](#).

The retro changelog is created in the directory tree under a special suffix, **cn=changelog**.

The procedure for enabling the retro changelog plug-in from Directory Server Console is the same as for all Directory Server plug-ins. For information, see [Section 1.8.1, “Enabling Plug-ins in the Directory Server Console”](#).

#### 11.21.2. Trimming the Retro Changelog

The size of the retro changelog is automatically reduced if you lower the maximum age of records set in the **nsslapd-changelogmaxage** parameter and the next trim interval, set in **nsslapd-changelog-trim-interval**, is executed.

For example, to set maximum age of records in the retro changelog to two days:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=Retro Changelog Plugin,cn=plugins,cn=config
changetype: modify
replace: nsslapd-changelogmaxage
nsslapd-changelogmaxage: 2d
```

#### 11.21.3. Searching and Modifying the Retro Changelog

The changelog supports search operations and is optimized for searches that include filters of the form **(&(changeNumber>=X)(changeNumber<=Y))**.

As a general rule, do not perform add or modify operations on the retro changelog entries, although entries can be deleted to trim the size of the changelog. Only modify the retro changelog entry to modify the default access control policy.

#### 11.21.4. Retro Changelog and the Access Control Policy

When the retro changelog is created, the following access control rules apply by default:

- Read, search, and compare rights are granted to all authenticated users (**userdn=anyone**, not to be confused with anonymous access where **userdn=all**) to the retro changelog top entry **cn=changelog**.
- Write and delete access are not granted, except implicitly to the Directory Manager.

Do not grant read access to anonymous users because the changelog entries can contain modifications to sensitive information, such as passwords. Only authenticated applications and users should be allowed to access this information.

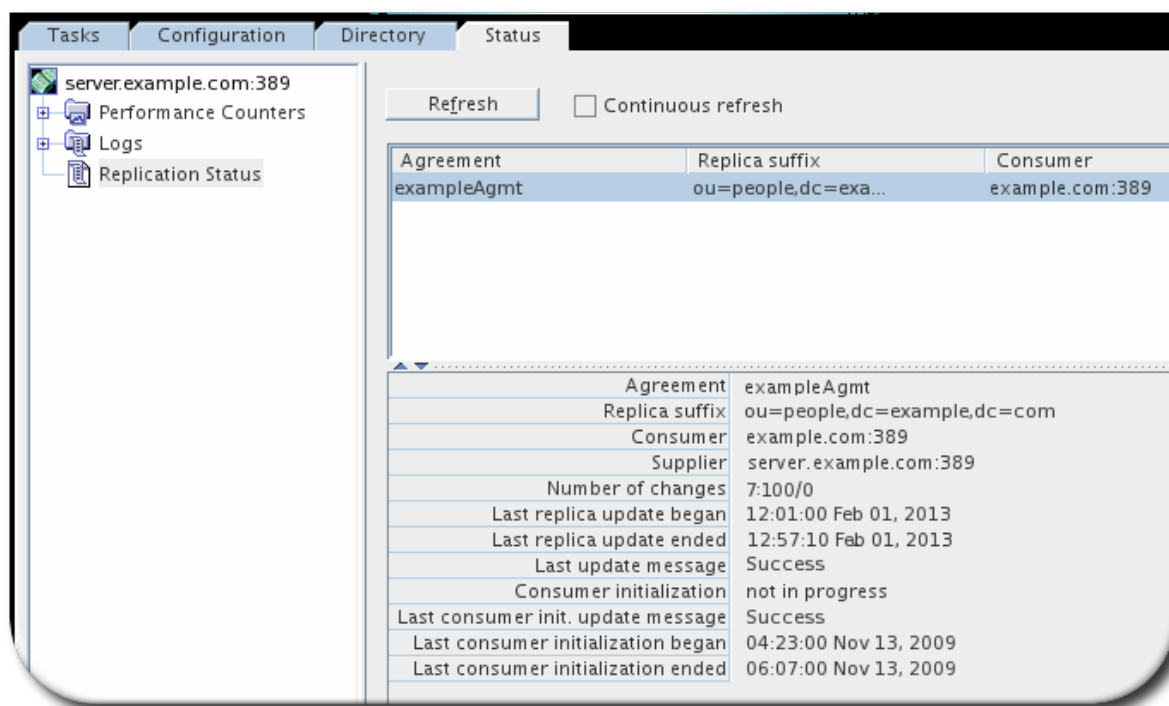
To modify the default access control policy which applies to the retro changelog, modify the **aci** attribute of the **cn=changelog** entry.

## 11.22. MONITORING REPLICATION STATUS

The replication status can be viewed in the Directory Server Console or Red Hat Administration Express (Section 11.22.2, “Monitoring Replication from Admin Express”).

### 11.22.1. Monitoring Replication Status from the Console

1. Select the **Status** tab, and then, in the left navigation tree, select **Replication Status**.



In the right pane, a table appears that contains information about each of the replication agreements configured for this server.

2. Click **Refresh** to update the contents of the tab.

The status information displayed is described in Table 11.6, “Directory Server Console Replication Status”.

**Table 11.6. Directory Server Console Replication Status**

| Table Header   | Description                            |
|----------------|----------------------------------------|
| Agreement      | The name of the replication agreement. |
| Replica suffix | The suffix that is replicated.         |
| Supplier       | The supplier server in the agreement.  |
| Consumer       | The consumer server in the agreement.  |

| Table Header                                | Description                                                                                                                                                                                                                                                                                  |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Number of changes                           | A ratio showing the changes sent to this replica since the server started. This value has the format <i>replica_id:changes_sent/changes_skipped</i> . So, if the replica ID is 7, 100 changes were sent, and no changes were skipped, the value of the number of changes is <b>7:100/0</b> . |
| Last replica update began                   | The time when the most recent replication update started.                                                                                                                                                                                                                                    |
| Last replica update ended                   | The time when the most recent replication update ended.                                                                                                                                                                                                                                      |
| Last update message                         | The status for the most recent replication updates.                                                                                                                                                                                                                                          |
| Consumer initialization                     | The current status on consumer initialization (in progress or not).                                                                                                                                                                                                                          |
| Last consumer initialization update message | The status on the last initialization of the consumer.                                                                                                                                                                                                                                       |
| Last consumer initialization began          | The time when the initialization of the consumer replica started.                                                                                                                                                                                                                            |
| Last consumer initialization ended          | The time when the initialization of the consumer replica ended.                                                                                                                                                                                                                              |

### 11.22.2. Monitoring Replication from Admin Express

Admin Express has an option to monitor replication status in real-time, meaning that it shows the number of updates, times the most recent updates were sent, error and success messages, replication schedule, the replicated directory suffix, and other information. Unlike other ways of checking replication status, the Admin Express **Replication Status** page shows the real-time status of replication, including updates in progress, current changes sequence numbers, and the lag between when a change is made on the supplier and when that change is sent to the consumer.

Monitoring replication is set up using a simple configuration file which specifies which server to monitor and what supplier and consumer replicas to include in the status page.

When trying to monitor replication status through Admin Express, remember two things:

- The **Replication Status** page is only available for supplier servers. (It can be opened for other types of replicas; there is just no information available and has the message *The server is not a master or it has no replication agreement*.)
- The configuration file must be in a directory that is accessible to Admin Server, and the file must be readable by the Admin Server user. By default, the user is **nobody**. If you set a different account during the installation, like Red Hat recommends, use this account instead for a better security.

The user is set in the **console.conf** file. To check the user, use **grep** to return the value:

```
grep \^User /etc/dirsrv/admin-serv/console.conf
```

The configuration file should be readable by the Admin Server user and no other users, so consider resetting the permissions on the file:

```
chmod 0400 filename
```

To view in-progress status of replication in Admin Express:

1. Create a configuration file. The configuration file lists all of the servers to monitor for replication, giving their host name or IPv4 or IPv6 address, port, the bind credentials to use, and then optional settings for aliases and time lag colors.

```
#Configuration File for Monitoring Replication Via Admin Express
[connection] Required. Gives the server host (or IPv4 or IPv6
address), port, supplier bind DN, and password.
host1.example.com:389:cn=replication manager:mypassword
host2.example.com:3891:cn=replication manager:altpassword

[alias] Optional. Gives a friendly-name alias to the servers and
consumers.
M1 = host1.example.com:389
M2 = host2.example.com:3891
C1 = host3.example.com:3892
C2 = host4.example.com:3890

[color] Optional. Sets the color for the time lag boxes.
0 = #ccffcc
5 = #ffffcc
60 = #ffcccc
```

The configuration file must be in a directory that is accessible to the Admin Server, and the file must be readable by the Admin Server user. By default, the user is **nobody**.

The user is set in the **console.conf** file. To check the user, use **grep** to return the value:

```
grep \^User /etc/dirsrv/admin-serv/console.conf
```

The configuration file should be readable by the Admin Server user and no other users, so consider resetting the permissions on the file:

```
chmod 0400 filename
```

2. In the Admin Server web page, click the **Admin Express** link, and log in.
3. Click the **Replication Status** link by the supplier server name.
4. Type the path to the configuration file in the **Configuration file** field. Also, set the refresh rate, which is how frequently the replication status page updates; the default is 300 seconds.





Figure 11.5. Viewing Replication Status

5. Click **OK**.

The **Replication Status** page shows the status for sending updates to every consumer listed in the configuration file.

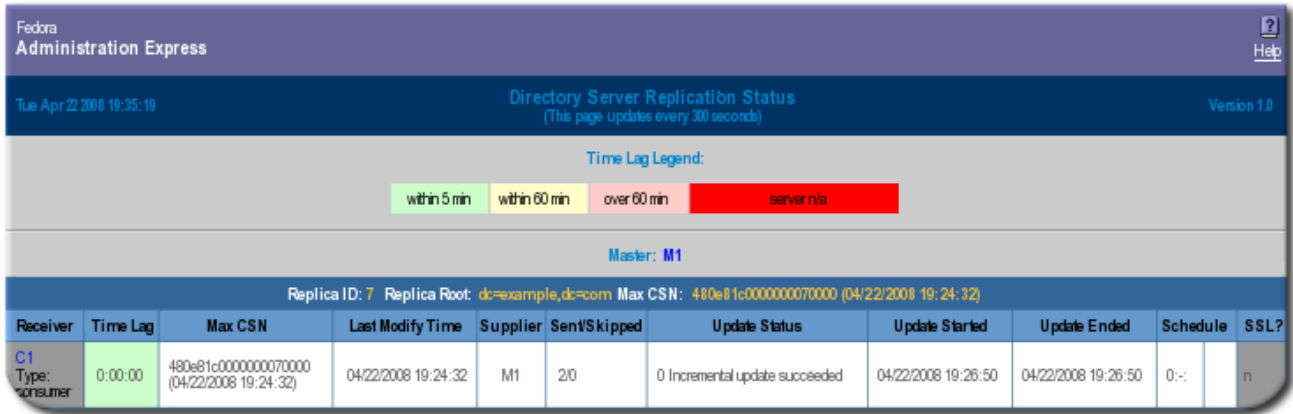


Figure 11.6. Viewing Replication Status

| Table        | Description                                                                                                                                                                                                                                                                                                                |
|--------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Table header | The table header shows the replica ID of the supplier replica, the replicated suffix root (such as <b>dc=example, dc=com</b> ), and the maximum change state number (CSN) on the supplier. (The CSN is the ID of the latest change on the supplier, while the max CSN for the supplier shows the last update it received.) |
| Max CSN      | The ID number of the most recent CSN the consumer has received that originated from the supplier.                                                                                                                                                                                                                          |
| Time lag     | How long it takes for the consumer to receive updates from the supplier; this is the time difference between the supplier and the consumer's max CSNs. When a consumer is in sync with its supplier, the time lag is <b>0</b> .                                                                                            |

| Table                | Description                                                                                                                                                                                                                                                           |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Last Modify Time     | Gives the time of the last update for the consumer (the time the last CSN entry was sent).                                                                                                                                                                            |
| Supplier             | Gives the name of the supplier sending updates to that consumer; this can be useful if a consumer receives updates from multiple suppliers or there are multiple suppliers being monitored on the <b>Replication Status</b> page.                                     |
| Sent/Skipped         | The number of changes that were sent from the supplier and the number skipped in the replication update. The numbers are kept in suppliers' memory only and are cleared if the supplier is restarted.                                                                 |
| Update Status        | The status code (and meaning) for the last update. This column can indicate a possible deadlock if <i>all</i> the suppliers complain that they cannot acquire a busy replica. It is normal for there to be a busy message if one of the suppliers is doing an update. |
| Update Start and End | The timestamps for when the most recent update process started and ended.                                                                                                                                                                                             |
| Schedule             | The configured replication schedule. <b>0: - :</b> means that the consumer is continually updated by the supplier.                                                                                                                                                    |
| SSL?                 | Indicates whether the supplier connects to the consumer over SSL.                                                                                                                                                                                                     |

## 11.23. SETTING REPLICATION SESSION HOOKS

Client applications can have some control over when the Directory Server performs replication by using custom plug-ins that define *replication session hooks*. The hooks apply to both the master sending the information and the consumer receiving the information. Basically, the master sends a preliminary message to the consumer and the consumer sends a response message. If either side sends a message that does not meet the expected criteria (such as the server version or some schema requirement), then the replication operation is terminated.

The intent of replication session hooks is to require some kind of parity between the servers involved in replication. If there is a situation that could cause conflicts or data corruption — such as different server versions which use different sets of schema — then the session hooks identify that potential problem. This prevents replication when it could hurt server performance.

The session hooks are implemented through callback functions in the Directory Server replication functions.

The call backs are defined in an ordered array that follows the replication operation process from initializing the replication agreement to receiving the response from the consumer.

```
static void *test_repl_session_api[] = {
 NULL, /* reserved for api broker use, must be zero */
 test_repl_session_plugin_agmt_init_cb,
 test_repl_session_plugin_pre_acquire_cb,
 test_repl_session_plugin_reply_acquire_cb,
 test_repl_session_plugin_post_acquire_cb,
 test_repl_session_plugin_recv_acquire_cb,
 test_repl_session_plugin_destroy_cb
};
```

Then, the custom replication plug-in registers the callbacks. In this example, it requires a certain version of the server.

```
int test_repl_session_plugin_init(Slapi_PBlock *pb)
{
 ...
 if(slapi_apib_register(REPL_SESSION_v1_0_GUID, test_repl_session_api)
) {
 slapi_log_error(SLAPI_LOG_FATAL, test_repl_session_plugin_name,
 "<-- test_repl_session_plugin_start -- failed to
register repl_session api -- end\n");
 return -1;
 }
 ...
}
```



## NOTE

The callbacks themselves are described in the *Plug-in Programmer's Guide*. This example is provided to make administrators and programmers aware that the replication process can be controlled by using these session hooks.

## 11.24. SOLVING COMMON REPLICATION CONFLICTS

Multi-master replication uses a loose consistency replication model. This means that the same entries can be changed on different servers. When replication occurs between the two servers, the conflicting changes need to be resolved. Mostly, resolution occurs automatically, based on the timestamp associated with the change on each server. The most recent change takes precedence.

However, there are some cases where change conflicts require manual intervention in order to reach a resolution. Entries that have a change conflict that cannot be resolved automatically by the replication process contain a conflict marker attribute ***nsds5ReplConflict***. The ***nsds5ReplConflict*** attribute is an operational attribute which is indexed for presence and equality, so it is simple to search for entries that contain this attribute. For example:

```
ldapsearch -D adminDN -W
-b "dc=example,dc=com" "nsds5ReplConflict=*" * nsds5ReplConflict
```

The ***nsds5ReplConflict*** attribute is already indexed for presence and equality, but for performance reasons, if there are many conflicting entries every day, index the ***nsds5ReplConflict*** attribute in other indexes. For information on indexing, see [Chapter 9, Managing Indexes](#).

- [Section 11.24.1, “Solving Naming Conflicts”](#)

- [Section 11.24.2, “Solving Orphan Entry Conflicts”](#)
- [Section 11.24.3, “Solving Potential Interoperability Problems”](#)

### 11.24.1. Solving Naming Conflicts

When two entries are created with the same DN on different servers, the automatic conflict resolution procedure during replication renames the last entry created, including the entry's unique identifier in the DN. Every directory entry includes a unique identifier given by the operational attribute *nsuniqueid*. When a naming conflict occurs, this unique ID is appended to the non-unique DN.

For example, the entry **uid=adamss,ou=people,dc=example,dc=com** is created on Server A at time **t1** and on Server B at time **t2**, where **t2** is greater (or later) than **t1**. After replication, Server A and Server B both hold the following entries:

- **uid=adamss,ou=people,dc=example,dc=com** (created at time **t1**)
- **nsuniqueid=66446001-1dd211b2+uid=adamss,dc=example,dc=com** (created at time **t2**)

The second entry needs to be renamed in such a way that it has a unique DN. The renaming procedure depends on whether the naming attribute is single-valued or multi-valued.

#### 11.24.1.1. Renaming an Entry with a Multi-Valued Naming Attribute

To rename an entry that has a multi-valued naming attribute:

1. Rename the entry using a new value for the naming attribute, and keep the old RDN. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x
dn: nsuniqueid=66446001-1dd211b2+uid=adamss,dc=example,dc=com
changetype: modrdn
newrdn: uid=NewValue
deleteoldrdn: 0
```

2. Remove the old RDN value of the naming attribute and the conflict marker attribute. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x
dn: uid=NewValue,dc=example,dc=com
changetype: modify
delete: uid
uid: adamss
-
delete: nsds5ReplConflict
-
```



#### NOTE

The unique identifier attribute *nsuniqueid* cannot be deleted.

The Console does not support editing multi-valued RDNs. For example, if there are two servers in a multi-master mode, an entry can be created on each server with the same user ID, and then the new entries' RDN changed to the ***nsuniqueid uid*** value. Attempting to modify this entry from the Console returns the error *Changes cannot be saved for entries with multi-valued RDNs*.

Opening the entry in the advanced mode shows that the naming attribute has been set to ***nsuniqueid uid***. However, the entry cannot be changed or corrected by changing the user ID and RDN values to something different. For example, if ***jdoe*** was the user ID and it should be changed to ***jdoe1***, it cannot be done from the Console. Instead, use the ***ldapmodify*** command:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x

dn: cn=John Doe
changetype: modify
replace: uid
uid: jdoe

dn: cn=John Doe
changetype: modrdn
newrdn: uid=jdoe1
deleteoldrdn: 1
```

#### 11.24.1.2. Renaming an Entry with a Single-Valued Naming Attribute

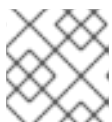
To rename an entry that has a single-valued naming attribute:

1. Rename the entry using a different naming attribute, and keep the old RDN. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: nsuniqueid=66446001-1dd211b2+dc=pubs,dc=example,dc=com
changetype: modrdn
newrdn: cn=TempValue
deleteoldrdn: 0
```

2. Remove the old RDN value of the naming attribute and the conflict marker attribute. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=TempValue,dc=example,dc=com
changetype: modify
delete: dc
dc: pubs
-
delete: nsds5ReplConflict
-
```



#### NOTE

The unique identifier attribute ***nsuniqueid*** cannot be deleted.

3. Rename the entry with the intended attribute-value pair. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x
dn: cn=TempValue,dc=example,dc=com
changetype: modrdn
newrdn: dc=NewValue
deleteoldrdn: 1
```

Setting the value of the ***deleteoldrdn*** attribute to **1** deletes the temporary attribute-value pair **cn=TempValue**. To keep this attribute, set the value of the ***deleteoldrdn*** attribute to **0**.

### 11.24.2. Solving Orphan Entry Conflicts

When a delete operation is replicated and the consumer server finds that the entry to be deleted has child entries, the conflict resolution procedure creates a **glue** entry to avoid having orphaned entries in the directory.

In the same way, when an add operation is replicated and the consumer server cannot find the parent entry, the conflict resolution procedure creates a glue entry representing the parent so that the new entry is not an orphan entry.

*Glue entries* are temporary entries that include the object classes **glue** and **extensibleObject**. Glue entries can be created in several ways:

- If the conflict resolution procedure finds a deleted entry with a matching unique identifier, the glue entry is a resurrection of that entry, with the addition of the **glue** object class and the ***nsds5Rep1Conflict*** attribute.

In such cases, either modify the glue entry to remove the **glue** object class and the ***nsds5Rep1Conflict*** attribute to keep the entry as a normal entry or delete the glue entry and its child entries.

- The server creates a minimalistic entry with the **glue** and **extensibleObject** object classes.

In such cases, modify the entry to turn it into a meaningful entry or delete it and all of its child entries.

### 11.24.3. Solving Potential Interoperability Problems

For reasons of interoperability with applications that rely on attribute uniqueness, such as a mail server, it may be necessary to restrict access to the entries which contain the ***nsds5Rep1Conflict*** attribute. If access is not restricted to these entries, then the applications requiring one attribute only pick up both the original entry and the conflict resolution entry containing the ***nsds5Rep1Conflict***, and operations will fail.

To restrict access, modify the default ACI that grants anonymous read access:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x

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

```
aci: (target="ldap:///dc=example,dc=com")(targetattr!="userPassword")
 (targetfilter="(! (nsds5ReplConflict=*))")(version 3.0;acl
 "Anonymous read-search access";allow (read, search, compare)
 (userdn="ldap:///anyone");)
-
```

The new ACI filters out all entries that contain the *nsds5ReplConflict* attribute from search results.

#### 11.24.4. Resolving Errors for Obsolete/Missing Suppliers

Information about the replication topology — all of the suppliers which are supplying updates to each other and other replicas within the same replication group — is contained in a set of metadata called the *replica update vector (RUV)*. The RUV contains information about the supplier like its ID and URL, its latest change state number for changes made on the local server, and the CSN of the first change. Both suppliers and consumers store RUV information, and they use it to control replication updates.

When one supplier is removed from the replication topology, it may remain in another replica's RUV. When the other replica is restarted, it can record errors in its log that the replication plug-in does not recognize the (removed) supplier.

```
[09/Sep/2017:09:03:43 -0600] NSMMReplicationPlugin - ruv_compare_ruv: RUV
[changelog max RUV] does not
 contain element [{replica 55 ldap://server.example.com:389}
4e6a27ca000000370000 4e6a27e8000000370000]
 which is present in RUV [database RUV]
.....
[09/Sep/2017:09:03:43 -0600] NSMMReplicationPlugin -
replica_check_for_data_reload: Warning: for replica
dc=example,dc=com there were some differences between the changelog max
RUV and the database RUV. If
 there are obsolete elements in the database RUV, you should remove them
using the CLEANRUV task. If they
 are not obsolete, you should check their status to see why there are no
changes from those servers in the changelog.
```

When the supplier is permanently removed from the topology, then any lingering metadata about that supplier should be purged from every other supplier's RUV entry.

There are three ways to do this:

- Removed from *all* suppliers in the topology using the **CLEANALLRUV** replication task.
- Removed from a single supplier in the topology (because of local errors) in using the **CLEANRUV** replication task.
- Removed from all suppliers in the topology using the directory task.

##### 11.24.4.1. Removing an Obsolete Replica from a Single Supplier

If a server is offline or unavailable when a supplier is removed from the topology, it may not receive any updates that the other supplier was removed. In that case, the supplier's RUV would still contain entries about the missing supplier and would return missing element errors.

This can be cleaned up on a single supplier using the **CLEANRUV** replication task. The *nsds5Task* attribute identifies a replication-related task in a replica configuration entry. This attribute is generally

added and removed automatically by the server as it performs regular replication tasks. However, the attribute can be added to a replica configuration entry to manually initiate a task, and it is used to clean obsolete supplier data from the local server's RUV.

To purge an obsolete supplier from the RUV:

1. Running the **CLEANRUV** replication task requires that old replica configuration DN and the old replica ID.

1. Get the replica configuration entry DN by checking for replica entries in the **cn=mapping tree,cn=config** entry.

```
ldapsearch -xLLL -D "cn=directory manager" -W -s sub -b cn=config
objectclass=nsds5replica

dn: cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping
tree,cn=config
cn: replica
...
```

2. Get the replica ID. This is in the **nsds5replicaid** attribute in the configuration entry. The ID is also in the error message about the server being unable to find the replica, identified in the *element [{replica ID URL} uniqueid]* line. For example:

```
[09/Sep/2011:09:03:43 -0600] NSMMReplicationPlugin -
ruv_compare_ruv: RUV [changelog max RUV] does not
contain element [{replica 55 ldap://server.example.com:389}
4e6a27ca000000370000 4e6a27e8000000370000]
...
```

2. Use **ldapmodify** to replace the **nsds5Task** attribute in the configuration entry with **CLEANRUV** and the replica ID, in the form **CLEANRUV#**. For example, for a replica with the ID of 55, the **nsds5Task** value is **CLEANRUV55**:

```
ldapmodify -x -D "cn=directory manager" -W

dn: cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping tree,cn=config
changetype: modify
replace: nsds5task
nsds5task: CLEANRUV55
```

#### 11.24.4.2. Removing an Obsolete/Missing Supplier from All Servers in the Topology

If a supplier is taken offline without cleaning up its RUV entries, then all suppliers and hubs in the topology can register missing element errors in their replication logs.

This can be cleaned up on a single supplier using the **CLEANALLRUV** replication task. The **nsds5Task** attribute identifies a replication-related task in a replica configuration entry. This attribute is generally added and removed automatically by the server as it performs regular replication tasks. However, the attribute can be added to a replica configuration entry to manually initiate a task, and it is used to clean obsolete supplier data from all RUV stores in the topology.





## NOTE

The **CLEANALLRUV** task is replicated to all suppliers and hubs in the replication topology.

1. Running the **CLEANALLRUV** replication task requires that old replica configuration DN and the old replica ID.

1. Get the replica configuration entry DN by checking for replica entries in the **cn=mapping tree,cn=config** entry.

```
ldapsearch -xLLL -D "cn=directory manager" -W -s sub -b cn=config
objectclass=nsds5replica

dn: cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping
tree,cn=config
cn: replica
...
```

2. Get the replica ID. This is in the **nsds5replicaid** attribute in the configuration entry.

```
ldapsearch -xLLL -D "cn=directory manager" -W -s sub -b
cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping tree,cn=config
objectclass=nsds5replica nsds5replicaid

dn: cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping
tree,cn=config
nsds5replicaid: 55
...
```

The ID is also in the error message about the server being unable to find the replica, identified in the *element [{replica ID URL} uniqueId]* line. For example:

```
[09/Sep/2011:09:03:43 -0600] NSMMReplicationPlugin -
ruv_compare_ruv: RUV [changelog max RUV] does not
contain element [{replica 55 ldap://server.example.com:389}
4e6a27ca000000370000 4e6a27e8000000370000]
...
```

2. Use **ldapmodify** to add the **nsds5Task** attribute in the configuration entry with a value of **CLEANALLRUV** and the replica ID, in the form **CLEANALLRUV#**. For example, for a replica with the ID of 55, the **nsds5Task** value is **CLEANALLRUV55**:

```
ldapmodify -x -D "cn=directory manager" -W

dn: cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping tree,cn=config
changetype: modify
replace: nsds5task
nsds5task: CLEANALLRUV55
```

### 11.24.4.3. Removing an Obsolete/Missing Supplier Using a Task Operation

If a supplier is taken offline without cleaning up its RUV entries, then all suppliers and hubs in the topology can register missing element errors in their replication logs.

There may be times when it is preferable to launch a directory task operation rather than a replication task. This can be done by creating an instance of the **cn=cleanallruv,cn=tasks,cn=config** task.



## NOTE

As with the **CLEANALLRUV** replication task, this **cn=cleanruv,cn=tasks** operation is replicated to all suppliers and hubs in the replication topology.

1. Obtain the old replica configuration DN and the old replica ID.

1. Get the replica configuration entry DN by checking for replica entries in the **cn=mapping tree,cn=config** entry.

```
ldapsearch -xLLL -D "cn=directory manager" -W -s sub -b cn=config
objectclass=nsds5replica

dn: cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping
tree,cn=config
cn: replica
...
```

2. Get the replica ID. This is in the **nsds5replicaid** attribute in the configuration entry.

```
ldapsearch -xLLL -D "cn=directory manager" -W -s sub -b
cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping tree,cn=config
objectclass=nsds5replica nsds5replicaid

dn: cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping
tree,cn=config
nsds5replicaid: 55
...
```

2. Use **ldapmodify** to create the **cn=cleanallruv,cn=tasks,cn=config** entry. This task requires information on the replication configuration:

- o The base DN of the replicated database (**replica-base-dn**).
- o The replica ID (**replica-id**).
- o Whether to catch up to the max change state number (CSN) from the missing supplier or just remove all RUV entries and miss any updates (**replica-force-cleaning**); setting this to **no** means that the task catches up with all changes first, and then removes the RUV.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=clean 55,cn=cleanallruv,cn=tasks,cn=config
objectclass: extensibleObject
replica-base-dn: dc=example,dc=com
replica-id: 55
replica-force-cleaning: no
cn: clean 55
```

This task replicates to all servers in the topology. This task can also be aborted, since it can take several minutes to run, and the abort task is also propagated to all suppliers.

```
ldapmodify -a -D "cn=directory manager" -w -p 389 -h server.example.com -x
dn: cn=abort 55,cn=abort cleanallruv,cn=tasks,cn=config
objectclass: extensibleObject
cn: abort 55
replica-base-dn: dc=example,dc=com
replica-id: 55
replica-certify-all: yes
```

The ***replica-certify-all*** attribute sets whether to wait for the task to be sent to all servers before completing on the local server.

## 11.25. TROUBLESHOOTING REPLICATION-RELATED PROBLEMS

This section lists some error messages, explains possible causes, and offers remedies.

It is possible to get more debugging information for replication by setting the error log level to **8192**, which is replication debugging. See [Section 15.3.5, “Configuring Log Levels”](#).

To change the error log level to **8192** with **ldapmodify**:

```
ldapmodify -D "cn=directory manager" -w -p 389 -h server.example.com -x
dn: cn=config
changetype: modify
replace: nsslapd-errorlog-level
nsslapd-errorlog-level: 8192
```

Because log level is additive, running the above command will result in excessive messages in the error log. So, use it judiciously.

To turn off replication debugging log, set the same attribute to **0**.

The **c1-dump.pl** script, which is explained in detail in the *Directory Server Configuration and Command-Line Tool Reference* can also help troubleshoot replication-related problems. Depending on the usage options, the script can selectively dump a particular replica:

- Dump the contents of a **replication-change-log** file and in-memory variables **purge RUV** and **maxRUV**.
- Grep and interpret change state numbers (CSNs) in the changelog.
- Get the base-64 encoded changelog from the Directory Server, and then decode the changelog.

Many common replication problems are described in [Table 11.7, “Replication Errors”](#).

**Table 11.7. Replication Errors**

| Error/Symptom                                                                                                                                                                                                                             | Reason                                                                                                                                                                                                                                                                                                       | Impact                                                                                                                                               | Remedy                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| agmt=%s (%s:%d)<br>Replica has a different generation ID than the local data                                                                                                                                                              | The consumer specified at the beginning of this message has not been (successfully) initialized yet, or it was initialized from a different root supplier.                                                                                                                                                   | The local supplier will not replicate any data to the consumer.                                                                                      | Ignore this message if it occurs before the consumer is initialized. Otherwise, reinitialize the consumer if the message is persistent. In a multi-master environment, all the servers should be initialized only once from a root supplier, directly or indirectly. For example, M1 initializes M2 and M4, M2 then initializes M3, and so on. The important thing to note is that M2 must not start initializing M3 until M2's own initialization is done (check the total update status from the M1's Console or M1 or M2's error log). Also, M2 should not initialize M1 back. |
| Warning: data for replica's was reloaded, and it no longer matches the data in the changelog. Recreating the changelog file. This could affect replication with replica's consumers, in which case the consumers should be reinitialized. | This message may appear only when a supplier is restarted. It indicates that the supplier was unable to write the changelog or did not flush out its RUV at its last shutdown. The former is usually because of a disk-space problem, and the latter because a server crashed or was ungracefully shut down. | The server will not be able to send the changes to a consumer if the consumer's <b>maxcsn</b> no longer exists in the server's changelog.            | Check the disk space and the possible core file (under the server's logs directory). If this is a single-master replication, reinitialize the consumers. Otherwise, if the server later complains that it can not locate some CSN for a consumer, see if the consumer can get the CSN from other suppliers. If not, reinitialize the consumer.                                                                                                                                                                                                                                    |
| agmt=%s(%s:%d):<br>Cannot locate CSN %s in the changelog (DB rc=%d). The consumer may need to be reinitialized.                                                                                                                           | Most likely the changelog was recreated because of the disk is full or the server ungracefully shutdown.                                                                                                                                                                                                     | The local server will not be able to send any more change to that consumer until the consumer is reinitialized or gets the CSN from other suppliers. | If this is a single-master replication, reinitialize the consumers. Otherwise, see if the consumer can get the CSN from other suppliers. If not, reinitialize the consumer.                                                                                                                                                                                                                                                                                                                                                                                                       |

| Error/Symptom                                                                     | Reason                                                                                                                                               | Impact                                                                                                                                                                                                                                                                                                                                                                                  | Remedy                                                                                                                                                                                                                                                                                   |
|-----------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Too much time skew                                                                | The system clocks on the host machines are extremely out of sync.                                                                                    | The system clock is used to generate a part of the CSN. In order to reflect the change sequence among multiple suppliers, suppliers would forward-adjust their local clocks based on the remote clocks of the other suppliers. Because the adjustment is limited to a certain amount, any difference that exceeds the permitted limit will cause the replication session to be aborted. | Synchronize the system clocks on the Directory Server host machines. If applicable, run the network time protocol ( <b>ntp</b> ) daemon on those hosts.                                                                                                                                  |
| agmt=%s(%s:%d):<br>Warning: Unable to send endReplication extended operation (%s) | The consumer is not responding.                                                                                                                      | If the consumer recovers without being restarted, there is a chance that the replica on the consumer will be locked forever if it did not receive the release lock message from the supplier.                                                                                                                                                                                           | Watch if the consumer can receive any new change from any of its suppliers, or start the replication monitor, and see if all the suppliers of this consumer warn that the replica is busy. If the replica appears to be locked forever and no supplier can get in, restart the consumer. |
| Changelog is getting too big.                                                     | Either changelog purge is turned off, which is the default setting, or changelog purge is turned on, but some consumers are way behind the supplier. |                                                                                                                                                                                                                                                                                                                                                                                         | By default changelog purge is turned off. To turn it on from the command line, run <b>ldapmodify</b> as follows: <pre> ldapmodify -D "cn=directory manager" -W -p 389 -h server.example .com -x  dn: cn=changelog5, cn=config changetype: modify add: nsslapd- changelogmaxag </pre>     |

| Error/Symptom                                                                                                                                             | Reason                                                                                                                                                                                                                                           | Impact | Remedy                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                                                                                                                           |                                                                                                                                                                                                                                                  |        | <p><code>nsslapd-changelogmaxage: 1d</code></p> <p>where <b>1d</b> means 1 day. Other valid time units are <b>s</b> for seconds, <b>m</b> for minutes, <b>h</b> for hours, and <b>w</b> for weeks. A value of <b>0</b> turns off the purge.</p> <p>With changelog purge turned on, a purge thread that wakes up every five minutes will remove a change if its age is greater than the value of <b><i>nsslapd-changelogmaxage</i></b> and if it has been replayed to all the direct consumers of this supplier (supplier or hub).</p> <p>If it appears that the changelog is not purged when the purge threshold is reached, check the maximum time lag from the replication monitor among all the consumers. Irrespective of what the purge threshold is, no change will be purged before it is replayed by all the consumers.</p> |
| The Replication Monitor is not responding. (For information on Replication Monitor, see <a href="#">Section 11.22, “Monitoring Replication Status”</a> .) | The SSL port is specified in some replication agreement, but the certificate database is not specified or not accessible by the Replication Monitor. If there is no SSL port problem, one of the servers in the replication topology might hang. |        | <p>Map the SSL port to a non-SSL port in the configuration file of the Replication Monitor. For example, if 636 is the SSL port and 389 is the non-SSL port, add the following line in the <b>[connection]</b> section:</p> <pre>*:636=389:*:password</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

| Error/Symptom                                                                                                                                                                                | Reason                                                                                                                                  | Impact | Remedy                                                                   |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|--------|--------------------------------------------------------------------------|
| In the Replication Monitor, some consumers show just the header of the table. (For information on Replication Monitor, see <a href="#">Section 11.22</a> , “Monitoring Replication Status”.) | No change has originated from the corresponding suppliers. In this case, the <b>MaxCSN</b> in the header part should be <b>"None"</b> . |        | There is nothing wrong if there is no change originated from a supplier. |

## CHAPTER 12. SYNCHRONIZING RED HAT DIRECTORY SERVER WITH MICROSOFT ACTIVE DIRECTORY

Windows Sync carries over changes in a directory — adds, deletes, and changes in groups, users, and passwords — between Red Hat Directory Server and Microsoft Active Directory. This makes it much more efficient and effective to maintain consistent information across directories.

### 12.1. ABOUT WINDOWS SYNC

Synchronization allows the user and group entries in Active Directory to be matched with the entries in the Red Hat Directory Server. As entries are created, modified, or deleted, the corresponding change is made to the sync peer server, allowing two-way synchronization of users, passwords, and groups.

The synchronization process is analogous to the replication process: the synchronization is enabled by a plug-in, configured and initiated through a sync agreement, and record of directory changes is maintained and updates are sent according to that changelog. This synchronizes users and groups between Directory Server and a Windows server.

Windows Sync has two parts is configured in two parts, one for user and group entries and the other for passwords:

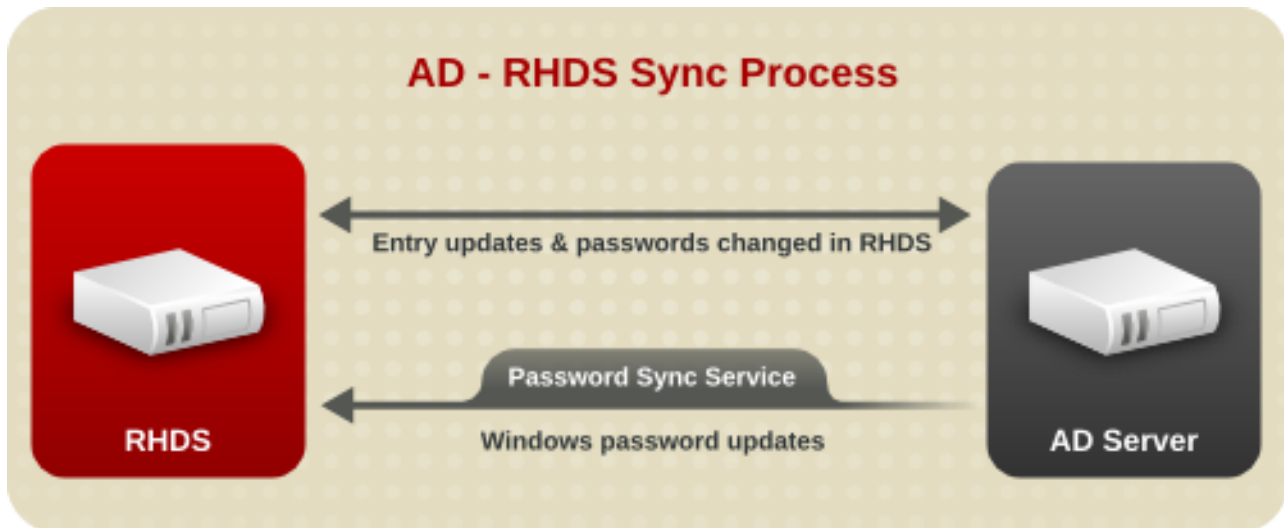
- *Directory Server Windows Sync.* Synchronization for user and group entries is configured in a synchronization agreement, much like replication is configured in a replication agreement. A sync agreement defines what kinds of entries are synchronized (users, groups, or both) and which direction changes are synced (from the Directory Server to Active Directory, from Active Directory to Directory Server, or both).

The Directory Server relies on the Multi-Master Replication Plug-in to synchronize user and group entries. The same changelog that is used for multi-master replication is also used to send updates from the Directory Server to Active Directory as LDAP operations. The server also performs LDAP search operations against its Windows server to synchronize changes made to Windows entries to the corresponding Directory Server entry.

- *Password Sync Service.* Password changes made on Directory Server are automatically synced over to Active Directory, but there must be a special hook to recognize and transmit password changes on Active Directory over to Directory Server. This is done by the Password Sync Service. This application captures password changes on the Windows machines and send them to the Directory Server over LDAPS.

**The Password Sync Service must be installed on every Active Directory domain controller.**





**Figure 12.1. Active Directory — Directory Server Synchronization Process**

Synchronization is configured and controlled by one or more *synchronization agreements*, which establishes synchronization between *sync peers*, the directory servers being synced. These are similar in purpose to replication agreements and contain a similar set of information, including the host name (or IPv4 or IPv6 address) and port number for Active Directory. The Directory Server connects to its peer Windows server using LDAP/LDAPS to both send and receive updates.

LDAP, a standard connection, can be used for syncing user and group entries alone, but to synchronize passwords, some sort of secure connection is required. If a secure connection is not used, the Windows domain will not accept password changes from the Directory Server and the Password Sync Service will not send passwords from the Active Directory domain to the Directory Server. Win Sync allows both LDAPS using SSL/TLS and Start TLS.

A single Active Directory subtree is synchronized with a single Directory Server subtree, and vice versa. Unlike replication, which connects *databases*, synchronization is between *suffixes*, parts of the directory tree structure. The synced Active Directory and Directory Server suffixes are both specified in the sync agreement. All entries within the respective subtrees are candidates for synchronization, including entries that are not immediate children of the specified suffix DN.



## NOTE

Any descendant container entries need to be created separately in Active Directory by an administrator; Windows Sync does not create container entries.

The Directory Server maintains a *changelog*, a database that records modifications that have occurred. The changelog is used by Windows Sync to coordinate and send changes made to the Active Directory peer. Changes to entries in Active Directory are found by using Active Directory's Dirsync search feature. The Dirsync search is issued periodically, every five minutes, to check for changes on the Active Directory server. Using Dirsync ensures that only those entries that have changed since the previous search are retrieved.

In some situations, such as when synchronization is configured or there have been major changes to directory data, a total update, or *resynchronization*, can be run. This examines every entry in both sync peers and sends any modifications or missing entries. A full Dirsync search is initiated whenever a total update is run. See [Section 12.9, “Sending Synchronization Updates”](#) for more information.

Windows Sync provides some control over which entries are synchronized to grant administrators fine-grained control of the entries that are synchronized and to give sufficient flexibility to support different deployment scenarios. This control is set through different configuration attributes set in the Directory

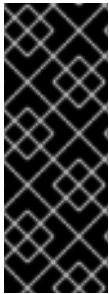
Server:

- When creating the sync agreement, there is an option to synchronizing new Windows entries (***nsDS7NewWinUserSyncEnabled*** and ***nsDS7NewWinGroupSyncEnabled***) as they are created. If these attributes are set to **on**, then existing Windows users/groups are synchronized to the Directory Server, and users/groups as they are created are synchronized to the Directory Server.

Within the Windows subtree, only entries with user or group object classes can be synchronized to Directory Server.

- On the Directory Server, only entries with the **ntUser** or **ntGroup** object classes and attributes can be synchronized.

The placement of the sync agreement depends on what suffixes are synchronized; for a single suffix, the sync agreement is made for that suffix alone; for multiple suffixes, the sync agreement is made at a higher branch of the directory tree. To propagate Windows entries and updates throughout the Directory Server deployment, make the agreement between a master in a multi-master replication environment, and use that master to replicate the changes across the Directory Server deployment, as shown in [Figure 12.2, “Multi-Master Directory Server — Windows Domain Synchronization”](#).



### IMPORTANT

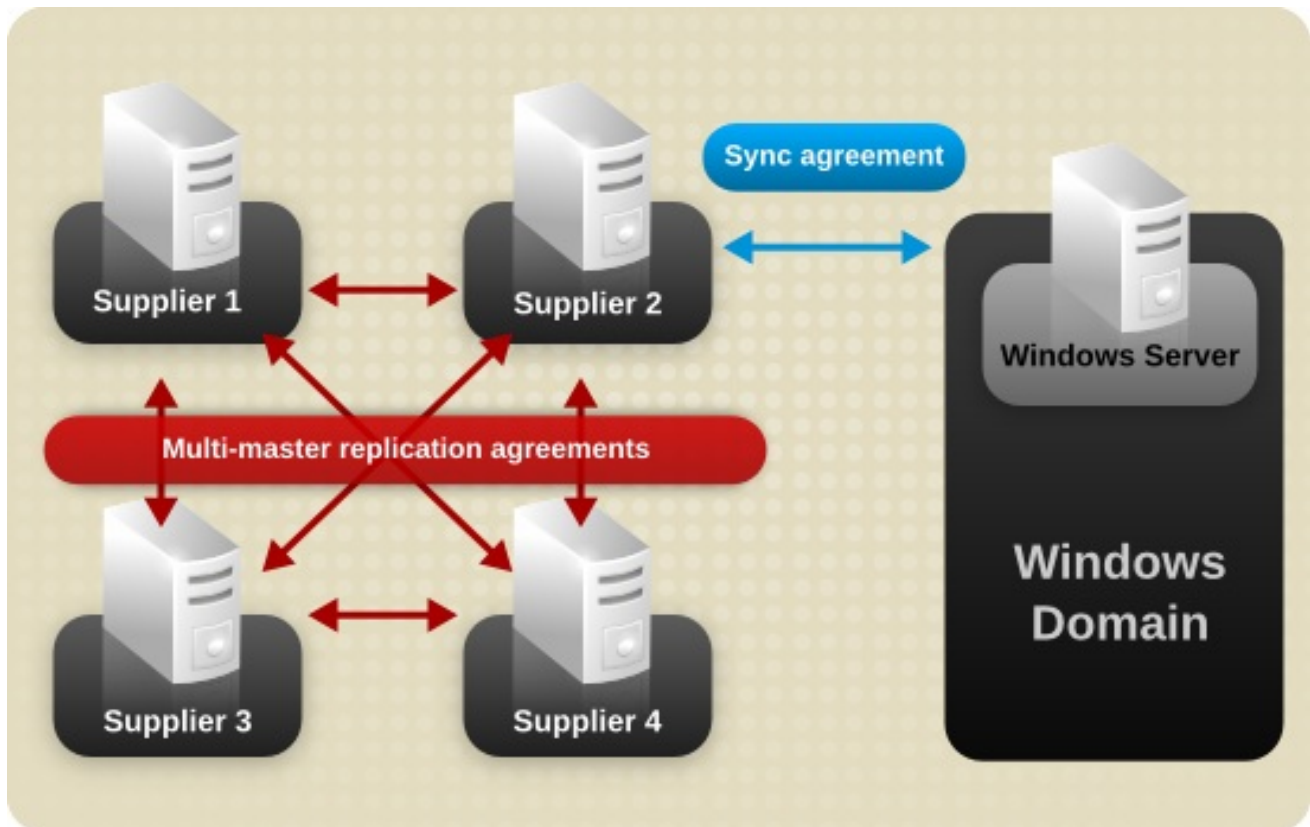
While it is possible to configure a sync agreement on a hub server, this only allows uni-directional synchronization, from Red Hat Directory Server to Active Directory. The Active Directory server cannot sync any changes back to the hub.

It is strongly recommended that only masters in multi-master replication be used to configure synchronization agreements.



### WARNING

There can only be a single sync agreement between the Directory Server environment and the Active Directory environment. Multiple sync agreements to the same Active Directory domain can create entry conflicts.



**Figure 12.2. Multi-Master Directory Server — Windows Domain Synchronization**

Directory Server passwords are synchronized along with other entry attributes because plain-text passwords are retained in the Directory Server changelog. The Password Sync service is needed to catch password changes made on Active Directory. Without the Password Sync service, it would be impossible to have Windows passwords synchronized because passwords are hashed in Active Directory, and the Windows hashing function is incompatible with the one used by Directory Server.

## 12.2. SUPPORTED ACTIVE DIRECTORY VERSIONS

Windows Synchronization and the Password Sync Service are supported on Windows 2008 R2 on both 32-bit and 64-bit platforms.

## 12.3. STEPS FOR CONFIGURING WINDOWS SYNC

Configuring synchronization is very similar to configuring replication. It requires configuring the database as a master with a changelog and creating an agreement to define synchronization. A common user identity, a sync user, connects to the Windows sync peer to send updates from the Directory Server and to check for updates to sync back to the Directory Server.



### NOTE

To synchronize passwords (which is the only way for users to be active on both Directory Server and Active Directory), synchronization must be configured to run over SSL/TLS. Therefore, this configuration section assumes that SSL/TLS must also be configured.

Configuring synchronization over SSL/TLS is also similar to configuring replication over SSL/TLS. Both sync peers must be configured to trust each other for encrypted sessions (all password operations are performed over TLS/SSL).

All synchronization for user and group entries is passive from the Active Directory side; it is the Directory Server which sends updates on its side and polls for updates on the Active Directory domain. For passwords, the Active Directory server requires a separate password service; this service actively sends password changes from the Active Directory domain to Directory Server.

### 12.3.1. Step 1: Configure SSL on Directory Server

The full instructions for configuring the Directory Server to run in SSL are at [Section 7.4.2, “Enabling TLS/SSL Only in the Directory Server”](#). Basically, the Directory Server needs to have the appropriate SSL certificates installed, be configured to run over an SSL port, and allow client authentication from other servers.

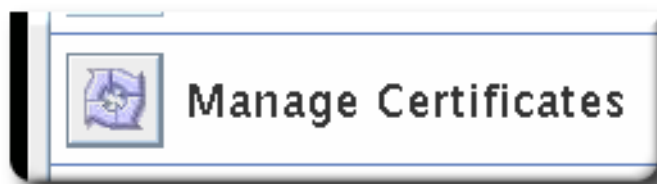
Two certificates must be issued and installed on both the Directory Server and the Active Directory sync peer:

- CA certificate, shared between the Directory Server and Active Directory
- Server certificates for the Directory Server and Active Directory sync peers, which are accessible by the sync services

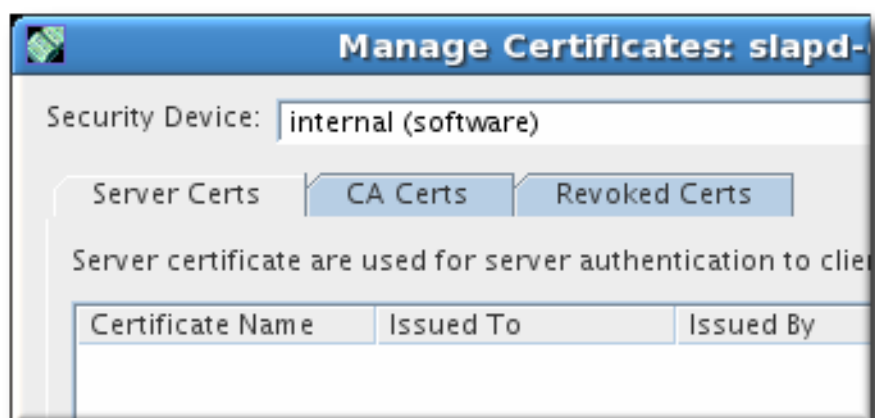
To set up SSL:

1. Generate a certificate request.

1. In the Directory Server Console, select the **Tasks** tab, and click **Manage Certificates**.



2. Select the **Server Certs** tab, and click the **Request** button at the bottom.

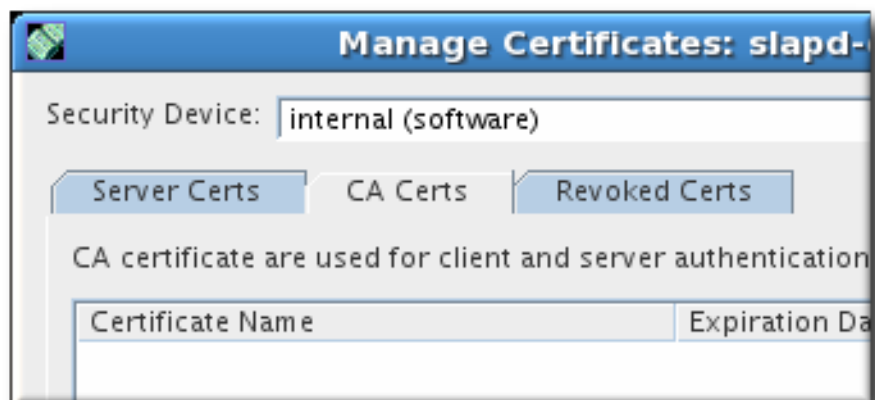


3. Fill in the certificate information, and save the certificate request to a file.
2. Submit the certificate to a certificate authority, and retrieve it once it is issued.

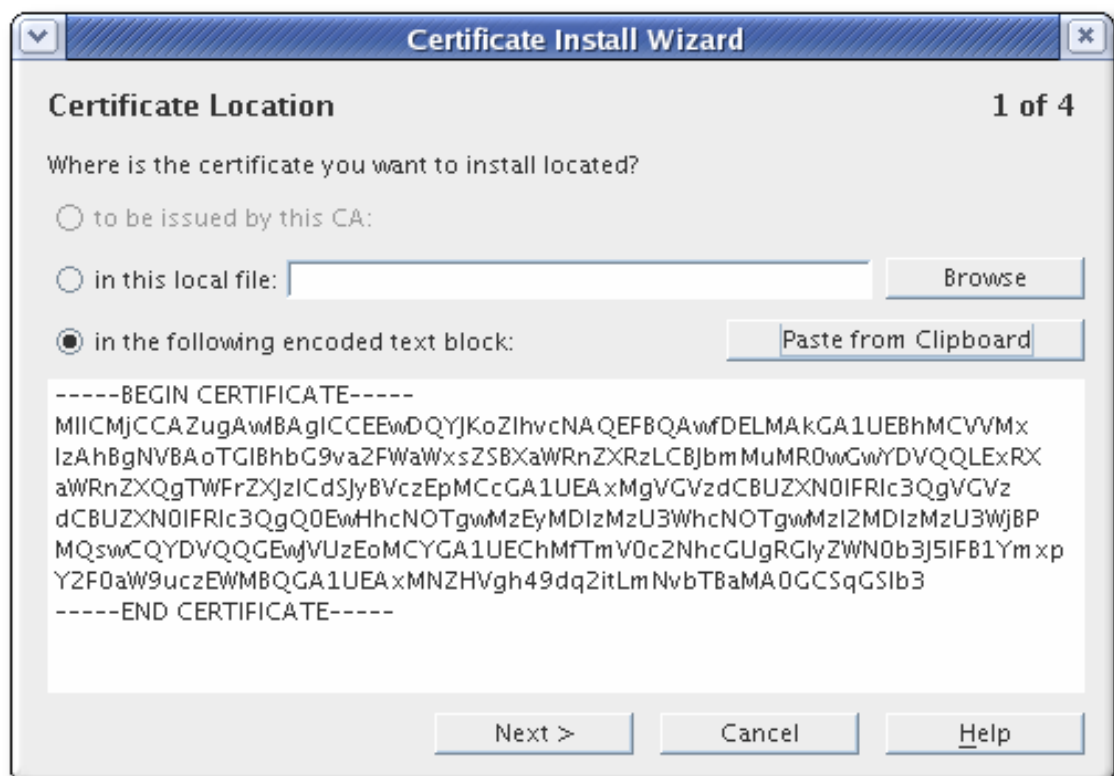
The method for submitting certificate requests and retrieving certificates varies for each CA.

3. Install the new certificate.

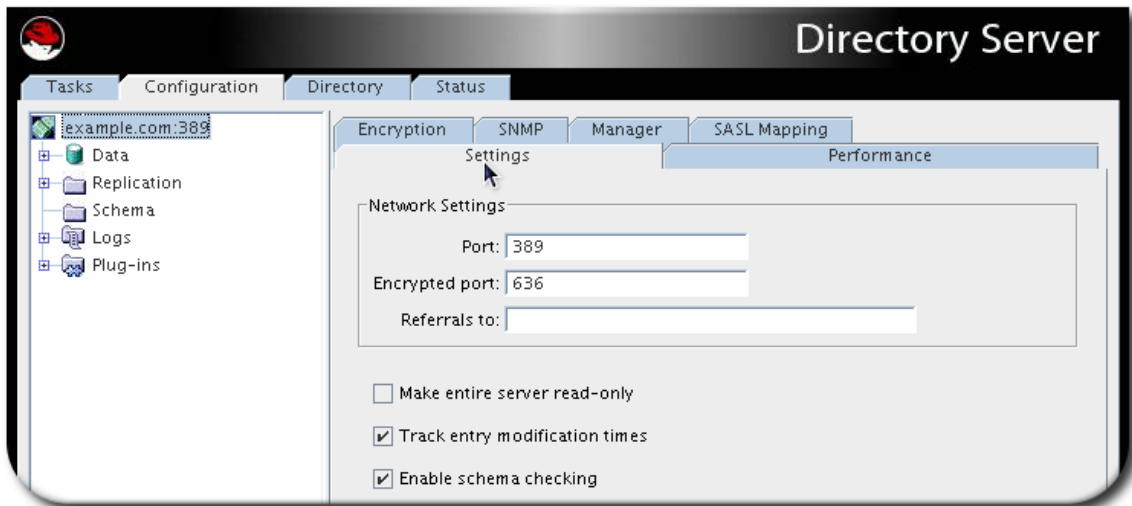
1. In the Directory Server Console, select the **Tasks** tab, and click **Manage Certificates**.
2. Select the **Server Certs** tab, and click **Install** at the bottom of the window.
3. Paste in the certificate, and set the password for the token database.
4. Install the CA certificate for the issuing CA.
  1. Download and save the CA certificate from the CA's site. Each CA has a slightly different way of making its CA certificate available.
  2. In the Directory Server Console, select the **Tasks** tab, and click **Manage Certificates**.
  3. Go to the **CA Certs** tab, and click **Install** at the bottom of the window.



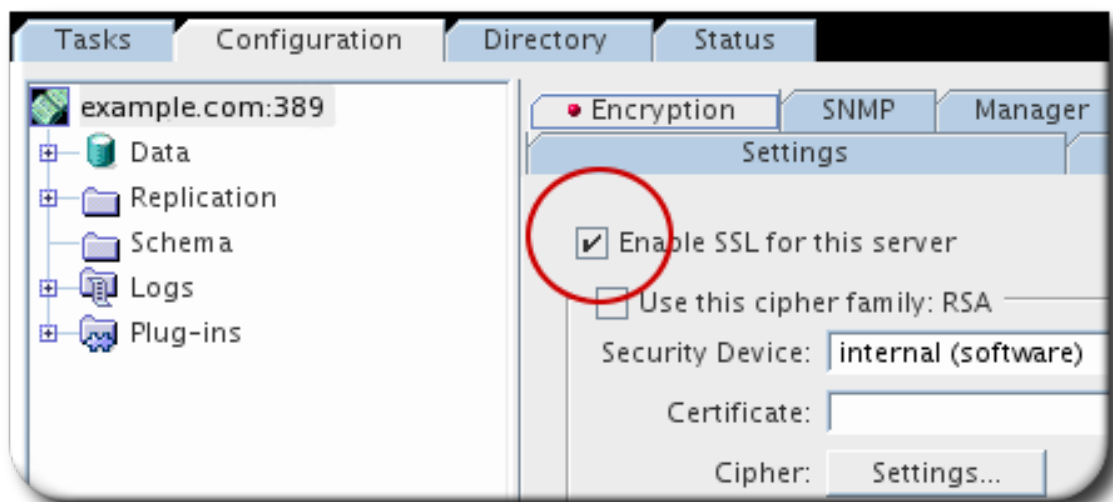
4. Paste in the CA certificate or point to the downloaded file, and go through the certificate installer.



5. Change the server to the SSL port; this is described in much more detail in [Section 1.6](#), “Changing Directory Server Port Numbers”.
1. Open the Directory Server Console, and open the **Configuration** tab for the Directory Server.
2. In the **Settings** tab, set the secure port for the server to use for TLS/SSL communications, such as **636**. Click **Save**.



3. Select the **Encryption** tab in the right pane.
4. Select the **Enable SSL for this Server** check box, then select the certificate to use from the drop-down menu. Click **Save**.



5. Restart the Directory Server. The Directory Server must be restarted from the command line.

```
service dirsrv restart example
```

To restart the Directory Server without the password prompt, create a PIN file or use a hardware crypto device. See [Section 7.4.4](#), “Creating a Password File for the Directory Server” for information on how to create a PIN file.

### 12.3.2. Step 2: Configure the Active Directory Domain



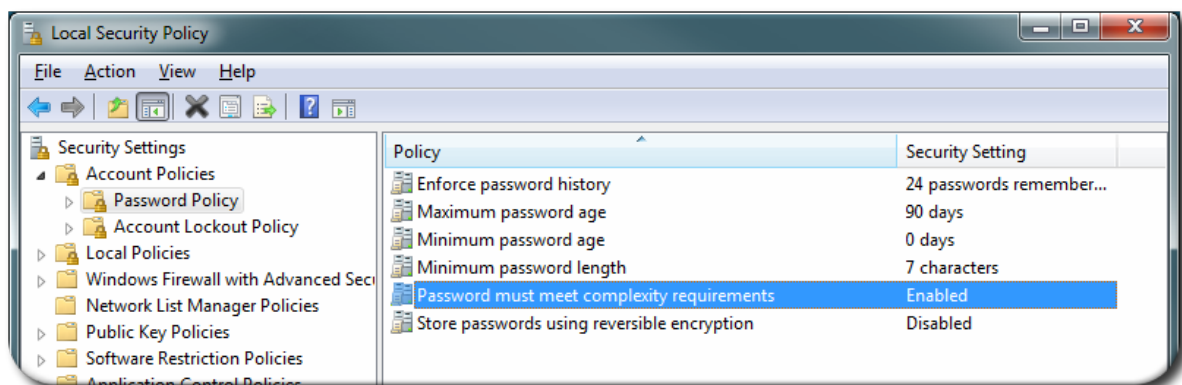


## NOTE

Synchronization can only be configured with an Active Directory domain controller, so make sure that the domain is properly installed and configured.

The first configuration step is to make sure that the Active Directory password complexity policies are enabled so that the Password Sync service will run.

1. Run **secpol.msc** from the command line.
2. Select **Security Settings**.
3. Open **Account Policies**, and then open **Password Policy**.
4. Enable the **Password must meet complexity requirements** option and save.



Configure SSL and set up a root CA on the Active Directory server, as described in the Microsoft knowledgebase at [http://technet.microsoft.com/en-us/library/cc772393%28v=ws.10%29.aspx#BKMK\\_AS1](http://technet.microsoft.com/en-us/library/cc772393%28v=ws.10%29.aspx#BKMK_AS1).

1. Install a certificate authority.
  1. In the **Administrative Tools** area, open **Server Manager** and add a role.
  2. Select the **Active Directory Certificate Services** check box.
  3. Click through to the **Select Role Services** page, and select the **Certification Authority** check box.
  4. When configuring the CA, select the following options on the appropriate screens:
    - **Enterprise** for the setup type
    - **Certification Authority Web Enrollment** in the optional configuration
  5. Reboot the Active Directory server.
2. Set up the Active Directory server to use the SSL server certificate.
  1. Create a certificate request **.inf**, using the fully-qualified domain name of the Active Directory as the certificate subject. For example:

```
;------ request.inf -----
```

```

[Version]

Signature="$Windows NT$

[NewRequest]

Subject = "CN=ad.server.example.com, O=Engineering, L=Raleigh,
S=North Carolina, C=US"
KeySpec = 1
KeyLength = 2048
Exportable = TRUE
MachineKeySet = TRUE
SMIME = False
PrivateKeyArchive = FALSE
UserProtected = FALSE
UseExistingKeySet = FALSE
ProviderName = "Microsoft RSA SChannel Cryptographic Provider"
ProviderType = 12
RequestType = PKCS10
KeyUsage = 0xa0

[EnhancedKeyUsageExtension]

OID=1.3.6.1.5.5.7.3.1

;-----

```

For more information on the **.inf** request file, see the Microsoft documentation, such as <http://technet.microsoft.com/en-us/library/cc783835.aspx>.

2. Generate the certificate request.

```
certreq -new request.inf request.req
```

3. Submit the request to the Active Directory CA. For example:

```
certreq -submit request.req certnew.cer
```



#### NOTE

If the command-line tool returns an error message, then use the Web browser to access the CA and submit the certificate request. If IIS is running, then the CA URL is **http://servername/certsrv**.

4. Accept the certificate request. For example:

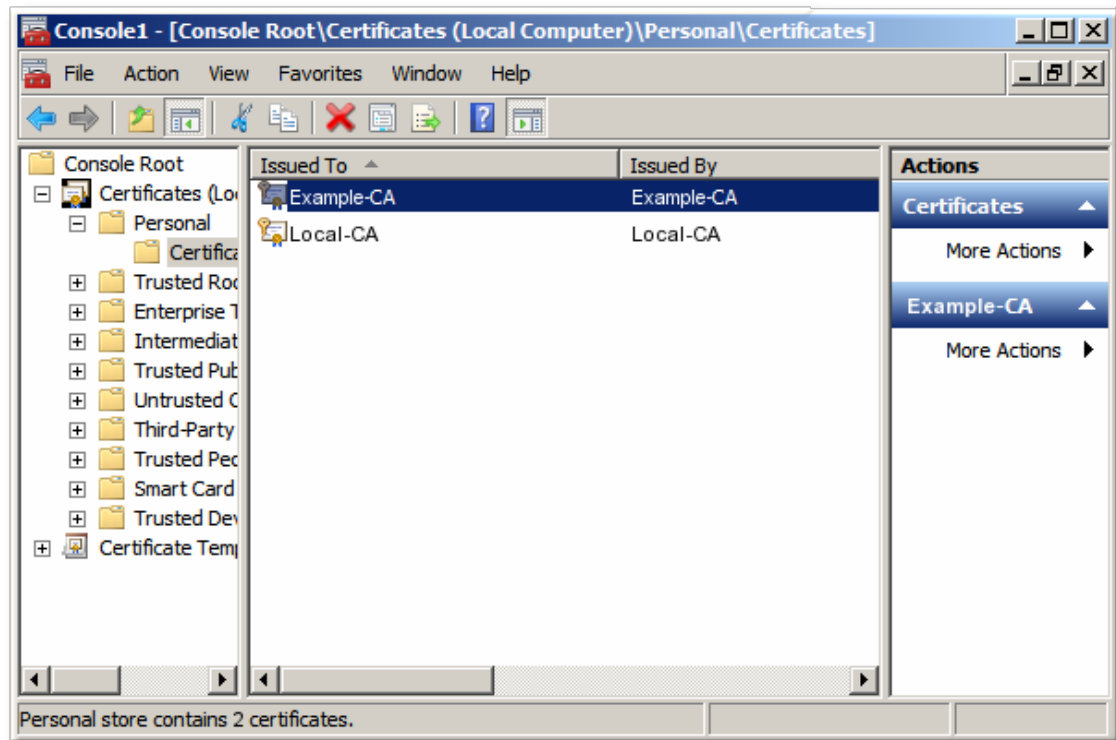
```
certreq -accept certnew.cer
```

3. Make sure that the server certificate is present on the Active Directory server.

1. In the **Run** menu, open the MMC console.
2. In the **File** menu, click **Add/Remove Snap-in...**



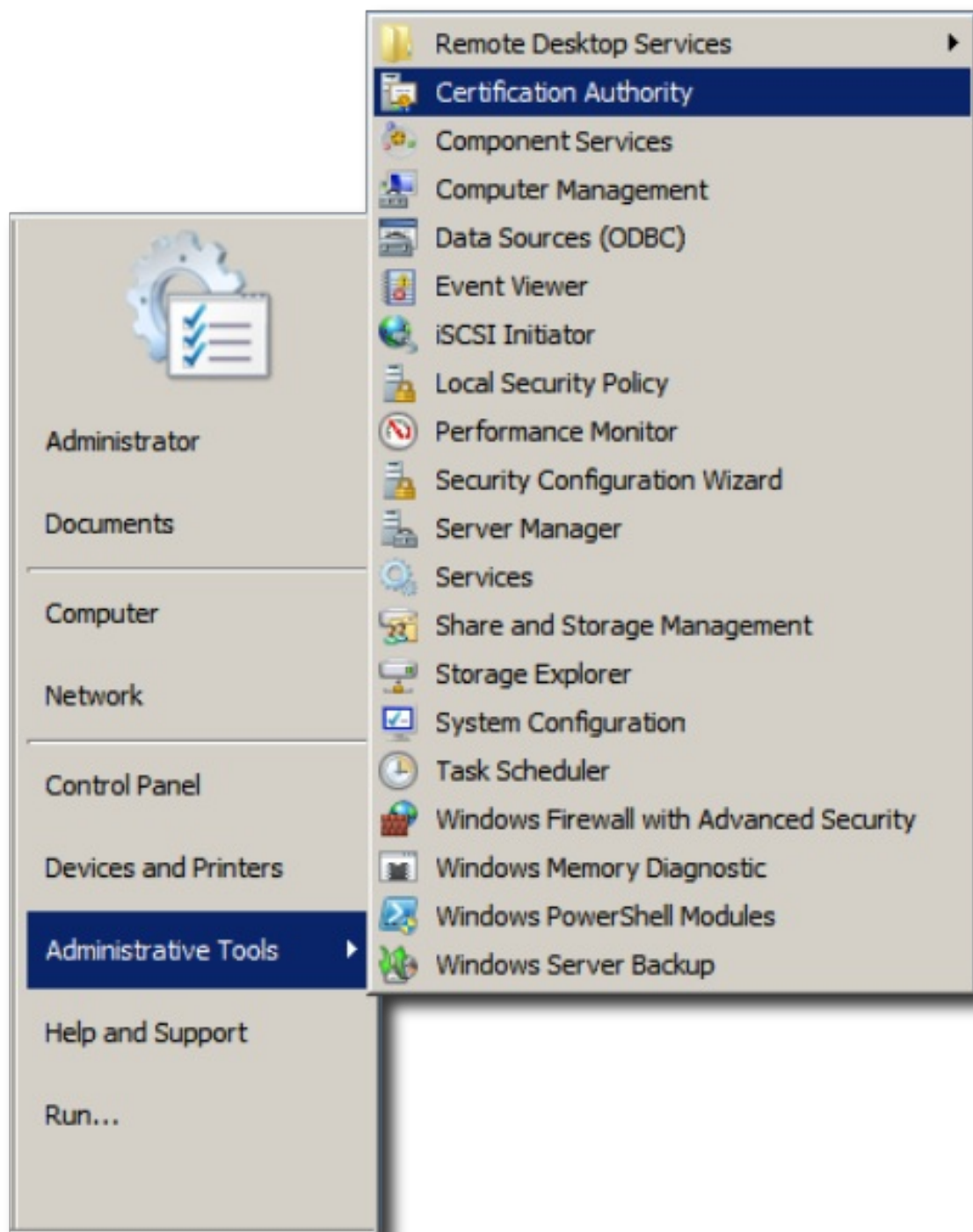
3. Select the **Certificates** snap-in, and click **Add** to add it, and then click **Next**.
4. Expand the **Certificates (Local)** menu on the left. Expand the **Personal** item and click **Certificates**.



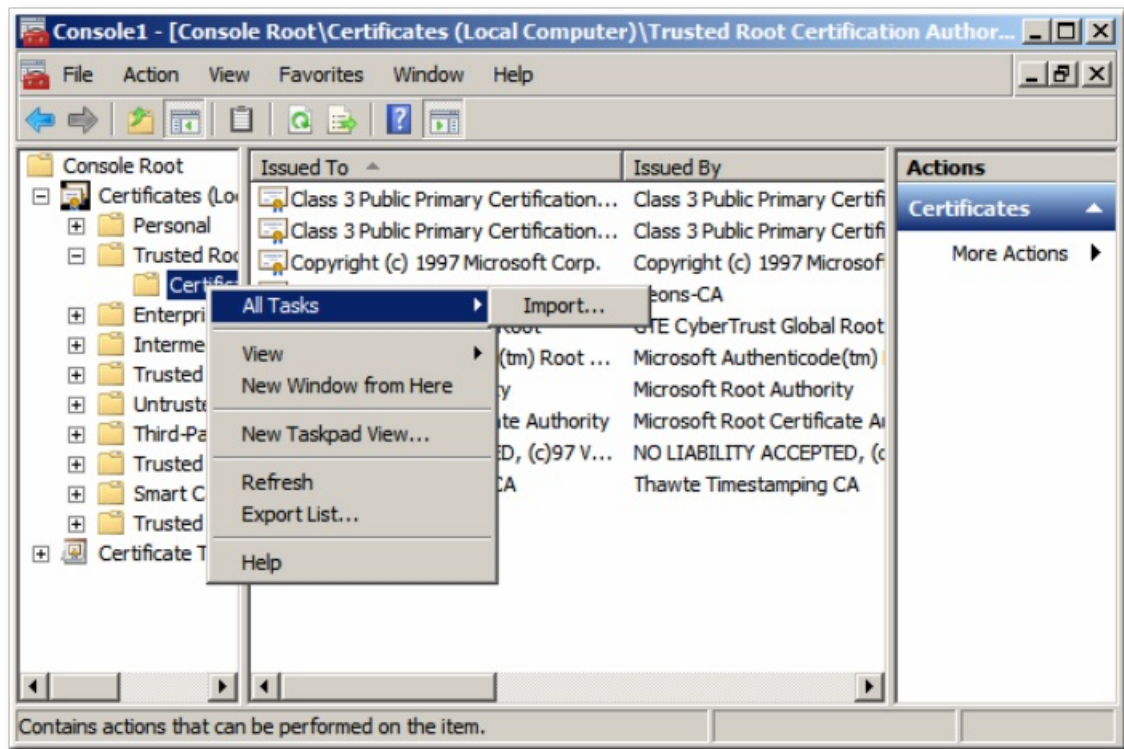
5. The new certificate should be listed with the other certificates.
4. On the Directory Server, export the CA certificate.

```
[root@server ~]# cd /etc/dirsrv/slapd-instance_name
[root@server slapd-instance_name]# certutil -d . -L -n "CA
certificate" -a > dsca.crt
```

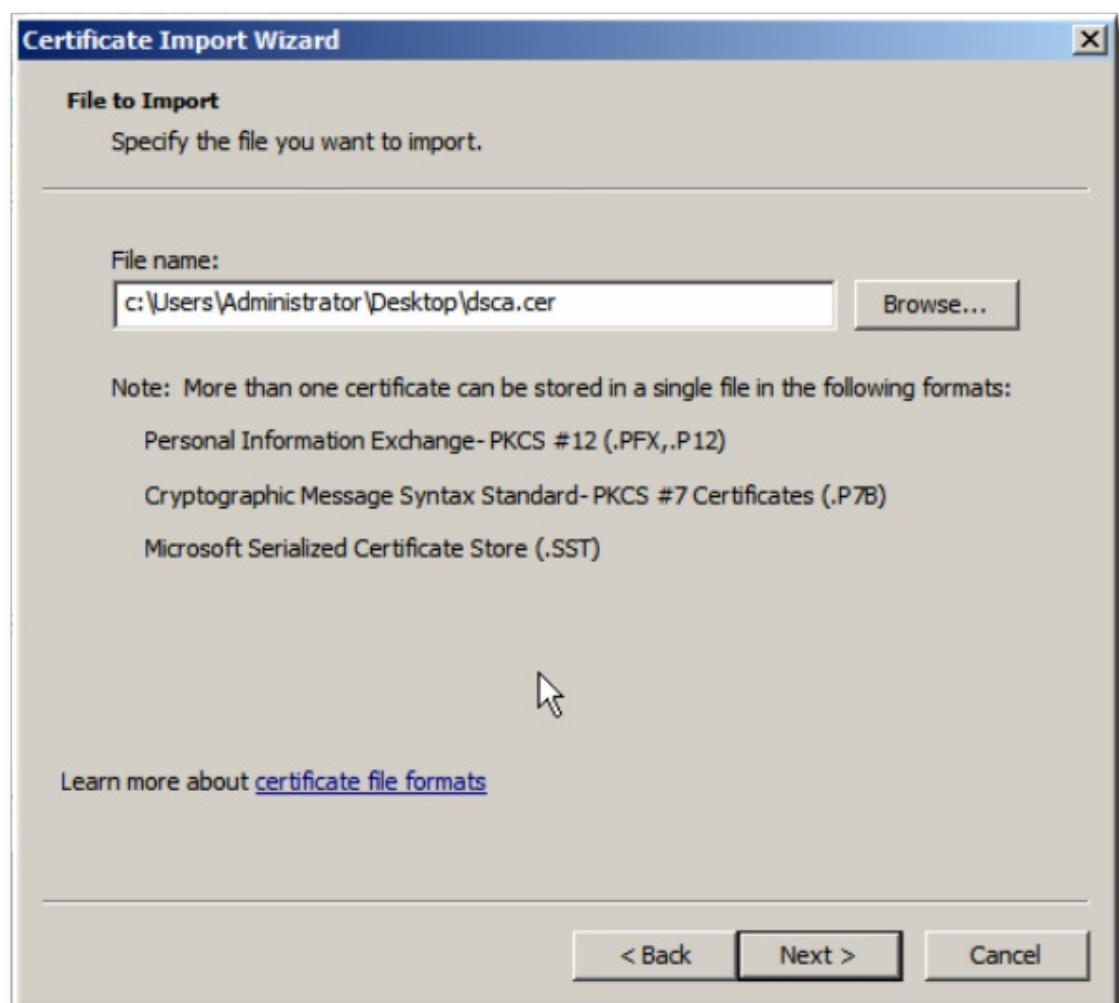
5. Copy the exported certificate from the Directory Server to the Windows machine.
6. Import the CA certificate from Directory Server into Active Directory.
  1. Open **Administrative Tools** and select the **Certificate Authority** item.
  2. Expand **Trusted Root Certification Authorities**.



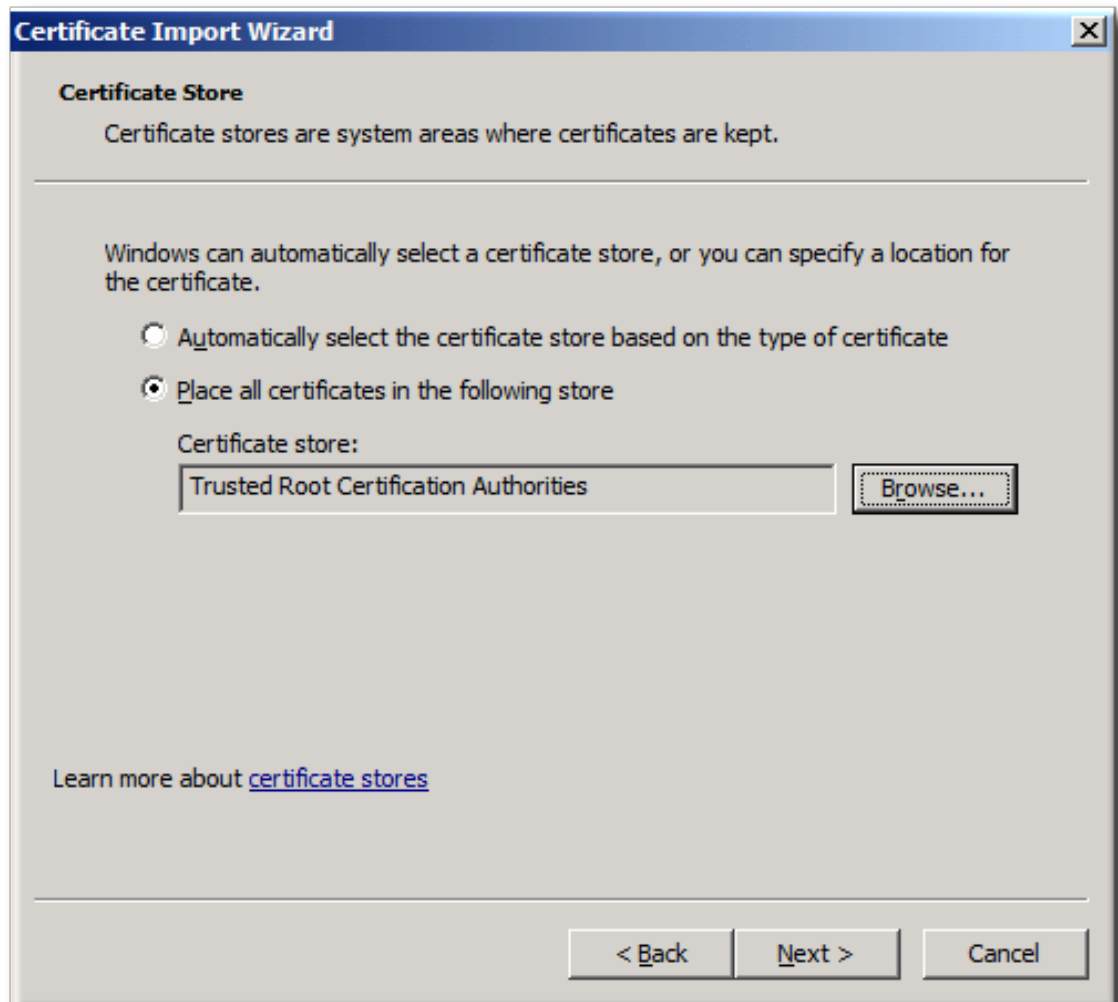
3. Right-click the **Certificates** item and select **Import**.



4. Browse to the downloaded Directory Server CA certificate, and click **Next**.



5. Save the CA certificate in the **Trusted Root Certification Authorities** store.



7. Reboot the domain controller.

To test that the server is running in SSL correctly, try searching Active Directory over LDAPS.

### 12.3.3. Step 3: Select or Create the Sync Identity

There are two users used to configure Windows Sync:

- *An Active Directory user, specified in the sync agreement.*

The user specified in the sync agreement is the entity as whom the Directory Server binds to Active Directory to send and receive updates. The Active Directory user should be a member of the Domain Admins group, or have equivalent rights, and must have rights to replicate directory changes.

For information on adding users and setting privileges in Active Directory, see the Microsoft documentation.

- *A Directory Server user, specified in the Password Sync Service.*

The user referenced in the Password Sync Service must have read and write permissions to every entry within the synchronized subtree and absolutely must have write access to password attributes in Directory Server so that Password Sync can update password changes.

**NOTE**

The user cited in the sync agreement (the supplier DN) exists on the Active Directory server. The user cited in the Password Sync configuration exists on Directory Server.

To create a sync user on Directory Server:

1. Create a new entry, such as **cn=sync user,cn=config**, with a password. For example:

```
ldapmodify -a -D "cn=directory manager" -w -p 389 -h
server.example.com -x

dn: cn=sync user,cn=config
changetype: add
objectClass: inetorgperson
objectClass: person
objectClass: top
cn: sync user
sn: SU
userPassword: secret
passwordExpirationTime: 20380119031407Z
```

2. Set an ACL that grants the sync user access to compare and write user passwords.

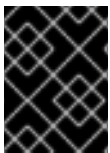
The ACL must be set at the top of the subtree which will be synchronized. For example:

```
ldapmodify -D "cn=directory manager" -w -p 389 -h server.example.com
-x

dn: ou=people,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="userPassword")(version 3.0;acl "password
sync";allow (write,compare) userdn="ldap:///cn=sync
user,cn=config";)
```

For security reasons, the Password Sync user should not be Directory Manager and should not be part of the synchronized subtree.

### 12.3.4. Step 4: Install the Password Sync Service

**IMPORTANT**

**Password Sync must be installed on every domain controller in the Active Directory domain in order to synchronize Windows passwords.**

Passwords can only be synchronized if both the Directory Server and Windows server are running in SSL, the sync agreement is configured over an SSL connection, and certificate databases are configured for Password Sync to access.

The Password Sync Service is supported on Microsoft Windows Server 2008 R2 (32-bit and 64-bit).

1. Go to <http://access.redhat.com>.

- Click the **Downloads** tab, and select the Red Hat Enterprise Linux channels, then filter for the Directory Server product and architecture.

The screenshot shows the Red Hat Downloads page with the 'Downloads' tab selected. On the left, there's a sidebar with 'Software Channels' and 'Download Software' sections. The main content area is titled 'Full Software Channel List'. It includes tabs for 'All Release Channels', 'All Beta channels', and 'Retired channels'. Below this, there's a filter section with dropdowns for 'Filter by Product Channel' (set to 'Red Hat Directory Server'), 'Latest Version', and 'All Architectures'. A table lists the channels:

| Channel Name                                                  | Architecture  |
|---------------------------------------------------------------|---------------|
| <input type="checkbox"/> Red Hat Enterprise Linux Server 6    | IA-32, x86_64 |
| <input type="checkbox"/> Red Hat Directory Server 9           | IA-32, x86_64 |
| <input type="checkbox"/> Red Hat Directory Server Debuginfo 9 | IA-32, x86_64 |

Below the table, it says 'See also: [Retired Channels](#)'.

- Open the **Downloads** tab for the Directory Server channel.

The screenshot shows the Red Hat Downloads page for 'Red Hat Directory Server 9 (for RHEL 6 for 64-bit x86\_64)'. The 'Downloads' tab is selected. The page includes sections for 'ISO Image Downloads', 'Latest Release', and a table of ISO images.

**ISO Image Downloads**

NOTE: By downloading this software, you agree to the terms and conditions of the applicable License Agreement (available at <http://www.redhat.com/licenses/>)

Not sure how to download and use these images? [Check out our ISO Download Help.](#)

**Latest Release**

Below please find the complete set of ISO images for the **latest release** of Red Hat Directory Server 9 (for RHEL 6 for 64-bit x86\_64). Depending on the variant of Red Hat Directory Server 9 (for RHEL 6 for 64-bit x86\_64) you'd like to install, you may only need a subset of these discs. ([more information](#))

**Red Hat Directory Server 9 (for RHEL 6 for 64-bit x86\_64)**

| ISO                                                 | Size  | Checksum                                                                                                              |
|-----------------------------------------------------|-------|-----------------------------------------------------------------------------------------------------------------------|
| WinSync Installer <a href="#">↗</a>                 | 3 MB  | MD5: 438ba864390d1042f6204021e83ddd75<br>SHA-256:<br>4e2cc41db6de1404dfbcbb7115aec2d1be50659bf2af536cb32804aa6a4002d  |
| Console Installer <a href="#">↗</a>                 | 4 MB  | MD5: d1e180c487b4eacd58ea1e236b129a24<br>SHA-256:<br>40bf687256c9b54a37ed89c0fd9599930780e19bda2f966d13edb1ec3d7d22b6 |
| Directory Server 9 for x86_64 DVD <a href="#">↗</a> | 15 MB | MD5: ecba5123969a5643b61ed8f25f8ef04c<br>SHA-256:<br>51ce66a92d9f9597b9539e89ffdc668268bbf310a72bbaabc627e3f75d7b4a2  |

- On the Directory Server page, download the appropriate version of the WinSync Installer. This is the Password Sync MSI file (**RedHat - PassSync - 1.1.5 - arch.msi**). Save it to the Active Directory machine.



**NOTE**

There are two PassSync packages available, one for 32-bit Windows servers and one for 64-bit. Make sure to select the appropriate packages for your Windows platform.

5. Double-click the Pass Sync MSI file to install it.
6. The **Password Sync Setup** window appears. Hit **Next** to begin installing.
7. Fill in the Directory Server host name (or IPv4 or IPv6 address), secure port number, user name (such as **cn=sync user,cn=config**), the certificate token (password), and the search base (e.g., **ou=People,dc=example,dc=com**).

Hit **Next**, then **Finish** to install Password Sync.

8. Reboot the Windows machine to start Password Sync.

**NOTE**

The Windows machine must be rebooted. Without the rebooting, **PasswordHook.dll** is not enabled, and password synchronization will not function.

The first attempt to synchronize passwords, which happened when the Password Sync application is installed, will always fail because of the SSL connection between the Directory Server and Active Directory sync peers. The tools to create the certificate and key databases is installed with the **.msi**.

Password Sync and many of its libraries are installed in **C:\Program Files\Red Hat Directory Password Synchronization**. All of the files installed with Password Sync are listed in [Table 12.1, “Installed Password Sync Libraries”](#).

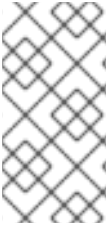
**Table 12.1. Installed Password Sync Libraries**

| Directory                                                                         | Library           | Directory                                                   | Library            |
|-----------------------------------------------------------------------------------|-------------------|-------------------------------------------------------------|--------------------|
| C:\WINDOWS\system32                                                               | passhook.dll      | C:\WINDOWS\system32                                         | libnspr4.dll       |
| C:\WINDOWS\system32                                                               | nss3.dll          | C:\WINDOWS\system32                                         | sqlite3.dll        |
| C:\WINDOWS\system32                                                               | softokn3.dll      | C:\WINDOWS\system32                                         | nssdbm3.dll        |
| C:\WINDOWS\system32                                                               | nssutil3.dll      |                                                             |                    |
| C:\WINDOWS\system32                                                               | smime3.dll        | C:\WINDOWS\system32                                         | freebl3.dll        |
| C:\Program Files\Red Hat Directory Password Synchronization                       | nsldap32v60.dll   | C:\Program Files\Red Hat Directory Password Synchronization | certutil.exe       |
| C:\Program Files\Red Hat Directory Password Synchronization                       | nsldappr32v60.dll | C:\Program Files\Red Hat Directory Password Synchronization | nsldapssl32v60.dll |
| C:\WINDOWS\system32                                                               | ssl3.dll          | C:\WINDOWS\system32                                         | libplc4.dll        |
| C:\Program Files\Red Hat Directory Password Synchronization                       | nssckbi.dll       | C:\Program Files\Red Hat Directory Password Synchronization | nsldif32v60.dll    |
| C:\Program Files\Red Hat Directory Password Synchronization                       | passsync.log[a]   | C:\Program Files\Red Hat Directory Password Synchronization | passsync.exe       |
| C:\Program Files\Red Hat Directory Password Synchronization                       | pk12util.exe      | C:\Program Files\Red Hat Directory Password Synchronization | msvcr71.dll        |
| C:\WINDOWS\system32                                                               | libplds4.dll      |                                                             |                    |
| [a] This log file is not an installed library, but it is created at installation. |                   |                                                             |                    |

### 12.3.5. Step 5: Configure the Password Sync Service

Next, set up certificates that Password Sync uses to access the Directory Server over SSL:



**NOTE**

SSL is required for Password Sync to send passwords to Directory Server. The service will not send the passwords except over SSL to protect the clear text password sent from the Active Directory machine to the Directory Server machine. This means that Password Sync will not work until SSL is configured.

1. On the Directory Server, export the CA certificate.

```
[root@server ~]# cd /etc/dirsrv/slapd-instance_name
[root@server ~]# certutil -d . -L -n "CA certificate" -a > dsca.crt
```

2. Copy the exported certificate from the Directory Server to the Windows machine.
3. Open a command prompt on the Windows machine, and open the **Password Sync** installation directory.

```
C:\Windows\system32>cd "C:\Program Files\Red Hat Directory Password Synchronization"
```

4. Create new **cert8.db** and **key.db** databases on the Windows machine.

```
C:\C:\Program Files\Red Hat Directory Password Synchronization>certutil.exe -d . -N
```

5. Import the CA certificate from the Directory Server into the new certificate database.

```
certutil.exe -d . -A -n "DS CA cert" -t CT,, -a -i \path\to\dsca.crt
```

6. Verify that the CA certificate was correctly imported.

```
certutil.exe -d . -L -n "DS CA cert"
```

7. Reboot the Windows machine. The Password Sync service is not available until after a system reboot.

**NOTE**

If any Active Directory user accounts exist when Password Sync is first installed, then the passwords for those user accounts cannot be synchronized until they are changed because Password Sync cannot decrypt a password once it has been hashed in Active Directory.

### 12.3.6. Step 6: Configure the Directory Server Database for Synchronization

Just as with replication, there must be a changelog available to track and send directory changes and the Directory Server database being synchronized must be configured as a replica.

**NOTE**

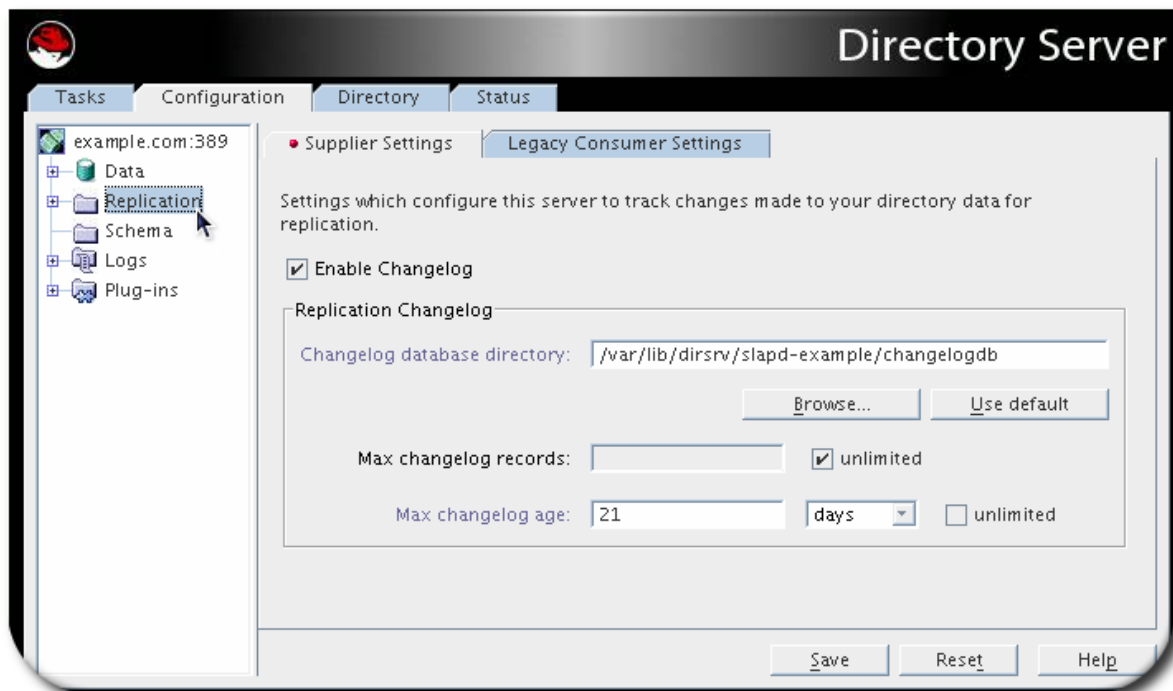
If the Directory Server database is already configured for replication, this step is not necessary.

Setting up a database for replication is described in [Section 11.5.1, “Configuring the Read-Write Replicas on the Supplier Servers”](#).

### 12.3.6.1. Setting up the Directory Server from the Console

First, enable the changelog:

1. In the Directory Server Console, select the **Configuration** tab.
2. In the left-hand navigation tree, click the **Replication** folder.
3. In the main window, click the **Supplier Settings** tab.
4. Check the **Enable Changelog** database.



5. Set the changelog database directory. Click the **Use default** button to use the default or **Browse . . .** to select a custom directory.
6. Save the changelog settings.

After setting up the changelog, then configure the database that will be synchronized as a replica. The replica role should be either a single-master or multi-master.



## IMPORTANT

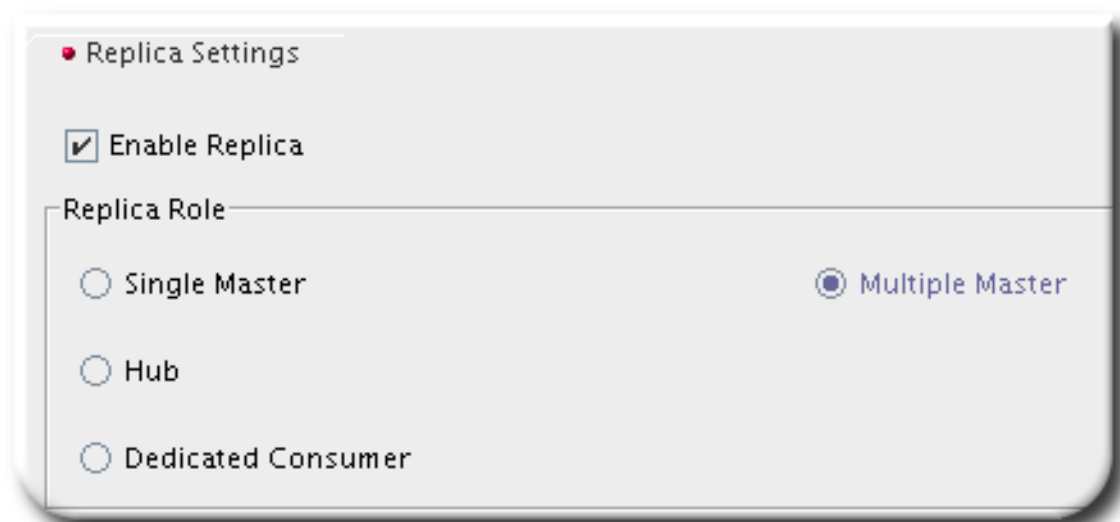
While it is possible to configure a sync agreement on a hub server, this only allows uni-directional synchronization, from Red Hat Directory Server to Active Directory. The Active Directory server cannot sync any changes back to the hub.

It is strongly recommended that only masters in multi-master replication be used to configure synchronization agreements.

1. In the Directory Server Console, select the **Configuration** tab.
2. In the left-hand navigation tree, click the **Replication** folder, then click the name of the database to synchronize.

By default, there are two databases, **NetscapeRoot** for directory configuration and **userRoot** for directory entries. Other databases may be listed if they have been added to Directory Server.

3. Check the **Enable Replica** check box, and select the radio button by the type of replica which the database is.



4. In the **Update Settings** section, either select or add a supplier DN. This is the user account as which synchronization process will be run. As mentioned in [Section 12.3.3, “Step 3: Select or Create the Sync Identity”](#), this user must be on the Active Directory server.

5. Save the replication settings for the database.



#### NOTE

For more information on replication settings, see [Chapter 11, Managing Replication](#).

### 12.3.6.2. Setting up the Directory Server for Sync from the Command Line

First, enable the changelog:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=changelog5,cn=config
changetype: add
objectclass: top
objectclass: extensibleObject
cn: changelog5
nsslapd-changelogdir: /var/lib/dirsrv/slapd-instance_name/changelogdb
```

Then, create the supplier replica entry:

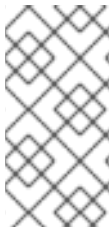
```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=sync replica,cn=dc=example\,dc=com,cn=mapping tree,cn=config
changetype: add
objectclass: top
objectclass: nsds5replica
objectclass: extensibleObject
cn: sync replica
nsds5replicaroot: dc=example,dc=com
nsds5replicaid: 7
nsds5replicatype: 3
```

```
nsds5flags: 1
nsds5ReplicaPurgeDelay: 604800
nsds5ReplicaBindDN: cn=sync user,cn=config
```

These different parameters are described in more detail in the *Configuration and Command-Line Tool Reference* and [Section 11.7.1, “Configuring Suppliers from the Command Line”](#).

### 12.3.7. Step 7: Create the Synchronization Agreement

Create the synchronization agreement.



#### NOTE

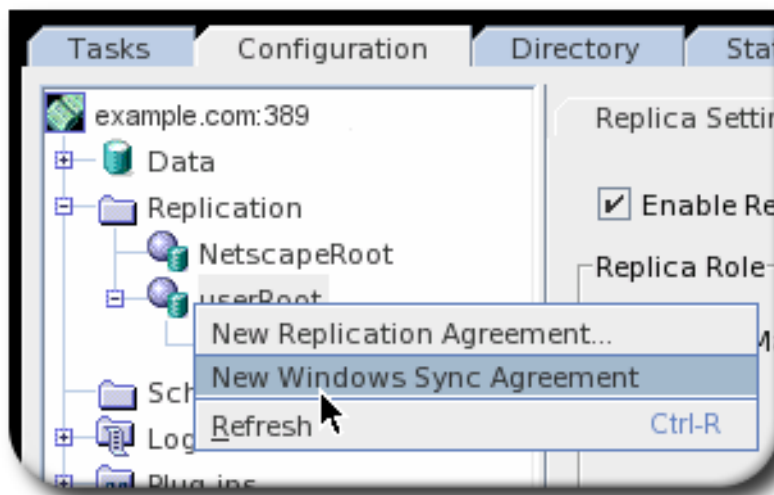
If secure binds are required for simple password authentication ([Section 14.8.1, “Requiring Secure Binds”](#)), then any replication operations will fail unless they occur over a secure connection. Using a secure connection (SSL/TLS and Start TLS connections or SASL authentication) is recommended, anyway.

#### 12.3.7.1. Creating the Sync Agreement from the Console

1. In the Directory Server Console, select the **Configuration** tab.
2. In the left-hand navigation tree, click **Replication**, then right-click on the database to sync. The default user database is **userRoot**, but additional databases are added as new suffixes are added to the Directory Server.

Alternatively, highlight the database, and in the top tool bar, click **Object**.


3. Select **New Windows Sync Agreement** from the menu.



4. In the two fields, supply a name and description of the synchronization agreement. Hit **Next**.
5. In the **Windows Sync Server Info** window, fill in the Active Directory information in the **Windows Domain Information** area.

Windows Sync Server Info

Provide server and content information:

Supplier:  example.com:389

Windows Domain Information

Windows Domain Name: ad1

Sync New Windows Users: ☒ Sync New Windows Groups: ☒

Windows Subtree: cn=Users,dc=ad1

DS Subtree: ou=People,dc=example,dc=com

Domain Controller Host: ad-server

Port Num: 389

Connection

☐ Use LDAP (no encryption)

☐ Use TLS/SSL (TLS/SSL encryption with LDAPS)

☒ Use StartTLS (TLS/SSL encryption with LDAP)

Bind as: cn=sync,cn=Users,dc=domain,dc=com

Password: .....

Subtree: dc=example,dc=com

Back Next Cancel Help

- The name of the Windows domain.
  - What kinds of entries to synchronize; users and groups are synchronized independently. When a type of entry is chosen, then all of the entries of that type that are found in the Windows subtree are created in the Directory Server.
  - The Windows and Directory Server subtree information; this is automatically filled in.
  - The host name, IPv4 address, or IPv6 address of the domain controller
  - The Windows server's port number
6. Set the connection type. There are three options:

- *Use LDAP.* This sets either a standard, unencrypted connection.
- *Use TLS/SSL.* This uses a secure connection over the server's secure LDAPS port, such as **636**. Both the Directory Server and the Windows server must be properly configured to run in TLS/SSL for this connection and must have installed each other's CA certificates in order to trust their server certificates.
- *Use Start TLS.* This uses Start TLS to establish a secure connection over the server's standard port. Like regular SSL, these peer servers must be able to trust each other's certificates.

Using either TLS/SSL or Start TLS is recommended for security reasons. TLS/SSL or Start TLS is required for synchronizing passwords because Active Directory refuses to modify passwords unless the connection is SSL-protected.

7. Fill in the authentication information in the **Bind as...** and **Password** fields with the sync ID information. This user must exist in the Active Directory domain.
8. Save the sync agreement.



## NOTE

By default, Win Sync polls the Active Directory peer every five (5) minutes to check for changes. In the sync agreement summary, this is displayed as the **Update Interval**. The update interval can be changed by editing the *winSyncInterval* attribute manually. See [Section 12.10.2, “Adding and Editing the Sync Agreement in the Command Line”](#).

When the agreement is complete, the new sync agreement is listed under the suffix.

### 12.3.7.2. Creating the Sync Agreement from the Command Line

It is also possible to add the sync agreement through the command line.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=ExampleSyncAgreement,cn=sync
replica,cn=dc=example\,dc=com,cn=mapping tree,cn=config
changetype: add
objectclass: top
objectclass: nsDSWindowsReplicationAgreement
cn: ExampleSyncAgreement
nsds7WindowsReplicaSubtree: cn=Users,dc=ad1
nsds7DirectoryReplicaSubtree: ou=People,dc=example,dc=com
nsds7NewWinUserSyncEnabled: on
nsds7NewWinGroupSyncEnabled: on
nsds7WindowsDomain: ad1
nsDS5ReplicaRoot: dc=example,dc=com
nsDS5ReplicaHost: ad1.windows-server.com
nsDS5ReplicaPort: 389
nsDS5ReplicaBindDN: cn=sync user,cn=config
nsDS5ReplicaCredentials: {DES}ffGad646dT0nnsT8nJOaMA==
nsDS5ReplicaTransportInfo: TLS
winSyncInterval: 1200
```

All of the different parameters used in the sync agreement are listed in [Table 12.6, “Sync Agreement Attributes”](#). These different parameters are described in more detail in the *Configuration and Command-Line Tool Reference*.

### 12.3.8. Step 8: Configure Directory Server User and Group Entries for Synchronization

Add the **ntUser** and **ntGroup** object classes to any user and group entries, respectively, which will be synchronized, along with any required attributes. Only Directory Server entries with those object classes are synchronized. Active Directory entries which are synced over to Directory Server have those object classes automatically.

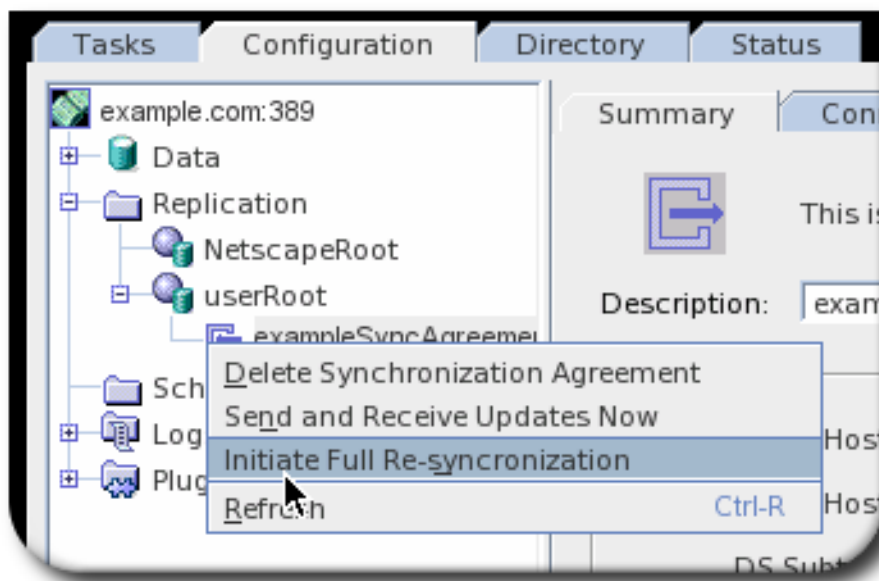
Whenever the appropriate object classes are added to an entry, both for new entries and existing entries, the entry is synced over at the next incremental update.

Configuring Directory Server user entries for synchronization is described in [Section 12.4.3, “Configuring User Sync for Directory Server Users”](#), and configuring Directory Server group entries for synchronization is described in [Section 12.5.4, “Configuring Group Sync for Directory Server Groups”](#).

### 12.3.9. Step 9: Begin Synchronization

After the sync agreement is created, begin the synchronization process.

1. Go to the **Configuration** tab in the Console.
2. Open the **Replication** folder and expand the appropriate database.
3. Select the sync agreement.
4. Right-click on the agreement or open the **Object** menu.
5. Select **Initiate Full Re-synchronization**.



If synchronization stops for any reason, begin another total update (resynchronization) by selecting this from the sync agreement menu. Beginning a total update (resynchronization) will not delete or overwrite the databases.



## 12.4. SYNCHRONIZING USERS

Users are not automatically synced between Directory Server and Active Directory. Synchronization both directions has to be configured:

- Users in the Active Directory domain are synced if it is configured in the sync agreement by selecting the **Sync New Windows Users** option. All of the Windows users are copied to the Directory Server when synchronization is initiated and then new users are synced over when they are created.
- A Directory Server user account is synchronized to Active Directory through specific attributes that are present on the Directory Server entry. Any Directory Server entry must have the **ntUser** object class and the **ntUserCreateNewAccount** attribute; the **ntUserCreateNewAccount** attribute (even on an existing entry) signals Directory Server Win Syn to write the entry over to the Active Directory server.

New or modified user entries with the **ntUser** object class added are created and synced over to the Windows machine at the next regular update, which is a standard poll of entry.



### NOTE

A user is not active on the Active Directory domain until it has a password. When an existing user is modified to have the required Windows attributes, that user entry will be synced over to the Active Directory domain, but will not be able to log in until the password is changed on the Directory Server side or an administrator sets the password on Active Directory. This is because passwords stored in the Directory Server are encrypted, and Password Sync cannot sync already encrypted passwords.

To make the user active on the Active Directory domain, reset the user's password.

All synchronized entries *in the Directory Server*, whether they originated in the Directory Server or in Active Directory, have special synchronization attributes:

- **ntUserDomainId**. This corresponds to the **sAMAccountName** attribute for Active Directory entries.
- **ntUniqueld**. This contains the value of the **objectGUID** attribute for the corresponding Windows entry. This attribute is set by the synchronization process and should not be set or modified manually.
- **ntUserDeleteAccount**. This attribute is set automatically when a Windows entry is synced over but must be set manually for Directory Server entries. If **ntUserDeleteAccount** has the value **true**, the corresponding Windows entry be deleted when the Directory Server entry is deleted. Otherwise, the entry remains in Active Directory, but is removed from the Directory Server database if it is deleted in the Directory Server.

Setting **ntUserCreateNewAccount** and **ntUserDeleteAccount** on Directory Server entries allows the Directory Manager precise control over which users within the synchronized subtree are synced on Active Directory.

### 12.4.1. User Attributes Synchronized between Directory Server and Active Directory

Only a subset of Directory Server and Active Directory attributes are synchronized. These attributes are

hard-coded and are defined regardless of which way the entry is being synchronized. Any other attributes present in the entry, either in Directory Server or in Active Directory, remain unaffected by synchronization.

Some attributes used in Directory Server and Active Directory are identical. These are usually attributes defined in an LDAP standard, which are common among all LDAP services. These attributes are synchronized to one another exactly. [Table 12.3, “User Schema That Are the Same in Directory Server and Windows Servers”](#) shows attributes that are the same between the Directory Server and Windows servers.

Some attributes define the same information, but the names of the attributes or their schema definitions are different. These attributes are mapped between Active Directory and Directory Server, so that attribute A in one server is treated as attribute B in the other. For synchronization, many of these attributes relate to Windows-specific information. [Table 12.2, “User Schema Mapped between Directory Server and Active Directory”](#) shows the attributes that are mapped between the Directory Server and Windows servers.

For more information on the differences in ways that Directory Server and Active Directory handle some schema elements, see [Section 12.4.2, “User Schema Differences between Red Hat Directory Server and Active Directory”](#).

**Table 12.2. User Schema Mapped between Directory Server and Active Directory**

| Directory Server   | Active Directory |
|--------------------|------------------|
| cn[a]              | name             |
| ntUserDomainId     | sAMAccountName   |
| ntUserHomeDir      | homeDirectory    |
| ntUserScriptPath   | scriptPath       |
| ntUserLastLogon    | lastLogon        |
| ntUserLastLogoff   | lastLogoff       |
| ntUserAcctExpires  | accountExpires   |
| ntUserCodePage     | codePage         |
| ntUserLogonHours   | logonHours       |
| ntUserMaxStorage   | maxStorage       |
| ntUserProfile      | profilePath      |
| ntUserParms        | userParameters   |
| ntUserWorkstations | userWorkstations |

| Directory Server                                                                                                                                                                                                                                                                                                                                               | Active Directory |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| [a] The <b>cn</b> is treated differently than other synced attributes. It is mapped directly ( <b>cn</b> to <b>cn</b> ) when syncing from Directory Server to Active Directory. When syncing from Active Directory to Directory Server, however, <b>cn</b> is mapped from the <b>name</b> attribute on Windows to the <b>cn</b> attribute in Directory Server. |                  |

**Table 12.3. User Schema That Are the Same in Directory Server and Windows Servers**

|                                                                                                                                                                                                                                                                                                                                                                |                            |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|
| cn[a]                                                                                                                                                                                                                                                                                                                                                          | physicalDeliveryOfficeName |
| description                                                                                                                                                                                                                                                                                                                                                    | postOfficeBox              |
| destinationIndicator                                                                                                                                                                                                                                                                                                                                           | postalAddress              |
| facsimileTelephoneNumber                                                                                                                                                                                                                                                                                                                                       | postalCode                 |
| givenname                                                                                                                                                                                                                                                                                                                                                      | registeredAddress          |
| homePhone                                                                                                                                                                                                                                                                                                                                                      | sn                         |
| homePostalAddress                                                                                                                                                                                                                                                                                                                                              | st                         |
| initials                                                                                                                                                                                                                                                                                                                                                       | street                     |
| l                                                                                                                                                                                                                                                                                                                                                              | telephoneNumber            |
| mail                                                                                                                                                                                                                                                                                                                                                           | teletexTerminalIdentifier  |
| mobile                                                                                                                                                                                                                                                                                                                                                         | telexNumber                |
| o                                                                                                                                                                                                                                                                                                                                                              | title                      |
| ou                                                                                                                                                                                                                                                                                                                                                             | usercertificate            |
| pager                                                                                                                                                                                                                                                                                                                                                          | x121Address                |
| [a] The <b>cn</b> is treated differently than other synced attributes. It is mapped directly ( <b>cn</b> to <b>cn</b> ) when syncing from Directory Server to Active Directory. When syncing from Active Directory to Directory Server, however, <b>cn</b> is mapped from the <b>name</b> attribute on Windows to the <b>cn</b> attribute in Directory Server. |                            |

### 12.4.2. User Schema Differences between Red Hat Directory Server and Active Directory

Although Active Directory supports the same basic X.500 object classes as Directory Server, there are a few incompatibilities of which administrators should be aware.

#### 12.4.2.1. Values for **cn** Attributes

In Directory Server, the **cn** attribute can be multi-valued, while in Active Directory this attribute must have only a single value. When the Directory Server **cn** attribute is synchronized, then, only one value is sent to the Active Directory peer.

What this means for synchronization is that, potentially, if a **cn** value is added to an Active Directory entry and that value is not one of the values for **cn** in Directory Server, then all of the Directory Server **cn** values are overwritten with the single Active Directory value.

One other important difference is that Active Directory uses the **cn** attribute as its naming attribute, where Directory Server uses **uid**. This means that there is the potential to rename the entry entirely (and accidentally) if the **cn** attribute is edited in the Directory Server. If that **cn** change is written over to the Active Directory entry, then the entry is renamed, and the new named entry is written back over to Directory Server.

#### 12.4.2.2. Password Policies

Both Active Directory and Directory Server can enforce password policies such as password minimum length or maximum age. Windows Sync makes no attempt to ensure that the policies are consistent, enforced, or synchronized. If password policy is not consistent in both Directory Server and Active Directory, then password changes made on one system may fail when synced to the other system. The default password syntax setting on Directory Server mimics the default password complexity rules that Active Directory enforces.

#### 12.4.2.3. Values for **street** and **streetAddress**

Active Directory uses the attribute **streetAddress** for a user or group's postal address; this is the way that Directory Server uses the **street** attribute. There are two important differences in the way that Active Directory and Directory Server use the **streetAddress** and **street** attributes, respectively:

- In Directory Server, **streetAddress** is an alias for **street**. Active Directory also has the **street** attribute, but it is a separate attribute that can hold an independent value, not an alias for **streetAddress**.
- Active Directory defines both **streetAddress** and **street** as single-valued attributes, while Directory Server defines **street** as a multi-valued attribute, as specified in RFC 4519.

Because of the different ways that Directory Server and Active Directory handle **streetAddress** and **street** attributes, there are two rules to follow when setting address attributes in Active Directory and Directory Server:

- Windows Sync maps **streetAddress** in the Windows entry to **street** in Directory Server. To avoid conflicts, the **street** attribute should not be used in Active Directory.
- Only one Directory Server **street** attribute value is synced to Active Directory. If the **streetAddress** attribute is changed in Active Directory and the new value does not already exist in Directory Server, then all **street** attribute values in Directory Server are replaced with the new, single Active Directory value.

#### 12.4.2.4. Constraints on the initials Attribute

For the **initials** attribute, Active Directory imposes a maximum length constraint of six characters, but Directory Server does not have a length limit. If an **initials** attribute longer than six characters is added to Directory Server, the value is trimmed when it is synchronized with the Active Directory entry.

### 12.4.3. Configuring User Sync for Directory Server Users

For Directory Server users to be synchronized over to Active Directory, the user entries must have the appropriate sync attributes set.

#### 12.4.3.1. Configuring User Sync in the Console

1. In the Directory Server Console, select the **Directory** tab.
2. For an existing entry, right-click the entry, and click **Properties** to open the property editor for the entry.  
  
For a new entry, right-click the main entry in the left window to add the new entry, select **User**, and then fill in the required entry attributes.
3. On the left side of the **Property Editor**, click the **NT User** link.
4. In the **NT User** tab, check the **Enable NT Attributes** check box.

The screenshot shows the 'Property Editor' window for a user named 'Sam Carter' in the 'Accounting' group. The left sidebar lists 'User', 'Languages', 'NT User' (selected), 'Posix User', and 'Account'. The main area shows the 'NT User' configuration. The 'Enable NT User Attributes' checkbox is checked. Below it, the '\* NT User ID' field contains 'Sam Carter'. The 'Create New NT Account' and 'Delete NT Account If Person Deleted' checkboxes are also checked. Other fields include 'Comment', 'User Profile Path', 'Logon Script', 'Home Drive' (set to 'C'), 'Home Directory', 'Logon Server', 'Logon Hours' (button), 'User Workstations List', and 'Account Expiration Date' (set to 'Never Expires' with a 'Change' button). A note at the bottom states '\* Indicates a required field'. At the bottom of the window are buttons for 'Advanced...', 'OK' (with a mouse cursor), 'Cancel', and 'Help'.

5. To enable synchronization, two fields are required:
  - Setting a **NT User ID**
  - Selecting the **Create New NT Account** check box

6. Selecting the **Delete NT Account** check box means that the corresponding Windows user is deleted if the Directory Server entry is deleted.
7. Set the other Windows attributes. These attributes are mapped to relevant Windows attributes.

Additional **ntUser** attributes can be created either by using the **Advanced** button; see [Section 3.2.4.2, “Modifying Entries Using ldapmodify”](#).



## NOTE

Reset the user's password.

A user is not active on the Active Directory domain until it has a password. When an existing user is modified to have the required Windows attributes, that user entry will be synced over to the Active Directory domain, but will not be able to log in until the password is changed on the Directory Server side or an administrator sets the password on Active Directory. Password Sync cannot sync encrypted passwords.

So, to make the user active on the Active Directory domain, reset the user's password.

### 12.4.3.2. Configuring User Sync in the Command Line

To enable synchronization through the command line, add the required sync attributes to an entry or create an entry with those attributes.

Three schema elements are required for synchronization:

- The **ntUser** object class
- The **ntUserDomainId** attribute, to give the Windows ID
- The **ntUserCreateNewAccount** attribute, to signal to the synchronization plug-in to sync the Directory Server entry over to Active Directory

For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: uid=scarter,ou=People,dc=example,dc=com
changetype: modify
add: objectClass
objectClass:ntUser

add: ntUserDomainId
ntUserDomainId: Sam Carter

add: ntUserCreateNewAccount
ntUserCreateNewAccount: true

add: ntUserDeleteAccount
ntUserDeleteAccount: true
```

Many additional Windows and user attributes can be added to the entry. All of the schema which is synchronized is listed in [Section 12.4.1, “User Attributes Synchronized between Directory Server and Active Directory”](#). Windows-specific attributes, belonging to the **ntUser** object class, are described in

more detail in the [Red Hat Directory Server 9 Configuration, Command, and File Reference](#).



## NOTE

Reset the user's password.

A user is not active on the Active Directory domain until it has a password. When an existing user is modified to have the required Windows attributes, that user entry will be synced over to the Active Directory domain, but will not be able to log in until the password is changed on the Directory Server side or an administrator sets the password on Active Directory. Password Sync cannot sync encrypted passwords.

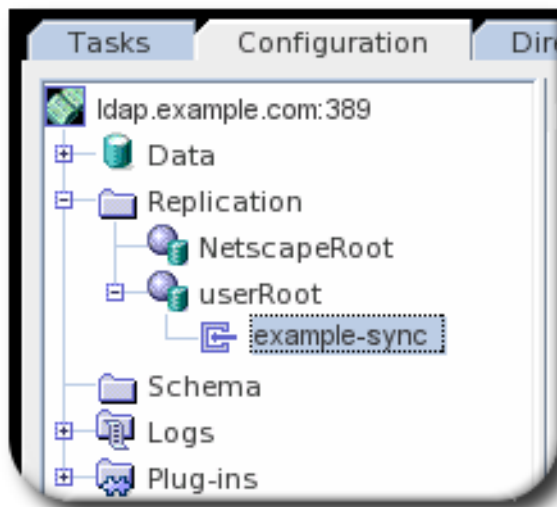
So, to make the user active on the Active Directory domain, reset the user's password.

### 12.4.4. Configuring User Sync for Active Directory Users

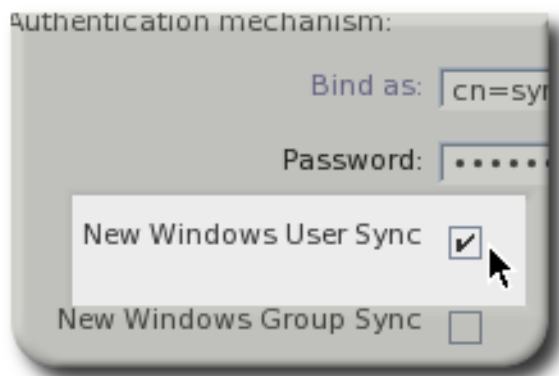
Synchronization for Windows users (users which originate in the Active Directory domain) is configured in the sync agreement.

#### 12.4.4.1. Configuring User Sync in the Console

1. Open the **Configuration** tab and expand the **Replication** folder.
2. Open the appropriate database, and select the sync agreement.



3. Open the **Connection** tab.
4. Check the **New Windows User Sync** check box to enable users sync. To disable sync, uncheck the box.



For new sync agreements, select the corresponding users sync check box in the sync agreement creation wizard.

#### 12.4.4.2. Configuring User Sync in the Command Line

The attribute to set Active Directory user sync is ***nsds7NewWinUserSyncEnabled*** and is set on the sync agreement. To enable user sync, add this attribute to the sync agreement or create a sync agreement with this attribute set to **on**:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=ExampleSyncAgreement,cn=userRoot,cn=dc=example\,dc=com,cn=mapping
tree,cn=config
changetype: modify
replace: nsds7NewWinUserSyncEnabled
nsds7NewWinUserSyncEnabled: on
```

To disable user sync, set ***nsds7NewWinUserSyncEnabled***: **off**.

## 12.5. SYNCHRONIZING GROUPS

Like user entries, groups are not automatically synced between Directory Server and Active Directory. Synchronization both directions has to be configured:

- Groups in the Active Directory domain are synced if it is configured in the sync agreement by selecting the **Sync New Windows Groups** option. All of the Windows groups are copied to the Directory Server when synchronization is initiated and then new groups are synced over as they are created.
- A Directory Server group account is synchronized to Active Directory through specific attributes that are present on the Directory Server entry. Any Directory Server entry must have the **ntGroup** object class and the **ntGroupCreateNewGroup** attribute; the **ntGroupCreateNewGroup** attribute (even on an existing entry) signals Directory Server Win Sync to write the entry over to the Active Directory server.

New or modified groups that have the **ntGroup** object class are created and synced over to the Windows machine at the next regular update.





## IMPORTANT

When a group is synced, the list of all of its members is also synced. However, the member entries themselves are not synced unless user sync is enabled and applies to those entries.

This could create a problem if an application or service tries to do a modify operation on all members in a group on the Active Directory server, and some of those users do not exist.

Additionally, groups have a few other common attributes:

- Two attributes control whether Directory Server groups are created and deleted on Active Directory, ***ntGroupCreateNewGroup*** and ***ntGroupDeleteGroup***.

***ntGroupCreateNewGroup*** is required to sync Directory Server groups over to Active Directory.

- ***ntUserDomainId*** contains the unique ID for the entry on the Active Directory domain. This is the only required attribute for the ***ntGroup*** object class.
- ***ntGroupType*** is the type of Windows group. Windows group types are global/security, domain local/security, global/distribution, or domain local/distribution. This is set automatically for Windows groups that are synchronized over, but this attribute must be set manually on Directory Server entries before they can be synced.

### 12.5.1. About Windows Group Types

In Active Directory, there are two major types of groups: security and distribution. Security groups are most similar to groups in Directory Server, since security groups can have policies configured for access controls, resource restrictions, and other permissions. Distribution groups are for mailing distribution. These are further broken down into global and local groups. The Directory Server ***ntGroupType*** supports all four group types:

- **-21483646** for global/security (the default)
- **-21483644** for domain local/security
- **2** for global/distribution
- **4** for domain local/distribution

### 12.5.2. Group Attributes Synchronized between Directory Server and Active Directory

Only a subset of Directory Server and Active Directory attributes are synchronized. These attributes are hard-coded and are defined regardless of which way the entry is being synchronized. Any other attributes present in the entry, either in Directory Server or in Active Directory, remain unaffected by synchronization.

Some attributes used in Directory Server and Active Directory group entries are identical. These are usually attributes defined in an LDAP standard, which are common among all LDAP services. These attributes are synchronized to one another exactly. [Table 12.5, “Group Entry Attributes That Are the Same between Directory Server and Active Directory”](#) shows attributes that are the same between the Directory Server and Windows servers.

Some attributes define the same information, but the names of the attributes or their schema definitions are different. These attributes are mapped between Active Directory and Directory Server, so that attribute A in one server is treated as attribute B in the other. For synchronization, many of these attributes relate to Windows-specific information. [Table 12.4, “Group Entry Attribute Mapping between Directory Server and Active Directory”](#) shows the attributes that are mapped between the Directory Server and Windows servers.

For more information on the differences in ways that Directory Server and Active Directory handle some schema elements, see [Section 12.5.3, “Group Schema Differences between Red Hat Directory Server and Active Directory”](#).

**Table 12.4. Group Entry Attribute Mapping between Directory Server and Active Directory**

| Directory Server                                                                                                        | Active Directory |
|-------------------------------------------------------------------------------------------------------------------------|------------------|
| cn                                                                                                                      | name             |
| ntUserDomainID                                                                                                          | name             |
| ntGroupType                                                                                                             | groupType        |
| <div><div>uniqueMember</div><div>member</div></div>                                                                     | Member[a]        |
| [a] The <b>Member</b> attribute in Active Directory is synced to the <b>uniqueMember</b> attribute in Directory Server. |                  |

**Table 12.5. Group Entry Attributes That Are the Same between Directory Server and Active Directory**

|             |         |
|-------------|---------|
| cn          | o       |
| description | ou      |
| l           | seeAlso |
| mail        |         |

**12.5.3. Group Schema Differences between Red Hat Directory Server and Active Directory**

Although Active Directory supports the same basic X.500 object classes as Directory Server, there are a few incompatibilities of which administrators should be aware.

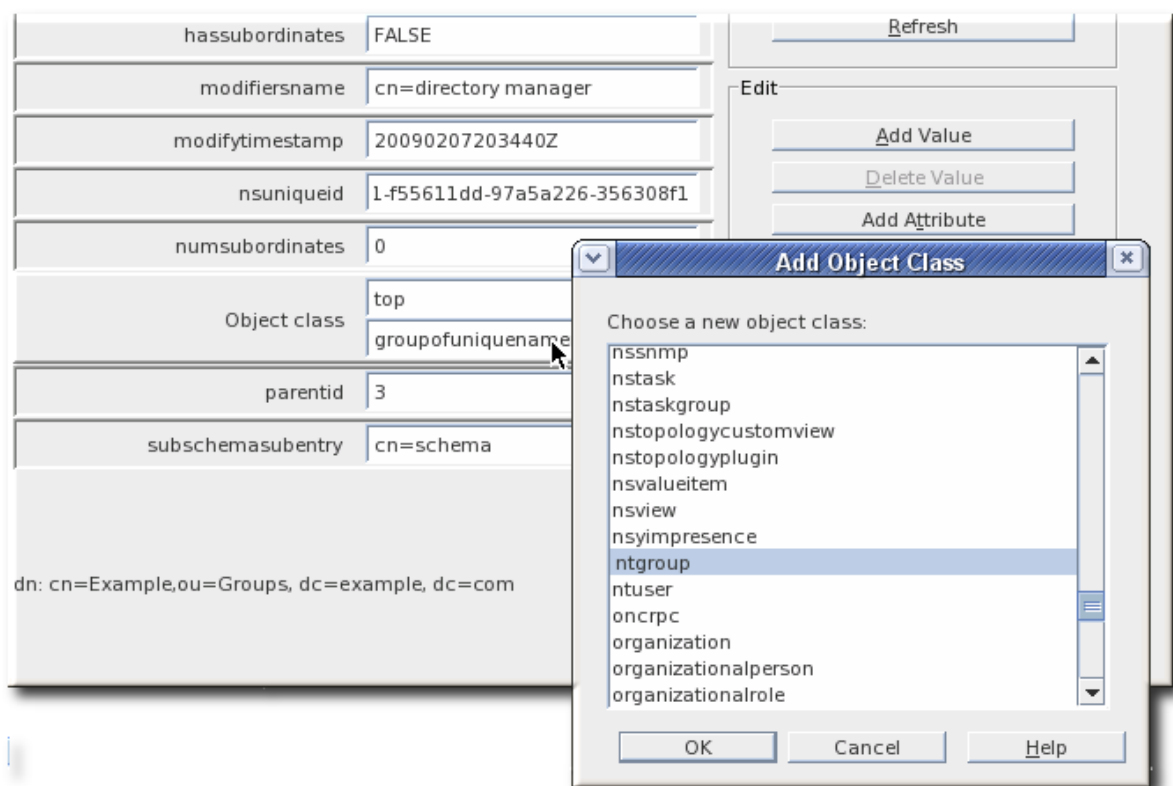
Nested groups (where a group contains another group as a member) are supported and for WinSync are synchronized. However, Active Directory imposes certain constraints as to the composition of nested groups. For example, a global group contain a domain local group as a member. Directory Server has no concept of local and global groups, and, therefore, it is possible to create entries on the Directory Server side that violate Active Directory's constraints when synchronized.

### 12.5.4. Configuring Group Sync for Directory Server Groups

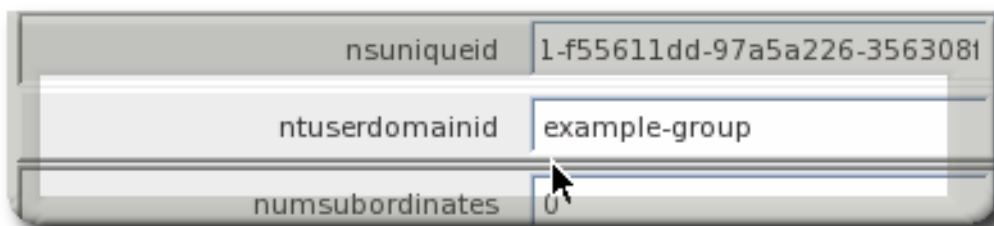
For Directory Server groups to be synchronized over to Active Directory, the group entries must have the appropriate sync attributes set.

#### 12.5.4.1. Configuring Group Sync in the Console

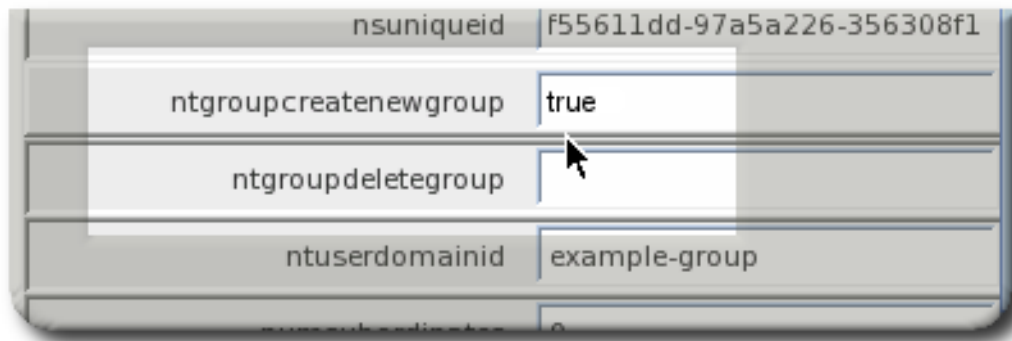
1. In the Directory Server Console, select the **Directory** tab.
2. Right-click the group entry, and click **Advanced** to open the advanced property editor for the entry. All of the sync-related attributes must be added manually, so only the advanced property editor can set the attributes.
3. Click the **objectClasses** field, and then click the **Add Value** button.
4. Select the **ntGroup** object class.



5. Setting the **ntGroup** object class automatically adds the **ntUserDomainId** attribute. This attribute is required, so add a value.



6. To enable synchronization, click the **Add Attribute** button, and select the **ntGroupCreateNewGroup** attribute from the list. Then, set its value to **true**. This signals to the sync plug-in that the entry should be added to the Active Directory directory.



To delete the group entry from the Active Directory domain if it is deleted from the Directory Server database, set the ***ntGroupDeleteGroup*** attribute and set it to **true**.

7. Add any other Windows attributes for the Directory Server entry. The available attributes are listed in [Section 12.5.2, “Group Attributes Synchronized between Directory Server and Active Directory”](#).

If the ***ntGroupType*** is not added, then the group is automatically added as a global security group (***ntGroupType***: -21483646).

#### 12.5.4.2. Configuring Group Sync in the Command Line

To enable synchronization through the command line, add the required sync attributes to an entry or create an entry with those attributes.

Three schema elements are required for synchronization:

- The ***ntGroup*** object class.
- The ***ntUserDomainId*** attribute, to give the Windows ID for the entry.
- The ***ntGroupCreateNewGroup*** attribute, to signal to the synchronization plug-in to sync the Directory Server entry over to Active Directory.

The ***ntGroupDeleteGroup*** attribute is optional, but this sets whether to delete the entry automatically from the Active Directory domain if it is deleted in the Directory Server.

It is also recommended to the ***ntGroupType*** attribute. If this attribute is not specified, then the group is automatically added as a global security group (***ntGroupType***: -21483646).

For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x

dn: cn=Example Group,ou=Groups,dc=example,dc=com
changetype: modify
add: objectClass
objectClass:ntGroup

add: ntUserDomainId
ntUserDomainId: example-group

add: ntGroupCreateNewGroup
ntGroupCreateNewGroup: true
```

```
add: ntGroupDeleteGroup
ntGroupDeleteGroup: true
```

```
add: ntGroupType
ntGroupType: 2
```

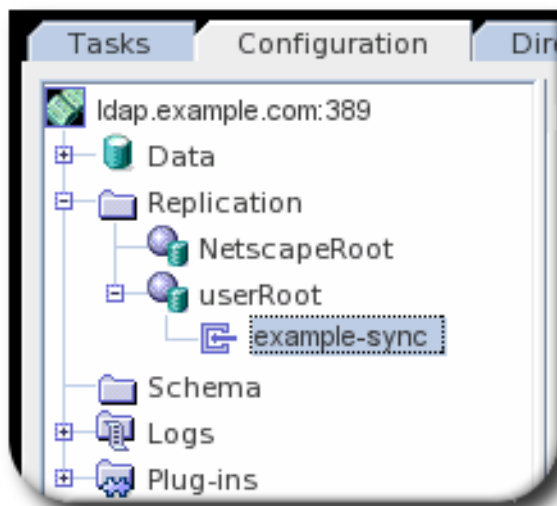
Many additional Windows and group attributes can be added to the entry. All of the schema which is synchronized is listed in [Section 12.5.2, “Group Attributes Synchronized between Directory Server and Active Directory”](#). Windows-specific attributes, belonging to the **ntGroup** object class, are described in more detail in the [Red Hat Directory Server 9 Configuration, Command, and File Reference](#).

## 12.5.5. Configuring Group Sync for Active Directory Groups

Synchronization for Windows users (users which originate in the Active Directory domain) is configured in the sync agreement.

### 12.5.5.1. Configuring Group Sync in the Console

1. Open the **Configuration** tab and expand the **Replication** folder.
2. Open the appropriate database, and select the sync agreement.



3. Open the **Connection** tab.
4. Check the **New Windows Group Sync** check box to enable group sync. To disable sync, uncheck the box.



For new sync agreements, select the corresponding group sync check box in the sync agreement creation wizard.

### 12.5.5.2. Configuring Group Sync in the Command Line

The attribute to set Active Directory group sync is ***nsds7NewWinGroupSyncEnabled*** and is set on the sync agreement. To enable group sync, add this attribute to the sync agreement or create a sync agreement with this attribute set to **on**:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=ExampleSyncAgreement,cn=userRoot,cn=dc=example\,dc=com,cn=mapping
tree,cn=config
changetype: modify
replace: nsds7NewWinGroupSyncEnabled
nsds7NewWinGroupSyncEnabled: on
```

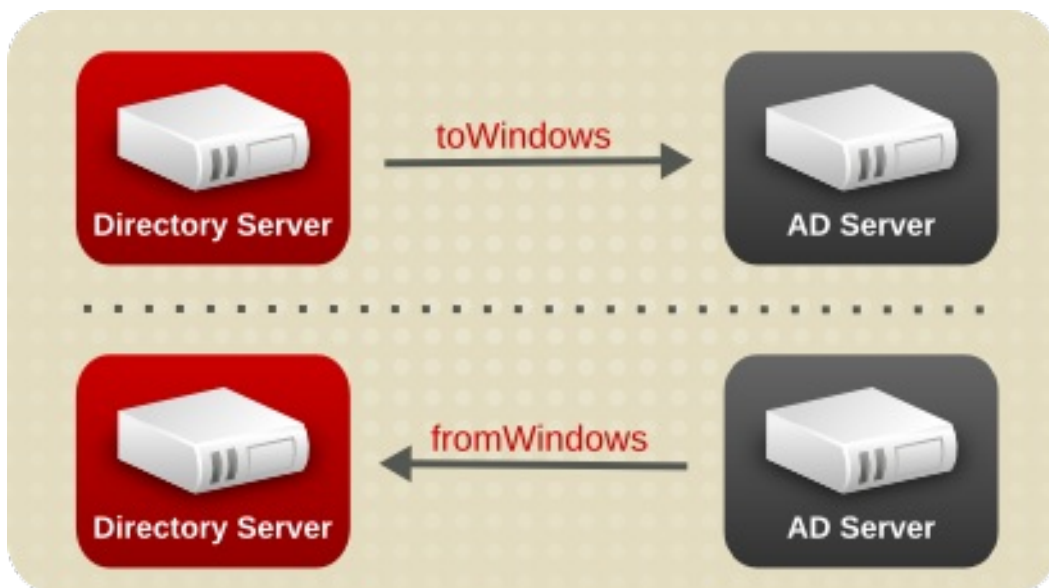
To disable group sync, set ***nsds7NewWinGroupSyncEnabled***: **off**.

## 12.6. CONFIGURING UNI-DIRECTIONAL SYNCHRONIZATION

As [Figure 12.1, “Active Directory — Directory Server Synchronization Process”](#) illustrates, synchronization is *bi-directional* by default. That means that changes in Active Directory are sent to Directory Server and changes on Directory Server are sent to Active Directory.

It is possible to create *uni-directional* synchronization, where changes are only sent one-way. This is similar to a master-consumer relationship<sup>[3]</sup> as opposed to multi-master.

An additional attribute for the sync agreement, ***oneWaySync***, enables uni-directional synchronization and specifies the direction to send changes. The possible values are ***fromWindows*** (for Active Directory to Directory Server sync) and ***toWindows*** (for Directory Server to Active Directory sync). If this attribute is absent, then synchronization is bi-directional.



**Figure 12.3. Uni-Directional Synchronization**

The synchronization process itself is the mostly same for bi-directional and uni-directional synchronization. It uses the same sync interval and configuration. The only difference is in how sync information is requested.

For Windows-only sync, during the regular synchronization update interval, the Directory Server contacts the Active Directory server and sends the DirSync control to request updates. However, the Directory

Server does not send any changes or entries from its side. So, the sync update consists of the Active Directory changes being sent to and updating the Directory Server entries.

For Directory Server only sync, the Directory Server sends entry modifications to the Active Directory server in a normal update, but it does not include the DirSync control so that it does not request any updates from the Active Directory side.

To enable uni-directional sync:

1. Create the synchronization agreement, as in [Section 12.3.7, “Step 7: Create the Synchronization Agreement”](#).
2. There is no option in the Directory Server Console to set uni-directional sync when the agreement is initially created. Edit the sync agreement to contain the ***oneWaySync*** attribute.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=ExampleSyncAgreement,cn=sync
replica,cn=dc=example\,dc=com,cn=mapping tree,cn=config
changetype: modify
add: oneWaySync
oneWaySync: fromWindows
```

## NOTE

Enabling uni-directional sync does *not* automatically prevent changes on the unsynchronized server, and this can lead to inconsistencies between the sync peers between sync updates. For example, uni-directional sync is configured to go from Active Directory to Directory Server, so Active Directory is (in essence) the data master. If an entry is modified or even deleted on the Directory Server, then the Directory Server information is different than the information and those changes are never carried over to Active Directory. During the next sync update, the edits are overwritten on the Directory Server and the deleted entry is re-added.

To prevent data inconsistency, use access control rules to prevent editing or deleting entries within the synchronized subtree on the *unsynced* server. Access controls for Directory Server are covered in [Section 13.3, “Creating ACLs Manually”](#). For Active Directory, see the appropriate Windows documentation.

## 12.7. SYNCHRONIZING POSIX ATTRIBUTES FOR USERS AND GROUPS

A subset of all possible user and attributes are synchronized between Active Directory and Red Hat Directory Server. Some attributes are mapped, where there are differences between Active Directory and Directory Server schemas, and some attributes are matched directly. The attributes (matched and mapped) which are synchronized are listed in [Section 12.4.1, “User Attributes Synchronized between Directory Server and Active Directory”](#) and [Section 12.5.2, “Group Attributes Synchronized between Directory Server and Active Directory”](#).

By default, only those attributes are synchronized.

One type of attribute that is missing from that sync list is any POSIX-related attribute. On Linux systems, system users and groups are identified as POSIX entries, and LDAP POSIX attributes contain that



required information. However, when Windows users are synced over, they have **ntUser** and **ntGroup** attributes automatically added which identify them as Windows accounts, but no POSIX attributes are synced over (even if they exist on the Active Directory entry) and no POSIX attributes are added on the Directory Server side.

The Posix Winsync API Plug-in synchronizes POSIX attributes between Active Directory and Directory Server entries.



## NOTE

All POSIX attributes (such as **uidNumber**, **gidNumber**, and **homeDirectory**) are synchronized between Active Directory and Directory Server entries. However, if a new POSIX entry or POSIX attributes are added to an existing entry in the Directory Server, *only the POSIX attributes are synchronized over to the Active Directory corresponding entry*. The POSIX object class (**posixAccount** for users and **posixGroup** for groups) is not added to the Active Directory entry.

### 12.7.1. Enabling POSIX Attribute Sync

The Posix Winsync API Plug-in is disabled by default and must be enabled for POSIX attributes to be synchronized from Active Directory user and group entries to the corresponding Directory Server entries.

1. Set the **nsslapd-pluginEnabled** attribute to **on**.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=Posix Winsync API,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
```



## NOTE

The precedence must be below 50 so that the Posix sync plug-in is loaded first. In the default configuration, the precedence is 25, and this value can remain the same in most deployments.

2. Restart the Directory Server to load the new configuration.

```
service dirsrv restart instance_name
```

### 12.7.2. Changing Posix Group Attribute Sync Settings

There are multiple plug-in attributes that can be set to control how the POSIX group attributes and group members are synced from the Active Directory entry to the corresponding Directory Server group and user entries. For details, see the corresponding section in the [Red Hat Directory Server Configuration, Command, and File Reference](#).

The defaults can be used for most deployments, but the settings can be changed depending on the Active Directory environment. For example, to enable nested group mappings:

1. Use **ldapmodify** to change the attribute to the appropriate setting:



```
ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=Posix Winsync API,cn=plugins,cn=config
changetype: modify
replace: posixWinsyncMapNestedGrouping
posixWinsyncMapNestedGrouping: true
```

2. Restart the Directory Server to load the new configuration.

### 12.7.3. Changing to Older Versions of Windows Posix Attributes

By default, the Posix Winsync API Plug-in uses Posix schema for modern Active Directory servers: 2005, 2008, and later versions. There are slight differences between the modern Active Directory Posix schema and the Posix schema used by Windows Server 2003 and older Windows servers. If an Active Directory domain is using the older-style schema, then the Posix Winsync API Plug-in can be configured to use the older Microsoft System Services for Unix 3.0 (msSFU30) schema.

To switch to Windows 2003-style Posix schema:

1. Use **ldapmodify** to change the ***posixWinsyncMsSFUSchema*** attribute to **true**.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=Posix Winsync API,cn=plugins,cn=config
changetype: modify
replace: posixWinsyncMsSFUSchema
posixWinsyncMsSFUSchema: true
```

2. Restart the Directory Server to load the new configuration.

```
service dirsrv restart instance_name
```

## 12.8. DELETING AND RESURRECTING ENTRIES

This section describes how enabling synchronization affects deleted entries on the sync peers and how resurrected entries are handled.

### 12.8.1. Deleting Entries

All changes on an Active Directory peers are always synced back to the Directory Server. This means that when an Active Directory group or user account is deleted on the Active Directory domain, the deletion is automatically synced back to the Directory Server sync peer server.

On Directory Server, on the other hand, when a Directory Server account is deleted, the corresponding entry on Active Directory is only deleted if the Directory Server entry has the ***ntUserDeleteAccount*** or ***ntGroupDeleteGroup*** attribute set to **true**.



## NOTE

When a Directory Server entry is synchronized over to Active Directory for the first time, Active Directory automatically assigns it a unique ID. At the next synchronization interval, the unique ID is synchronized back to the Directory Server entry and stored as the **ntUniqueId** attribute. If the Directory Server entry is deleted on Active Directory *before* the unique ID is synchronized back to Directory Server, the entry *will not* be deleted on Directory Server. Directory Server uses the **ntUniqueId** attribute to identify and synchronize changes made on Active Directory to the corresponding Directory Server entry; without that attribute, Directory Server will not recognize the deletion.

To delete the entry on Active Directory and then synchronize the deletion over to Directory Server, wait the length of the **winSyncInterval** (by default, five minutes) after the entry is created before deleting it so that the **ntUniqueId** attribute is synchronized.

### 12.8.2. Resurrecting Entries

It is possible to add deleted entries back in Directory Server; the deleted entries are called *tombstone* entries. When a deleted entry which was synced between Directory Server and Active Directory is re-added to Directory Server, the resurrected Directory Server entry has all of its original attributes and values. This is called *tombstone reanimation*. The resurrected entry includes the original **ntUniqueId** attribute which was used to synchronize the entries, which signals to the Active Directory server that this new entry is a tombstone entry.

Active Directory resurrects the old entry and preserves the original unique ID for the entry.

For Active Directory entries, when the tombstone entry is resurrected on Directory Server, all of the attributes of the original Directory Server are retained and are still included in the resurrected Active Directory entry.

## 12.9. SENDING SYNCHRONIZATION UPDATES

Synchronization occurs as frequently as is set in the **winSyncInterval** setting (for retrieving changes from the Active Directory domain) or **nsds5replicaupdateschedule** setting (for pushing changes from the Directory Server). By default, changes are retrieved from Active Directory every five minutes, and changes from the Directory Server are sent immediately.

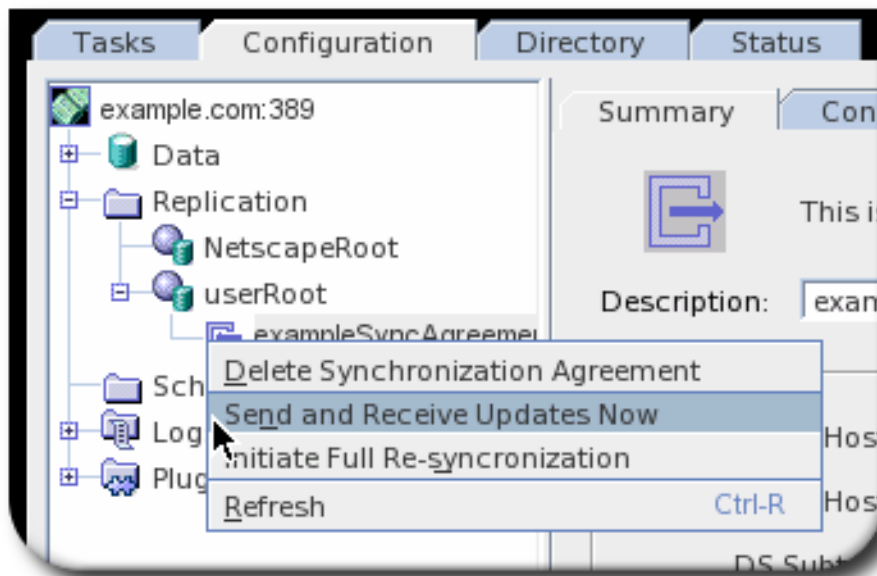
A sync update can be triggered manually. It is also possible to do a full resynchronization, which sends and pulls every entry in the Directory Server and Active Directory as if it were new. A full resynchronization includes existing Directory Server entries which may not have previously been synchronized.

### 12.9.1. Performing a Manual Sync Update

During normal operations, all the updates made to entries in the Directory Server that need to be sent to Active Directory are collected the changelog and then replayed during an incremental update.

1. Go to the **Configuration** tab in the Console.
2. Open the **Replication** folder and expand the appropriate database.
3. Select the sync agreement.
4. Right-click on the agreement or open the **Object** menu.

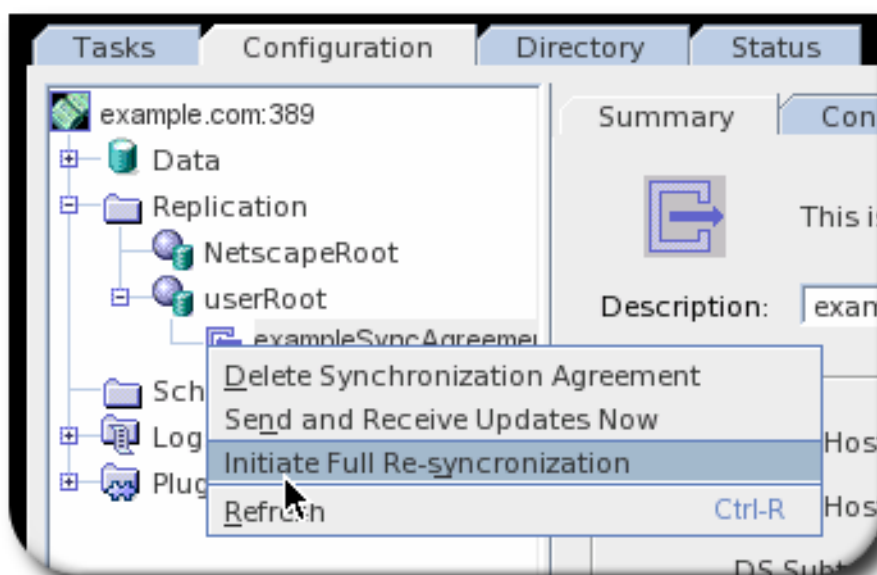
5. Select **Send and Receive Updates** from the drop down menu.



### 12.9.2. Sending a Total Update (Full Synchronization)

If there have been major changes to data, or synchronization attributes are added to pre-existing Directory Server entries, it is necessary to initiate a *resynchronization*. Resynchronization is a total update; the entire contents of synchronized subtrees are examined and, if necessary, updated. Resynchronization is done without using the changelog. This is similar to initializing or reinitializing a consumer in replication.

1. Go to the **Configuration** tab in the Console.
2. Open the **Replication** folder and expand the appropriate database.
3. Select the sync agreement.
4. Right-click on the agreement or open the **Object** menu.
5. Select **Initialize Full Re-synchronization** from the drop down menu.



Resynchronizing will not delete data on the sync peer; it sends and receives all updates and add any new or modified Directory Server entries; for example, it adds a pre-existing Directory Server user that had the **ntUser** object class added.

### 12.9.3. Sending Sync Updates in the Command Line

To send sync updates through the command line, add the ***nsDS5BeginReplicaRefresh*** attribute to the sync agreement. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x

dn: cn=ExampleSyncAgreement,cn=sync
replica,cn=dc=example\,dc=com,cn=mapping tree,cn=config
changetype: modify
add: nsDS5BeginReplicaRefresh
nsDS5BeginReplicaRefresh: start
```

This attribute is removed from the entry as soon as the update is complete.



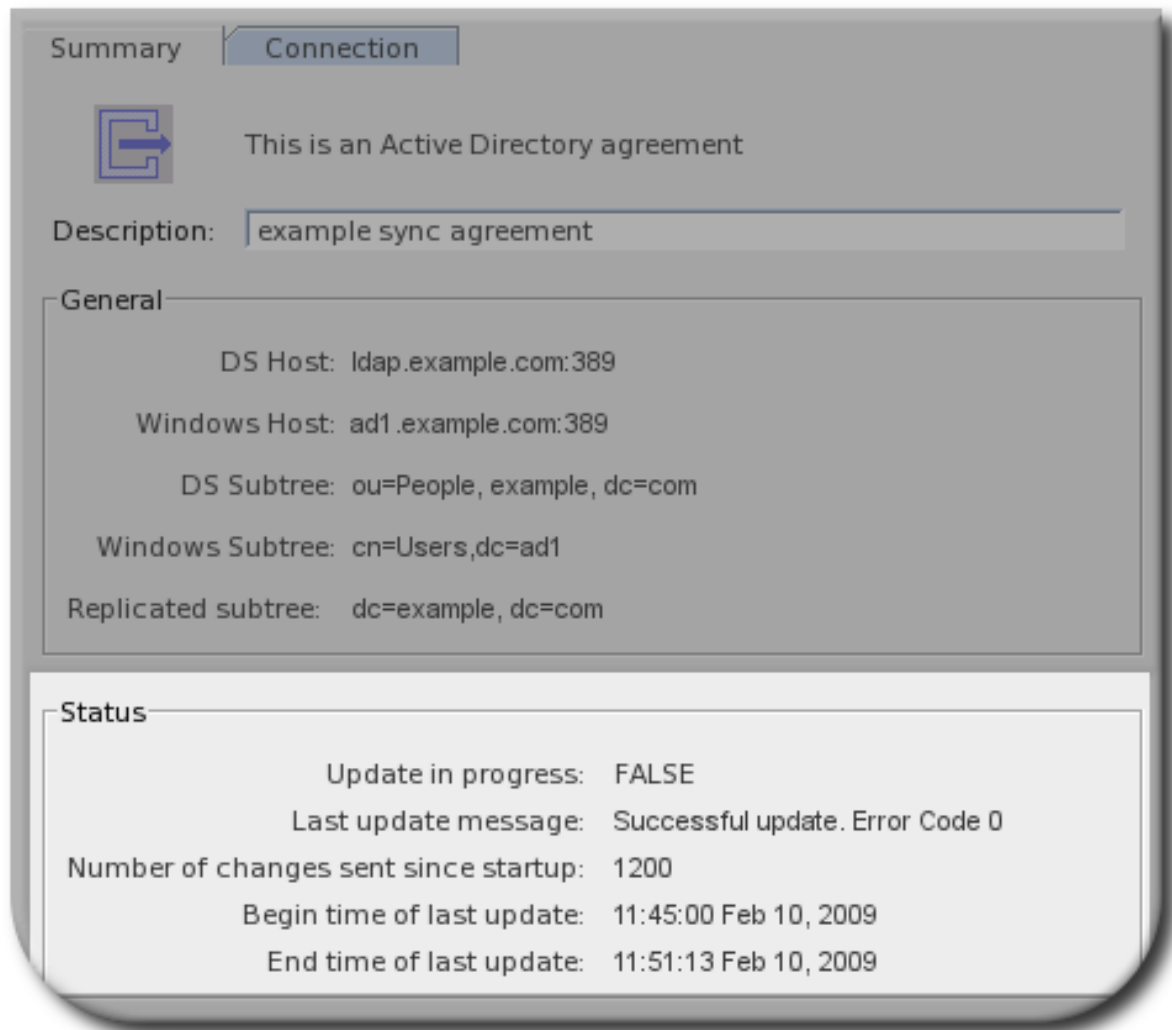
#### NOTE

This initiates a full synchronization for the entire database, not an incremental update of recent changes.

### 12.9.4. Checking Synchronization Status

Check synchronization status in the **Replication** tab in the **Status** of the Console. Highlight the synchronization agreement to monitor, and the relevant information should appear in the right-hand pane. The **Status** area shows whether the last incremental and total updates were successful and when they occurred.

1. Go to the **Configuration** tab in the Console.
2. Open the **Replication** folder and expand the appropriate database.
3. Select the sync agreement.
4. In the **Summary** tab, the status of the latest sync process is shown at the bottom.



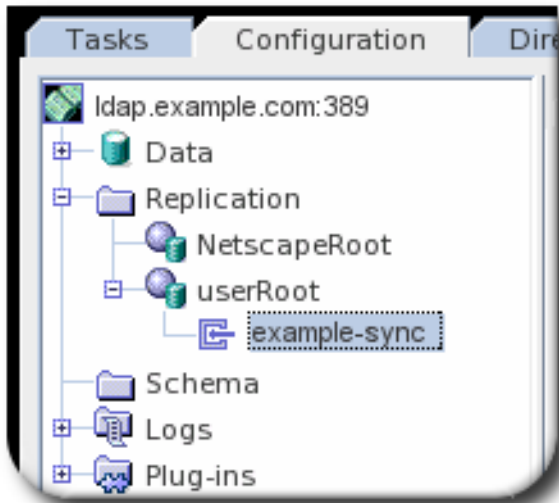
## 12.10. MODIFYING THE SYNC AGREEMENT

Certain attributes of the sync agreement can be modified, including the connection information. Using the command line, many additional parameters can be created with or added to the sync agreement, including changing the sync interval and setting a sync schedule.

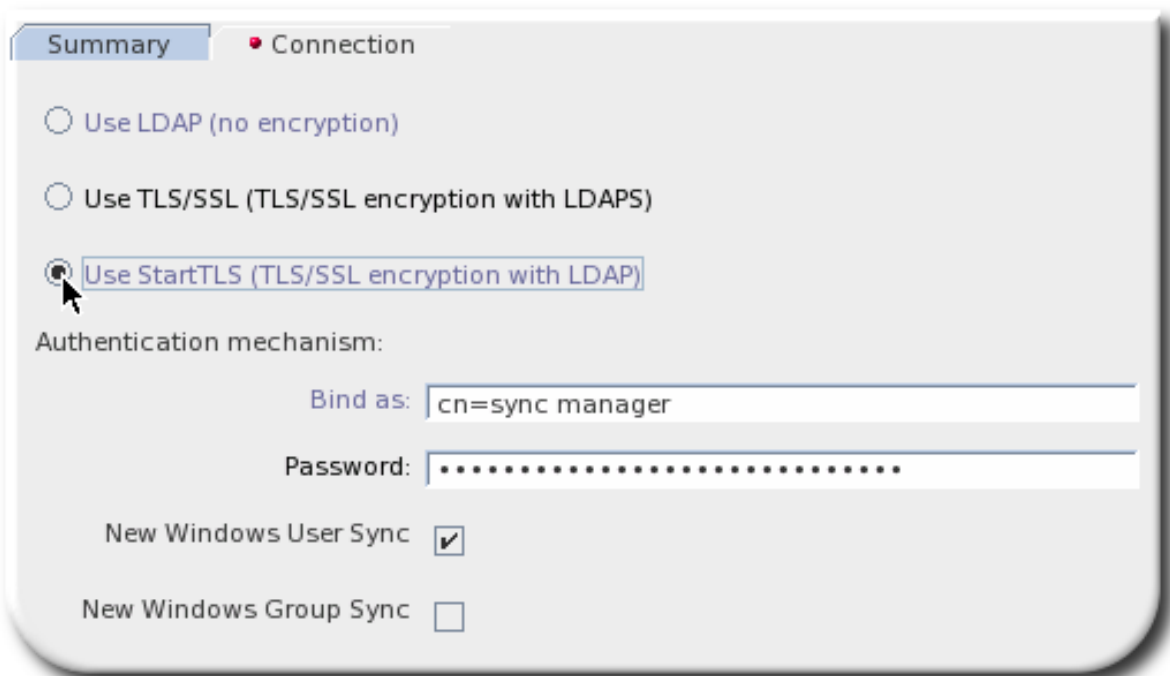
### 12.10.1. Editing the Sync Agreement in the Console

Most of the information which can be edited in the Console is limited to connection information, including the protocol to use and the bind credentials. It is also possible to edit the sync agreement description.

1. In the **Configuration** tab, expand the **Replication** folder.
2. Expand the database being synced. All of the synchronization agreements are listed below the database. Double-click the sync agreement to open it in the main window.



3. Click the **Connection** tab.



There are three areas of information that can be edited.

- The connection type (standard, TLS/SSL, and Start TLS).
- The bind user, both DN and password.
- Whether to sync new Directory Server users and new Directory Server groups automatically.

There are three options for the connection type — standard, TLS/SSL, and Start TLS — but there are really only two connection protocols, LDAP and LDAPS. Both a standard connection and Start TLS connection use LDAP (Start TLS creates a secure connection over an insecure port).

It is *not* possible to change the connection protocol because it is not possible to change the port number used to connect to the Windows sync peer.

It is possible to change the connection type between the standard connection and Start TLS, but

it is not possible to change from TLS/SSL to either the standard or Start TLS connections. Likewise, it is not possible to go from standard or Start TLS to TLS/SSL. If you need to change the connection protocol or the port number, delete the sync agreement and create a new one.

## 12.10.2. Adding and Editing the Sync Agreement in the Command Line

Creating or editing the sync agreement through the command line is more flexible and provides more options than using the Directory Server Console. The full list of sync agreement attributes are listed in [Section 12.10.2.5, “Sync Agreement Attributes”](#) and described in the *Configuration and Command-Line Tool Reference*.

### 12.10.2.1. Creating a Basic Sync Agreement

The most basic sync agreement defines the Directory Server database and the Active Directory sync peer:

- For the Directory Server database:
  - The synchronized subtree in the directory (***nsds7DirectoryReplicaSubtree***)
  - The Directory Server root DN (***nsDS5ReplicaRoot***)
  - The synchronized subtree in the directory (***nsds7DirectoryReplicaSubtree***)
- For the Active Directory domain:
  - The synchronized subtree in the Active Directory domain (***nsds7WindowsReplicaSubtree***)
  - The Active Directory domain name (***nsds7WindowsDomain***)

It also defines the connection information that the Directory Server uses to bind to the Active Directory domain:

- The Active Directory host name, IPv4 address, or IPv6 address (***nsDS5ReplicaHost***).
- The Active Directory port (***nsDS5ReplicaPort***).
- The type of connection (***nsDS5ReplicaTransportInfo***), which can be standard (***LDAP***), SSL (***SSL***), or Start TLS (***TLS***), which is a secure connection over a standard port.
- The user name (***nsDS5ReplicaBindDN***) and password (***nsDS5ReplicaCredentials***) for the Directory Server to use to bind to the Active Directory server.

For example:

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=ExampleSyncAgreement,cn=sync
replica,cn=dc=example\,dc=com,cn=mapping tree,cn=config
changetype: add
objectclass: top
objectclass: nsDSWindowsReplicationAgreement
cn: ExampleSyncAgreement
nsds7WindowsReplicaSubtree: cn=Users,dc=ad1
nsds7DirectoryReplicaSubtree: ou=People,dc=example,dc=com
```

```

nsds7WindowsDomain: ad1
nsDS5ReplicaRoot: dc=example,dc=com
nsDS5ReplicaHost: ad1.windows-server.com
nsDS5ReplicaPort: 389
nsDS5ReplicaBindDN: cn=sync user,cn=config
nsDS5ReplicaCredentials: {DES}ffGad646dT0nnsT8nJOaMA==
nsDS5ReplicaTransportInfo: TLS
nsds7NewWinUserSyncEnabled: on
nsds7NewWinGroupSyncEnabled: on

```

### 12.10.2.2. Setting Sync Schedules

Synchronization works two ways. The Directory Server sends its updates to Active Directory on a configurable schedule, similar to replication, using the ***nsds5replicaupdateschedule*** attribute. The Directory Server polls the Active Directory to check for changes; the frequency that it checks the Active Directory server is set in the ***winSyncInterval*** attribute.

By default, the Directory Server update schedule is to always be in sync. The Active Directory interval is to poll the Active Directory every five minutes.

To change the schedule the Directory Server uses to send its updates to the Active Directory, edit the ***nsds5replicaupdateschedule*** attribute. The schedule is set with start (*SSSS*) and end (*EEEE*) times in the form *HHMM*, using a 24-hour clock. The days to schedule sync updates are use ranging from **0** (Sunday) to **6** (Saturday).

```
nsds5replicaupdateschedule: SSSS EEEE DDDDDDD
```

For example, this schedules synchronization to run from noon to 2:00pm on Sunday, Tuesday, Thursday, and Saturday:

```
nsds5replicaupdateschedule: 1200 1400 0246
```



#### NOTE

The synchronization times cannot wrap around midnight, so the setting **2300 0100** is not valid.

To change how frequently the Directory Server checks the Active Directory for changes to Active Directory entries, reset the ***winSyncInterval*** attribute. This attribute is set in seconds, so the default of **300** means that the Directory Server polls the Active Directory server every 300 seconds, or five minutes. Setting this to a higher value can be useful if the directory searches are taking too long and affecting performance.

```
winSyncInterval: 1000
```

### 12.10.2.3. Changing Sync Connections

Two aspects of the connection for the sync agreement can be altered:

- The bind user name and password (***nsDS5ReplicaBindDN*** and ***nsDS5ReplicaCredentials***).



- The connection method (*nsDS5ReplicaTransportInfo*).

It is only possible to change the *nsDS5ReplicaTransportInfo* from **LDAP** to **TLS** and vice versa. It is not possible to change to or from **SSL** because it is not possible to change the port number, and switching between LDAP and LDAPS requires changing the port number.

For example:

```
nsDS5ReplicaBindDN: cn=sync user,cn=config
nsDS5ReplicaCredentials: {DES}ffGad646dT0nnsT8nJ0aMA==
nsDS5ReplicaTransportInfo: TLS
```



### WARNING

It is not possible to change the port number of the Active Directory sync peer. Therefore, it is also not possible to switch between standard/Start TLS connections and SSL connections, since that requires changing between standard and insecure ports.

To change to or from SSL/TLS, delete the sync agreement and add it again with the updated port number and new transport information.

#### 12.10.2.4. Handling Entries That Move Out of the Synced Subtree

The sync agreement defines what subtrees in both Active Directory and Directory Server are synchronized between each other. Entries *within* the scope (the subtree) are synchronized; other entries are ignored.

However, the synchronization process actually starts at the root DN to begin evaluating entries for synchronization. Entries are correlated based on the *samAccount* in the Active Directory and the *uid* attribute in Directory Server. The synchronization plug-in notes if an entry (based on the *samAccount/uid* relationship) is removed from the synced subtree either because it is deleted or moved. That is the signal to the synchronization plug-in that the entry is no longer to be synced.

The issue is that the sync process needs some configuration to determine how to handle that moved entry. There are three options: delete the corresponding entry, ignore the entry (the default), or unsync the entry.

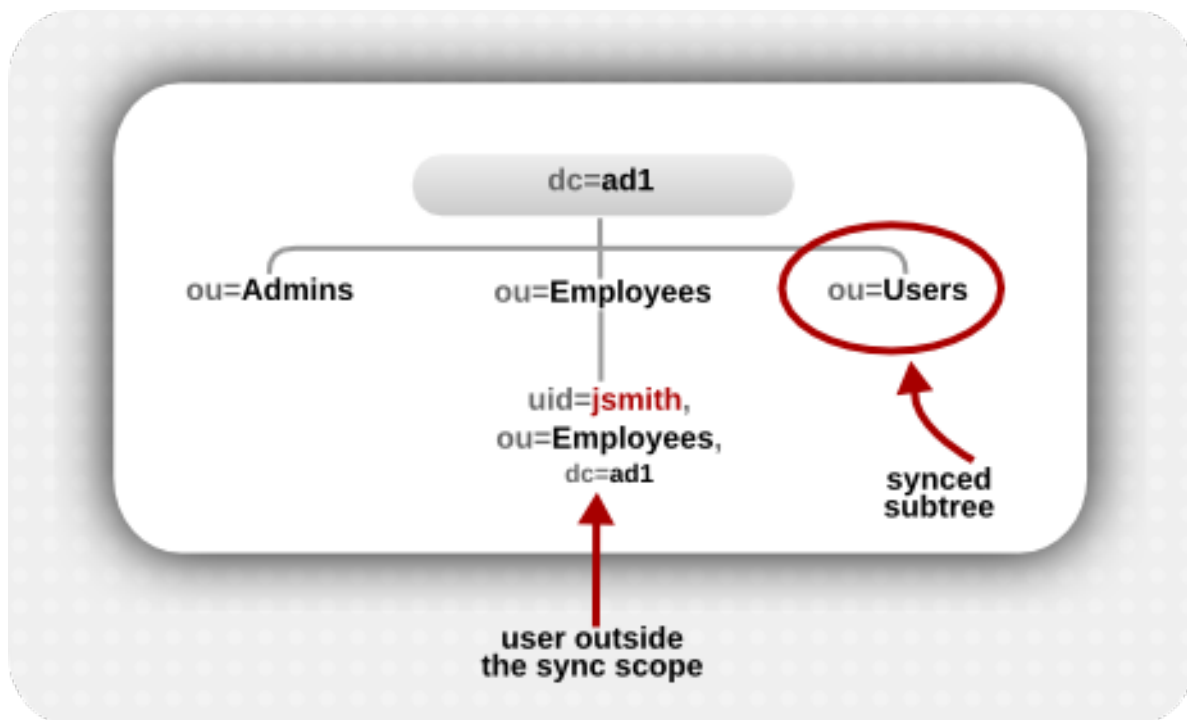


### NOTE

These sync actions only relate to how to handle *on the Directory Server side* when an entry is moved out of scope on the Active Directory side. This does not affect any Active Directory entry if an entry is moved out of the synced subtree on the Directory Server side.

**The default behavior in Directory Server 9.0 was to delete the corresponding Directory Server entry. This was true even if the entry on the Active Directory side was never synchronized over to the Directory Server side.** Starting in Directory Server 9.1, the default behavior is to ignore the entry and take no action.

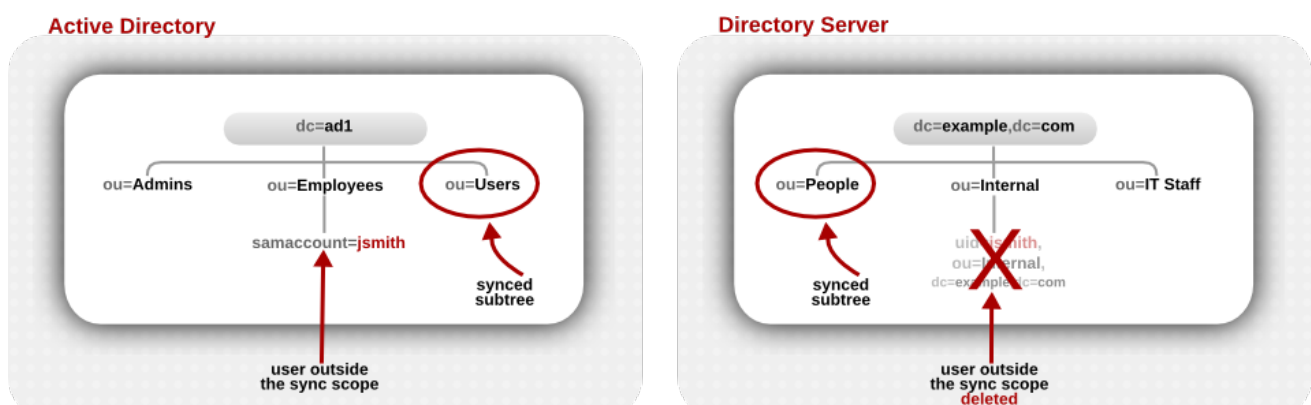
For example, a user with the **samAccount** ID of **jsmith** was created in the **ou=Employees** subtree on Active Directory. The synchronized subtree is **ou=Users**, so the **jsmith** user was never synchronized over to Directory Server.



**Figure 12.4. Active Directory Tree**

For 7.x and 8.x versions of Directory Server, synchronization simply ignored that user, since it was outside the synced subtree.

Starting in Directory Server 9.0, Directory Server began supporting subtree renames — which means that existing entries could be moved between branches of the directory tree. The synchronization plug-in, then, assumes that entries in the Active Directory tree which correspond to a Directory Server user (**samAccount/uid** relationship) but are outside the synced subtree are *intentionally* moved outside the synced subtree — essentially, a rename operation. The assumption then was that the "corresponding" Directory Server entry should be deleted.

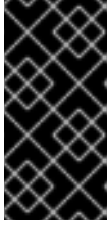


**Figure 12.5. Active Directory and Directory Server Trees Compared**

This assumption is not necessarily an accurate one, particularly for user entries which always existed outside the synced subtree.

The **winSyncMoveAction** attribute for the synchronization agreement sets instructions on how to handle these moved entries:

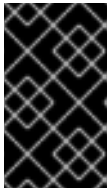
- **none** takes no action, so if a synced Directory Server entry exists, it may be synced over to or create an Active Directory entry *within* scope. If no synced Directory Server entry exists, nothing happens at all (this is the default behavior in Directory Server 9.1).
- **unsync** removes any sync-related attributes (*ntUser* or *ntGroup*) from the Directory Server entry but otherwise leaves the Directory Server entry intact.



### IMPORTANT

There is a risk when unsyncing entries that the Active Directory entry may be deleted at a later time, and the Directory Server entry will be left intact. This can create data inconsistency issues, especially if the Directory Server entry is ever used to recreate the entry on the Active Directory side later.

- **delete** deletes the corresponding entry on the Directory Server side, regardless of whether it was ever synced with Active Directory (this was the default behavior in 9.0).



### IMPORTANT

You almost never want to delete a Directory Server entry without deleting the corresponding Active Directory entry. This option is available only for compatibility with Directory Server 9.0 systems.

If it is necessary to change the default behavior from **none**, then edit the synchronization agreement to add the *winSyncMoveAction* attribute:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=ExampleSyncAgreement,cn=sync
replica,cn=dc=example\,dc=com,cn=mapping tree,cn=config
changetype: modify
add: winSyncMoveAction
winSyncMoveAction: unsync
```

#### 12.10.2.5. Sync Agreement Attributes

The common sync agreement attributes are listed in [Table 12.6, “Sync Agreement Attributes”](#). All of the possible sync agreement attributes are described in detail in the [Red Hat Directory Server 9 Configuration, Command, and File Reference](#).

**Table 12.6. Sync Agreement Attributes**

| Object Class or Attribute       | Description                                                               |
|---------------------------------|---------------------------------------------------------------------------|
| nsDSWindowsReplicationAgreement | An operational object class which contains the sync agreement attributes. |
| cn                              | Gives the name for the sync agreement.                                    |
| nsds7WindowsReplicaSubtree      | Specifies the Windows server suffix (root or sub) that is synced.         |

| Object Class or Attribute    | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| nsds7DirectoryReplicaSubtree | Specifies the Directory Server suffix (root or sub) that is synced.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| nsds7NewWinUserSyncEnabled   | Sets whether new Windows user accounts are automatically created on the Directory Server.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| nsds7NewWinGroupSyncEnabled  | Specifies whether new Windows group accounts are automatically created on the Directory Server.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| nsds7WindowsDomain           | Identifies the Windows domain being synchronized; analogous to <b><i>nsDS5ReplicaHost</i></b> in a replication agreement.                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| nsds5replicaport             | <p>Gives the LDAP port for the Windows server.</p> <p>To use TLS/SSL, give the secure port number (<b>636</b> by default) and set the <b><i>nsds5ReplicaTransportInfo</i></b> attribute to <b>SSL</b>.</p> <p>To use Start TLS, which initiates a secure connection over a standard port, use the standard port, <b>389</b>, with the <b><i>nsds5ReplicaTransportInfo</i></b> attribute to <b>TLS</b>.</p>                                                                                                                                                                                             |
| nsds5replicatransportinfo    | <p>To use TLS/SSL, set this parameter to <b>SSL</b>.</p> <p>To use Start TLS, which initiates a secure connection over a standard port, set this parameter to <b>TLS</b>.</p> <p>To use simple authentication, set this parameter to <b>LDAP</b>.</p>                                                                                                                                                                                                                                                                                                                                                  |
| nsds5ReplicaBindDN           | The sync user DN used by the Directory Server instance to bind to the Windows server.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| nsds5replicabindmethod       | <p>The connection type for replication between the servers. The connection type defines how the supplier authenticates to the consumer.</p> <p>Leaving the bind method empty or setting it to <b>SIMPLE</b> means that the server uses basic password-based authentication. This requires the <b><i>nsds5ReplicaBindDN</i></b> and <b><i>nsds5ReplicaCredentials</i></b> attributes to give the bind information.</p> <p>The <b>SSLCLIENTAUTH</b> option uses a secure connection. This requires setting the <b><i>nsds5ReplicaTransportInfo</i></b> attribute be set to <b>SSL</b> or <b>TLS</b>.</p> |

| Object Class or Attribute  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| nsDS5ReplicaCredentials    | <i>Only for simple authentication.</i> Stores the hashed password used with the bind DN given for simple authentication.                                                                                                                                                                                                                                                                                                                                                      |
| nsds5replicaroot           | Sets which Directory Server subtree is synchronized. Usually, it is recommended that the synchronized subtree be high in the directory tree so that the entire database is synchronized. For example:<br><br><div>dc=example, dc=com</div>                                                                                                                                                                                                                                    |
| description                | A text description of the replication agreement. Make this a useful description so it is easier to manage sync agreements.                                                                                                                                                                                                                                                                                                                                                    |
| nsds5replicaupdateschedule | Sets the start and end time for the replication updates and the days on which replication occurs in the form <i>start_time end_time days</i> . If the schedule is omitted, synchronization occurs all of the time.                                                                                                                                                                                                                                                            |
| oneWaySync                 | <i>Optional.</i> Configures synchronization updates to come from only one direction, either from Active Directory to Directory Server ( <b>fromWindows</b> ) or from Directory Server to Active Directory ( <b>toWindows</b> ). This attribute is not set by default, so synchronization is bi-directional.                                                                                                                                                                   |
| winSyncInterval            | <i>Optional.</i> Sets how frequently, in seconds, the Directory Server polls the Windows server for updates to write over. If this is not set, the default is <b>300</b> , which is 300 seconds or five (5) minutes.                                                                                                                                                                                                                                                          |
| winSyncMoveAction          | <i>Optional.</i> Sets how the sync plug-in handles corresponding entries that are discovered in Active Directory outside of the synced subtree. The sync process can ignore these entries (none, the default) or it can assume that the entries were moved intentionally to remove them from synchronization, and it can then either delete the corresponding Directory Server entry (delete) or remove the synchronization attributes and no longer sync the entry (unsync). |
| nsds5BeginReplicaRefresh   | <i>Optional.</i> Performs an online (immediate) initialization of the sync peer. If this is set, the attribute is only present as long as the sync peer is being updated initialized; when the initialization is complete, the attribute is deleted automatically. The only value when adding this attribute is <b>start</b> .                                                                                                                                                |

| Object Class or Attribute | Description |
|---------------------------|-------------|
|---------------------------|-------------|

## 12.11. MANAGING THE PASSWORD SYNC SERVICE



### IMPORTANT

**Password Sync must be installed on every domain controller in the Active Directory domain in order to synchronize Windows passwords.**

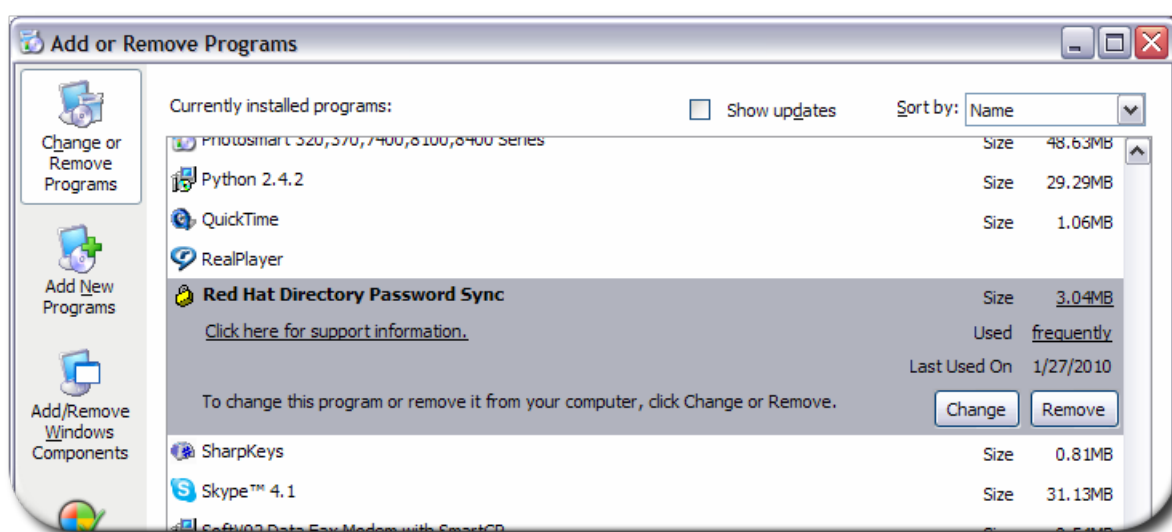
The service synchronizes password changes made on Active Directory with the corresponding entries' passwords on the Directory Server. Like any Windows service, it can be modified, started and stopped, and uninstalled, depending on how synchronization between Directory Server and Active Directory changes.

The Password Sync Service is supported on Microsoft Windows Server 2008 R2 (32-bit and 64-bit).

### 12.11.1. Modifying Password Sync

To reconfigure Password Sync:

1. Open **Control Panel**, and double-click **Add/Remove Programs**.
2. Click the **Change** button to relaunch the installer to change the settings.

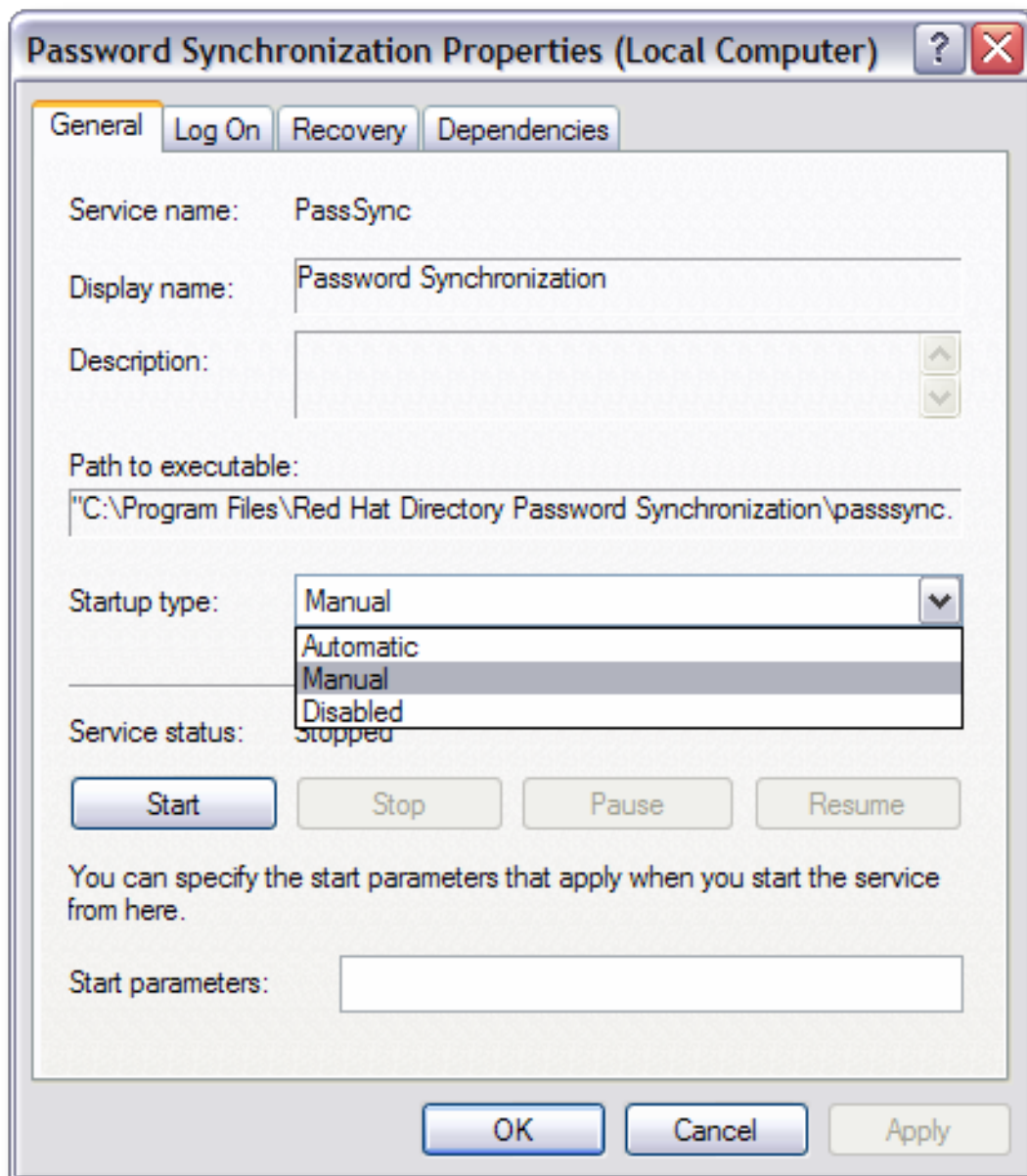


3. Go back through the configuration screens to make any changes to the configuration.

### 12.11.2. Starting and Stopping the Password Sync Service

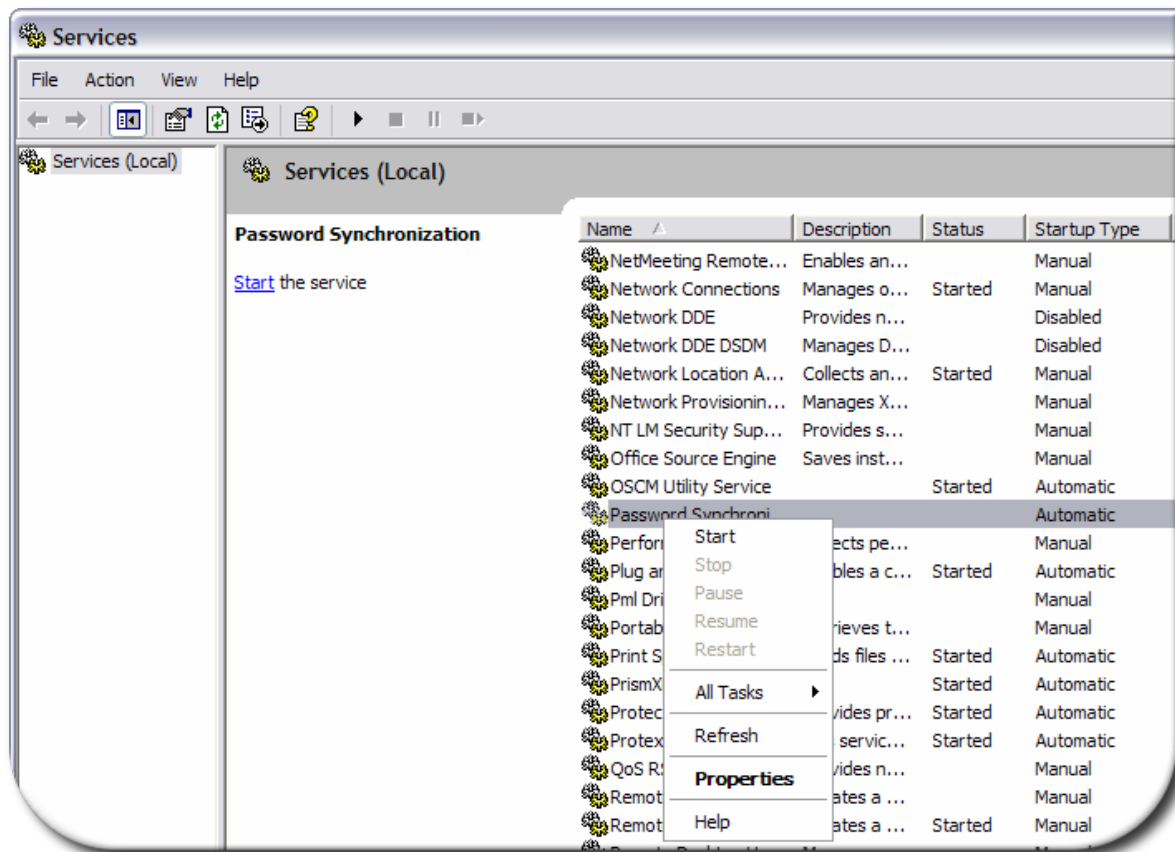
The Password Sync Service is configured to start whenever the Active Directory host is started. To reconfigure the service so that it does not start when Windows reboots:

1. Go to the **Control Panel**, and select **Services**.
2. Scroll through the list of services for the Password Sync Service. The **Startup** field is set to **Automatic**.
3. Double-click Password Sync.
4. Select the **Manual** radio button, and then click **OK**.



To start and stop Password Sync:

1. Go to the **Control Panel**, and select **Services**.
2. Scroll through the list of services for Password Sync, and right-click.
3. Select **Stop**, **Start**, or **Restart**, and hit okay.

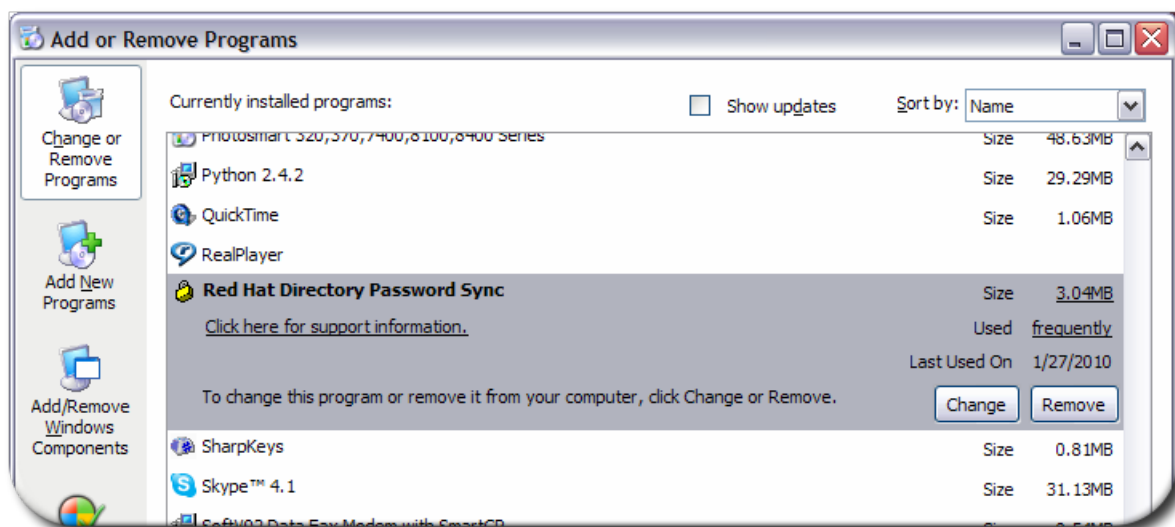


It's also possible to select the sync service and then click the start or stop links in the upper left of the **Services** window.

Changed passwords are captured even if Password Sync is not running. If Password Sync is restarted, the password changes are sent to Directory Server at the next synchronization.

### 12.11.3. Uninstalling Password Sync Service

1. Open **Control Panel**, and double-click **Add/Remove Programs**.
2. Select click **Remove** to uninstall the Password Sync Service.





3. If SSL was configured for the Password Sync, then the **cert8.db** and **key3.db** databases that were created were not removed when Password Sync was uninstalled. Delete these files by hand.

#### 12.11.4. Upgrading Password Sync

For details, see the corresponding section in the [Red Hat Directory Server Installation Guide](#).

## 12.12. TROUBLESHOOTING

If synchronization does not seem to function properly, see the Windows event log or Directory Server error log for information on any potential problems.

### Enable replication logging to record synchronization errors

Enable replication logging for more detailed information on synchronization to be recorded in the error logs. The replication log level produces more verbose logs from the sync code. Messages related to synchronization traffic (which is the same as replication traffic) can help in diagnosing problems.

1. In the Console, click the **Configuration** tab.
2. Select **Logs** from the navigation menu on the right, and open the error log.
3. Scroll down to error log level, and select **Replication** from the menu.
4. Hit save.

### Error #1: The message box when creating the sync agreement indicates that the it cannot connect to Active Directory.

Make sure that the directory suffixes, Windows domain and domain host, and the administrator DN and password are correct. Also verify that the port number used for LDAPS is correct. If all of the connection information is correct, make sure that Active Directory machine is running.

### Error #2: After synchronization, the status returns error 81.

One of the sync peer servers has not been properly configured for SSL communication. Examine the Directory Server access log file to see if the connection attempt was received by the Directory Server. There are also helpful messages in the Directory Server's error log file.

To narrow down the source of the misconfiguration, try to establish an LDAPS connection to the Directory Server. If this connection attempt fails, check all values (including the port number, host name or IPv4/IPv6 address, search base, and user credentials) to see if any of these are the problem. If all else fails, reconfigure the Directory Server with a new certificate.

If the LDAPS connection to the Directory Server is successful, it is likely that the misconfiguration is on Active Directory. Examine the Windows event log file for error messages.



### NOTE

A common problem is that the certificate authority was not configured as trusted when the Windows sync services certificate database was configured.

### Error #3: An entry is moved from one subtree on Active Directory to another subtree, but the user is not moved to the corresponding subtree on Directory Server.

This is a known issue with synchronizing modrdn operations on Active Directory with entries on Directory Server. To work around it, delete the entry on Active Directory and then add it anew to the new subtree. The deletion and the addition will be properly synchronized to the Directory Server peer.

---

[3] Unlike a consumer, changes can still be made on the un-synced server. Use ACLs to prevent editing or deleting entries on the un-synced server to maintain data integrity.

## CHAPTER 13. MANAGING ACCESS CONTROL

Red Hat Directory Server allows you to control access to your directory. This chapter describes the how to implement *access control*. To take full advantage of the power and flexibility of access control, while you are in the planning phase for your directory deployment, define an access control strategy as an integral part of your overall security policy.

### 13.1. ACCESS CONTROL PRINCIPLES

The mechanism which defines user access is called *access control*. When the server receives a request, it uses the authentication information provided by the user in the bind operation and the access control instructions (ACIs) defined in the server to allow or deny access to directory information. The server can allow or deny permissions for actions on entries like read, write, search, and compare. The permission level granted to a user may depend on the authentication information provided.

Access control in Directory Server is flexible enough to provide very precise rules on when the ACIs are applicable:

- For the entire directory, a subtree of the directory, specific entries in the directory (including entries defining configuration tasks), or a specific set of entry attributes.
- For a specific user, all users belonging to a specific group or role, or all users of the directory.
- For a specific location such as an IP address or a DNS name.

#### 13.1.1. ACI Structure

Access control instructions are stored in the directory as attributes of entries. The **aci** attribute is an operational attribute; it is available for use on every entry in the directory, regardless of whether it is defined for the object class of the entry. It is used by the Directory Server to evaluate what rights are granted or denied when it receives an LDAP request from a client. The **aci** attribute is returned in an **ldapsearch** operation if specifically requested.

The three main parts of an ACI statement are:

- Target
- Permission
- Bind Rule

The permission and bind rule portions of the ACI are set as a pair, also called an *access control rule* (ACR). The specified permission is granted or denied depending on whether the accompanying rule is evaluated to be true.

#### 13.1.2. ACI Placement

If an entry containing an ACI does not have any child entries, the ACI applies to that entry only. If the entry has child entries, the ACI applies to the entry itself and all entries below it. As a direct consequence, when the server evaluates access permissions to any given entry, it verifies the ACIs for every entry between the one requested and the directory suffix, as well as the ACIs on the entry itself.

The **aci** attribute is multi-valued, which means that you can define several ACIs for the same entry or subtree.

An ACI created on an entry can be set so it does not apply directly to that entry but to some or all of the entries in the subtree below it. The advantage of this is that general ACIs can be placed at a high level in the directory tree that effectively apply to entries more likely to be located lower in the tree. For example, an ACI that targets entries that include the **inetorgperson** object class can be created at the level of an **organizationalUnit** entry or a **locality** entry.

Minimize the number of ACIs in the directory tree by placing general rules at high level branch points. To limit the scope of more specific rules, place them as close as possible to leaf entries.

**NOTE**

ACIs placed in the root DSE entry apply only to that entry.

### 13.1.3. ACI Evaluation

To evaluate the access rights to a particular entry, the server compiles a list of the ACIs present on the entry itself and on the parent entries back up to the top level entry stored on the Directory Server. ACIs are evaluated across all of the databases for a particular Directory Server but not across all Directory Server instances.

The evaluation of this list of ACIs is done based on the semantics of the ACIs, not on their placement in the directory tree. This means that ACIs that are close to the root of the directory tree do not take precedence over ACIs that are closer to the leaves of the directory tree.

For Directory Server ACIs, the *precedence rule* is that ACIs that deny access take precedence over ACIs that allow access. Between ACIs that allow access, union semantics apply, so there is no precedence.

For example, if you deny write permission at the directory's root level, then none of the users can write to the directory, regardless of the specific permissions you grant them. To grant a specific user write permissions to the directory, you have to restrict the scope of the original denial for write permission so that it does not include the user.

### 13.1.4. ACI Limitations

When creating an access control policy for your directory service, you need to be aware of the following restrictions:

- If your directory tree is distributed over several servers using the chaining feature, some restrictions apply to the keywords you can use in access control statements:
  - ACIs that depend on group entries (**groupdn** keyword) must be located on the same server as the group entry. If the group is dynamic, then all members of the group must have an entry on the server, too. If the group is static, the members' entries can be located on remote servers.
  - ACIs that depend on role definitions (**roledn** keyword) must be located on the same server as the role definition entry. Every entry that is intended to have the role must also be located on the same server.

However, you can match values stored in the target entry with values stored in the entry of the bind user; for example, using the **userattr** keyword. Access is evaluated normally even if the bind user does not have an entry on the server that holds the ACI.

For more information on how to chain access control evaluation, see [Section 2.3.6, "Database Links and Access Control Evaluation"](#).

- Attributes generated by class of service (CoS) cannot be used in all ACI keywords. Specifically, you should not use attributes generated by CoS with the following keywords:
  - `targetfilter` ([Section 13.3.2.4, “Targeting Entries or Attributes Using LDAP Filters”](#))
  - `targetattrfilters` ([Section 13.3.2.2, “Targeting Attributes”](#))
  - `userattr` ([Section 13.4.5.1, “Using the userattr Keyword”](#))

If you create target filters or bind rules that depend on the value of attributes generated by CoS, the access control rule will not work. For more information on CoS, see [Chapter 6, \*Organizing and Grouping Entries\*](#).

- Access control rules are always evaluated on the local server. Therefore, it is not necessary to specify the host name or port number of the server in LDAP URLs used in ACI keywords. If you do, the LDAP URL is not taken into account at all. For more information on LDAP URLs, see [Appendix C, \*LDAP URLs\*](#).

## 13.2. DEFAULT ACIS

When the Admin Server is set up, the following default ACIs apply to the directory information stored in the **userRoot** database:

- Users can modify a list of common attributes in their own entries, including the ***mail***, ***telephoneNumber***, ***userPassword***, and ***seeAlso*** attributes. Operational and most of the security attributes, such as ***aci***, ***nsroledn***, and ***passwordExpirationTime***, cannot be modified by users.
- Users have anonymous access to the directory for search, compare, and read operations.
- The administrator (by default ***uid=admin,ou=Administrators,ou=TopologyManagement,o=NetscapeRoot***) has all rights except proxy rights.
- All members of the **Configuration Administrators** group have all rights except proxy rights.
- All members of the **Directory Administrators** group have all rights except proxy rights.
- **Server Instance Entry** (SIE) group.

The **NetscapeRoot** subtree has its own set of default ACIs:

- All members of the **Configuration Administrators** group have all rights on the **NetscapeRoot** subtree except proxy rights.
- Users have anonymous access to the **NetscapeRoot** subtree for search and read operations.
- All authenticated users have search, compare, and read rights to configuration attributes that identify the Admin Server.
- Group expansion.

The following sections explain how to modify these default settings.

## 13.3. CREATING ACIS MANUALLY

You can create access control instructions manually using LDIF statements and add them to the directory tree using the **ldapmodify** utility, similar to the instructions in [Section 3.3, “Using LDIF Update Statements to Create or Modify Entries”](#).



### NOTE

LDIF ACI statements can be very complex. However, if you are setting access control for a large number of directory entries, using LDIF is the preferred because it is faster than using the Console. To familiarize yourself with LDIF ACI statements, however, you may want to use the Directory Server Console to set the ACI and then click the **Edit Manually** button on the **Access Control Editor**. This shows you the correct LDIF syntax. If your operating system allows it, you can even copy the LDIF from the **Access Control Editor** and paste it into your LDIF file.

### 13.3.1. The ACI Syntax

The **aci** attribute uses the following syntax:

```
aci: (target)(version 3.0;acl "name";permissionbind_rules;)
```

- *target* specifies the entry, attributes, or set of entries and attributes for which to control access. The target can be a distinguished name, one or more attributes, or a single LDAP filter. The target is an optional part of the ACI.
- **version 3.0** is a required string that identifies the ACI version.
- *name* is a name for the ACI. The name can be any string that identifies the ACI. The ACI name is required.
- *permission* specifically outlines what rights are being allowed or denied; for example, read or search rights.
- *bind\_rules* specify the credentials and bind parameters that a user has to provide to be granted access. Bind rules can also specifically deny access to certain users or groups of users.

You can have multiple permission-bind rule pairs for each target. This allows you to set multiple access controls for a given target efficiently. For example:

```
target(permissionbind_rule)(permissionbind_rule)...
```

If you have several ACRs in one ACI statement, the syntax is in the following form:

```
aci: (target)(version 3.0;acl "name";permissionbind_rule;
permissionbind_rule; ... permissionbind_rule;)
```

The following is an example of a complete LDIF ACI:

```
aci: (target="ldap:///uid=bjensen,dc=example,dc=com")(targetattr=*)
(version 3.0;acl "aci1";allow (write) userdn="ldap:///self";)
```

In this example, the ACI states that the user **bjensen** has rights to modify all attributes in her own directory entry.

### 13.3.2. Defining Targets

The target identifies to what the ACI applies. If the target is not specified, the ACI applies to the entry containing the **aci** attribute and to the entries below it. A target can be any of the following:

- A directory entry or all of the entries in a subtree, as described in [Section 13.3.2.1, “Targeting a Directory Entry”](#).
- Attributes of an entry, as described in [Section 13.3.2.2, “Targeting Attributes”](#).
- A set of entries or attributes that match a specified LDAP filter, as described in [Section 13.3.2.4, “Targeting Entries or Attributes Using LDAP Filters”](#).
- An attribute value, or a combination of values, that match a specified LDAP filter, as described in [Section 13.3.2.5, “Targeting Attribute Values Using LDAP Filters”](#).

The general syntax for a target is as follows:

```
(keyword = "expression")
(keyword != "expression")
```

- *keyword* indicates the type of target.
- equal (=) indicates that the target is the object specified in the *expression*, and not equal (!=) indicates the target is not the object specified in the *expression*.
- *expression* identifies the target.

The quotation marks (") around *expression* are required. What you use for *expression* is dependent upon the *keyword* that you supply.

[Table 13.1, “LDIF Target Keywords”](#) lists each keyword and the associated expressions.

**Table 13.1. LDIF Target Keywords**

| Keyword                                               | Valid Expressions                                           | Wildcard Allowed |
|-------------------------------------------------------|-------------------------------------------------------------|------------------|
| target                                                | <code>ldap:///distinguished_name</code>                     | Yes              |
| targetattr                                            | <code>attribute</code>                                      | Yes              |
| targetfilter                                          | <code>LDAP_filter</code>                                    | Yes              |
| targetattrfilters                                     | <code>operation=attribute:LDAP_filter</code> <sup>[a]</sup> | Yes              |
| [a] Supported operations: <b>add</b> and <b>del</b> . |                                                             |                  |

In all cases, you must keep in mind that when you place an ACI on an entry, if it is not a leaf entry, the ACI also applies to all entries below it. For example, if you target the entry

**ou=accounting,dc=example,dc=com**, the permissions you set apply to all entries in the accounting branch of the **example** tree.

As a counter example, if you place an ACL on the **ou=accounting,dc=example,dc=com** entry, you cannot target the **uid=sarette,ou=people,dc=example,dc=com** entry because it is not located under the accounting tree.

Be wary of using **!=** when specifying an attribute to deny. ACLs are treated as a logical OR, which means that if you created two ACLs as shown below, the result allows all values of the target attribute.

```
acl1: (target=...)(targetattr!=a)(version 3.0; acl "name";allow (...)..
acl2: (target=...)(targetattr!=b)(version 3.0; acl "name";allow (...)..
```

The first ACL (**acl1**) allows **b** and the second ACL (**acl2**) allows **a**. The result of these two ACLs is the same as the one resulting from using an ACL of the following form:

```
acl3: (targetattr="*") allow (...) ...
```

In the second example, nothing is denied, which could give rise to security problems.

When you want to deny access to a particular attribute, use **deny** in the permissions clause rather than using **allow** with ( **targetattr != value** ). For example, usages such as these are recommended:

```
acl1: (target=...)(targetattr=a)(version 3.0; acl "name";deny (...)..
acl2: (target=...)(targetattr=b)(version 3.0; acl "name";deny (...)..
```

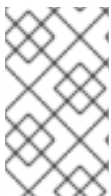
### 13.3.2.1. Targeting a Directory Entry

To target a directory entry (and the entries below it), you must use the **target** keyword. The **target** keyword can accept a value of the following format:

```
target="ldap:///distinguished_name
```

This identifies the distinguished name of the entry to which the access control rule applies. For example:

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



#### NOTE

If the DN of the entry to which the access control rule applies contains a comma, escape the comma with a single backslash (\), such as (**target="ldap:///uid=lfuentes,dc=example.com Bolivia\,S.A."**).

Wildcards can be used when targeting a distinguished name using the **target** keyword. The wildcard indicates that any character or string or substring is a match for the wildcard. Pattern matching is based on any other strings that have been specified with the wildcard.

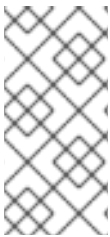
The following are legal examples of wildcard usage:

- (**target="ldap:///uid=\*,dc=example,dc=com"**) — Matches every entry in the entire **example** tree that has the **uid** attribute in the entry's RDN.



- **(target="ldap:///uid=\*Anderson,dc=example,dc=com")** — Matches every entry directly under the **example** node with a **uid** ending in Anderson.
- **(target="ldap:///uid=C\*A,dc=example,dc=com")** — Matches every entry directly under the **example** node with a **uid** beginning with C and ending with A.
- **(target="ldap:///uid=\*,dc=example,dc=com")** — Matches every entry in the entire **example** tree that has the **uid** attribute in the entry's RDN.
- **(target="ldap:///uid=\*,ou=\*,dc=example,dc=com")** — Matches every entry in the **example** tree whose distinguished name contains the **uid** and **ou** attributes. Thus, **uid=fchen,ou=Engineering,dc=example,dc=com** or **uid=claire,ou=Engineering,ou=people,dc=example,dc=com** would match, but **uid=bjensen,dc=example,dc=com ou=Engineering,dc=example,dc=com** would not.

Depending on the position of the wildcard, it can apply to the full DN, not only to attribute values. Therefore, the wildcard can be used as a substitute for portions of the DN. For example, **uid=andy\*,dc=example,dc=com** targets all the directory entries in the entire **example** tree with a matching uid attribute and not just the entries that are immediately below the **dc=example,dc=com** node. In other words, this target matches with longer expressions such as **uid=andy,ou=eng,dc=example,dc=com** or **uid=andy,ou=marketing,dc=example,dc=com**.



#### NOTE

You cannot use wildcards in the suffix part of a distinguished name. That is, if your directory uses the suffixes **c=US** and **c=GB**, then you cannot use **(target="ldap:///dc=example,c=\*")** as a target to reference both suffixes. Neither can you use a target such as **uid=bjensen,dc=\*.com**.

### 13.3.2.2. Targeting Attributes

In addition to targeting directory entries, you can also target one or more attributes included in the targeted entries. This is useful to deny or allow access to partial information about an entry. For example, you could allow access to only the common name, surname, and telephone number attributes of a given entry while denying access to sensitive information such as passwords.

You can specify that the target is equal (=) or is not equal (!=) to a specific attribute. The attributes you supply do not need to be defined in the schema. This absence of schema checking makes it possible to implement an access control policy when you set up your directory service for the first time, even if the ACLs you create do not apply to the current directory content.

To target attributes, use the **targetattr** keyword. The keyword uses the following syntax:

```
(targetattr = "attribute")
```

For example, to target the common name (**cn**) attribute:

```
(targetattr = "cn")
```

You can target multiple attributes by using the **targetattr** keyword with the following syntax:

```
(targetattr = "attribute1 || attribute2 || ...")
```

To target all attributes:

```
(targetattr = "*")
```



#### NOTE

If the **targetattr** keyword is not set, the rights granted by the ACI applies to no attributes.

The attributes specified in the **targetattr** keyword apply to the entry that the ACI is targeting and to all the entries below it. If you target the password attribute on the entry **uid=bjensen,ou=Marketing,dc=example,dc=com**, only the password attribute on the **bjensen** entry is affected by the ACI because it is a leaf entry.

If, however, you target the tree's branch point **ou=Marketing,dc=example,dc=com**, then all the entries beneath the branch point that can contain a password attribute are affected by the ACI.

### 13.3.2.3. Targeting Both an Entry and Attributes

By default, the entry targeted by an ACI containing a **targetattr** keyword is the entry on which the ACI is placed. That is, putting an ACI such as **aci: (targetattr = "uid")(access\_control\_rules;)** on the **ou=Marketing,dc=example,dc=com** entry means that the ACI applies to the entire **Marketing** subtree. However, you can also explicitly specify a target using the **target** keyword:

```
aci: (target="ldap:///ou=Marketing,dc=example,dc=com")(targetattr="uid")
(access_control_rules;)
```

The order in which you specify the **target** and the **targetattr** keywords is not important.

### 13.3.2.4. Targeting Entries or Attributes Using LDAP Filters

You can use LDAP filters to target a group of entries that match certain criteria. To do this, you must use the **targetfilter** keyword with an LDAP filter. The syntax of the **targetfilter** keyword is as follows:

```
(targetfilter = "LDAP_filter")
```

*LDAP\_filter* is a standard LDAP search filter. For more information on the syntax of LDAP search filters, see [Chapter 10, Finding Directory Entries](#).

For example, suppose that all entries in the accounting department include the attribute-value pair **ou=accounting**, and all entries in the engineering department include the attribute-value pair **ou=engineering** subtree. The following filter targets all the entries in the accounting and engineering branches of the directory tree:

```
(targetfilter = "(|(ou=accounting)(ou=engineering))")
```

This type of filter targets whole entries. You can associate the **targetfilter** and the **targetattr** keywords to create ACIs that apply to a subset of attributes in the targeted entries.

The following LDIF example allows members of the Engineering Admins group to modify the **departmentNumber** and **manager** attributes of all entries in the **Engineering** business category.

This example uses LDAP filtering to select all entries with **businessCategory** attributes set to **Engineering**:

```
dn: dc=example,dc=com
objectClass: top
objectClass: organization
aci: (targetattr="departmentNumber || manager")
 (targetfilter="(businessCategory=Engineering)")
 (version 3.0; acl "eng-admins-write"; allow (write)
 groupdn ="ldap:///cn=Engineering Admins,dc=example,dc=com";)
```



## NOTE

Although using LDAP filters can be useful when you are targeting entries and attributes that are spread across the directory, the results are sometimes unpredictable because filters do not directly name the object for which you are managing access. The set of entries targeted by a filtered ACL is likely to change as attributes are added or deleted. Therefore, if you use LDAP filters in ACLs, you should verify that they target the correct entries and attributes by using the same filter in an **ldapsearch** operation.

### 13.3.2.5. Targeting Attribute Values Using LDAP Filters

You can use access control to target specific attribute values. This means that you can grant or deny permissions on an attribute if that attribute's value meets the criteria defined in the ACL. An ACL that grants or denies access based on an attribute's value is called a value-based ACL.

For example, you might grant all users in your organization permission to modify the **nsroledn** attribute in their own entry. However, you would also want to ensure that they do not give themselves certain key roles, such as **Top Level Administrator**. LDAP filters are used to check that the conditions on attribute values are satisfied.

To create a value-based ACL, use the **targetattrfilters** keyword with the following syntax:

- For one operation with one attribute and filter combination:

```
(targetattrfilters="operation=attribute:filter")
```

- For one operation with multiple attribute and filter combinations:

```
(targetattrfilters="operation=attribute_1:filter_1 &&
attribute_2:filter_2 ... && attribute_m:filter_m")
```

- For two operations, each with multiple attribute and filter combinations:

```
(targetattrfilters="operation_1=attribute_1_1:filter_1_1 &&
attribute_1_2:filter_1_2 ... && attribute_1_m:filter_1_m ,
operation_2=attribute_2_1:filter_2_1 && attribute_2_2:filter_2_2 ...
& attribute_2_n:filter_2_n ")
```

In the previous syntax examples, you can set the operations either to **add** or **del**. The **attribute:filter** combination sets the filter and the attribute the filter is applied to.

When creating an entry, if a filter applies to an attribute in the new entry, then each instance of that attribute must satisfy the filter. When deleting an entry, if a filter applies to an attribute in the entry, then each instance of that attribute must also satisfy the filter.

When modifying an entry, if the operation adds an attribute, then the add filter that applies to that attribute must be satisfied; if the operation deletes an attribute, then the delete filter that applies to that attribute must be satisfied. If individual values of an attribute already present in the entry are replaced, then both the add and delete filters must be satisfied.

For example, consider the following attribute filter:

```
(targetattrfilters="add=nsroledn:(! (nsroledn=cn=superAdmin)) &&
telephoneNumber:(telephoneNumber=123*)")
```

This filter can be used to allow users to add any role (***nsroledn*** attribute) to their own entry, except the **superAdmin** role. It also allows users to add a telephone number with a 123 prefix.



#### NOTE

You cannot create value-based ACIs from the Directory Server Console.

### 13.3.2.6. Targeting a Single Directory Entry

Targeting a single directory entry is not straightforward because it goes against the design philosophy of the access control mechanism. However, it can be done in either of two ways:

- By creating a bind rule that matches user input in the bind request with an attribute value stored in the targeted entry. For more details, see [Section 13.4.5, “Defining Access Based on Value Matching”](#).
- By using the **targetattr** and **targetfilter** keywords.

You can use the **targetattr** keyword to specify an attribute that is only present in the entry you want to target, and not in any of the entries below your target. For example, if you want to target **ou=people, dc=example, dc=com**, and there are not any organizational units (**ou**) defined below that node, you could specify an ACI that contains **targetattr=ou**.

A safer method is to use the **targetfilter** keyword and to specify explicitly an attribute value that appears in the entry alone. For example, during the installation of the Directory Server, the following ACI is created:

```
aci: (targetattr="*)(targetfilter=(o=NetscapeRoot))(version 3.0;
 acl "Default anonymous access"; allow (read, search)
 userdn="ldap:///anyone";)
```

This ACI can apply only to the **o=NetscapeRoot** entry.

The risk associated with these method is that your directory tree might change in the future, and you would have to remember to modify this ACI.

### 13.3.3. Defining Permissions

Permissions specify the type of access you are allowing or denying. You can either allow or deny permission to perform specific operations in the directory. The various operations that can be assigned are known as *rights*.

There are two parts to setting permissions:

- Allowing or denying access
- Assigning rights

### 13.3.3.1. Allowing or Denying Access

You can either explicitly allow or deny access permissions to the directory tree.



#### NOTE

From the Directory Server Console, you cannot explicitly deny access, only grant permissions.

### 13.3.3.2. Assigning Rights

Rights detail the specific operations a user can perform on directory data. You can allow or deny all rights, or you can assign one or more of the following rights:

**Table 13.2. User Rights**

| Right  | Description                                                                                                                                                                                                            |
|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Read   | Indicates whether users can read directory data. This permission applies only to the search operation.                                                                                                                 |
| Write  | Indicates whether users can modify an entry by adding, modifying, or deleting <i>attributes</i> . This permission applies to the modify and modrdn operations.                                                         |
| Add    | Indicates whether users can create an <b>entry</b> . This permission applies only to the add operation.                                                                                                                |
| Delete | Indicates whether users can delete an <b>entry</b> . This permission applies only to the delete operation.                                                                                                             |
| Search | Indicates whether users can search for the directory data. Users must have Search and Read rights in order to view the data returned as part of a search result. This permission applies only to the search operation. |

| Right     | Description                                                                                                                                                                                                                                                                                                        |
|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Compare   | Indicates whether the users can compare data they supply with data stored in the directory. With compare rights, the directory returns a success or failure message in response to an inquiry, but the user cannot see the value of the entry or attribute. This permission applies only to the compare operation. |
| Selfwrite | Indicates whether users can add or delete their own DN from a group. This right is used only for group management.                                                                                                                                                                                                 |
| Proxy     | Indicates whether the specified DN can access the target with the rights of another entry.                                                                                                                                                                                                                         |
| All       | Indicates that the specified DN has all rights ( <b>read</b> , <b>write</b> , <b>search</b> , <b>delete</b> , <b>compare</b> , and <b>selfwrite</b> ) to the targeted entry, <i>excluding</i> proxy rights.                                                                                                        |

Rights are granted independently of one another. This means, for example, that a user who is granted add rights can create an entry but cannot delete it if delete rights have not been specifically granted. Therefore, when planning the access control policy for your directory, you must ensure that you grant rights in a way that makes sense for users. For example, it does not usually make sense to grant write permission without granting read and search permissions.



## NOTE

The proxy mechanism is very powerful and must be used sparingly. Proxy rights are granted within the scope of the ACL, and there is no way to restrict who an entry that has the proxy right can impersonate; that is, when you grant a user proxy rights, that user has the ability to proxy for any user under the target; there is no way to restrict the proxy rights to only certain users. For example, if an entity has proxy rights to the **dc=example, dc=com** tree, that entity can do anything. Make sure you set the proxy ACI at the lowest possible level of the DIT; see [Section 13.9.12, “Proxied Authorization ACI Example”](#).

### 13.3.3.3. Rights Required for LDAP Operations

This section describes the rights you need to grant to users depending on the type of LDAP operation you want to authorize them to perform.

- Adding an entry:
  - Grant add permission on the entry being added.
  - Grant write permission on the value of each attribute in the entry. This right is granted by default but could be restricted using the **targetattrfilters** keyword.
- Deleting an entry:

- Grant delete permission on the entry to be deleted.
- Grant write permission on the value of each attribute in the entry. This right is granted by default but could be restricted using the **targetattrfilters** keyword.
- Modifying an attribute in an entry:
  - Grant write permission on the attribute type.
  - Grant write permission on the value of each attribute type. This right is granted by default but could be restricted using the **targetattrfilters** keyword.
- Modifying the RDN of an entry:
  - Grant write permission on the entry.
  - Grant write permission on the attribute type used in the new RDN.
  - Grant write permission on the attribute type used in the old RDN, if you want to grant the right to delete the old RDN.
  - Grant write permission on the value of attribute type used in the new RDN. This right is granted by default but could be restricted using the **targetattrfilters** keyword.
- Comparing the value of an attribute:
  - Grant compare permission on the attribute type.
- Searching for entries:
  - Grant search permission on each attribute type used in the search filter.
  - Grant read permission on attribute types used in the entry.

The permissions granted on individual attributes or entries can affect a broad range of actions; for example, there are several different permissions users must have to search the directory like the following **ldapsearch** operation:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -b "uid=bkolics,dc=example,dc=com" objectclass=*
mail
```

The following ACL is used to determine whether user **bkolics** can be granted access:

```
aci: (targetattr = "mail")(version 3.0; acl "self access to
mail"; allow (read, search) userdn = "ldap:///self");
```

The search result list is empty because this ACL does not grant access to the **objectclass** attribute. If you want the search operation described above to be successful, modify the ACL to allow read and search access for the **mail** and **objectclass** attributes.

```
aci: (targetattr = "mail || objectclass")(version 3.0; acl "self
access to mail"; allow (read, search) userdn = "ldap:///self");
```

#### 13.3.3.4. Permissions Syntax

In an ACL statement, the syntax for permissions is **allow|deny** (*rights*). *rights* is a list of 1 to 8 comma-separated keywords enclosed within parentheses. Valid keywords are **read**, **write**, **add**, **delete**, **search**, **compare**, **selfwrite**, **proxy**, or **all**.

In the following example, read, search, and compare access is allowed, provided the bind rule is evaluated to be true:

```
aci: (target="ldap:///dc=example,dc=com") (version 3.0;acl "example";
 allow (read, search, compare) bind_rule;)
```

### 13.3.3.5. Access Control and the modrdn Operation

To explicitly deny **modrdn** rights using ACLs, target the relevant entries but omit the **targetattr** keyword. For example, to prevent the **cn=helpDeskGroup,ou=groups,dc=example,dc=com** group from renaming any entries in the set specified by the pattern **cn=\*,ou=people,dc=example,dc=com**, add the following ACL:

```
aci: (target="ldap:///cn=*,ou=people,dc=example,dc=com")
 (version 3.0; acl "Deny modrdn rights to the helpDeskGroup";
 deny(write)
groupdn="ldap:///cn=helpDeskGroup,ou=groups,dc=example,dc=com";)
```

## 13.4. BIND RULES

Depending on the ACLs defined for the directory, for certain operations, you need to *bind* to the directory. *Binding* means logging in or authenticating yourself to the directory by providing credentials (for example, a keytab for SASL or a client certificate for SSL). The credentials provided in the bind operation and the circumstances of the bind determine whether access to the directory is allowed or denied.

Every permission set in an ACL has a corresponding bind rule that details the required credentials and bind parameters.

Bind rules can be simple, such as stating that the person accessing the directory must belong to a specific group. Bind rules can also be more complex, such as requiring that a person must belong to a specific group, must log in from a machine with a specific IP address, and is restricted to access between 8 a.m. and 5 p.m.

Bind rules define who can access the directory, when, and from where by defining any of the following:

- Users, groups, and roles that are granted access.
- Locations from which an entity must bind.
- Times or days on which binding must occur.
- Types of authentication that must be in use during binding.

Additionally, bind rules can be complex constructions that combine these criteria by using Boolean operators. See [Section 13.4.11, “Using Boolean Bind Rules”](#) for more information.

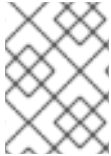
### 13.4.1. Bind Rule Syntax



Whether access is allowed or denied depends on whether an ACI's bind rule is evaluated to be true. Bind rules use one of the two following patterns:

```
keyword = "expression"; or
keyword != "expression";
```

Equal (=) indicates that *keyword* and *expression* must match in order for the bind rule to be true, and not equal (!=) indicates that *keyword* and *expression* must not match in order for the bind rule to be true.



## NOTE

The **timeofday** keyword also supports the inequality expressions (<, <=, >, >=). This is the only keyword that supports these expressions.

The quotation marks (") around *expression* and the delimiting semicolon (;) are required. The expressions you can use depend on the associated *keyword*.

Table 13.3, “LDIF Bind Rule Keywords” lists each keyword and the associated expressions and indicates whether wildcard characters are allowed in the expression.

**Table 13.3. LDIF Bind Rule Keywords**

| Keyword  | Valid Expressions                                                                                                                                                                  | Wildcard Allowed |
|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| userdn   | <div>ldap:///distinguished_name</div> <div>ldap:///all</div> <div>ldap:///anyone</div> <div>ldap:///self</div> <div>ldap:///parent</div> <div>ldap:///suffix??scope?(filter)</div> | Yes, in DN only  |
| groupdn  | <div>ldap:///DN  DN</div> <div>ldap:///suffix??scope?(filter)</div>                                                                                                                | No               |
| roledn   | ldap:///DN  DN                                                                                                                                                                     | No               |
| userattr | attribute#bindType<br>or attribute#value                                                                                                                                           | No               |
| ip       | IP_address                                                                                                                                                                         | Yes              |
| dns      | DNS_host_name                                                                                                                                                                      | Yes              |

| Keyword    | Valid Expressions                                                                      | Wildcard Allowed |
|------------|----------------------------------------------------------------------------------------|------------------|
| dayofweek  | sun mon tue wed thu fri sat                                                            | No               |
| timeofday  | 0 - 2359                                                                               | No               |
| authmethod | <div>none</div> <div>simple</div> <div>ssl</div> <div>sasl <i>sasl_mechanism</i></div> | No               |

### 13.4.2. Defining User Access - userdn Keyword

User access is defined using the **userdn** keyword. The **userdn** keyword requires one or more valid distinguished names in the following format:

```
userdn = "ldap:///dn [| ldap:///dn]...[| ldap:///dn]"
```

*dn* can be a DN or one of the expressions **anyone**, **all**, **self**, or **parent**:

```
userdn = "ldap:///anyone" Defines anonymous access
userdn = "ldap:///all" Defines general access
userdn = "ldap:///self" Defines self access
userdn = "ldap:///parent" Defines access for the parent entry
```

The **userdn** keyword can also be expressed as an LDAP filter:

```
ldap:///suffix??scope?(filter)
```



#### NOTE

If a DN contains a comma, the comma must be preceded by a backslash (\) escape character.

#### 13.4.2.1. Anonymous Access (anyone Keyword)

Granting anonymous access to the directory means that anyone can access it without providing a bind DN or password and regardless of the circumstances of the bind. You can limit anonymous access to specific types of access (for example, read or search access) or to specific subtrees or individual entries within the directory.

From the Directory Server Console, you define anonymous access through the **Access Control Editor**. See [Section 13.5, "Creating ACIs from the Console"](#).

#### 13.4.2.2. General Access (all Keyword)

You can use bind rules to indicate that a permission applies to anyone who has successfully bound to the directory; that is, all authenticated users. This allows general access while preventing anonymous access.

From the Directory Server Console, you define general access on the **Access Control Editor**. For more information, see [Section 13.5, “Creating ACLs from the Console”](#).

#### 13.4.2.3. Self Access (self Keyword)

Specifies that users are granted or denied access to their own entries. In this case, access is granted or denied if the bind DN matches the DN of the targeted entry.

Self access can be configured in the **Access Control Editor** in the Directory Server Console. For more information, see [Section 13.5, “Creating ACLs from the Console”](#).

#### 13.4.2.4. Parent Access (parent Keyword)

Specifies that users are granted or denied access to the entry only if their bind DN is the parent of the targeted entry.

You cannot set up parent access control using the Directory Server Console.

#### 13.4.2.5. LDAP URLs

You can dynamically target users in ACLs using a URL with an LDAP filter:

```
userdn = "ldap:///suffix??scope?(filter)"
```

For example, all users in the accounting and engineering branches of the **example** tree would be granted or denied access to the targeted resource dynamically based on the following URL:

```
userdn = "ldap:///dc=example,dc=com??sub?(|(ou=engineering)
(ou=accounting))"
```



#### NOTE

Do not specify a host name or port number within the LDAP URL. LDAP URLs always apply to the local server.

It is possible to string multiple LDAP URLs together so that the bind user must match both filter A *and* filter B. This is done by using multiple **userdn** keyword definitions. For example:

```
userdn="ldap:///dc=example,dc=com??sub?(ou=engineering)" and
userdn="ldap:///dc=example,dc=com??sub?
(manager="uid=bjensen,ou=managers,dc=example,dc=com")"
```

Using a connector such as **&&** is not allowed. For example, this is not an acceptable bind rule:

```
groupdn="ldap:///dc=example,dc=com??sub?(ou=engineering) &&
ldap:///dc=example,dc=com??sub?
(manager="uid=bjensen,ou=managers,dc=example,dc=com")"
```

For more information about LDAP URLs, see [Appendix C, LDAP URLs](#).

### 13.4.2.6. Wildcards

You can also specify a set of users by using the wildcard character (\*). For example, specifying a user DN of **uid=u\*,dc=example,dc=com** indicates that only users with a bind DN beginning with the letter **u** are allowed or denied access based on the permissions you set.

From the Directory Server Console, you set user access from the **Access Control Editor**. For more information, see [Section 13.5, “Creating ACIs from the Console”](#).

### 13.4.2.7. Examples

**Table 13.4. userdn Keyword Examples**

| Scenario                                          | Example                                                                                   | Description                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Userdn keyword containing an LDAP URL             | <pre>userdn = "ldap:///uid=*,dc=example,dc=com";</pre>                                    | <p>The bind rule is evaluated to be true if the user binds to the directory using any distinguished name of the specified pattern. For example, both of the following bind DN's would be evaluated to be true:</p> <div> <div>uid=ssarette,dc=example,dc=com</div> <div>uid=tjaz,ou=Accounting,dc=example,dc=com</div> </div> <p>This bind DN would be evaluated to be false:</p> <div>cn=Babs Jensen,dc=example,dc=com</div> |
| Userdn keyword containing logical OR of LDAP URLs | <pre>userdn="ldap:///uid=bj,dc=example,dc=com    ldap:///uid=kc,dc=example,dc=com";</pre> | The bind rule is evaluated to be true if the client binds as either of the two supplied distinguished names.                                                                                                                                                                                                                                                                                                                  |
| Userdn keyword excluding a specific LDAP URL      | <pre>userdn != "ldap:///uid=*,ou=Accounting,dc=example,dc=com";</pre>                     | The bind rule is evaluated to be true if the client is not binding as a UID-based distinguished name in the accounting subtree. This bind rule only makes sense if the targeted entry is not under the accounting branch of the directory tree.                                                                                                                                                                               |

| Scenario                                  | Example                  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------------------------------------|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Userdn keyword containing self keyword    | userdn = "ldap:///self"; | <p>The bind rule is evaluated to be true if the user is accessing the entry represented by the DN with which the user bound to the directory. That is, if the user has bound as <b>uid=ssarette, dc=example, dc=com</b> and the user is attempting an operation on the <b>uid=ssarette, dc=example, dc=com</b> entry, then the bind rule is true.</p> <p>If you want to grant all users in the <b>example</b> tree write access to their <b>userPassword</b> attribute, you would create the following ACI on the <b>dc=example, dc=com</b> node.</p> <pre>aci: (targetattr = "userPassword") (version 3.0; acl "write-self"; allow (write) userdn = "ldap:///self";)</pre> |
| Userdn keyword containing the all keyword | userdn = "ldap:///all";  | <p>The bind rule is evaluated to be true for any valid bind DN. To be true, a valid distinguished name must be presented by the user for a successful bind operation.</p> <p>For example, if you want to grant read access to the entire tree to all authenticated users, you would create the following ACI on the <b>dc=example, dc=com</b> node:</p> <pre>aci:(version 3.0; acl "all-read"; allow (read) userdn="ldap:///all";)</pre>                                                                                                                                                                                                                                    |

| Scenario                                     | Example                    | Description                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------------------------------|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Userdn keyword containing the anyone keyword | userdn = "ldap:///anyone"; | <p>The bind rule is evaluated to be true for anyone; use this keyword to provide anonymous access to your directory.</p> <p>For example, if you want to allow anonymous read and search access to the entire <b>example</b> tree, you would create the following ACL on the <b>dc=example, dc=com</b> node:</p> <pre>aci: (version 3.0; acl "anonymous-read-search"; allow (read,search) userdn = "ldap:///anyone";)</pre> |
| Userdn keyword containing the parent keyword | userdn = "ldap:///parent"; | <p>The bind rule is evaluated to be true if the bind DN is the parent of the targeted entry.</p> <p>For example, if you want to grant write access to every user's child entries, you would create the following ACL on the <b>dc=example, dc=com</b> node:</p> <pre>aci:(version 3.0; acl "parent access"; allow (write) userdn="ldap:///parent";)</pre>                                                                  |

### 13.4.3. Defining Group Access - groupdn Keyword

Members of a specific group can access a targeted resource. This is known as *group access*. Group access is defined using the **groupdn** keyword to specify that access to a targeted entry is granted or denied if the user binds using a DN that belongs to a specific group.

Group membership can be determined based on the user's DN or by using an LDAP filter to search for group members.

The **groupdn** keyword requires one or more valid distinguished names in the following format:

```
groupdn="ldap:///dn [| ldap:///dn]... [| ldap:///dn]"
```

The bind rule is evaluated to be true if the bind DN belongs to the named group.



#### NOTE

If a DN contains a comma, the comma must be escaped by a backslash (\).

The **groupdn** keyword can also be expressed with an LDAP filter:

```
groupdn="ldap:///suffix??scope?(filter)
```

With more complex **groupdn** syntax, the value of the **groupdn** expression is a single LDAP URL. Multiple **groupdns** can be grouped together within parentheses and use **or** or **and** connectors to define additional conditions on the group membership. For example:

```
(groupdn = "ldap:///ou=Groups,dc=example,dc=com??sub?(cn=*s_0)" or groupdn
= "ldap:///ou=Groups,dc=example,dc=com??sub?(cn=*s_1)") and groupdn =
"ldap:///ou=Groups,dc=example,dc=com??sub?(cn=*s_2)"
```

When stringing multiple **groupdn** URLs together, the keyword supports pipes to separate the URLs:

```
groupdn = "LDAPURI0 || LDAPURL1 || LDAPURL2"
```

However, it is not permissible to use ampersands (&), like **groupdn = "LDAPURI0 && LDAPURL1"**, or double quotes.

For example, to use two **groupdn** keywords so that the bind user must belong to both an Administrators group *and* a Managers group:

```
groupdn="ldap:///dc=example,dc=com??sub?(cn=*Administrators)" and
groupdn="ldap:///dc=example,dc=com??sub?(cn=*Managers)"
```

For more information about LDAP URLs, see [Appendix C, LDAP URLs](#).

The Directory Server Console defines specific groups through the **Access Control Editor**. For more information, see [Section 13.5, “Creating ACIs from the Console”](#).

**Table 13.5. groupdn Examples**

| Scenario                                             | Example                                                             | Description                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------------------------------------------------------|---------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Groupdn keyword containing an LDAP URL               | groupdn =<br>"ldap:///cn=Administrators,dc=example,dc=com";         | The bind rule is evaluated to be true if the bind DN belongs to the Administrators group. If you wanted to grant the Administrators group permission to write to the entire directory tree, you would create the following ACI on the <b>dc=example, dc=com</b> node:<br><br><pre>aci: (targetattr=*)(version 3.0; aci "Administrators-write"; allow (write) groupdn="ldap:///cn=Administrato rs,dc=example,dc=com");)</pre> |
| Groupdn keyword containing an LDAP URL with a filter | groupdn =<br>"ldap:///dc=example,dc=com??sub?(cn=*Administrators)"; | The bind rule is evaluated to be true if the bind DN belongs to any of the groups which are returned, meaning they match the filter.                                                                                                                                                                                                                                                                                         |

| Scenario                                           | Example                                                                                                        | Description                                                                                                                               |
|----------------------------------------------------|----------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| Groupdn keyword containing logical OR of LDAP URLs | groupdn = "ldap:///cn=Administrators,dc=example,dc=com"    "ldap:///cn=Mail Administrators,dc=example,dc=com"; | The bind rule is evaluated to be true if the bind DN belongs to either the <b>Administrators</b> or the <b>Mail Administrators</b> group. |

#### 13.4.4. Defining Role Access - roledn Keyword

Members of a specific role can access a targeted resource. This is known as *role access*. Role access is defined using the **roledn** keyword to specify that access to a targeted entry is granted or denied if the user binds using a DN that belongs to a specific role.

The **roledn** keyword requires one or more valid distinguished names in the following format:

```
roledn = "ldap:///dn [| ldap:///dn]... [| ldap:///dn]"
```

The bind rule is evaluated to be true if the bind DN belongs to the specified role.



#### NOTE

If a DN contains a comma, the comma must be escaped by a backslash (\).

The **roledn** keyword has the same syntax and is used in the same way as the **groupdn** keyword, with the exception of the LDAP filter, which is not implemented for role membership.

#### 13.4.5. Defining Access Based on Value Matching

You can set bind rules to specify that an attribute value of the entry used to bind to the directory must match an attribute value of the targeted entry.

For example, you can specify that the bind DN must match the DN in the **manager** attribute of a user entry in order for the ACI to apply. In this case, only the user's manager would have access to the entry.

This example is based on DN matching. However, you can match any attribute of the entry used in the bind with the targeted entry. For example, you could create an ACI that allowed any user whose **favoriteDrink** attribute is **beer** to read all the entries of other users that have the same value for **favoriteDrink**.

##### 13.4.5.1. Using the userattr Keyword

The **userattr** keyword can be used to specify which attribute values must match between the entry used to bind and the targeted entry. You can specify any of the following:

- A user DN
- A group DN
- A role DN
- An LDAP filter, in an LDAP URL



- Any attribute type

The LDIF syntax of the **userattr** keyword is as follows:

```
userattr = "attrName#bindType"
```

Using an attribute type that requires a value other than a user DN, group DN, role DN, or an LDAP filter has the following format:

```
userattr = "attrName#attrValue"
```

- *attrName* is the name of the attribute used for value matching.
- *bindType* is either **USERDN**, **GROUPDN**, or **LDAPURL**.
- *attrValue* is any string representing an attribute value.

#### 13.4.5.1.1. Example with USERDN Bind Type

The following associates the **userattr** keyword with a bind based on the user DN:

```
userattr = "manager#USERDN"
```

The bind rule is evaluated to be true if the bind DN matches the value of the **manager** attribute in the targeted entry. You can use this to allow a user's manager to modify employees' attributes. This mechanism only works if the **manager** attribute in the targeted entry is expressed as a full DN.

The following example grants a manager full access to his or her employees' entries:

```
aci: (target="ldap:///dc=example,dc=com")(targetattr=*)
 (version 3.0; acl "manager-write"; allow (all) userattr =
 "manager#USERDN";)
```

#### 13.4.5.1.2. Example with GROUPDN Bind Type

The following associates the **userattr** keyword with a bind based on a group DN:

```
userattr = "owner#GROUPDN"
```

The bind rule is evaluated to be true if the bind DN is a member of the group specified in the **owner** attribute of the targeted entry. For example, you can use this mechanism to allow a group to manage employees' status information. You can use an attribute other than **owner** as long as the attribute you use contains the DN of a group entry.

The group you point to can be a dynamic group, and the DN of the group can be under any suffix in the database. However, the evaluation of this type of ACI by the server is very resource intensive.

If you are using static groups that are under the same suffix as the targeted entry, you can use the following expression:

```
userattr = "ldap:///dc=example,dc=com?owner#GROUPDN"
```

In this example, the group entry is under the **dc=example, dc=com** suffix. The server can process this type of syntax more quickly than the previous example.

(By default, **owner** is not an allowed entry in a user's entry. You would have to extend your schema to allow this attribute in a **person** object.)

#### 13.4.5.1.3. Example with ROLEDN Bind Type

The following associates the **userattr** keyword with a bind based on a role DN:

```
userattr = "exampleEmployeeReportsTo#ROLEDN"
```

The bind rule is evaluated to be true if the bind DN belongs to the role specified in the **exampleEmployeeReportsTo** attribute of the targeted entry. For example, if you create a nested role for all managers in your company, you can use this mechanism to grant managers at all levels access to information about employees that are at a lower grade than themselves.



#### NOTE

This example assumes that you have added the **exampleEmployeeReportsToattribute** to the schema and that all employee entries contain this attribute. It also assumes that the value of this attribute is the DN of a role entry. For information on adding attributes to the schema, see [Section 8.4.2, "Creating Attributes"](#).

The DN of the role can be under any suffix in the database. If you are also using filtered roles, the evaluation of this type of ACI uses a lot of resources on the server.

If you are using a static role definition and the role entry is under the same suffix as the targeted entry, you can use the following expression:

```
userattr = "ldap:///dc=example,dc=com?employeeReportsTo#ROLEDN"
```

In this example, the role entry is under the **dc=example, dc=com** suffix. The server can process this type of syntax more quickly than the previous example.

#### 13.4.5.1.4. Example with LDAPURL Bind Type

The following associates the **userattr** keyword with a bind based on an LDAP filter:

```
userattr = "myfilter#LDAPURL"
```

The bind rule is evaluated to be true if the bind DN matches the filter specified in the *myfilter* attribute of the targeted entry. The *myfilter* attribute can be replaced by any attribute that contains an LDAP filter.

#### 13.4.5.1.5. Example with Any Attribute Value

The following associates the **userattr** keyword with a bind based on any attribute value:

```
userattr = "favoriteDrink#Beer"
```

The bind rule is evaluated to be true if the bind DN and the target DN include the ***favoriteDrink*** attribute with a value of **Beer**.

#### 13.4.5.1.6. Using the **userattr** Keyword with Inheritance

When you use the **userattr** keyword to associate the entry used to bind with the target entry, the ACI applies only to the target specified and not to the entries below it. In some circumstances, you might want to extend the application of the ACI several levels below the targeted entry. This is possible by using the **parent** keyword and specifying the number of levels below the target that should inherit the ACI.

When you use the **userattr** keyword in association with the **parent** keyword, the syntax is as follows:

```
userattr = "parent[inheritance_level].attrName#bindType"
```

Using an attribute type that requires a value other than a user DN, group DN, role DN, or an LDAP filter, the syntax is as follows:

```
userattr = "parent[inheritance_level].attrName#attrValue"
```

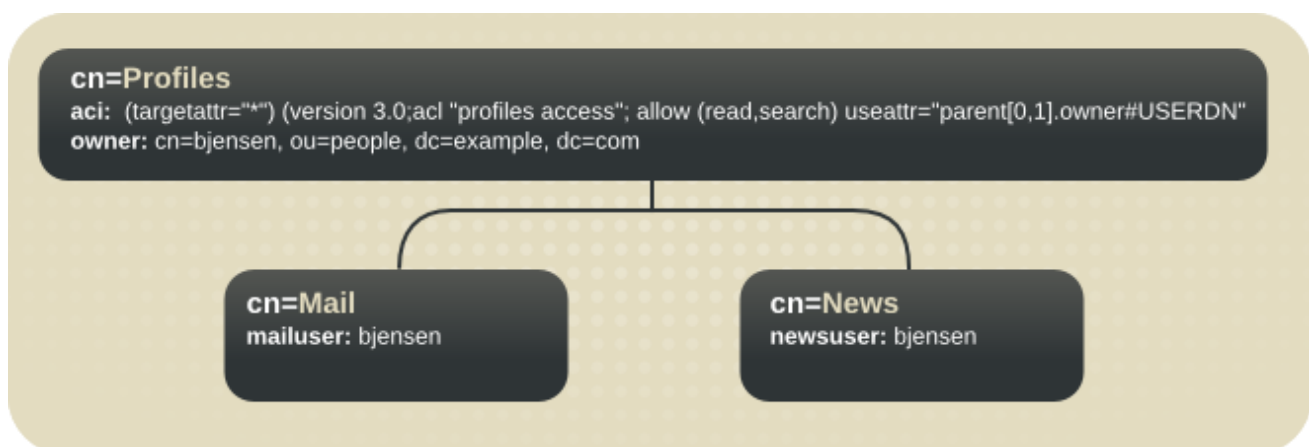
- *inheritance\_level* is a comma-separated list that indicates how many levels below the target inherits the ACI. You can include five levels (**0**, **1**, **2**, **3**, **4**) below the targeted entry; zero (**0**) indicates the targeted entry.
- *attribute* is the attribute targeted by the **userattr** or **groupattr** keyword.
- *bindType* can be one of **USERDN**, **GROUPDN**, or **LDAPURL**.

For example:

```
userattr = "parent[0,1].manager#USERDN"
```

This bind rule is evaluated to be true if the bind DN matches the manager attribute of the targeted entry. The permissions granted when the bind rule is evaluated to be true apply to the target entry *and* to all entries immediately below it.

The example in [Figure 13.1](#), “Using Inheritance With the **userattr** Keyword” indicates that user **bjensen** is allowed to read and search the **cn=Profiles** entry as well as the first level of child entries which includes **cn=mail** and **cn=news**, thus allowing her to search through her own mail and news IDs.



**Figure 13.1. Using Inheritance With the **userattr** Keyword**

In this example, if you did not use inheritance, you would have to do one of the following to achieve the same result:

- Explicitly set read and search access for user **bjensen** on the **cn=Profiles**, **cn=mail**, and **cn=news** entries in the directory.
- Add the owner attribute with a value of **bjensen** to the **cn=mail** and **cn=news** entries, and then add the following ACL to the **cn=mail** and **cn=news** entries.

```
aci: (targetattr="*") (version 3.0; acl "profiles access"; allow
(read,search)
 userattr="owner#USERDN";)
```

#### 13.4.5.1.7. Granting Add Permission Using the userattr Keyword

Using the **userattr** keyword in conjunction with **all** or **add** permissions does not behave as one would typically expect. Typically, when a new entry is created in the directory, Directory Server evaluates access rights on the entry being created and not on the parent entry. However, in the case of ACLs using the **userattr** keyword, this behavior could create a security hole, and the server's normal behavior is modified to avoid it.

Consider the following example:

```
aci: (target="ldap:///dc=example,dc=com")(targetattr=*) (version 3.0;
 acl "manager-write"; allow (all) userattr = "manager#USERDN";)
```

This ACL grants managers all rights on the entries of employees that report to them. However, because access rights are evaluated on the entry being created, this type of ACL would also allow any employee to create an entry in which the manager attribute is set to their own DN. For example, disgruntled employee Joe (**cn=Joe, ou=eng, dc=example, dc=com**) might want to create an entry in the Human Resources branch of the tree to use (or misuse) the privileges granted to Human Resources employees.

He could do this by creating the following entry:

```
dn: cn= Trojan Horse,ou=Human Resources,dc=example,dc=com
objectclass: top
...
cn: Trojan Horse
manager: cn=Joe,ou=eng,dc=example,dc=com
```

To avoid this type of security threat, the ACL evaluation process does not grant add permission at level 0, to the entry itself. You can, however, use the **parent** keyword to grant add rights below existing entries. You must specify the number of levels below the parent for add rights. For example, the following ACL allows child entries to be added to any entry in the **dc=example, dc=com** that has a **manager** attribute that matches the bind DN:

```
aci: (target="ldap:///dc=example,dc=com")(targetattr=*)
 (version 3.0; acl "parent-access"; allow (add)
 userattr = "parent[0,1].manager#USERDN";)
```

This ACL ensures that add permission is granted only to users whose bind DN matches the manager attribute of the parent entry.

### 13.4.6. Defining Access from a Specific IP Address



#### NOTE

Directory Server supports both IPv4 and IPv6 IP addresses.

Using bind rules, you can indicate that the bind operation must originate from a specific IP address. This is often used to force all directory updates to occur from a given machine or network domain.

The LDIF syntax for setting a bind rule based on an IP address is as follows:

```
ip = "IP_address" or ip != "IP_address"
```

The IP address must be expressed in dot notation. You can use the wildcard character (\*) to include multiple machines. For example, the following string is valid:

```
ip = "12.123.1.*";
```

The bind rule is evaluated to be true if the client accessing the directory is located at the named IP address. This can be useful for allowing certain kinds of directory access only from a specific subnet or machine.

For example, use a wildcard IP address such as **12.3.45.\*** to specify a specific subnetwork or **123.45.6.\*+255.255.255.115** to specify a subnetwork mask.

From the Directory Server Console, you can define specific machines to which the ACI applies through the **Access Control Editor**. For more information, see [Section 13.5, “Creating ACIs from the Console”](#).

### 13.4.7. Defining Access from a Specific Domain

A bind rule can specify that the bind operation must originate from a particular domain or host machine. This is often used to force all directory updates to occur from a given machine or network domain.

The LDIF syntax for setting a bind rule based on the DNS host name is as follows:

```
dns = "DNS_Hostname" or dns != "DNS_Hostname"
```



#### WARNING

The **dns** keyword requires that the naming service used on your machine is DNS. If the name service is not DNS, use the **ip** keyword instead.

The **dns** keyword requires a fully qualified DNS domain name. Granting access to a host without specifying the domain creates a potential security threat. For example, the following expression is allowed but not recommended:

```
dns = "legend.eng";
```

Instead, use a fully qualified name:

```
dns = "legend.eng.example.com";
```

The **dns** keyword allows wildcards. For example:

```
dns = "*.example.com";
```

The bind rule is evaluated to be true if the client accessing the directory is located in the named domain. This can be useful for allowing access only from a specific domain. Wildcards will not work if your system uses a naming service other than DNS. In such a case, if you want to restrict access to a particular domain, use the **ip** keyword, as described in [Section 13.4.6, “Defining Access from a Specific IP Address”](#).

### 13.4.8. Requiring a Certain Level of Security in Connections

A bind rule can specify that an operation must occur over a connection with a certain level of security. This forces operations — such as password change operations — to be performed over an SSL/TLS, Start TLS, or SASL connection. The security of the connection is determined by its *security strength factor*, which sets the minimum key strength required to process operations. The value for the SSF for any operation is the higher of the values between an SSL/TLS connection and a SASL bind; this means that if a server is configured to run over TLS and a replication agreement is configured for SASL/GSSAPI, the SSF for the operation is whichever available encryption type is more secure.

The SSF can be specified using any comparison pattern, like equals (=), less-than and greater-than, (< and >), is-not (!), less-than-or-equal-to (<=), or greater-than-or-equal-to (>=). The LDIF syntax for setting a bind rule based on the SSF is as follows:

```
ssf = "key_strength"
ssf >= "key_strength"
```

The **ssf** keyword accepts any positive whole number. If this is set to 0, then no secure connection is required for an operation.

The bind rule is evaluated to be true if the client accesses the directory using a connection of adequate strength.

### 13.4.9. Defining Access at a Specific Time of Day or Day of Week

You can use bind rules to specify that binding can only occur at a certain time of day or on a certain day of the week. For example, you can set a rule that allows access only if it is between the hours of 8 a.m. and 5 p.m. Monday through Friday. The time used to evaluate access rights is the time on the Directory Server, not the time on the client.

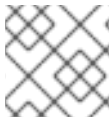
The LDIF syntax for setting a bind rule based on the time of day is as follows:

```
timeofday operator time
```

*operator* can be one of the following symbols:

|                               |
|-------------------------------|
| equal to (=)                  |
| not equal to (!=)             |
| greater than (>)              |
| greater than or equal to (>=) |
| less than (<)                 |
| less than or equal to (<=)    |

The **timeofday** keyword requires a time of day expressed in hours and minutes in the 24 hour clock (0 to 2359).



## NOTE

The time on the Directory Server is used for the evaluation, not the time on the client.

The LDIF syntax for setting a bind rule based on the day in the week is as follows:

```
dayofweek = "day1, day2 ..."
```

The possible values for the **dayofweek** keyword are the English three-letter abbreviations for the days of the week: **sun**, **mon**, **tue**, **wed**, **thu**, **fri**, **sat**.

### 13.4.9.1. Examples

The following are examples of the **timeofday** and **dayofweek** syntax:

- The bind rule is evaluated to be true if the client is accessing the directory at noon.

```
timeofday = "1200";
```

- The bind rule is evaluated to be true if the client is accessing the directory at any time other than 1 a.m.

```
timeofday != "0100";
```

- The bind rule is evaluated to be true if the client is accessing the directory at any time after 8 a.m.

```
timeofday > "0800";
```

- The bind rule is evaluated to be true if the client is accessing the directory at any time before 6 p.m.

```
timeofday < "1800";
```

- The bind rule is evaluated to be true if the client is accessing the directory at 8 a.m. or later.

```
timeofday >= "0800";
```

- The bind rule is evaluated to be true if the client is accessing the directory at 6 p.m. or earlier.

```
timeofday <= "1800";
```

- The bind rule is evaluated to be true if the client is accessing the directory on Sunday, Monday, or Tuesday.

```
dayofweek = "Sun, Mon, Tue";
```

### 13.4.10. Defining Access Based on Authentication Method

The **authmethod** keyword sets the specific method that a client uses to bind to the directory. There are four available authentication methods:

- *None*. Authentication is not required. This is the default. It represents anonymous access.
- *Simple*. The client must provide a user name and password to bind to the directory.
- *SSL*. The client must bind to the directory using some kind of PKI credentials, meaning a client must present an SSL certificate either in a database or on a smart card, token, or some other device.

Certificate-based authentication, as one method, is described in [Section 7.10, “Using Client \(Certificate-Based\) Authentication”](#).

- *SASL*. The client must bind to the directory over a Simple Authentication and Security Layer (SASL) connection. Directory Server supports several SASL mechanisms: **PLAIN**, **EXTERNAL**, **CRAM-MD5**, **DIGEST-MD5** (for Kerberos systems), and **GSS-API** (for Kerberos systems). For information on setting up SASL, see [Section 7.11, “Setting up SASL Identity Mapping”](#).



#### NOTE

You cannot set up authentication-based bind rules through the **Access Control Editor**.

The LDIF syntax for setting a bind rule based on an authentication method is as follows:

```
authmethod = "auth_mechanism"
```

*auth\_mechanism* can be **none**, **simple**, **ssl**, or **sasl** *sasl\_mechanism*.

#### 13.4.10.1. Examples

The following are examples of the **authmethod** keyword:

- Authentication is not checked during bind rule evaluation.

```
authmethod = "none";
```



- The bind rule is evaluated to be true if the client is accessing the directory using a user name and password.

```
authmethod = "simple";
```

- The bind rule is evaluated to be true if the client authenticates to the directory using a certificate over LDAPS. This is not evaluated to be true if the client authenticates using simple authentication (bind DN and password) over LDAPS. The **authmethod = "ssl"** means that a certificate must be presented to authenticate to the server. This does *not* configure a required connection type, even though SSL has to be used with certificate-based authentication.

```
authmethod = "ssl";
```

- The bind rule is evaluated to be true if the client is accessing the directory using the SASL DIGEST-MD5 mechanism.

```
authmethod = "sasl DIGEST-MD5";
```

### 13.4.11. Using Boolean Bind Rules

Bind rules can be complex expressions that use the Boolean expressions **AND**, **OR**, and **NOT** to set very precise access rules. You cannot use the Directory Server Console to create Boolean bind rules. You must create an LDIF statement.

The LDIF syntax for a Boolean bind rule is as follows:

```
bind_rule [boolean][bind_rule][boolean][bind_rule]...;)
```

For example, this bind rule is evaluated to be true if the bind DN is a member of either the administrator's group or the **Mail Administrator's** group and if the client is running from within the **example.com** domain:

```
(groupdn = "ldap:///cn=administrators,dc=example,dc=com" or
 groupdn = "ldap:///cn=mail administrators,dc=example,dc=com" and
 dns = "/*.example.com");)
```

The trailing semicolon (;) is a required delimiter that must appear after the final bind rule.

Boolean expressions are evaluated in the following order:

- Innermost to outermost parenthetical expressions first.
- All expressions from left to right.
- **NOT** before **AND** or **OR** operators.
- **OR** and **AND** operators have no order of precedence.

Consider the following Boolean bind rules:

```
(bind_rule_A) OR (bind_rule_B)
(bind_rule_B) OR (bind_rule_A)
```

Because Boolean expressions are evaluated from left to right, in the first case, bind rule A is evaluated before bind rule B, and, in the second case, bind rule B is evaluated before bind rule A.

However, the Boolean **NOT** is evaluated *before* the Boolean **OR** and Boolean **AND**. Thus, in the following example, bind rule B is evaluated before bind rule A despite the left-to-right rule.

```
(bind_rule_A) AND NOT (bind_rule_B)
```

## 13.5. CREATING ACIS FROM THE CONSOLE

You can use the Directory Server Console to view, create, edit, and delete access control instructions for your directory:

- [Section 13.5.1, “Displaying the Access Control Editor”](#)
- [Section 13.5.2, “Creating a New ACI”](#)
- [Section 13.5.3, “Editing an ACI”](#)
- [Section 13.5.4, “Deleting an ACI”](#)

See [Section 13.9, “Access Control Usage Examples”](#) for a collection of access control rules commonly used in Directory Server security policies, along with step-by-step instructions for using the Directory Server Console to create them.

The **Access Control Editor** prevents creating more complex ACIs in visual editing mode, especially ACIs with any of these characteristics:

- Deny access ([Section 13.3.3.4, “Permissions Syntax”](#)).
- Create value-based ACIs ([Section 13.3.2.2, “Targeting Attributes”](#)).
- Define parent access ([Section 13.4.2.4, “Parent Access \(parent Keyword\)”](#)).
- Create ACIs that contain Boolean bind rules ([Section 13.4.11, “Using Boolean Bind Rules”](#)).
- Create ACIs that use the **roledn**, **userattr**, **authmethod** keywords.

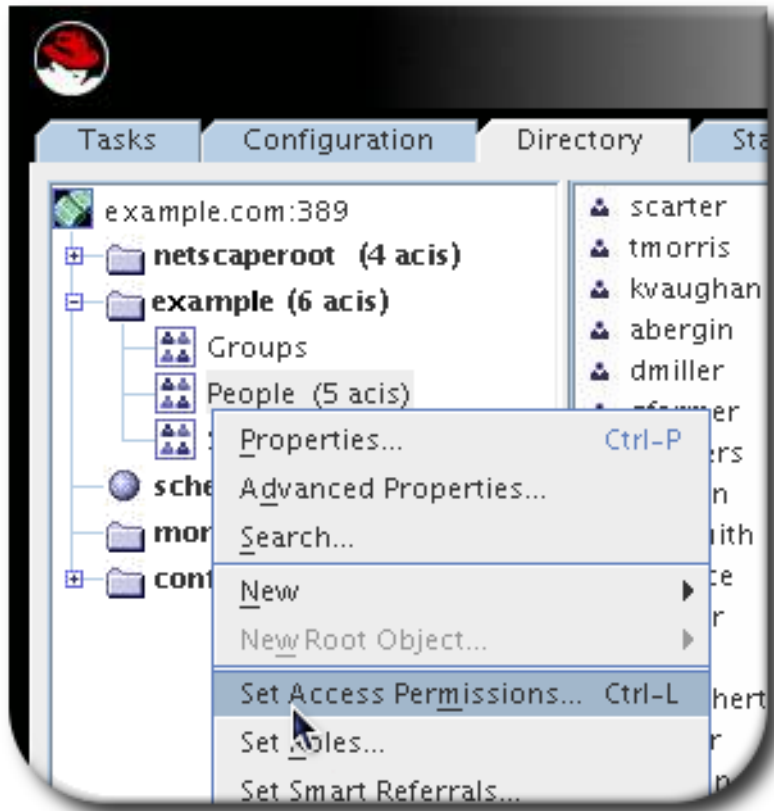


### NOTE

In the **Access Control Editor**, click the **Edit Manually** button at any time to check the LDIF representation of the ACI changes made through the graphical interface.

### 13.5.1. Displaying the Access Control Editor

1. Start the Directory Server Console. Log in using the bind DN and password of a privileged user, such as the Directory Manager, who has write access to the ACIs configured for the directory.
2. Select the **Directory** tab.
3. Right-click the entry in the navigation tree for which to set access control, and select **Set Access Permissions** from the pop-up menu.



Alternatively, highlight the entry, and select **Set Access Permissions** from the **Object** menu.

4. Click **New** to open the **Access Control Editor**.

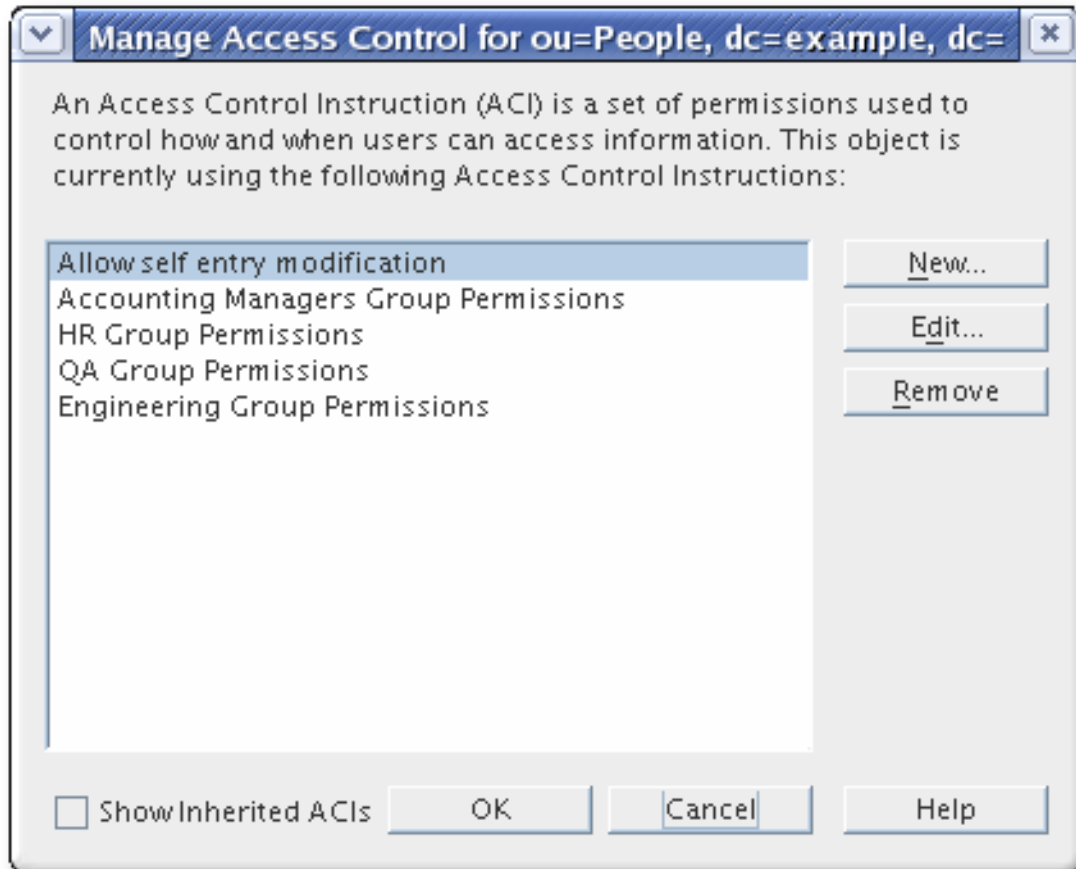


Figure 13.2. Access Control Editor Window

### 13.5.2. Creating a New ACI

To create a new ACI in the Directory Server Console:

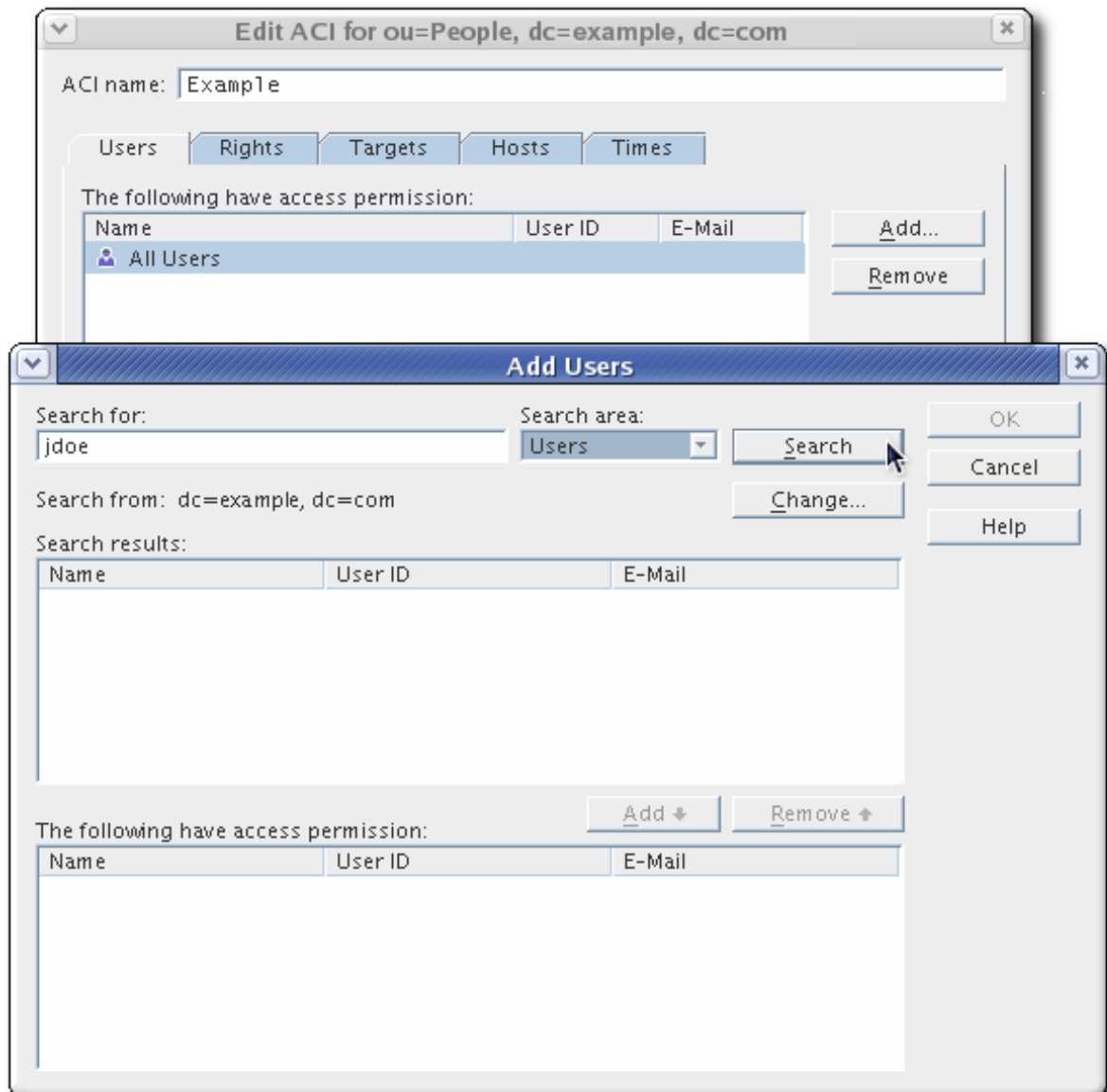
1. Open the **Access Control Editor**, as described in [Section 13.5.1, “Displaying the Access Control Editor”](#).

If the view displayed is different from [Figure 13.2, “Access Control Editor Window”](#), click the **Edit Visually** button.

2. Type the ACI name in the **ACI Name** field.

The name can be any unique string to identify the ACI. If you do not enter a name, the server uses **unnamed ACI**.

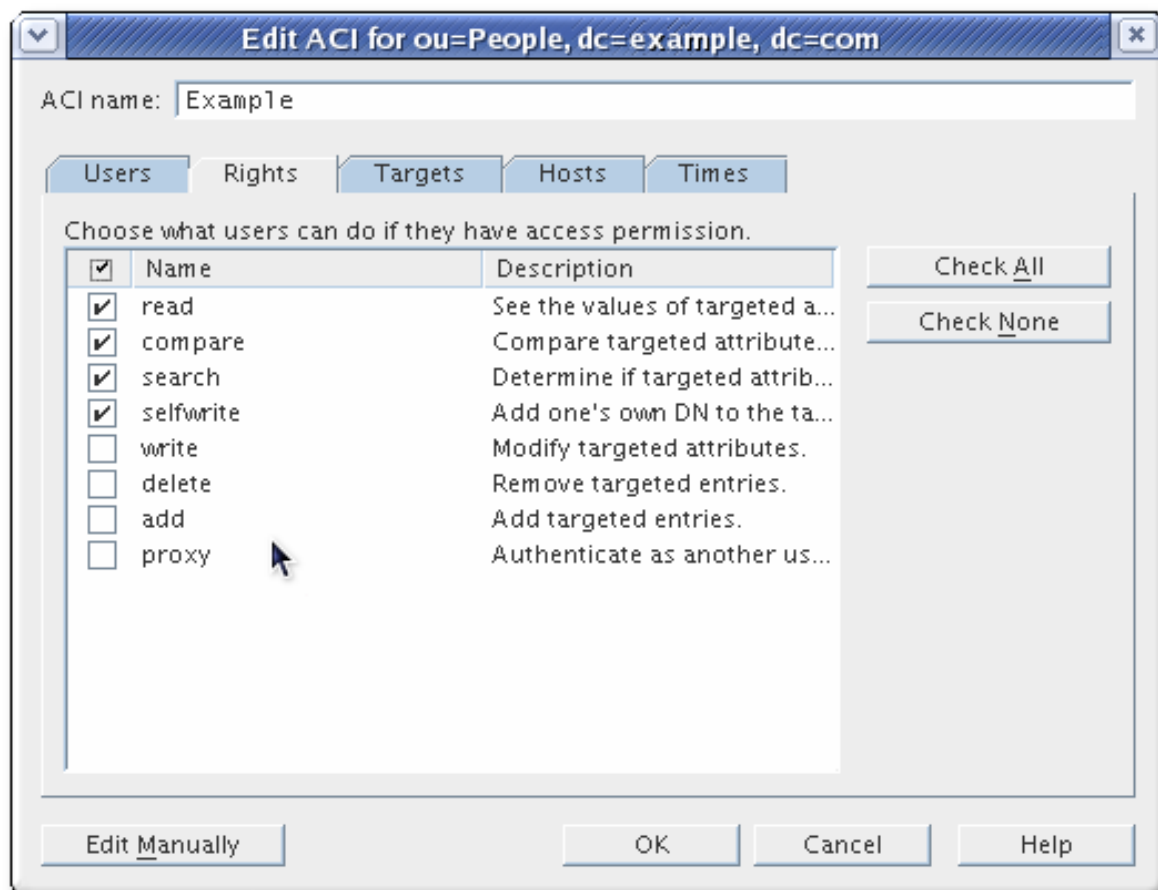
3. In the **Users/Groups** tab, select the users to whom you are granting access by highlighting **All Users** or clicking the **Add** button to search the directory for the users to add.



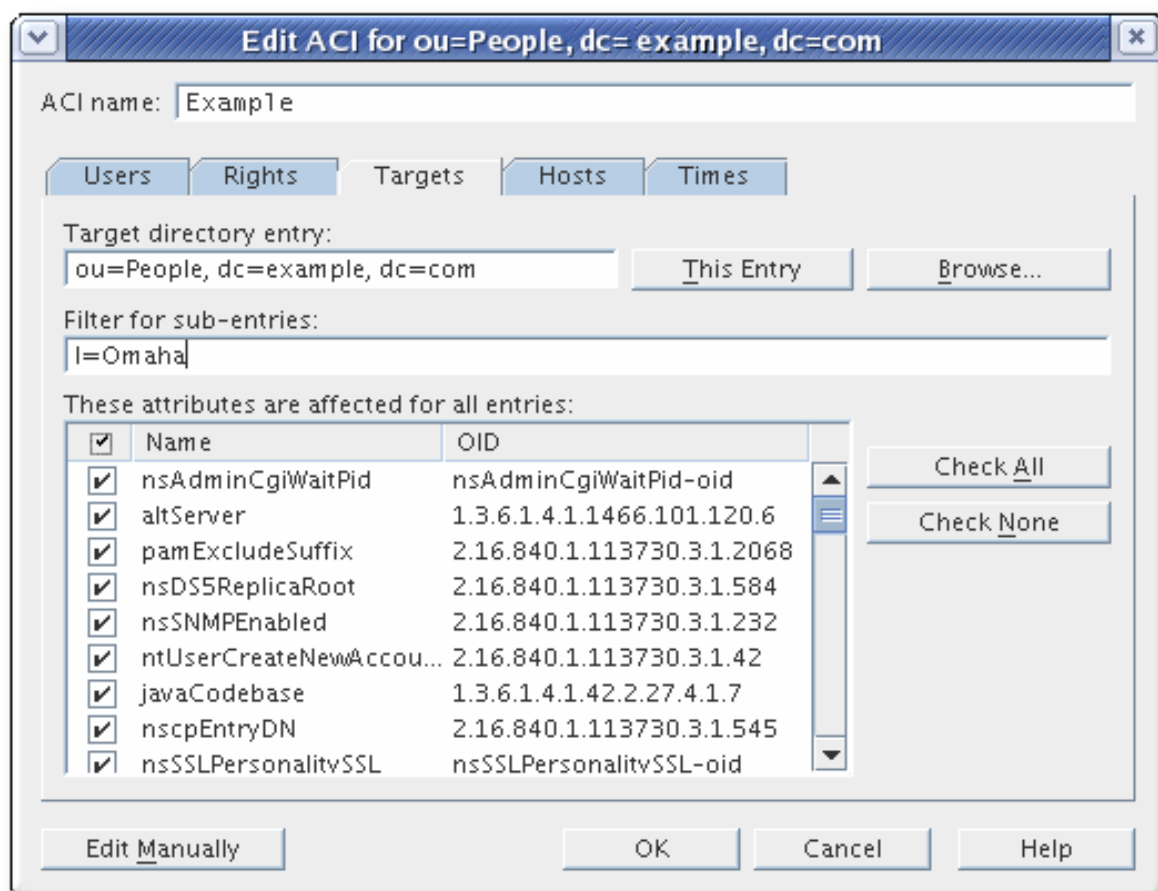
1. Select a search area from the drop-down list, enter a search string in the **Search** field, and click the **Search** button. You can use wildcards (an asterisk, **\***) to search for partial user names. The search results are displayed in the list below.
2. Highlight the entries you want in the search result list, and click the **Add** button to add them to the list of entries which have access permission.
3. Click **OK** to dismiss the **Add Users and Groups** window.

The selected entries are now listed on the **Users/Groups** tab in the ACI editor.

4. In the **Access Control Editor**, click the **Rights** tab, and use the check boxes to select the rights to grant.



- Click the **Targets** tab. Click **This Entry** to display the current node as the target for the ACI or click **Browse** to select a different suffix.



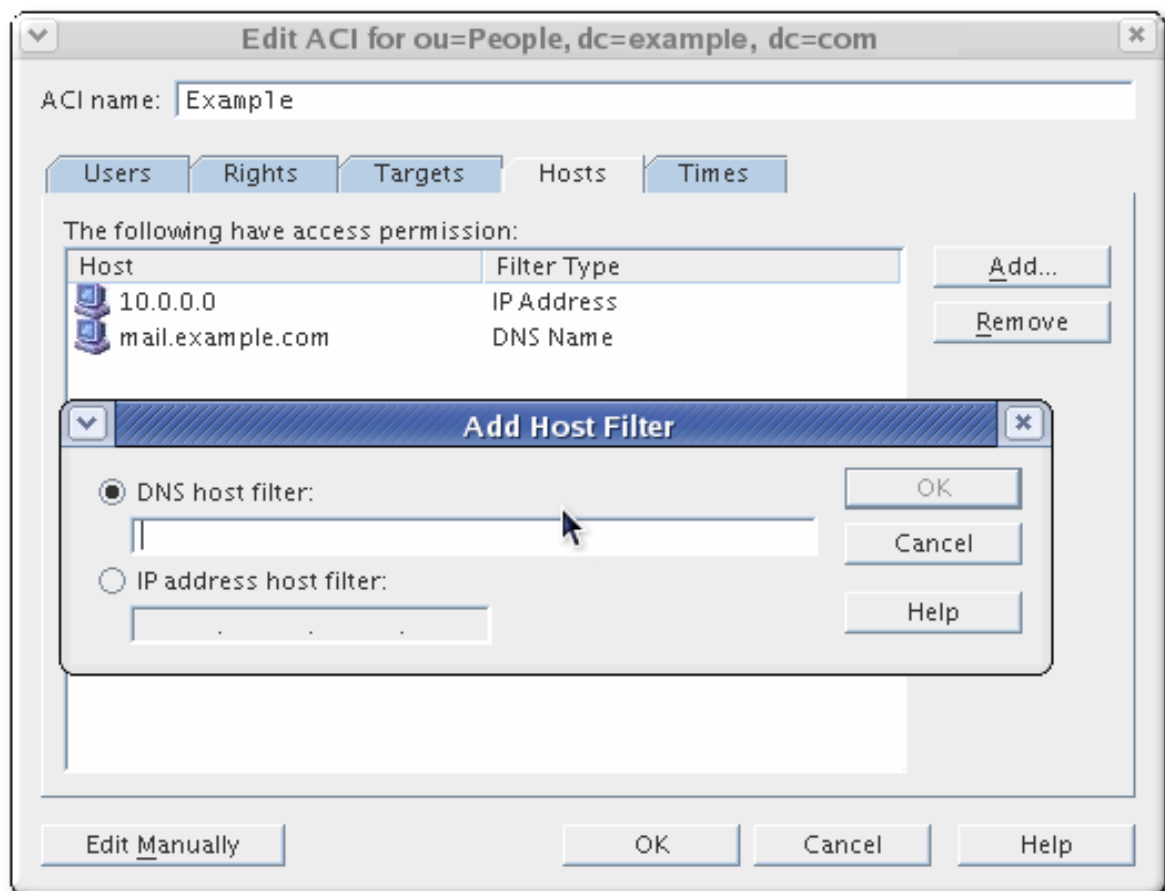
**NOTE**

You can change the value of the target DN, but the new DN must be a direct or indirect child of the selected entry.

If you do not want every entry in the subtree under this node to be targeted by the ACI, enter a filter in the **Filter for Sub-entries** field. The filter applies to every entry below the target entry; for example, setting a filter of **ou=Sales** means that only entries with **ou=Sales** in their DN are returned.

Additionally, you can restrict the scope of the ACI to only certain attributes by selecting the attributes to target in the attribute list.

- Click the **Hosts** tab, then the **Add** button to open the **Add Host Filter** dialog box.



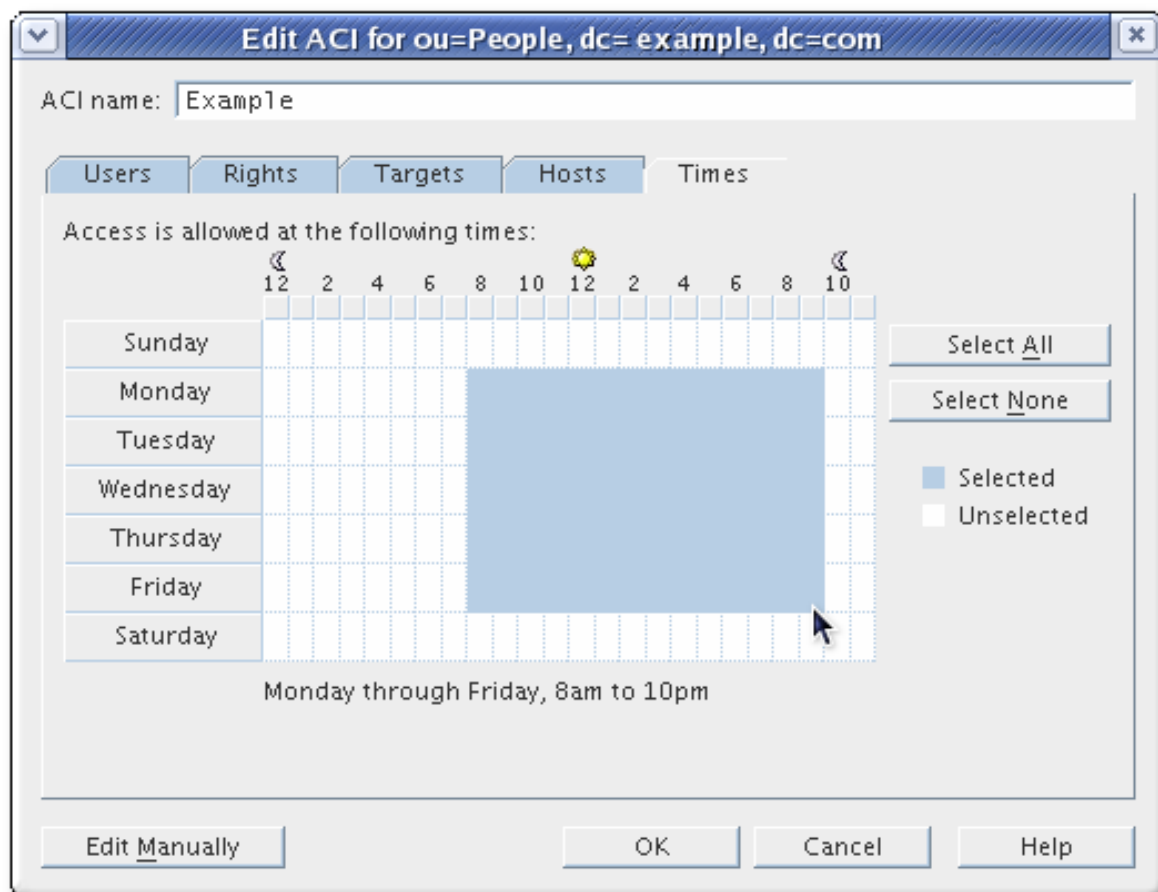
You can specify a host name or an IP address. With an IP address, you can use an asterisk (\*) as a wildcard.

**NOTE**

Directory Server supports both IPv4 and IPv6 IP addresses.

- Click the **Times** tab to display the table showing at what times access is allowed.

By default, access is allowed at all times. You can change the access times by clicking and dragging the cursor over the table. You cannot select discrete blocks of time, only continuous time ranges.



8. Click **OK** when all of the configuration is complete.

The **Access Control Editor** closes, and the new ACI is listed in the **Access Control Manager** window.



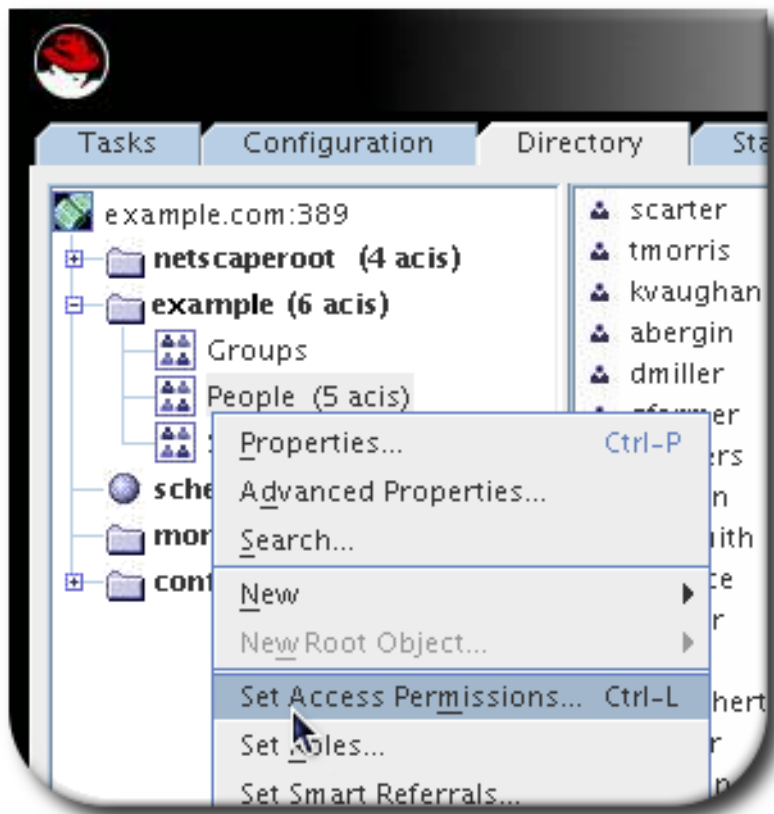
#### NOTE

For any point of creating the ACI, click the **Edit Manually** button to display the LDIF statement corresponding to the wizard input. This statement can be edited directly, but the changes may not be visible in the graphical interface.

### 13.5.3. Editing an ACI

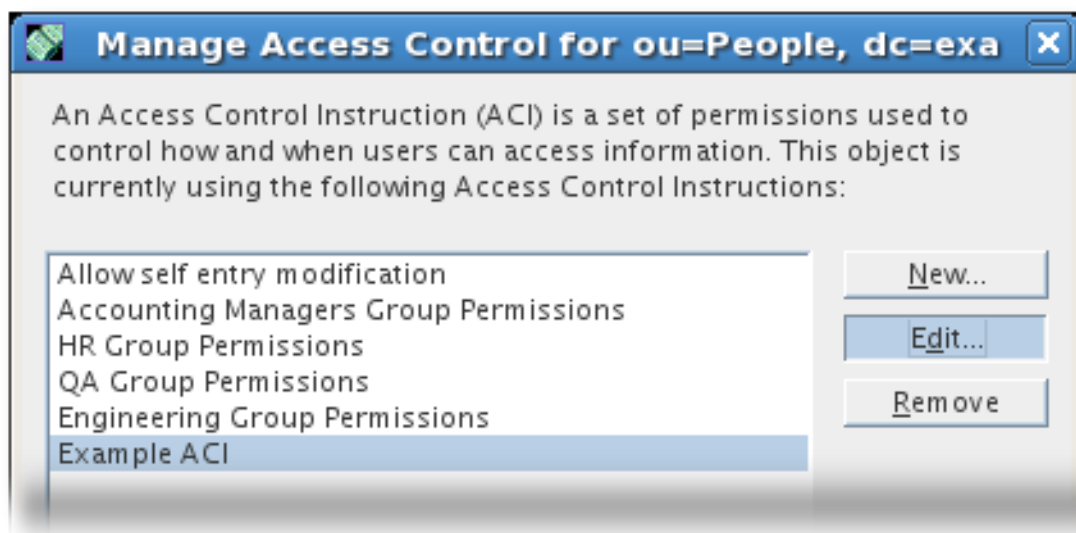
1. In the **Directory** tab, right-click the top entry in the subtree, and choose **Set Access Permissions** from the pop-up menu.





The **Access Control Manager** window opens, listing the ACIs belonging to the entry.

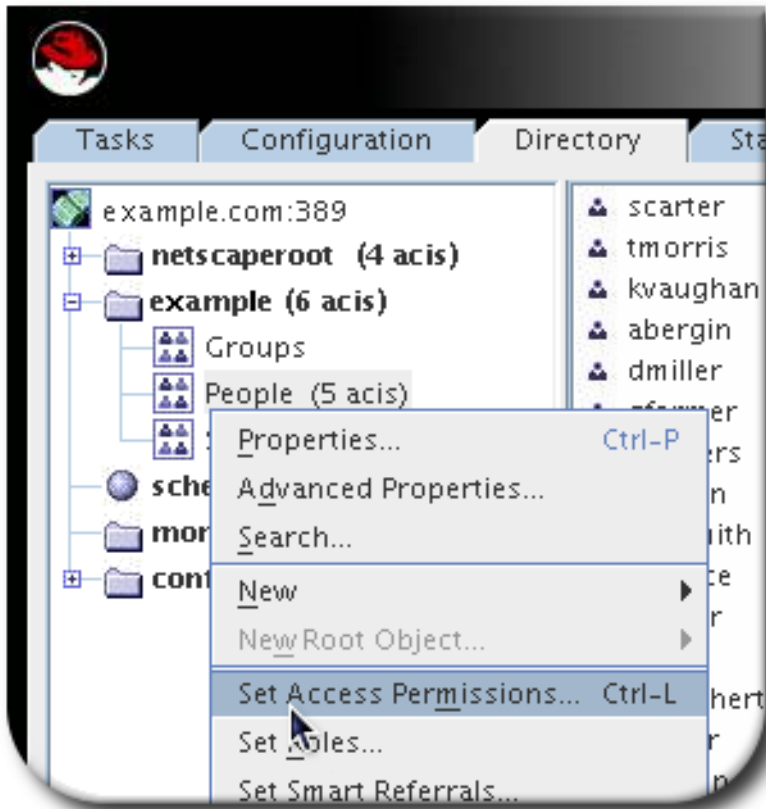
2. In the **Access Control Manager** window, highlight the ACI to edit, and click **Edit**.



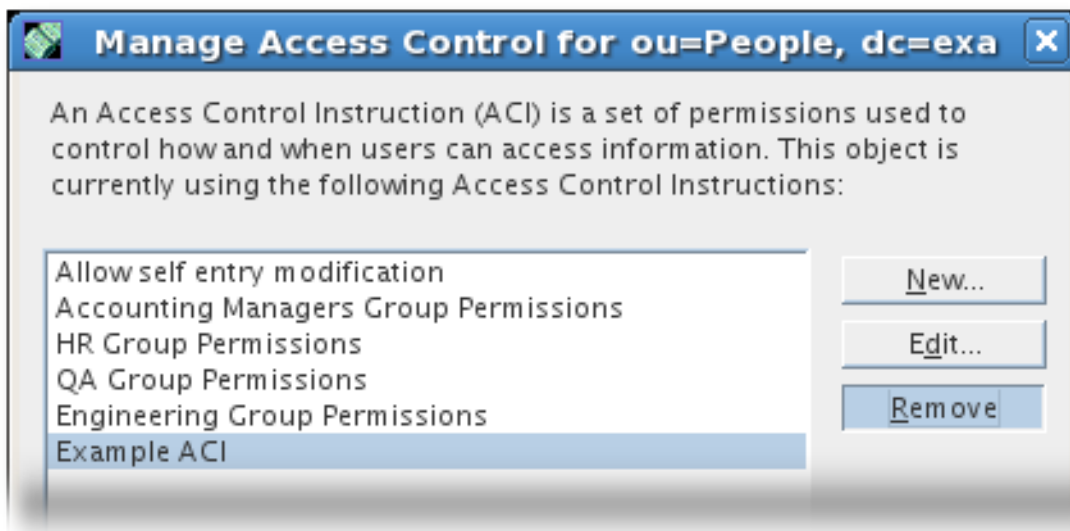
3. Make the edits to the ACI in the **Access Control Editor**; the different screens are described more in [Section 13.5.2, “Creating a New ACI”](#) and in the online help.
4. When the edits are complete, click **OK**.

### 13.5.4. Deleting an ACI

1. In the **Directory** tab, right-click the top entry in the subtree, and choose **Set Access Permissions** from the pop-up menu.



2. In the **Access Control Manager** window, select the ACI to delete.
3. Click **Remove**.



The ACI is no longer listed in the **Access Control Manager** window.

## 13.6. VIEWING ACIS

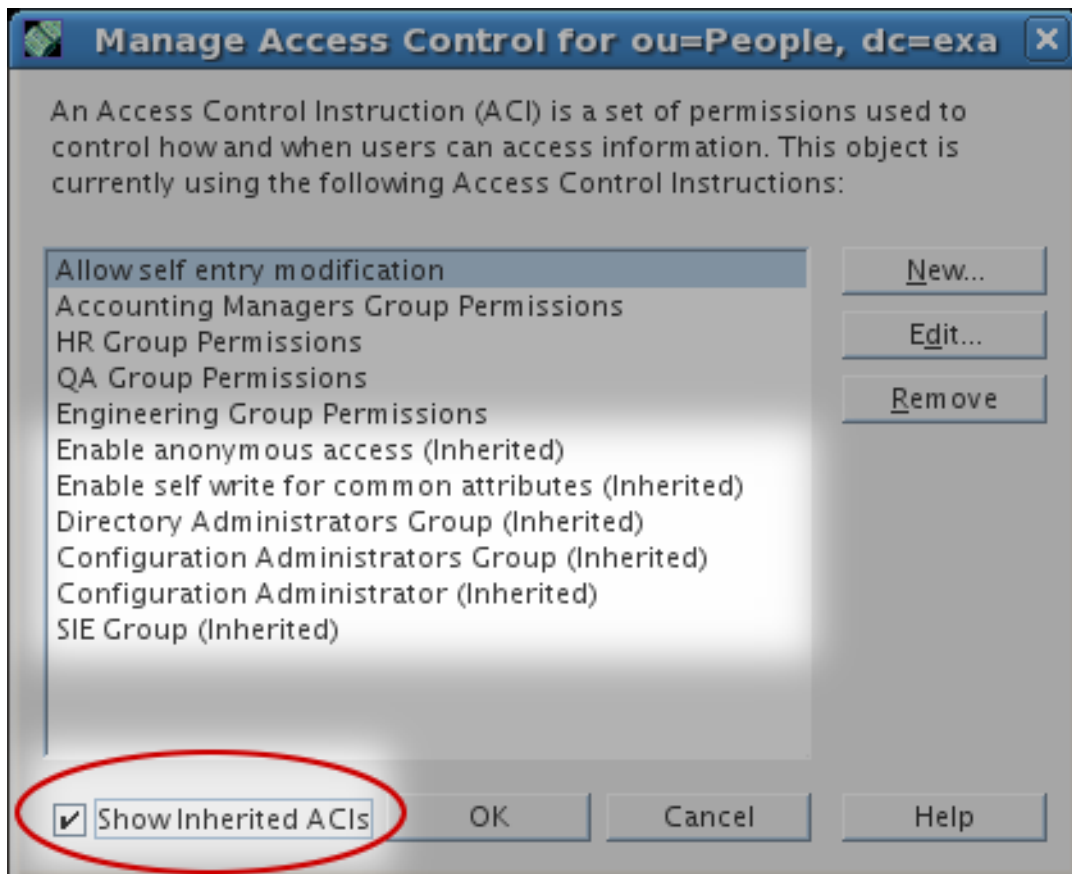
All the ACIs under a single suffix in the directory can be viewed from the command line by using the following **ldapsearch** command:

```
ldapsearch -x -D bind_dn -w password -p server_port -h server_hostname
(aci=*) aci
```

See [Chapter 10, Finding Directory Entries](#) for information on using the **ldapsearch** utility.

From the Directory Server Console, all of the ACIs that apply to a particular entry can be viewed through the **Access Control Manager**.

1. Start the Directory Server Console.
2. In the **Directory** tab, right-click the entry in the navigation tree, and select **Set Access Permissions**.
3. Check the **Show Inherited ACIs** check box to display all ACIs created on entries above the selected entry that also apply.



## 13.7. CHECKING ACCESS RIGHTS ON ENTRIES (GET EFFECTIVE RIGHTS)

Finding the access rights that a user has on attributes within a specific entry offers a convenient way for administrators to find and control the access rights.

*Get effective rights* is a way to extend directory searches to display what access rights — such as read, search, write and self-write, add, and delete — a user has to a specified entry.

In Directory Server, regular users can check their rights over entries which they can view and can check other people's access to their personal entries. The Directory Manager can check rights that one user has over another user.

There are two common situations where checking the effective rights on an entry are useful:

- An administrator can use the *get effective rights* command in order to better organize access control instructions for the directory. It is frequently necessary to restrict what one group of users

can view or edit versus another group. For instance, members of the **QA Managers** group may have the right to search and read attributes like *manager* and *salary* but only **HR Group** members have the rights to modify or delete them. Checking the effective rights for a user or group is one way to verify that the appropriate access controls are in place.

- A user can run the get effective rights command to see what attributes he can view or modify on his personal entry. For instance, a user should have access to attributes such as *homePostalAddress* and *cn* but may only have read access to *manager* and *salary* attributes.

There are three people involved in a get effective rights search. The first is the person running the search command, the *requester*. The rights are checked (with a variety of permutations) to see what rights Person A has over Entry B. The person whose rights are being checked (Person A) is the *GER subject*; as in, their rights are the subject of the search. The entry or entries to which the person has rights (Entry B) is the *target* of the search or the *search base*.

### 13.7.1. Rights Shown with a Get Effective Rights Search

Any get effective rights search, both when viewing an entry in the Directory Server Console and searching for it in the command line, shows the rights that User A has to User B's entry.

There are two kinds of access rights that can be allowed to any entry. The first are upper-level rights, *rights on the entry itself*, which means that kinds of operations that the User A can perform on User B's entry as a whole. The second level of access rights are more granular, show what *rights for a given attribute* User A has. In this case, User A may have different kinds of access permissions for different attributes in the same entry. Whatever access controls are allowed for a user are the *effective* rights over that entry.

For example:

```
entryLevelRights: vadm
attributeLevelRights: givenName:rscwO, sn:rscw, objectClass:rsc, uid:rsc,
cn:rscw
```

[Table 13.6, “Entry Rights”](#) and [Table 13.7, “Attribute Rights”](#) show the access rights to entries and attributes, respectively, that are returned by a get effective rights search.

**Table 13.6. Entry Rights**

| Permission | Description        |
|------------|--------------------|
| a          | Add an entry.      |
| d          | Delete this entry. |
| n          | Rename the DN.     |
| v          | View the entry.    |

**Table 13.7. Attribute Rights**

| Permission | Description                                          |
|------------|------------------------------------------------------|
| r          | Read.                                                |
| s          | Search.                                              |
| w          | Write ( <b>mod - add</b> ).                          |
| o          | Obliterate( <b>mod - del</b> ). Analogous to delete. |
| c          | Compare.                                             |
| W          | Self-write.                                          |
| O          | Self-delete.                                         |

### 13.7.2. The Format of a Get Effective Rights Search

Get effective rights (sometimes called GER) is an extended directory search; the GER parameters are defined with the **-E** option to pass an LDAP control with the **ldapsearch** command. (If an **ldapsearch** is run without the **-E** option, then, naturally, the entry is returned as normal, without any get effective rights information.)

```
ldapsearch -x -D bind_dn -w password -p server_port -h server_hostname -E
[!]1.3.6.1.4.1.42.2.27.9.5.2=:GER_subject (searchFilter) attributeList
```

- **-b** is the base DN of the subtree or entry used to search for the GER subject.

If the search base is a specific entry DN or if only one entry is returned, then the results show the rights the requester has over that specific entry. If multiple entries beneath the search base match the filter, then the search returns every matching entry, with the rights for the requester over each entry.

- **1.3.6.1.4.1.42.2.27.9.5.2** is the OID for the get effective rights control.
- The exclamation point (!) specifies whether the search operation should return an error if the server does not support this control (!) or if it should be ignored and let the search return as normal (nothing).
- The *GER\_subject* is the person whose rights are being checked. If the *GER\_subject* is left blank (**dn:**), then the rights of an anonymous user are returned.
- An optional *attributeList* limits the get effective rights results to the specified attribute or object class. As with a regular **ldapsearch**, this can give specific attributes, like *mail*. If no attributes are listed, then every present attribute for the entry is returned. Using an asterisk (\*) returns the rights for every possible attribute for the entry, both existing attribute and non-existent attributes. Using an plus sign (+) returns operational attributes for the entry. Examples for checking rights for specific attributes are given in [Section 13.7.3.2, “Examples of Get Effective Rights Searches for Non-Existent Attributes”](#) and [Section 13.7.3.3, “Examples of Get Effective Rights Searches for Specific Attributes or Object Classes”](#).

The crux of a get effective rights search is the ability to check what rights the GER subject (**-E**) has to the targets of the search (**-b**). The get effective rights search is a regular **ldapsearch**, in that it simply looks for entries that match the search parameters and returns their information. The get effective rights option adds extra information to those search results, showing what rights a specific user has over those results. That GER subject user can be the requester himself (**-D** is the same as **-E**) or someone else.

If the requester is a regular user (not the Directory Manager), then the requester can only see the effective that a GER subject has on the requester's own entry. That is, if John Smith runs a request to see what effective rights Babs Jensen has, then he can only get the effective rights that Babs Jensen has on his own entry. All of the other entries return an insufficient access error for the effective rights.

There are three general scenarios for a regular user when running a get effective rights search:

- User A checks the rights that he has over other directory entries.
- User A checks the rights that he has to his personal entry.
- User A checks the rights that User B has to User A's entry.

The get effective rights search has a number of flexible different ways that it can check rights on attributes.

### 13.7.3. Examples of GER Searches

There are a number of different ways to run GER searches, depending on the exact type of information that needs to be returned and the types of entries and attributes being searched.

#### 13.7.3.1. General Examples on Checking Access Rights

One common scenario for effective rights searches is for a regular user to determine what changes he can make to his personal entry.

For example, Ted Morris wants to check the rights he has to his entry. Both the **-D** and **-E** options give his entry as the requester. Since he is checking his personal entry, the **-b** option also contains his DN.

##### Example 13.1. Checking Personal Rights (User A to User A)

```
ldapsearch -x -p 389 -h server.example.com -D
"uid=tmorris,ou=people,dc=example,dc=com" -w secret -b
"uid=tmorris,ou=people,dc=example,dc=com" -E
"!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=tmorris,ou=people,dc=example,dc=com"
"(objectClass=*)"

dn: uid=tmorris,ou=People,dc=example,dc=com
givenName: Ted
sn: Morris
ou: IT
ou: People
l: Santa Clara
manager: uid=jsmith,ou=People,dc=example,dc=com
roomNumber: 4117
mail: tmorris@example.com
facsimileTelephoneNumber: +1 408 555 5409
objectClass: top
objectClass: person
```

```

objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: tmorris
cn: Ted Morris
userPassword: {SSHA}bz0uCmHZM5b357zwrCUCJs1IOHtMD6yqPyhxBA==
entryLevelRights: v
attributeLevelRights: givenName:rsc, sn:rsc, ou:rsc, l:rsc,
manager:rsc, roomNumber:rscwo, mail:rscwo,
facsimileTelephoneNumber:rscwo, objectClass:rsc, uid:rsc, cn:rsc,
userPassword:wo

```

Ted Morris may, for example, be a manager or work in a department where he has to edit other user's entries, such as IT or human resources. In this case, he may want to check what rights he has to another user's entry, as in [Example 13.2](#), “[Personally Checking the Rights of One User over Another \(User A to User B\)](#)”, where Ted (-D) checks his rights (-E) to Dave Miller's entry (-b):

### Example 13.2. Personally Checking the Rights of One User over Another (User A to User B)

```

ldapsearch -p 389 -h server.example.com -D
"uid=tmorris,ou=people,dc=example,dc=com" -w secret -b
"uid=dmiller,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=tmorris,ou=people,dc=example,dc=com'
"(objectClass=*)"

dn: uid=dmiller,ou=People,dc=example,dc=com
... snip ...
entryLevelRights: vad
attributeLevelRights: givenName:rscwo, sn:rscwo, ou:rscwo, l:rscwo,
manager:rsc, roomNumber:rscwo, mail:rscwo,
facsimileTelephoneNumber:rscwo, objectClass:rscwo, uid:rscwo, cn:rscwo,
userPassword:rswo

```

For all attributes, Ted Morris has read, search, compare, modify, and delete permissions to Dave Miller's entry. These results are different than the ones returned in checking Ted Morris's access to his own entry, since he personally had only read, search, and compare rights to most of these attributes.

The Directory Manager has the ability to check the rights that one user has over another user's entry. In [Example 13.3](#), “[The Directory Manager's Checking the Rights of One User over Another \(User A to User B\)](#)”, the Directory Manager is checking the rights that a manager, Jane Smith (-E), has over her subordinate, Ted Morris (-b):

### Example 13.3. The Directory Manager's Checking the Rights of One User over Another (User A to User B)

```

ldapsearch -p 389 -h server.example.com -D "cn=directory manager" -w
secret -b "uid=tmorris,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=jsmith,ou=people,dc=example,dc=com'
"(objectClass=*)"

dn: uid=tmorris,ou=People,dc=example,dc=com
... snip ...
entryLevelRights: vadm

```

```
attributeLevelRights: givenName:rscwo, sn:rscwo, ou:rscwo, l:rscwo,
manager:rscwo, roomNumber:rscwo, mail:rscwo,
facsimileTelephoneNumber:rscwo, objectClass:rscwo, uid:rscwo, cn:rscwo,
userPassword:rscwo
```

Only an administrator can retrieve the effective rights that a different user has on an entry. If Ted Morris tried to determine Dave Miller's rights to Dave Miller's entry, then he would receive an insufficient access error:

```
ldapsearch -p 389 -h server.example.com -D
"uid=dmiller,ou=people,dc=example,dc=com" -w secret -b
"uid=tmorris,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=tmorris,ou=people,dc=example,dc=com' "
(objectClass=*)"

ldap_search: Insufficient access
ldap_search: additional info: get-effective-rights: requester has no g
permission on the entry
```

However, a regular user can run a get effective rights search to see what rights another user has to his personal entry. In [Example 13.4, "Checking the Rights Someone Else Has to a Personal Entry"](#), Ted Morris checks what rights Dave Miller has on Ted Morris's entry.

#### Example 13.4. Checking the Rights Someone Else Has to a Personal Entry

```
ldapsearch -p 389 -h server.example.com -D
"uid=tmorris,ou=people,dc=example,dc=com" -w secret -b
"uid=tmorris,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=dmiller,ou=people,dc=example,dc=com'
"(objectClass=*)"

dn: uid=tmorris,ou=people,dc=example,dc=com
... snip ...
entryLevelRights: v
attributeLevelRights: givenName:rsc, sn:rsc, ou:rsc, l:rsc,manager:rsc,
roomNumber:rsc, mail:rsc, facsimileTelephoneNumber:rsc, objectClass:rsc,
uid:rsc, cn:rsc, userPassword:none
```

In this case, Dave Miller has the right to view the DN of the entry and to read, search, and compare the **ou**, **givenName**, **l**, and other attributes, and no rights to the **userPassword** attribute.

### 13.7.3.2. Examples of Get Effective Rights Searches for Non-Existent Attributes

By default, information is not given for attributes in an entry that do not have a value; for example, if the **userPassword** value is removed, then a future effective rights search on the entry above would not return any effective rights for **userPassword**, even though self-write and self-delete rights could be allowed.

Using an asterisk (\*) with the get effective rights search returns every attribute available for the entry, including attributes not set on the entry.



**Example 13.5. Returning Effective Rights for Non-Existent Attributes**

```
ldapsearch -D "cn=directory manager" -w secret -b
"uid=scarter,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com'
"(objectclass=*)" "*"

dn: uid=scarter,ou=People,dc=example,dc=com
givenName: Sam
telephoneNumber: +1 408 555 4798
sn: Carter
ou: Accounting
ou: People
l: Sunnyvale
manager: uid=dmiller,ou=People,dc=example,dc=com
roomNumber: 4612
mail: scarter@example.com
facsimileTelephoneNumber: +1 408 555 9700
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: scarter
cn: Sam Carter
userPassword: {SSHA}Xd9Jt8g1UsHC8enNDrEmxj3iJPKQLitlDYdD9A==
entryLevelRights: vadm
attributeLevelRights: objectClass:rscwo, aci:rscwo, sn:rscwo, cn:rscwo,
description:rscwo, seeAlso:rscwo, telephoneNumber:rscwo,
userPassword:rscwo, destinationIndicator:rscwo,
facsimileTelephoneNumber:rscwo, internationaliSDNNumber:rscwo, l:rscwo,
ou:rscwo, physicalDeliveryOfficeName:rscwo, postOfficeBox:rscwo,
postalAddress:rscwo, postalCode:rscwo, preferredDeliveryMethod:rscwo,
registeredAddress:rscwo, st:rscwo, street:rscwo,
teletexTerminalIdentifier:rscwo, telexNumber:rscwo, title:rscwo,
x121Address:rscwo, audio:rscwo, businessCategory:rscwo,
carLicense:rscwo, departmentNumber:rscwo, displayName:rscwo,
employeeType:rscwo, employeeNumber:rscwo, givenName:rscwo,
homePhone:rscwo, homePostalAddress:rscwo, initials:rscwo,
jpegPhoto:rscwo, labeledUri:rscwo, manager:rscwo, mobile:rscwo,
pager:rscwo, photo:rscwo, preferredLanguage:rscwo, mail:rscwo, o:rscwo,
roomNumber:rscwo, secretary:rscwo, uid:rscwo,x500UniqueIdentifier:rscwo,
userCertificate:rscwo, userSMIMECertificate:rscwo, userPKCS12:rscwo
```

All of the attributes available for the entry, such as **secretary**, are listed, even though that attribute is non-existent.

**13.7.3.3. Examples of Get Effective Rights Searches for Specific Attributes or Object Classes**

Taking the attribute-related GER searches further, it is possible to search for the rights to a specific attribute and set of attributes and to list all of the attributes available for one of the object classes set on the entry.

One of the options listed in the formatting example in [Section 13.7.2, “The Format of a Get Effective Rights Search”](#) is *attributeList*. To return the effective rights for only specific attributes, list the attributes, separated by spaces, at the end of the search command.

### Example 13.6. Get Effective Rights Results for Specific Attributes

```
ldapsearch -D "cn=directory manager" -w secret -b
"uid=scarter,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com'
"(objectclass=*)" cn mail initials

dn: uid=scarter,ou=People,dc=example,dc=com
cn: Sam Carter
mail: scarter@example.com
entryLevelRights: vadm
attributeLevelRights: cn:rscwo, mail:rscwo, initials:rscwo
```

It is possible to specify a non-existent attribute in the *attributeList*, as with the *initials* attribute in [Example 13.6, “Get Effective Rights Results for Specific Attributes”](#), to see the rights which are available, similar to using an asterisk to list all attributes.

The Directory Manager can also list the rights for all of the attributes available to a specific object class. This option has the format *attribute@objectClass*. This returns two entries; the first for the specified GER subject and the second for a template entry for the object class.

### Example 13.7. Get Effective Rights Results for an Attribute within an Object Class

```
ldapsearch -D "cn=directory manager" -w secret -b
"uid=scarter,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com'
"(objectclass=*)" uidNumber@posixAccount

... snip ...

dn:
cn=template_posixaccount_objectclass,uid=scarter,ou=people,dc=example,dc=com
uidnumber: (template_attribute)
entryLevelRights: v
attributeLevelRights: uidNumber:rsc
```



#### NOTE

Using the search format *attribute@objectClass* is only available if the requester (-D) is the Directory Manager.

Using an asterisk (\*) instead of a specific attribute returns all of the attributes (present and non-existent) for the specified GER subject and the full list of attributes for the object class template.

### Example 13.8. Get Effective Rights Results for All Attributes for an Object Class

```
ldapsearch -D "cn=directory manager" -w secret -b
"uid=scarter,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com'
"(objectclass=*)" *@posixaccount

... snip ...

dn:
cn=template_posixaccount_objectclass,uid=scarter,ou=people,dc=example,dc=com
objectClass: posixaccount
objectClass: top
homeDirectory: (template_attribute)
gidNumber: (template_attribute)
uidNumber: (template_attribute)
uid: (template_attribute)
cn: (template_attribute)
entryLevelRights: v
attributeLevelRights: cn:rsc, uid:rsc, uidNumber:rsc, gidNumber:rsc,
homeDirectory:rsc, objectClass:rsc, userPassword:none, loginShell:rsc,
gecos:rsc, description:rsc, aci:rsc
```

#### 13.7.3.4. Examples of Get Effective Rights Searches for Non-Existent Entries

An administrator may want to check what rights a specific user (**jsmith**) would have to a non-existent user, based on the existing access control rules. For checking non-existent entries, the server generates a fake entry within that subtree. For example, to check for the fake entry **cn=joe new user, cn=accounts, ou=people, dc=example, dc=com**, the server creates **cn=template, cn=accounts, ou=people, dc=example, dc=com**.

For checking a non-existent entry, the get effective rights search can use a specified object class to generate a template entry with all of the potential attributes of the (non-existent) entry. For **cn=joe new user, cn=accounts, ou=people, dc=example, dc=com** with a **person** object class (**@person**), the server generates **cn=template\_person\_objectclass, cn=accounts, ou=people, dc=example, dc=com**.

When the server creates the template entry, it uses the first **MUST** attribute in the object class definition to create the RDN attribute (or it uses **MAY** if there is no **MUST** attribute). However, this may result in an erroneous RDN value which, in turn, violates or circumvents established ACIs for the given subtree. In that case, it is possible to specify the RDN value to use by passing it with the object class. This has the form **@objectclass:rdn\_attribute**.

For example, to check the rights of **scarter** for a non-existent Posix entry with **uidNumber** as its RDN:

```
ldapsearch -D "cn=directory manager" -w secret -b
"ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com' "
(objectclass=*)" @posixaccount:uidnumber

dn:
uidNumber=template_posixaccount_objectclass,ou=people,dc=example,dc=com
entryLevelRights: v
attributeLevelRights: description:rsc, geocos:rsc, loginShell:rsc,
userPassword
```

```
:rsc, objectClass:rsc, homeDirectory:rsc, gidNumber:rsc, uidNumber:rsc,
uid:
 rsc, cn:rsc
```

### 13.7.3.5. Examples of Get Effective Rights Searches for Operational Attributes

Operational attributes are not returned in regular **ldapsearches**, including get effective rights searches. To return the information for the operational attributes, use the plus sign (+). This returns only the operational attributes that can be used in the entry.

#### Example 13.9. Get Effective Rights Results for Operational Attributes

```
ldapsearch -D "cn=directory manager" -w secret -x -b
"uid=scarter,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com'
"(objectclass=*)" "+"
```

```
dn: uid=scarter,ou=People,dc=example,dc=com
entryLevelRights: vadm
attributeLevelRights: nsICQStatusText:rscwo,
passwordGraceUserTime:rscwo, pwdGraceUserTime:rscwo,
nsYIMStatusText:rscwo, modifyTimestamp:rscwo, passwordExpWarned:rscwo,
pwdExpirationWarned:rscwo, entrydn:rscwo, aci:rscwo, nsSizeLimit:rscwo,
nsAccountLock:rscwo, passwordExpirationTime:rscwo, entryid:rscwo,
nsSchemaCSN:rscwo, nsRole:rscwo, retryCountResetTime:rscwo,
ldapSchemas:rscwo, nsAIMStatusText:rscwo, copiedFrom:rscwo,
nsICQStatusGraphic:rscwo, nsUniqueId:rscwo, creatorsName:rscwo,
passwordRetryCount:rscwo, dncomp:rscwo, nsTimeLimit:rscwo,
passwordHistory:rscwo, pwdHistory:rscwo, nscpEntryDN:rscwo,
subschemaSubentry:rscwo, nsYIMStatusGraphic:rscwo,
hasSubordinates:rscwo, pwdpolicysubentry:rscwo,
nsAIMStatusGraphic:rscwo, nsRoleDN:rscwo, createTimeStamp:rscwo,
accountUnlockTime:rscwo, copyingFrom:rscwo, nsLookThroughLimit:rscwo,
nsds5ReplConflict:rscwo, modifiersName:rscwo, parentid:rscwo,
passwordAllowChangeTime:rscwo, nsBackendSuffix:rscwo,
nsIdleTimeout:rscwo, ldapSyntaxes:rscwo, numSubordinates:rscwo
```

### 13.7.3.6. Examples of Get Effective Rights Results and Access Control Rules

Get effective rights are returned according to whatever ACLs are in effect for the get effective rights subject entry.

For example, this ACL is set and, for the purposes of this example, it is the only ACL set:

```
dn: dc=example,dc=com
objectClass: top
objectClass: domain
dc: example
aci: (target=ldap:///ou=Accounting,dc=example,dc=com)(targetattr="*")
(version
 3.0; acl "test acl"; allow (read,search,compare) (userdn =
 "ldap:///anyone") ;)
```

```
dn: ou=Accounting,dc=example,dc=com
objectClass: top
objectClass: organizationalUnit
ou: Accounting
```

Because the ACL does not include the **dc=example, dc=com** subtree, the get effective rights search shows that the user does not have any rights to the **dc=example, dc=com** entry:

#### Example 13.10. Get Effective Rights Results with No ACL Set (Directory Manager)

```
ldapsearch -D "cn=directory manager" -w secret -b "dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com'
"(objectclass=*)" "*"@person"
```

```
dn:
cn=template_person_objectclass,uid=scarter,ou=people,dc=example,dc=com
objectClass: person
objectClass: top
cn: (template_attribute)
sn: (template_attribute)
description: (template_attribute)
seeAlso: (template_attribute)
telephoneNumber: (template_attribute)
userPassword: (template_attribute)
entryLevelRights: none
attributeLevelRights: sn:none, cn:none, objectClass:none,
description:none, seeAlso:none, telephoneNumber:none, userPassword:none,
aci:none
```

If a regular user, rather than Directory Manager, tried to run the same command, the result would simply be blank.

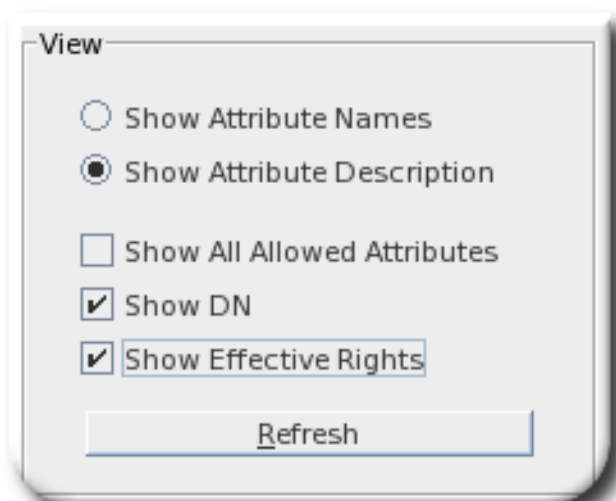
#### Example 13.11. Get Effective Rights Results with No ACL Set (Regular User)

```
$ ldapsearch -D "uid=scarter,ou=people,dc=example,dc=com" -w secret -b
"dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com'
"(objectclass=*)" "*"@person"

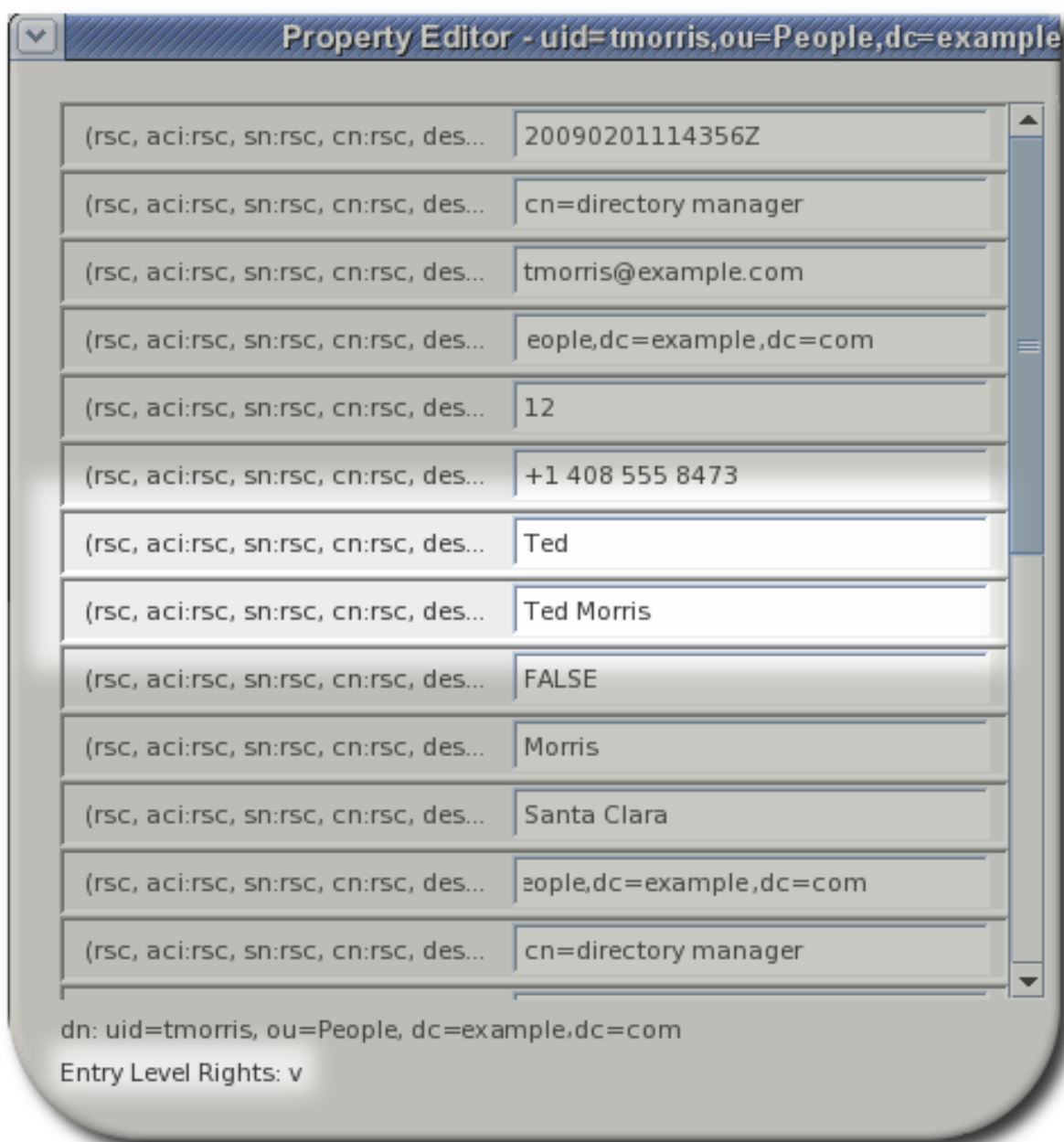
$
```

### 13.7.4. Using Get Effective Rights from the Console

1. Open the **Directory** tab, and right-click the entry of which to check the rights.
2. Select **Advanced Properties** from the drop-down menu.
3. Check the **Show effective rights** check box.



4. Beside each attribute, the attribute-level get effective rights are displayed. The entry-level rights are shown beneath the entry's DN.



The attribute-level effective rights (**r**, **s**, **c**, **w**, **o**) appear next to the attributes. The entry-level rights (**v**, **a**, **d**, **n**) appear under the full DN for the entry in the lower left-hand corner of the **Property Editor**.

If you check the **Show all allowed attributes** check box, then the effective rights for those attributes appear next to the additional attributes, even though they do not have values.

### 13.7.5. Get Effective Rights Return Codes

If the criticality is not set for a get effective rights search and an error occurs, the regular entry information is returned, but, in place of rights for **entryLevelRights** and **attributeLevelRights**, an error code is returned. This code can give information on the configuration of the entry that was queried. [Table 13.8, “Returned Result Codes”](#) summarizes the error codes and the potential configuration information they can relay.

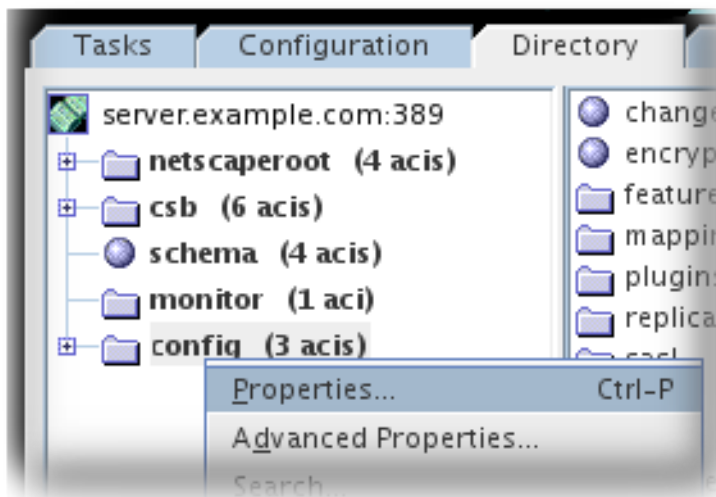
**Table 13.8. Returned Result Codes**

| Code | Description                                                                                                                                                                           |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0    | Successfully completed.                                                                                                                                                               |
| 1    | Operation error.                                                                                                                                                                      |
| 12   | The critical extension is unavailable. If the criticality expression is set to <b>true</b> and effective rights do not exist on the entry being queried, then this error is returned. |
| 16   | No such attribute. If an attribute is specifically queried for access rights but that attribute does not exist in the schema, this error is returned.                                 |
| 17   | Undefined attribute type.                                                                                                                                                             |
| 21   | Invalid attribute syntax.                                                                                                                                                             |
| 50   | Insufficient rights.                                                                                                                                                                  |
| 52   | Unavailable.                                                                                                                                                                          |
| 53   | Unwilling to perform.                                                                                                                                                                 |
| 80   | Other.                                                                                                                                                                                |

## 13.8. LOGGING ACCESS CONTROL INFORMATION

To obtain information on access control in the error logs, you must set the appropriate log level. To set the error log level from the Console:

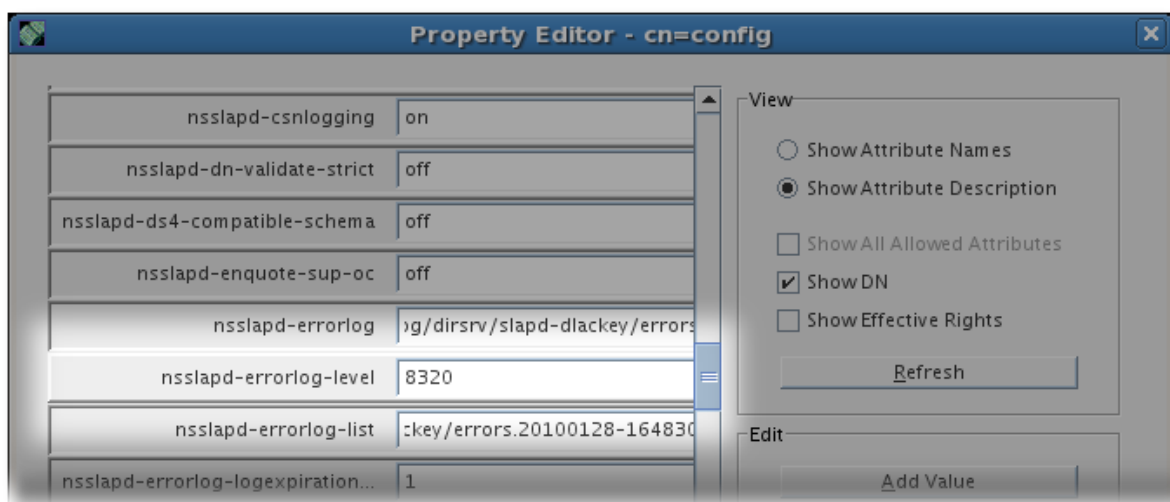
1. In the Console, click the **Directory** tab, right-click the config node, and choose **Properties** from the pop-up menu.



This displays the **Property Editor** for the **cn=config** entry.

2. Scroll down the list of attribute value pairs to locate the **nsslapd-errorlog-level** attribute.
3. Add **128** to the value already displayed in the **nsslapd-errorlog-level** value field.

For example, if the value already displayed is **8192** (replication debugging), change the value to **8320**. For complete information on error log levels, see the *Directory Server Configuration and Command-Line Tool Reference*.



4. Click **OK** to dismiss the **Property Editor**.

## 13.9. ACCESS CONTROL USAGE EXAMPLES

The examples provided in this section illustrate how an imaginary ISP company, Example Corp., would implement its access control policy. All the examples explain how to perform a given task from the Console and using an LDIF file.

Example Corp.'s business is to offer a web hosting service and Internet access. Part of Example Corp.'s web hosting service is to host the directories of client companies. Example Corp. actually hosts and partially manages the directories of two medium-sized companies, **HostedCompany1** and **HostedCompany2**. It also provides Internet access to a number of individual subscribers.

These are the access control rules that Example Corp. wants to put in place:



- Grant anonymous access for read, search, and compare to the entire **example** tree for Example Corp. employees ([Section 13.9.1, “Granting Anonymous Access”](#)).
- Grant write access to Example Corp. employees for personal information, such as **homePhone** and **homePostalAddress** ([Section 13.9.2, “Granting Write Access to Personal Entries”](#)).
- Grant Example Corp. employees the right to add any role to their entry, except certain critical roles ([Section 13.9.3, “Restricting Access to Key Roles”](#)).
- Grant the **example.com Human Resources** group all rights on the entries in the **People** branch ([Section 13.9.4, “Granting a Group Full Access to a Suffix”](#)).
- Grant all Example Corp. employees the right to create group entries under the **Social Committee** branch of the directory and to delete group entries that they own ([Section 13.9.5, “Granting Rights to Add and Delete Group Entries”](#)).
- Grant all Example Corp. employees the right to add themselves to group entries under the **Social Committee** branch of the directory ([Section 13.9.9, “Allowing Users to Add or Remove Themselves from a Group”](#)).
- Grant access to the directory administrator (role) of **HostedCompany1** and **HostedCompany2** on their respective branches of the directory tree, with certain conditions such as SSL authentication, time and date restrictions, and specified location ([Section 13.9.6, “Granting Conditional Access to a Group or Role”](#)).
- Deny individual subscribers access to the billing information in their own entries ([Section 13.9.7, “Denying Access”](#)).
- Grant anonymous access to the world to the individual subscribers subtree, except for subscribers who have specifically requested to be unlisted. (This part of the directory could be a consumer server outside of the firewall and be updated once a day.) See [Section 13.9.1, “Granting Anonymous Access”](#) and [Section 13.9.8, “Setting a Target Using Filtering”](#).

### 13.9.1. Granting Anonymous Access

Most directories are run such that you can anonymously access at least one suffix for read, search, or compare. For example, you might want to set these permissions if you are running a corporate personnel directory that you want employees to be able to search, such as a phonebook. This is the case at Example Corp. internally and is illustrated in [Section 13.9.1.1, “ACI “Anonymous example.com””](#).

As an ISP, Example Corp. also wants to advertise the contact information of all of its subscribers by creating a public phonebook accessible to the world. This is illustrated in [Section 13.9.1.2, “ACI “Anonymous World””](#).

#### 13.9.1.1. ACI “Anonymous example.com”

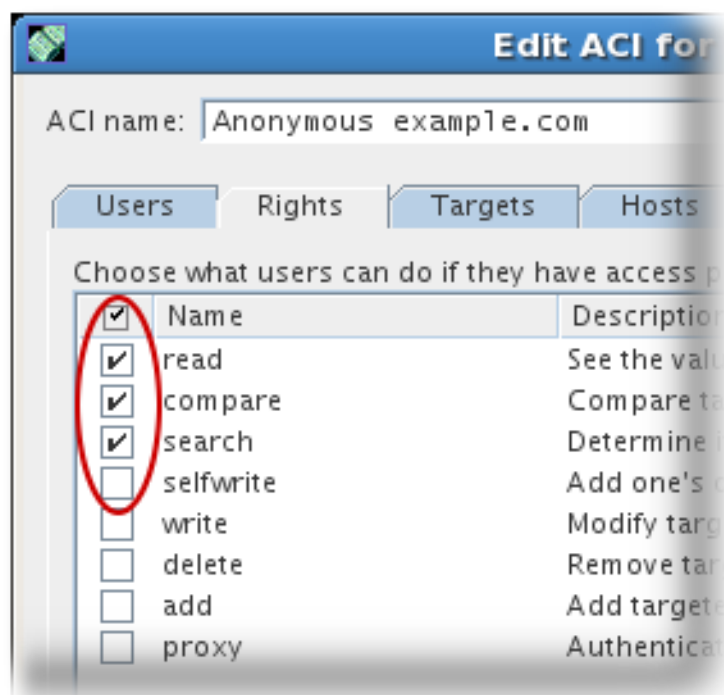
In LDIF, to grant read, search, and compare permissions to the entire Example Corp. tree to Example Corp. employees, write the following statement:

```
aci: (targetattr != "userPassword")(version 3.0; acl "Anonymous
 Example"; allow (read, search, compare) userdn= "ldap:///anyone"
 and dns="*.example.com";)
```

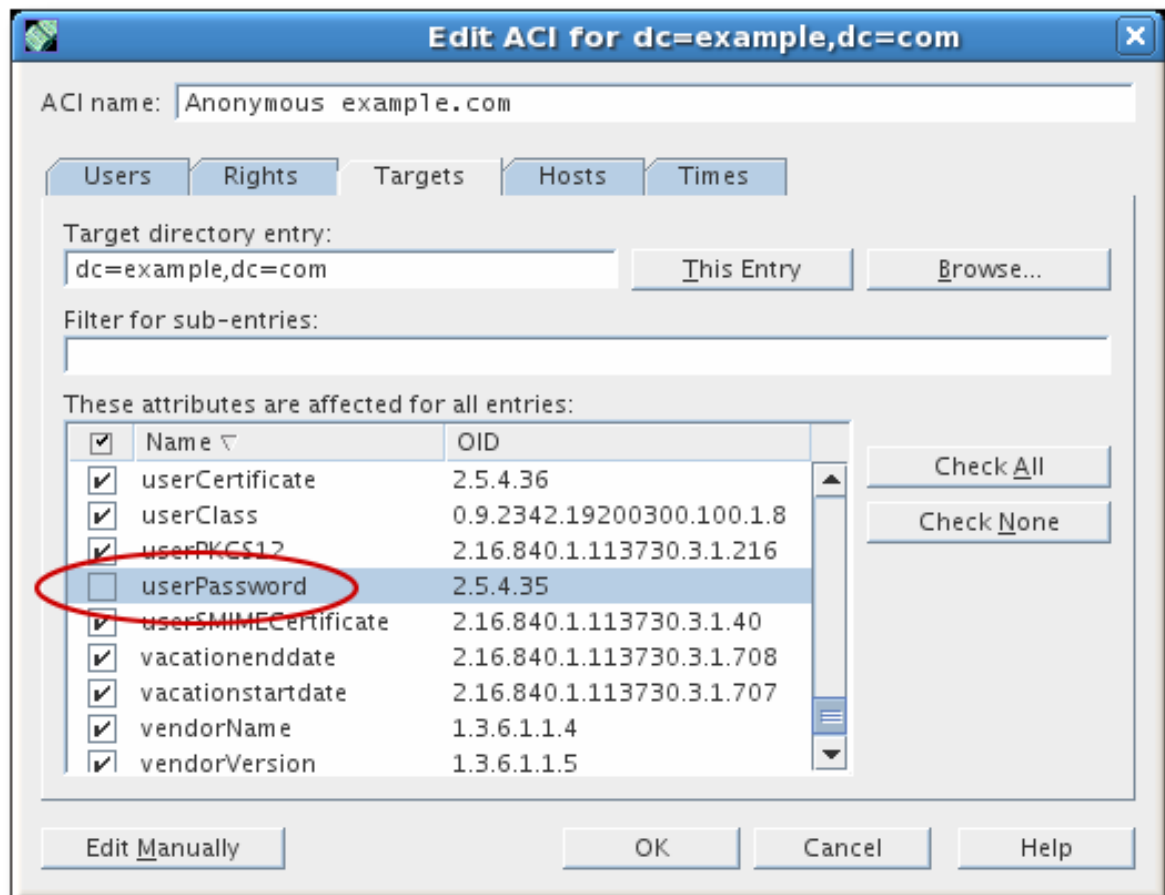
This example assumes that the **aci** attribute is added to the **dc=example,dc=com** entry. The **userPassword** attribute is excluded from the scope of the ACI.

From the Console:

1. In the **Directory** tab, right-click the **example** node in the left navigation tree, and choose **Set Access Permissions** from the pop-up menu to display the **Access Control Manager**.
2. Click **New** to display the **Access Control Editor**.
3. In the **Users/Groups** tab in the **ACI name** field, type **Anonymous example.com**. Check that **All Users** opens in the list of users granted access permission.
4. In the **Rights** tab, select the check boxes for **read**, **compare**, and **search** rights. Make sure the other check boxes are clear.



5. In the **Targets** tab, click **This Entry** to display the **dc=example,dc=com** suffix in the **Target directory entry** field. In the attribute table, locate the **userPassword** attribute, and clear the corresponding check box.



All other check boxes should be selected. This task is easier if you click the **Name** header to organize the list of attributes alphabetically.

6. In the **Hosts** tab, click **Add**, and in the **DNS host filter** field, type **\*.example.com**. Click **OK** to dismiss the dialog box.
7. Click **OK** in the **Access Control Editor** window.

### 13.9.1.2. ACI "Anonymous World"

In LDIF, to grant read and search access of the individual subscribers subtree to the world, while denying access to information on unlisted subscribers, write the following statement:

```
aci: (targetfilter= "(! (unlistedSubscriber=yes))")
 (targetattr="homePostalAddress || homePhone || mail") (version
 3.0; acl "Anonymous World"; allow (read, search)
 userdn="ldap:///anyone";)
```

This example assumes that the ACI is added to the **ou=subscribers,dc=example,dc=com** entry. It also assumes that every subscriber entry has an **unlistedSubscriber** attribute which is set to **yes** or **no**. The target definition filters out the unlisted subscribers based on the value of this attribute. For details on the filter definition, see [Section 13.9.8, "Setting a Target Using Filtering"](#).

From the Console:

1. In the **Directory** tab, right-click the **Subscribers** entry under the **example** node in the left navigation tree, and choose **Set Access Permissions** from the pop-up menu to display the **Access Control Manager**.

2. Click **New** to display the **Access Control Editor**.
3. In the **Users/Groups** tab, in the **ACI name** field, type **Anonymous World**. Check that **All Users** opens in the list of users granted access permission.
4. In the **Rights** tab, select the check boxes for **read** and **search** rights. Make sure the other check boxes are clear.
5. In the **Targets** tab, click **This Entry** to display the **ou=subscribers,dc=example,dc=com** suffix in the **Target directory entry** field.
6. In the **Filter for subentries** field, enter a filter which excludes unlisted subscribers (**(!(unlistedSubscriber=yes))**).

**Edit ACI for dc=example,dc=com**

ACI name:

Users Rights **Targets** Hosts Times

Target directory entry:

Filter for sub-entries:

These attributes are affected for all entries:

| <input checked="" type="checkbox"/> | Name ▾                 | OID                        |
|-------------------------------------|------------------------|----------------------------|
| <input type="checkbox"/>            | gidNumber              | 1.3.6.1.1.1.1.1            |
| <input type="checkbox"/>            | givenName              | 2.5.4.42                   |
| <input type="checkbox"/>            | governingStructureRule | 2.5.21.10                  |
| <input type="checkbox"/>            | hasSubordinates        | 2.5.18.9                   |
| <input type="checkbox"/>            | homeDirectory          | 1.3.6.1.1.1.1.3            |
| <input checked="" type="checkbox"/> | homePhone              | 0.9.2342.19200300.100.1.20 |
| <input checked="" type="checkbox"/> | homePostalAddress      | 0.9.2342.19200300.100.1.39 |
| <input type="checkbox"/>            | host                   | 0.9.2342.19200300.100.1.9  |
| <input type="checkbox"/>            | houseIdentifier        | 2.5.4.51                   |

7. In the attribute table, select the check boxes for the **homePhone**, **homePostalAddress**, and **mail** attributes.

All other check boxes should be clear; if it is easier, click the **Check None** button to clear the check boxes for all attributes in the table, then click the **Name** header to organize them alphabetically, and select the appropriate ones.

8. Click **OK**.

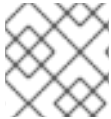
### 13.9.2. Granting Write Access to Personal Entries

Many directory administrators want to allow internal users to change some but not all of the attributes in their own entry. The directory administrators at Example Corp. want to allow users to change their own password, home telephone number, and home address, but nothing else. This is illustrated in

[Section 13.9.2.1, "ACI "Write example.com"."](#)

It is also Example Corp.'s policy to let their subscribers update their own personal information in the **example** tree, provided that they establish an SSL connection to the directory. This is illustrated in [Section 13.9.2.2, "ACI "Write Subscribers"."](#)

### 13.9.2.1. ACI "Write example.com"



#### NOTE

By setting this permission, users will also have the right to delete attribute values.

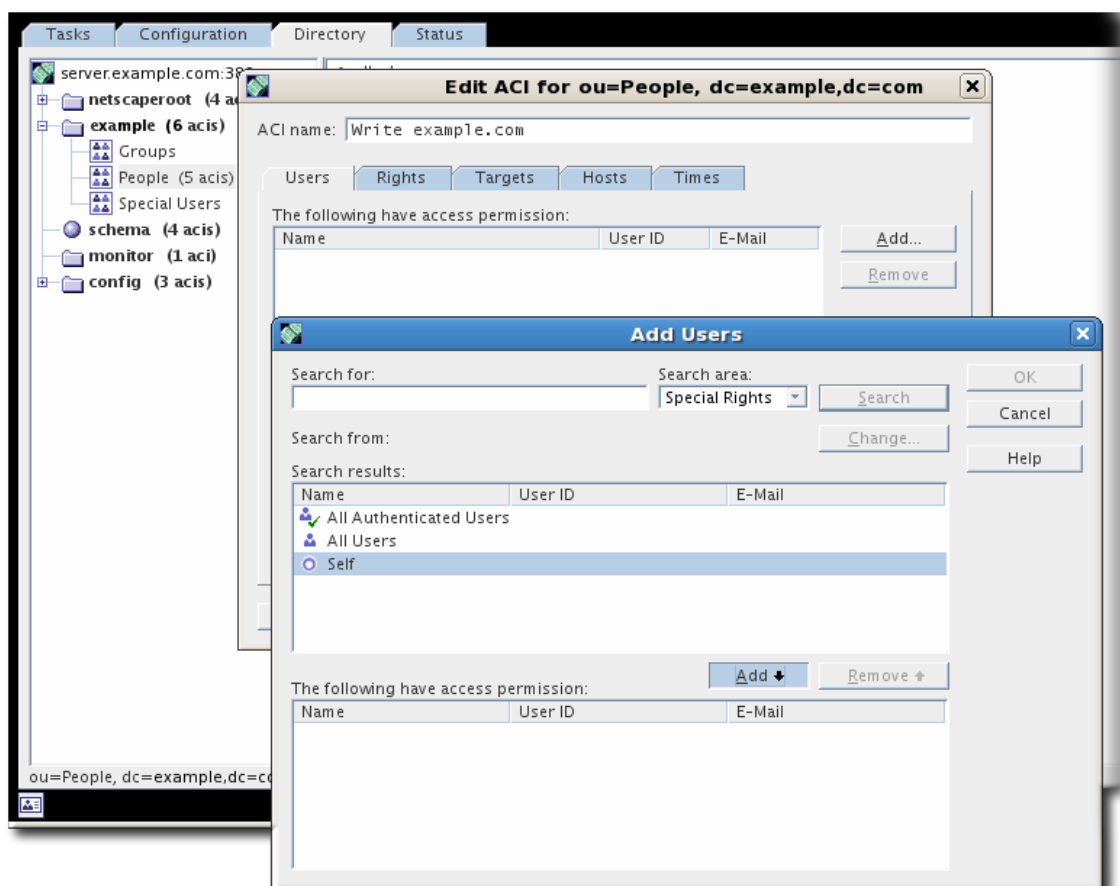
Granting Example Corp. employees the right to update their password, home telephone number, and home address has the following statement in LDIF:

```
aci: (targetattr="userPassword || homePhone ||
 homePostalAddress") (version 3.0; acl "Write example.com"; allow
 (write) userdn= "ldap:///self" and dns="*.example.com");)
```

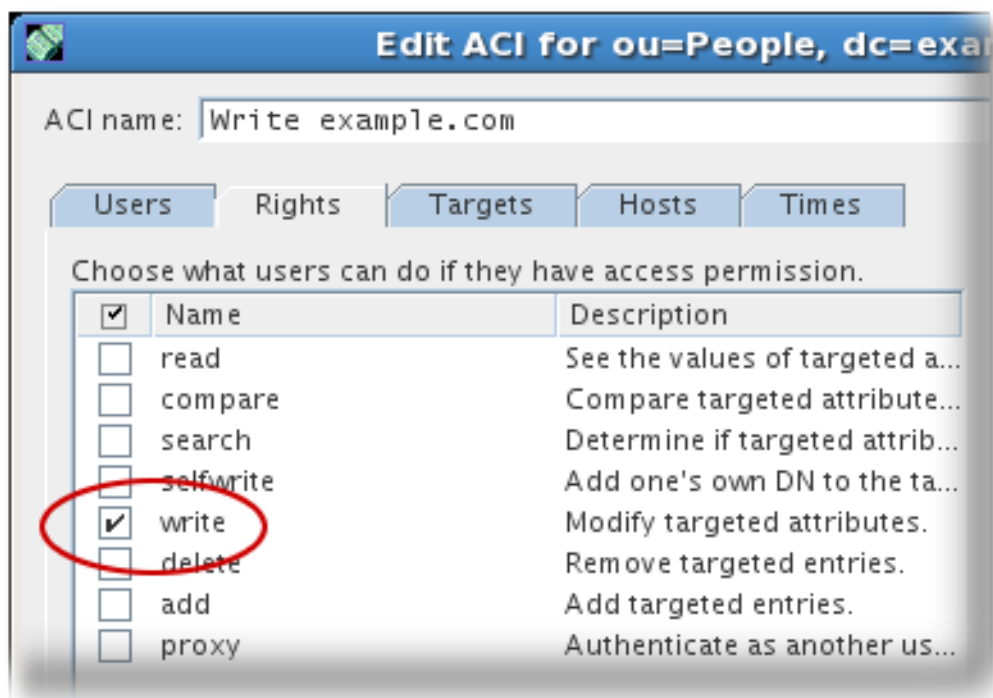
This example assumes that the ACI is added to the **ou=people, dc=example, dc=com** entry.

From the Console:

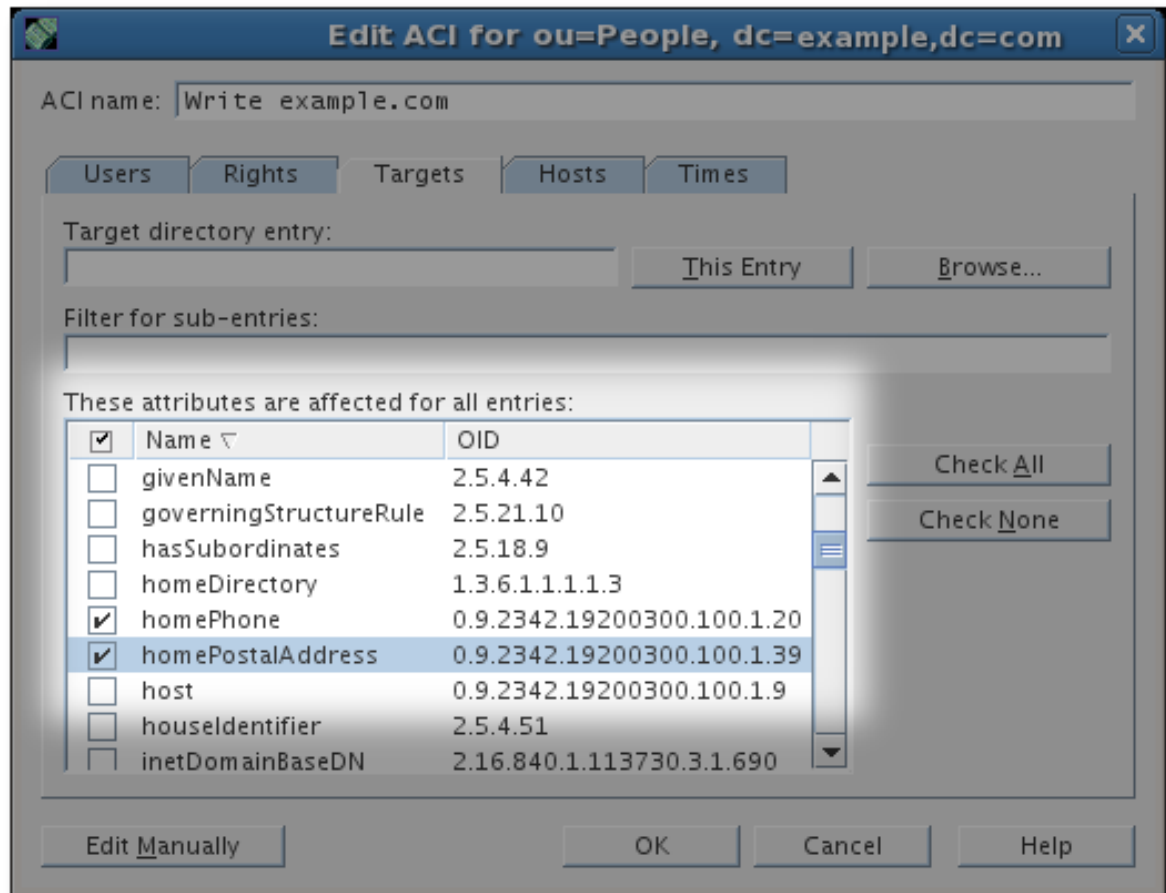
1. In the **Directory** tab, right-click the **people** entry under the **example** node in the left navigation tree, and choose **Set Access Permissions** from the pop-up menu to display the **Access Control Manager**.
2. Click **New** to display the **Access Control Editor**.
3. In the **Users/Groups** tab, in the **ACI name** field, type **Write example.com**. In the list of users granted access permission:
  1. Remove **All Users** from the targeted user list.
  2. Click **Add**.
  3. Set the **Search** area to **Special Rights**, and select **Self** from the search results list.
  4. Click the **Add** button to list **Self** in the list of users who are granted access permission.



5. Click **OK** to dismiss the **Add Users and Groups** dialog box.
4. In the **Rights** tab, select the check box for **write** right. Make sure the other check boxes are clear.

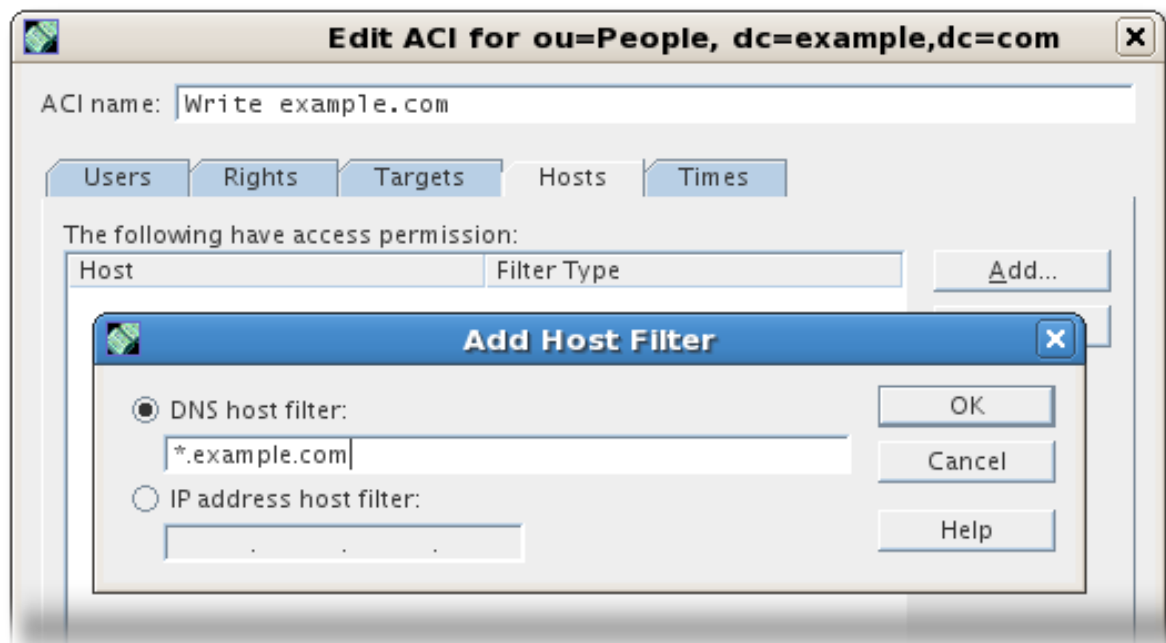


5. In the **Targets** tab, click **This Entry** to display the **ou=people, dc=example, dc=com** suffix in the **Target directory entry** field. In the attribute table, select the check boxes for the **homePhone**, **homePostalAddress**, and **userPassword** attributes.



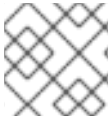
All other check boxes should be clear; if it is easier, click the **Check None** button to clear the check boxes for all attributes in the table, then click the **Name** header to organize them alphabetically, and select the appropriate ones.

6. In the **Hosts** tab, click **Add** to display the **Add Host Filter** dialog box. In the **DNS host filter** field, type **\*.example.com**. Click **OK** to dismiss the dialog box.



7. Click **OK** in the **Access Control Editor** window.

### 13.9.2.2. ACI "Write Subscribers"



#### NOTE

By setting this permission, you are also granting users the right to delete attribute values.

In LDIF, to grant Example Corp. subscribers the right to update their password and home telephone number, write the following statement:

```
aci: (targetattr = "homePhone || homePostalAddress || mail")
(target = "ldap:///ou=Subscribers,dc=example,dc=com")
(targetfilter = (!(unlistedSubscriber=yes)))
(version 3.0;
acl "Write Subscribers";
allow (write)
(userdn = "ldap:///self") and authmethod="ssl";
;)
```

This example assumes that the **aci** is added to the **ou=subscribers,dc=example,dc=com** entry.

Example Corp. subscribers do not have write access to their home address because they might delete the attribute, and Example Corp. needs that information for billing. Therefore, the home address is business-critical information.

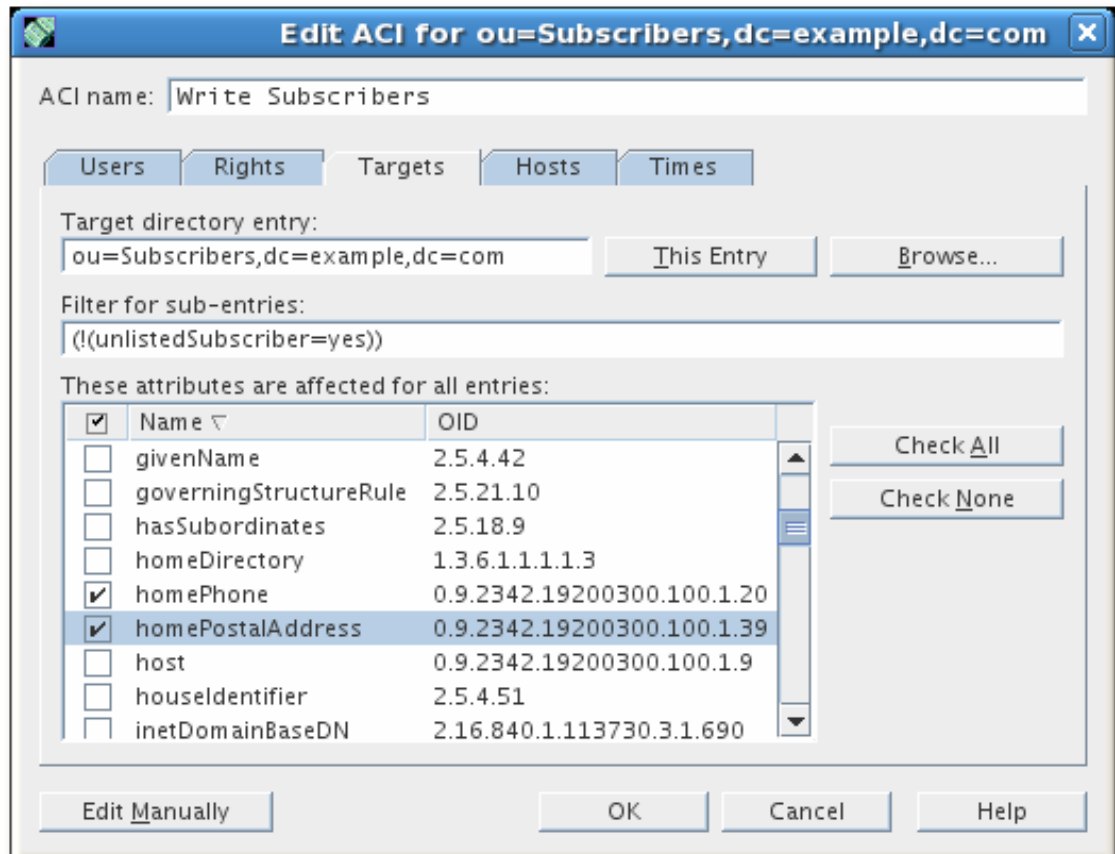
From the Console:

1. In the **Directory** tab, right-click the **subscribers** entry under the **example** node in the left navigation tree, and choose **Set Access Permissions** from the pop-up menu to display the **Access Control Manager**.
2. Click **New** to display the **Access Control Editor**.
3. In the **Users/Groups** tab, in the **ACI name** field, type **Write Subscribers**. In the list of users granted access permission:
  1. Remove **All Users** from the targeted user list.
  2. Click **Add**.
  3. Set the **Search** area to **Special Rights**, and select **Self** from the search results list.
  4. Click the **Add** button to list **Self** in the list of users who are granted access permission.
  5. Click **OK** to dismiss the **Add Users and Groups** dialog box.
4. In the **Rights** tab, select the check box for **write**. Make sure the other check boxes are clear.
5. In the **Targets** tab, click **This Entry** to display the **ou=subscribers,dc=example,dc=com** suffix in the **Target directory entry** field.
  1. In the **Filter for subentries** field, type a filter so that only listed subscribers are included:

```
(!(unlistedSubscriber=yes))
```



2. In the attribute table, select the check boxes for the **homePhone**, **homePostalAddress**, and **mail** attributes.



All other check boxes should be clear; if necessary, click the **Check None** button to clear the check boxes for all attributes in the table, then click the **Name** header to organize them alphabetically, and select the appropriate ones.

3. Optionally, to require users to authenticate using SSL, switch to manual editing by clicking the **Edit Manually** button, and add **authmethod=ssl** to the LDIF statement:

```
(targetattr = "homePhone || homePostalAddress || mail")
(target = "ldap:///ou=Subscribers,dc=example,dc=com")
(targetfilter = (!(unlistedSubscriber=yes)))
(version 3.0;
acl "Write Subscribers";
allow (write)
(userdn = "ldap:///self") and authmethod="ssl";
);
```

6. Click **OK**.

### 13.9.3. Restricting Access to Key Roles

You can use role definitions in the directory to identify functions that are critical to your business, the administration of your network and directory, or another purpose.

For example, you might create a **superAdmin** role by identifying a subset of your system administrators that are available at a particular time of day and day of the week at corporate sites worldwide, or you might want to create a **First Aid** role that includes all members of staff on a particular site that have

done first aid training. For information on creating role definitions, see [Section 6.2, "Using Roles"](#).

When a role gives any sort of privileged user rights over critical corporate or business functions, consider restricting access to that role. For example, at Example Corp., employees can add any role to their own entry except the **superAdmin** role. This is illustrated in [Section 13.9.3.1, "ACI "Roles"'"](#).

### 13.9.3.1. ACI "Roles"

In LDIF, to grant Example Corp. employees the right to add any role to their own entry except the **superAdmin** role, write the following statement:

```
aci: (targetattr = "nsroledn")
 (targetattrfilters="add=nsroledn:(nsroledn !=
 "cn=superAdmin,dc=example,dc=com")") (version 3.0; acl "Roles";
 allow (write) userdn= "ldap:///self" and dns="*.example.com";)
```

This example assumes that the ACI is added to the **ou=people,dc=example,dc=com** entry.

From the Console:

1. In the **Directory** tab, right-click the **people** entry under the **example** node in the left navigation tree, and choose **Set Access Permissions** from the pop-up menu to display the **Access Control Manager**.
2. Click **New** to display the **Access Control Editor**.
3. In the **Users/Groups** tab, in the **ACI name** field, type **Roles**. In the list of users granted access permission:
  1. Remove **All Users** from the targeted user list.
  2. Click **Add**.
  3. Set the **Search** area in the **Add Users and Groups** dialog box to **Special Rights**, and select **Self** from the search results list.
  4. Click the **Add** button to list **Self** in the list of users who are granted access permission.
  5. Click **OK** to dismiss the **Add Users and Groups** dialog box.
4. In the **Rights** tab, select the check box for **write**. Make sure the other check boxes are clear.
5. In the **Targets** tab, click **This Entry** to use the **ou=people,dc=example,dc=com** suffix in the **Target directory entry** field.
6. In the **Hosts** tab, click **Add** to display the **Add Host Filter** dialog box. In the **DNS host filter** field, type **\*.example.com**. Click **OK** to dismiss the dialog box.
7. To create the value-based filter for roles, switch to manual editing by clicking the **Edit Manually** button. Add the following to the beginning of the LDIF statement:

```
(targetattrfilters="add=nsroledn:(nsroledn !=
"cn=superAdmin,dc=example,dc=com")")
```

The LDIF statement should read as follows:

```
(targetattrfilters="add=nsroledn:(nsroledn != "cn=superAdmin,
dc=example,dc=com")") (targetattr = "*") (target = "ldap:///
ou=people,dc=example,dc=com") (version 3.0; acl "Roles";
allow (write) (userdn = "ldap:///self") and
(dns="*.example.com"));
```

8. Click **OK**.

### 13.9.4. Granting a Group Full Access to a Suffix

Most directories have a group that is used to identify certain corporate functions. These groups can be given full access to all or part of the directory. By applying the access rights to the group, you can avoid setting the access rights for each member individually. Instead, you grant users these access rights simply by adding them to the group.

For example, when the Directory Server is set up with a typical process, an administrators group with full access to the directory is created by default.

At Example Corp., the **Human Resources** group is allowed full access to the **ou=people** branch of the directory so that they can update the employee database. This is illustrated in [Section 13.9.4.1, “ACI “HR””](#).

#### 13.9.4.1. ACI “HR”

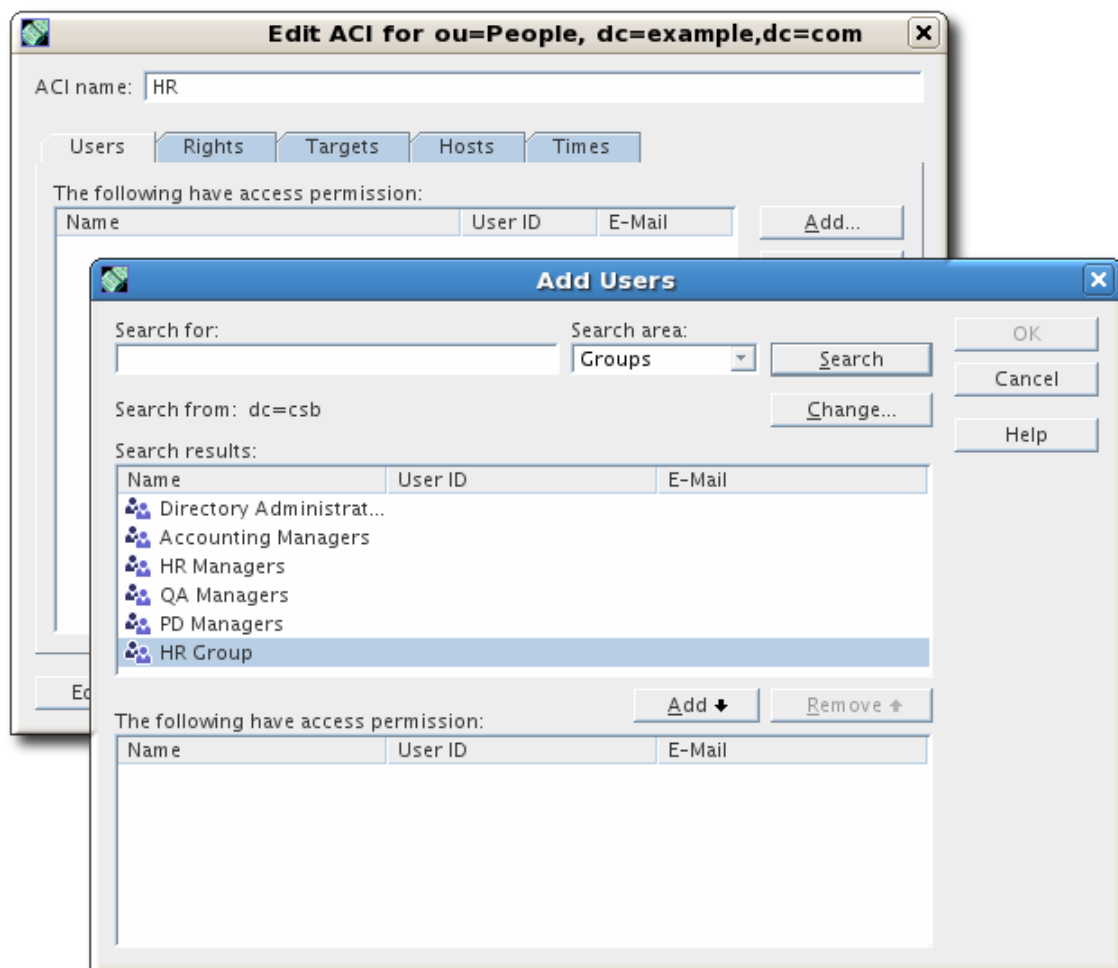
In LDIF, to grant the HR group all rights on the employee branch of the directory, use the following statement:

```
aci: (version 3.0; acl "HR"; allow (all) userdn=
"ldap:///cn=HRgroup,ou=people,dc=example,dc=com");)
```

This example assumes that the ACI is added to the **ou=people,dc=example,dc=com** entry.

From the Console:

1. In the **Directory** tab, right-click the **people** entry under the **example** node in the left navigation tree, and choose **Set Access Permissions** from the pop-up menu to display the **Access Control Manager**.
2. Click **New** to display the **Access Control Editor**.
3. In the **Users/Groups** tab, in the **ACI name** field, type **HR**. In the list of users granted access permission:
  1. Remove **All Users** from the targeted user list.
  2. Click **Add**.
  3. Set the **Search** area to **Users and Groups**, and type **HRgroup** in the **Search for** field.



This example assumes that you have created an HR group or role. For more information on groups and roles, see [Chapter 6, Organizing and Grouping Entries](#).

4. Click the **Add** button to list the HR group in the list of users who are granted access permission.
5. Click **OK** to dismiss the **Add Users and Groups** dialog box.
4. In the **Rights** tab, click the **Check All** button.

All check boxes are selected, except for proxy rights.



5. Click **OK**.

### 13.9.5. Granting Rights to Add and Delete Group Entries

Some organizations want to allow employees to create entries in the tree if it can increase their efficiency or if it can contribute to the corporate dynamics.

At Example Corp., there is an active social committee that is organized into various clubs, such as tennis, swimming, and skiing. Any Example Corp. employee can create a group entry representing a new club. This is illustrated in [Section 13.9.5.1, "ACI "Create Group"'](#). Any Example Corp. employee can become a member of one of these groups. This is illustrated in [Section 13.9.9.1, "ACI "Group Members"'](#) under [Section 13.9.9, "Allowing Users to Add or Remove Themselves from a Group"](#). Only the group owner can modify or delete a group entry. This is illustrated in [Section 13.9.5.2, "ACI "Delete Group"'](#).

#### 13.9.5.1. ACI "Create Group"

In LDIF, to grant Example Corp. employees the right to create a group entry under the **ou=Social Committee** branch, write the following statement:

```
aci: (target="ldap:///ou=social committee,dc=example,dc=com)
(targetattrfilters="add=objectClass:(objectClass=groupOfNames)")
(version 3.0; acl "Create Group"; allow (add)
(userdn= "ldap:///uid=*,ou=people,dc=example,dc=com")
and dns="*.example.com";)
```



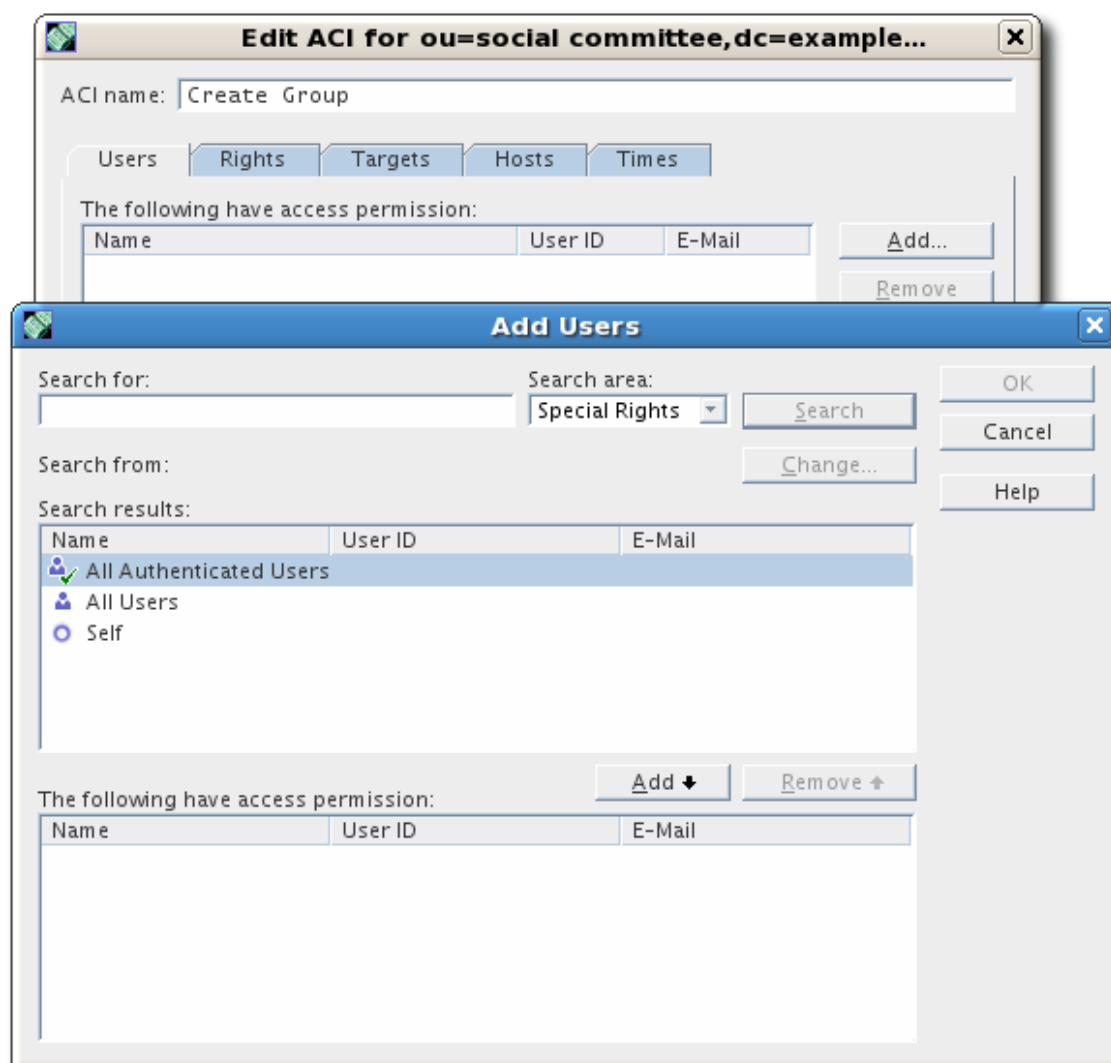
#### NOTE

This ACI does not grant write permission, which means that the entry creator cannot modify the entry.

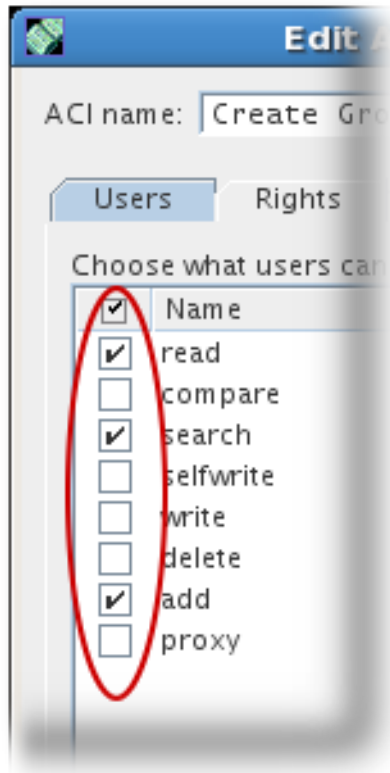
This example assumes that the ACI is added to the **ou=social committee,dc=example,dc=com** entry.

From the Console:

1. In the **Directory** tab, right-click the **Social Committee** entry under the **example** node in the left navigation tree, and choose **Set Access Permissions** from the pop-up menu to display the **Access Control Manager**.
2. Click **New** to display the **Access Control Editor**.
3. In the **Users/Groups** tab, in the **ACI name** field, type **Create Group**. In the list of users granted access permission:
  1. Remove **All Users** from the targeted user list.
  2. Click **Add**.
  3. Set the **Search** area to **Special Rights**, and select **All Authenticated Users** from the search results list.
  4. Click the **Add** button to list **All Authenticated Users** in the list of users who are granted access permission.



5. Click **OK** to dismiss the **Add Users and Groups** dialog box.
4. In the **Rights** tab, select the check boxes for add, search, and read. Make sure the other check boxes are clear.



5. In the **Targets** tab, click **This Entry** to display the **ou=social committee,dc=example,dc=com** suffix in the **Target directory entry** field.
6. In the **Hosts** tab, click **Add** to display the **Add Host Filter** dialog box. In the **DNS host filter** field, type **\*.example.com**. Click **OK** to dismiss the dialog box.
7. To create the value-based filter that allows employees to add only group entries to this subtree, click the **Edit Manually** button. Add the following to the beginning of the LDIF statement:

```
(targetattrfilters="add=objectClass:(objectClass=groupOfNames)")
```

The LDIF statement should read as follows:

```
(targetattrfilters="add=objectClass:(objectClass=groupOfNames)")
(targetattr = "*") (target="ldap:///ou=social
committee,dc=example,dc=com)
(version 3.0; acl "Create Group"; allow (read,search,add)
(userdn= "ldap:///all") and (dns="*.example.com");)
```

8. Click **OK**.

### 13.9.5.2. ACI "Delete Group"

In LDIF, to grant Example Corp. employees the right to modify or delete a group entry which they own under the **ou=Social Committee** branch, write the following statement:

```
aci: (target="ou=social committee,dc=example,dc=com)
(targetattrfilters="del=objectClass:(objectClass=groupOfNames)")
(version 3.0; acl "Delete Group"; allow (delete) userattr=
"owner#GROUPDN";)
```

This example assumes that the **aci** is added to the **ou=social committee,dc=example,dc=com** entry.



## NOTE

Using the Console is not an effective way of creating this ACI because it requires manually editing the ACI to create the target filter and to check group ownership.

### 13.9.6. Granting Conditional Access to a Group or Role

In many cases, when you grant a group or role privileged access to the directory, you want to ensure that those privileges are protected from intruders trying to impersonate your privileged users. Therefore, in many cases, access control rules that grant critical access to a group or role are often associated with a number of conditions.

Example Corp. has created a directory administrator role for each of its hosted companies, **HostedCompany1** and **HostedCompany2**. It wants these companies to be able to manage their own data and implement their own access control rules while securing it against intruders. For this reason, **HostedCompany1** and **HostedCompany2** have full rights on their respective branches of the directory tree, provided the following conditions are fulfilled:

- Connection authenticated using SSL
- Access requested between 8 a.m. and 6 p.m., Monday through Thursday
- Access requested from a specified IP address for each company

These conditions are illustrated in a single ACI for each company, **HostedCompany1** and **HostedCompany2**. Because the content of these ACIs is the same, the examples below illustrate the **HostedCompany1** ACI only.

#### 13.9.6.1. ACI "HostedCompany1"

In LDIF, to grant **HostedCompany1** full access to their own branch of the directory under the requisite conditions, write the following statement:

```
aci:(target="ou=HostedCompany1,ou=corporate-clients,dc=example,dc=com")
 (targetattr= "*") (version 3.0; acl "HostedCompany1";allow (all)
 (roledn="ldap:///cn=DirectoryAdmin,ou=HostedCompany1,
 ou=corporate-clients,dc=example,dc=com") and
 (authmethod="ssl") and (dayofweek="Mon,Tues,Wed,Thu") and (timeofday
 >= "0800" and
 timeofday <= "1800") and (ip="255.255.123.234"));)
```

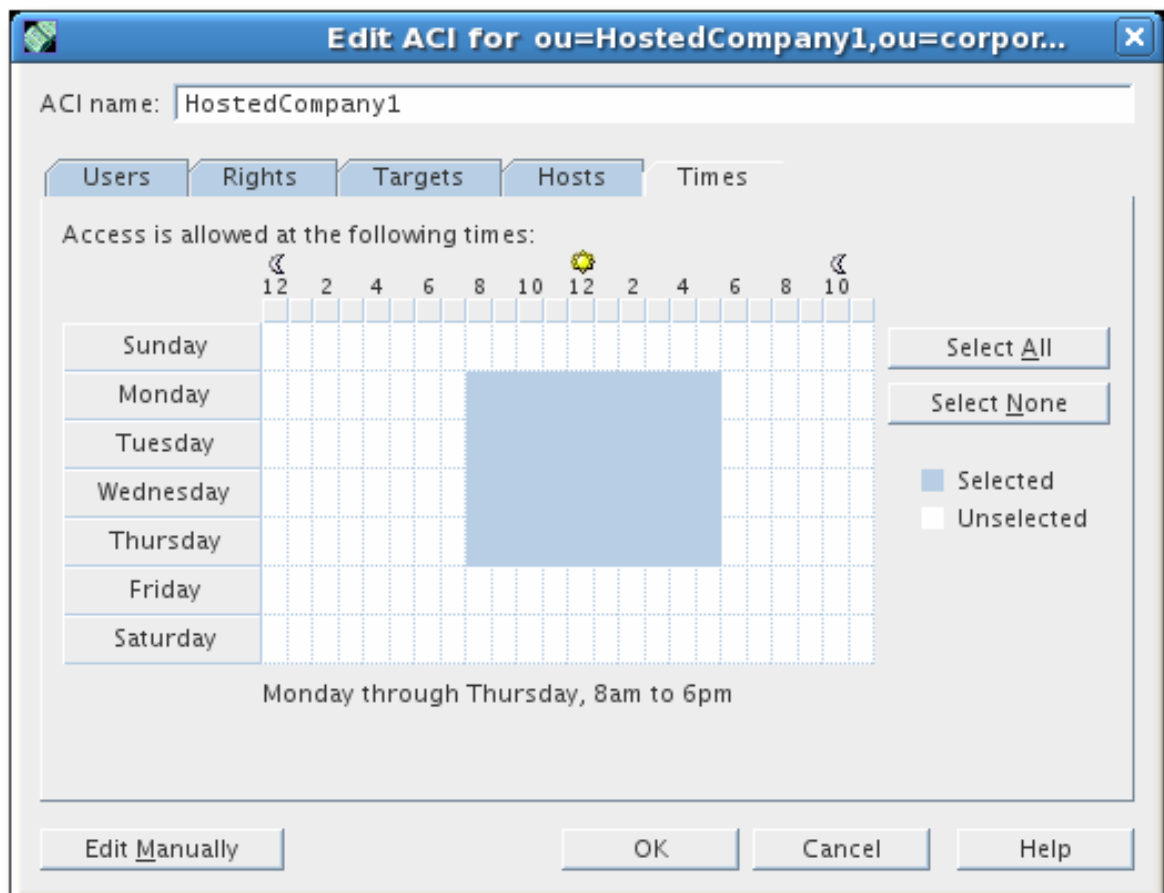
This example assumes that the ACI is added to the **ou=HostedCompany1,ou=corporate-clients,dc=example,dc=com** entry.

From the Console:

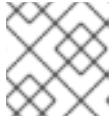
1. In the **Directory** tab, right-click the **HostedCompany1** entry under the **example** node in the left navigation tree, and choose **Set Access Permissions** from the pop-up menu to display the **Access Control Manager**.
2. Click **New** to display the **Access Control Editor**.



3. In the **Users/Groups** tab, type **HostedCompany1** in the **ACI name** field. In the list of users granted access permission:
  1. Remove **All Users** from the targeted user list.
  2. Click **Add**.
  3. Set the **Search** area to **Users and Groups**, and type **DirectoryAdmin** in the **Search For** field.  
  
(This assumes that there is an administrator's role with a **cn** of **DirectoryAdmin**.)
  4. Click the **Add** button to list the administrator's role in the list of users who are granted access permission.
  5. Click **OK** to dismiss the **Add Users and Groups** dialog box.
4. In the **Rights** tab, click the **Check All** button.
5. In the **Targets** tab, click **This Entry** to display the **ou=HostedCompany1,ou=corporate-clients,dc=example,dc=com** suffix in the **Target directory entry** field.
6. In the **Hosts** tab, click **Add** to display the **Add Host Filter** dialog box. In the **IP address host filter** field, type **255.255.123.234**. Click **OK**.

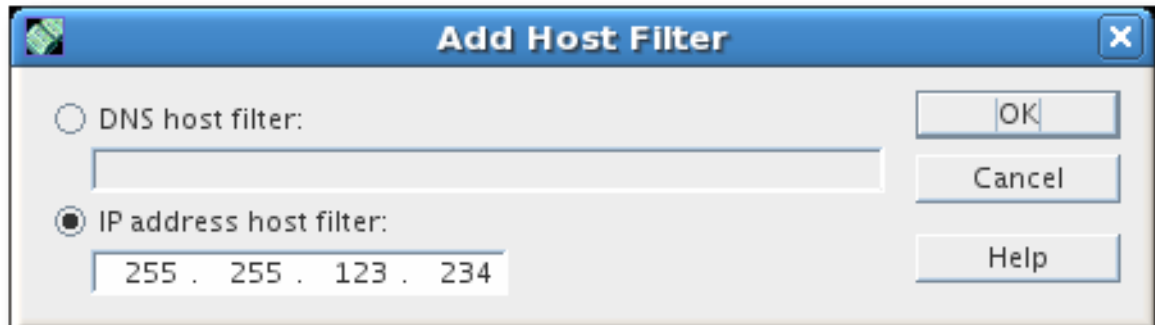


The IP address must be a valid IP address for the host machine that the **HostedCompany1** administrators use to connect to the **example** directory.

**NOTE**

Directory Server supports both IPv4 and IPv6 IP addresses.

7. In the **Times** tab, select the block time corresponding to Monday through Thursday and 8 a.m. to 6 p.m.



A message appears below the table that specifies the selected time block.

8. To enforce SSL authentication from **HostedCompany1** administrators, switch to manual editing by clicking the **Edit Manually** button. Add the following to the end of the LDIF statement:

```
and (authmethod="ssl")
```

The LDIF statement should be similar to the following:

```
(targetattr = "*") (target="ou=HostedCompany1,ou=corporate-
clients,dc=example,dc=com")
 (version 3.0; acl "HostedCompany1"; allow (all) (roledn=
 "ldap:///cn=DirectoryAdmin,ou=HostedCompany1,ou=corporate-
clients,dc=example,dc=com") and
 (dayofweek="Mon,Tues,Wed,Thu") and (timeofday >= "0800" and
timeofday <= "1800") and
 (ip="255.255.123.234") and (authmethod="ssl");)
```

9. Click **OK**.

### 13.9.7. Denying Access

If your directory holds business-critical information, it may be necessary to specifically deny access to it.

For example, Example Corp. wants all subscribers to be able to read billing information such as connection time or account balance under their own entries but explicitly wants to deny write access to that information. This is illustrated in [Section 13.9.7.1, “ACI “Billing Info Read”](#) and [Section 13.9.7.2, “ACI “Billing Info Deny”](#)”, respectively.

#### 13.9.7.1. ACI “Billing Info Read”

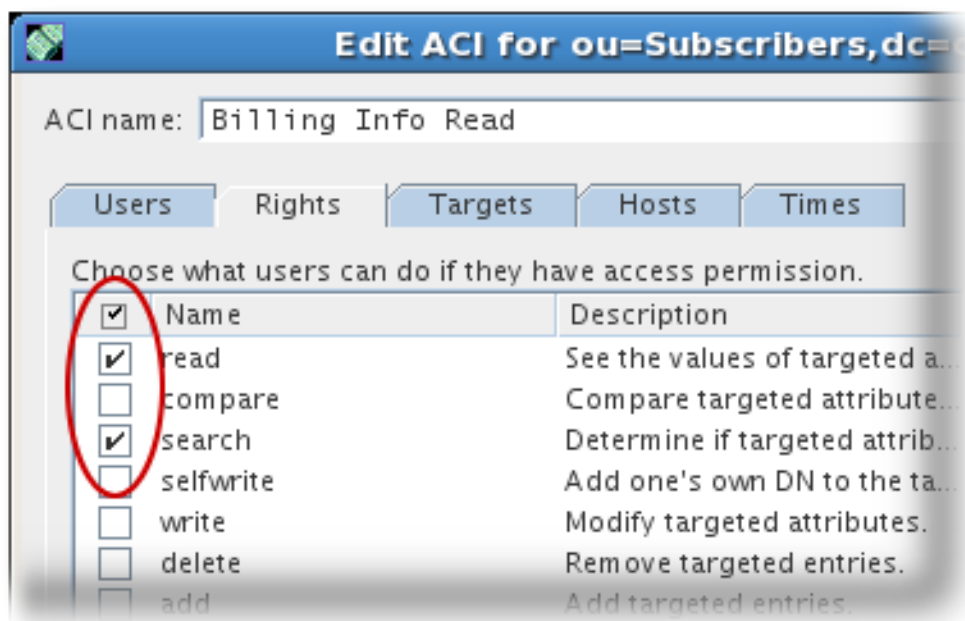
In LDIF, to grant subscribers permission to read billing information in their own entry, write the following statement:

```
aci: (targetattr="connectionTime || accountBalance") (version
 3.0; acl "Billing Info Read"; allow (search,read) userdn=
 "ldap:///self");
```

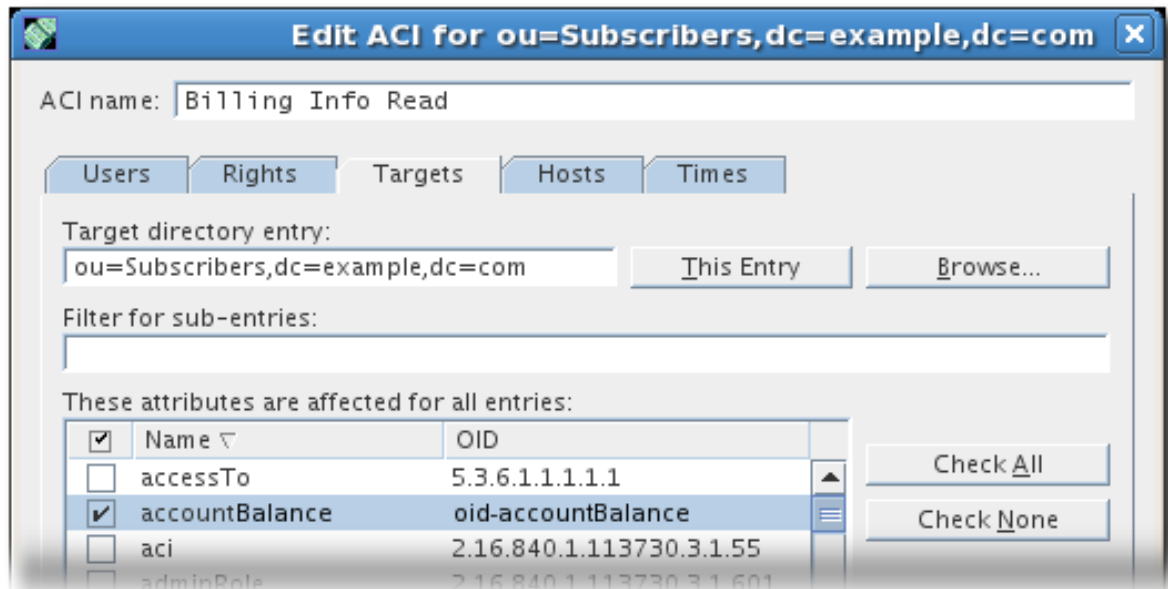
This example assumes that the relevant attributes have been created in the schema and that the ACI is added to the **ou=subscribers,dc=example,dc=com** entry.

From the Console:

1. In the **Directory** tab, right-click the **Subscribers** entry under the **example** node in the left navigation tree, and choose **Set Access Permissions** from the pop-up menu to display the **Access Control Manager**.
2. Click **New** to display the **Access Control Editor**.
3. In the **Users/Groups** tab, in the **ACI name** field, type **Billing Info Read**. In the list of users granted access permission:
  1. Remove **All Users** from the targeted user list.
  2. Click **Add**.
  3. Set the **Search** area in the **Add Users and Groups** dialog box to **Special Rights**, and select **Self** from the search results list.
  4. Click the **Add** button to list **Self** in the list of users who are granted access permission.
  5. Click **OK** to dismiss the **Add Users and Groups** dialog box.
4. In the **Rights** tab, select the check boxes for **search** and **read** rights. Make sure the other check boxes are clear.



5. In the **Targets** tab, click **This Entry** to display the **ou=subscribers,dc=example,dc=com** suffix in the **Target directory entry** field. In the attribute table, select the check boxes for the **connectionTime** and **accountBalance** attributes. (These are custom schema that Example Corp. uses for ISP account management.)



All other check boxes should be clear; if it is easier, click the **Check None** button to clear the check boxes for all attributes in the table, then click the **Name** header to organize them alphabetically, and select the appropriate ones.

This example assumes that you have added the *connectionTime* and *accountBalance* attributes to the schema.

6. Click **OK**.

### 13.9.7.2. ACI "Billing Info Deny"

In LDIF, to deny subscribers permission to modify billing information in their own entry, write the following statement:

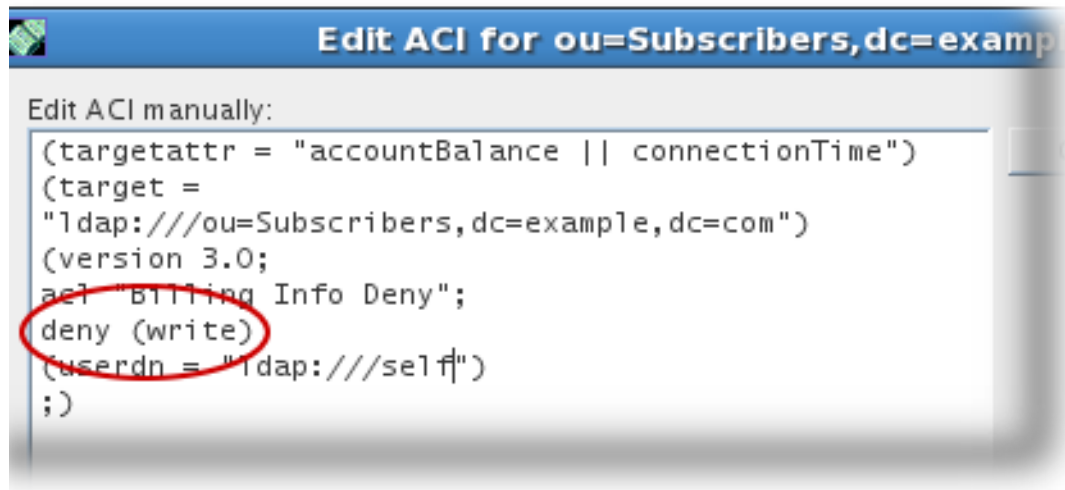
```
aci: (targetattr="connectionTime || accountBalance") (version
 3.0; acl "Billing Info Deny"; deny (write) userdn="ldap:///self";)
```

This example assumes that the relevant attributes have been created in the schema and that the ACI is added to the **ou=subscribers,dc=example,dc=com** entry.

From the Console:

1. In the **Directory** tab, right-click the **Subscribers** entry under the **example** node in the left navigation tree, and choose **Set Access Permissions** from the pop-up menu to display the **Access Control Manager**.
2. Click **New** to display the **Access Control Editor**.
3. In the **Users/Groups** tab, in the **ACI name** field, type **Billing Info Deny**. In the list of users granted access permission:
  1. Remove **All Users** from the targeted user list.
  2. Click **Add**.

3. Set the **Search** area in the **Add Users and Groups** dialog box to **Special Rights**, and select **Self** from the search results list.
4. Click the **Add** button to list **Self** in the list of users who are granted access permission.
5. Click **OK** to dismiss the **Add Users and Groups** dialog box.
4. In the **Rights** tab, select the check box for **write**. Make sure the other check boxes are clear.
5. Click the **Edit Manually** button, and, in the LDIF statement that opens, change the word **allow** to **deny**.



6. In the **Targets** tab, click **This Entry** to display the **ou=subscribers,dc=example,dc=com** suffix in the **Target directory entry** field. In the attribute table, select the check boxes for the **connectionTime** and **accountBalance** attributes.

All other check boxes should be clear; if it is easier, click the **Check None** button to clear the check boxes for all attributes in the table, then click the **Name** header to organize them alphabetically, and select the appropriate ones.

This example assumes that the **connectionTime** and **accountBalance** attributes were added to the schema.

7. Click **OK**.

### 13.9.8. Setting a Target Using Filtering

To set access controls that allow access to a number of entries that are spread across the directory, consider using a filter to set the target.



#### NOTE

Because search filters do not directly name the object for which you are managing access, it is easy to allow or deny access to the wrong objects unintentionally, especially as your directory becomes more complex. Additionally, filters can make it difficult to troubleshoot access control problems within your directory.

For example, the following ACI grants user **bjensen** write access to the department number, home phone number, home postal address, and manager attributes for all members of the accounting organization.

```
aci: (targetattr="departmentNumber || homePhone || homePostalAddress ||
manager")
(targetfilter="(uid=bjensen)") (version 3.0; acl "Filtered ACL"; allow
(write)
userdn = "ldap:///cn=*,ou=accounting,dc=example,dc=com";)
```

Before you can set these permissions, you must create the accounting branch point **ou=accounting,dc=example,dc=com**. You can create organizational unit branch points in the **Directory** tab on the Directory Server Console.

### 13.9.9. Allowing Users to Add or Remove Themselves from a Group

Many directories set ACIs that allow users to add or remove themselves from groups. This is useful, for example, for allowing users to add and remove themselves from mailing lists.

At Example Corp., employees can add themselves to any group entry under the **ou=social committee** subtree. This is illustrated in [Section 13.9.9.1, “ACI “Group Members””](#).

#### 13.9.9.1. ACI “Group Members”

In LDIF, to grant Example Corp. employees the right to add or delete themselves from a group, write the following statement:

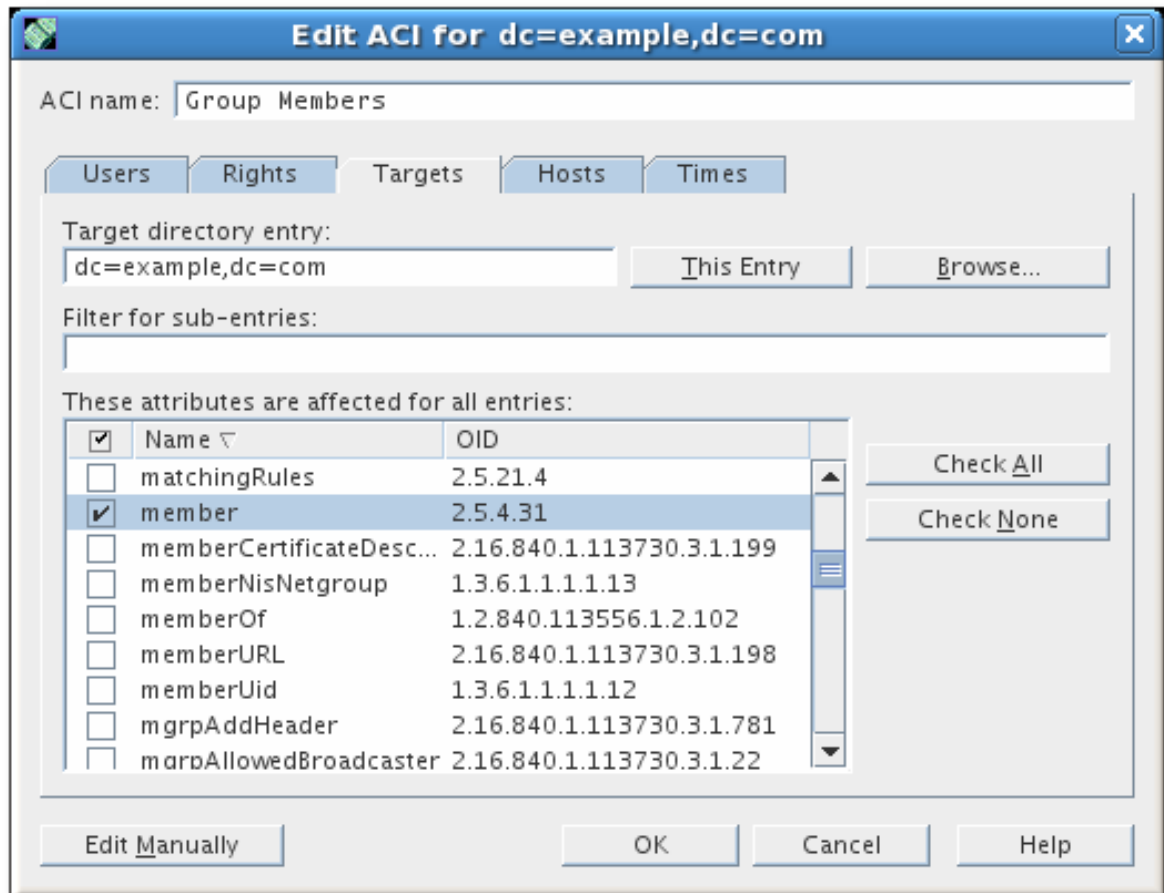
```
aci: (targetattr="member")(version 3.0; acl "Group Members"; allow
(selfwrite)
(userdn= "ldap:///uid=*,ou=people,dc=example,dc=com") ;)
```

This example assumes that the ACI is added to the **ou=social committee,dc=example,dc=com** entry.

From the Console:

1. In the **Directory** tab, right-click the **people** entry under the **example** node in the left navigation tree, and choose **Set Access Permissions** from the pop-up menu to display the **Access Control Manager**.
2. Click **New** to display the **Access Control Editor**.
3. In the **Users/Groups** tab, in the **ACI name** field, type **Group Members**. In the list of users granted access permission:
  1. Remove **All Users** from the targeted user list.
  2. Click **Add**.
  3. Set the **Search** area in the **Add Users and Groups** dialog box to **Special Rights**, and select **All Authenticated Users** from the search results list.
  4. Click the **Add** button to list **All Authenticated Users** in the list of users who are granted access permission.

5. Click **OK** to dismiss the **Add Users and Groups** dialog box.
4. In the **Rights** tab, select the check box for **selfwrite**. Make sure the other check boxes are clear.
5. In the **Targets** tab, type **dc=example,dc=com** suffix in the **Target directory entry** field. In the attribute table, select the check box for the **member** attribute.



All other check boxes should be clear; if it is easier, click the **Check None** button to clear the check boxes for all attributes in the table, then click the **Name** header to organize them alphabetically, and select the appropriate ones.

6. Click **OK**.

### 13.9.10. Setting an ACI to Require a Certain Security Strength Factor for Some Operations

As mentioned in [Section 13.4.8, “Requiring a Certain Level of Security in Connections”](#), the **ssf** keyword is used to require a secure connection and at a specific level of security. This can be a way to make changes to sensitive information only over a secure connection, like requiring that password changes be made over TLS or SASL:

```
aci: (targetattr="userPassword")(version 3.0; acl "Require secure password changes"; allow (write) userdn="ldap:///self" and ssf>="56";)
```

### 13.9.11. Defining Permissions for DNs That Contain a Comma

DNs that contain commas require special treatment within your LDIF ACI statements. In the target and bind rule portions of the ACI statement, commas must be escaped by a single backslash (\). For example:

```
dn: dc=example.com Bolivia\, S.A.,dc=com
objectClass: top
objectClass: organization
aci: (target="ldap:///dc=example.com Bolivia\,S.A.,dc=com")(targetattr=*)
 (version 3.0; acl "aci 2"; allow (all)
 groupdn = "ldap:///cn=Directory Administrators,dc=example.com
 Bolivia\, S.A.,dc=com";)
```

### 13.9.12. Proxied Authorization ACI Example

Proxied authorization allows one user to bind and perform operation as another user. For example, Example Corp. has an accounting program which must be able to bind to the directory as an accounting administrator in order to write data. This authorization assumes three things:

- The client application's bind DN is **"uid=MoneyWizAcctSoftware,ou=Applications,dc=example,dc=com"**.
- The targeted subtree to which the client application is requesting access is **ou=Accounting,dc=example,dc=com**.
- An accounting administrator with access permissions to the **ou=Accounting,dc=example,dc=com** subtree exists in the directory.

In order for the client application to gain access to the accounting subtree, using the same access permissions as the accounting administrator, two ACIs must be set:

- The accounting administrator must have access permissions to the **ou=Accounting,dc=example,dc=com** subtree, so the following ACI grants all rights to the accounting administrator entry:

```
aci: (target="ldap:///ou=Accounting,dc=example,dc=com")
 (targetattr="*")
 (version 3.0; acl "allowAll-AcctAdmin"; allow (all)

userdn="ldap://uid=AcctAdministrator,ou=Administrators,dc=example,dc=com")
```

- There must be an ACI granting proxy rights to the client application in the directory:

```
aci: (target="ldap:///ou=Accounting,dc=example,dc=com")
 (targetattr="*")
 (version 3.0; acl "allow proxy-accounting software"; allow
 (proxy)

userdn="ldap://uid=MoneyWizAcctSoftware,ou=Applications,dc=example,dc=com")
```

With this ACI in place, the **MoneyWizAcctSoftware** client application can bind to the directory and send an LDAP command such as **ldapsearch** or **ldapmodify** that requires the access rights of the proxy DN.



If the client performs an **ldapsearch** command, the command must include the **-e authzid=dn:** control to give the proxy account for a simple bind, as in the example, or use the **-X authzid** argument for proxy authentication with a SASL bind.

```
ldapsearch -x -D
"uid=MoneyWizAcctSoftware,ou=Applications,dc=example,dc=com" -w secret -e
authz="dn:uid=AcctAdministrator,ou=Administrators,dc=example,dc=com" ...
```

## NOTE

Both the real user and the proxy user are recorded in the access logs. The real user is recorded as the **dn**, and the proxied user is recorded as the authorization ID, **authzid**.

```
[01/Jul/2017:16:11:47 -0400] conn=1 op=0 BIND
dn="uid=jsmith,ou=people,dc=example,dc=com"
authzid="uid=admin,ou=corp-accounts,dc=example,dc=com" ...
```

The client or application (**MoneyWizAcctSoftware**) binds as itself but is granted the privileges of the proxy entry (**AcctAdministrator**). The client does not need the password of the proxy entry.

## NOTE

There are some restrictions on binding with proxy authorization. You cannot use the Directory Manager's DN (root DN) as a proxy DN.

Additionally, if Directory Server receives more than one proxied authentication control, an error is returned to the client application, and the bind attempt is unsuccessful.

## 13.10. ADVANCED ACCESS CONTROL: USING MACRO ACIS

In organizations that use repeating directory tree structures, it is possible to optimize the number of ACIs used in the directory by using macros. Reducing the number of ACIs in your directory tree makes it easier to manage your access control policy and improves the efficiency of ACI memory usage.

Macros are placeholders that are used to represent a DN, or a portion of a DN, in an ACI. You can use a macro to represent a DN in the target portion of the ACI or in the bind rule portion, or both. In practice, when Directory Server gets an incoming LDAP operation, the ACI macros are matched against the resource targeted by the LDAP operation. If there is a match, the macro is replaced by the value of the DN of the targeted resource. Directory Server then evaluates the ACI normally.

### 13.10.1. Macro ACI Example

Figure 13.3, “Example Directory Tree for Macro ACIs” shows a directory tree which uses macro ACIs to effectively reduce the overall number of ACIs. This illustration uses repeating pattern of subdomains with the same tree structure (**ou=groups**, **ou=people**). This pattern is also repeated across the tree because the Example Corp. directory tree stores the suffixes **dc=hostedCompany2, dc=example, dc=com** and **dc=hostedCompany3, dc=example, dc=com**.

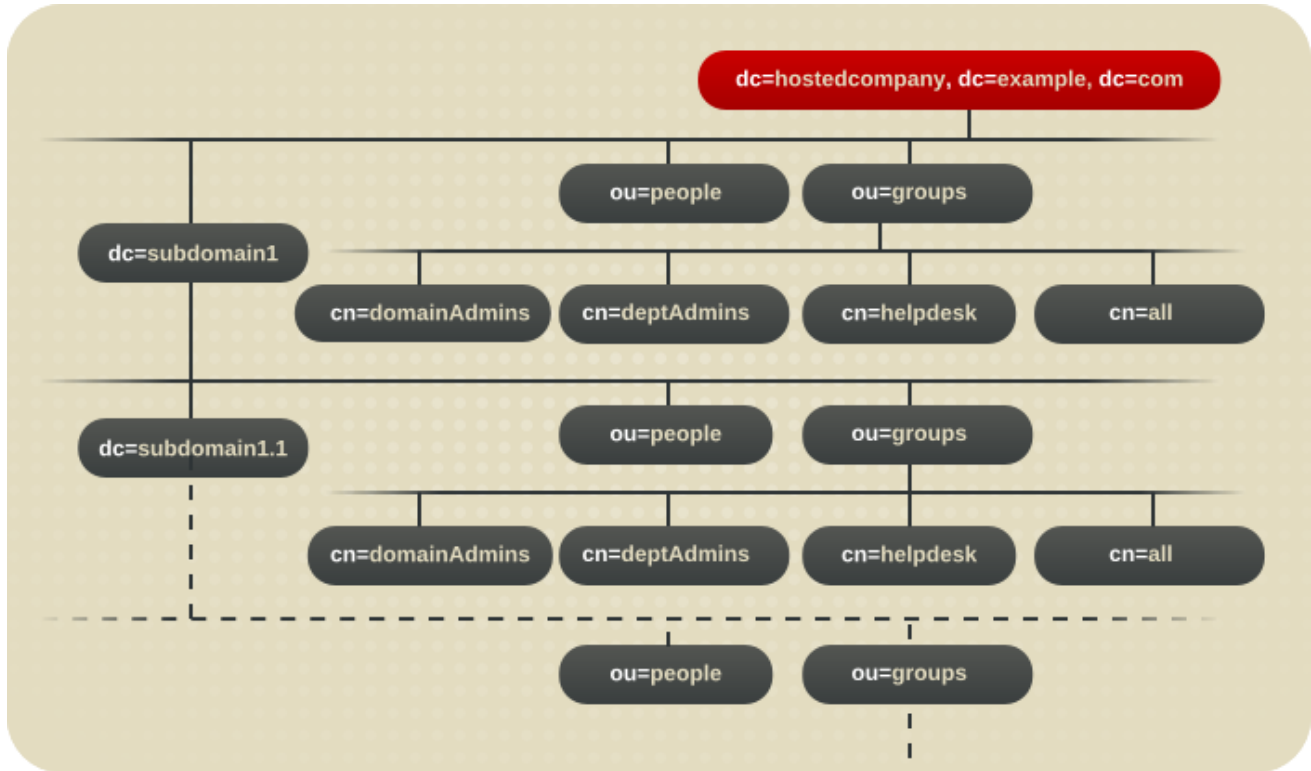
The ACIs that apply in the directory tree also have a repeating pattern. For example, the following ACI is located on the **dc=hostedCompany1, dc=example, dc=com** node:

```
aci: (targetattr="*)(targetfilter=(objectClass=nsManagedDomain))
```

```
(version 3.0; acl "Domain access"; allow (read,search)

groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany1,dc=example,dc=com";)
```

This ACL grants read and search rights to the **DomainAdmins** group to any entry in the **dc=hostedCompany1, dc=example, dc=com** tree.



**Figure 13.3. Example Directory Tree for Macro ACLs**

The following ACL is located on the **dc=hostedCompany1, dc=example, dc=com** node:

```
aci: (targetattr="*)(targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)

groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany1,dc=example,dc=com";)
```

The following ACL is located on the **dc=subdomain1, dc=hostedCompany1, dc=example, dc=com** node:

```
aci: (targetattr="*)(targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)

groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=subdomain1,dc=hostedCompany1,dc=example,dc=com";)
```

The following ACL is located on the **dc=hostedCompany2, dc=example, dc=com** node:

```
aci: (targetattr="*)(targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search))
```

```
groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany2,dc=example,dc=com";)
```

The following ACL is located on the **dc=subdomain1,dc=hostedCompany2,dc=example,dc=com** node:

```
aci: (targetattr="*)(targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)

groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=subdomain1,dc=hostedCompany2
,dc=example,dc=com";)
```

In the four ACLs shown above, the only differentiator is the DN specified in the **groupdn** keyword. By using a macro for the DN, it is possible to replace these ACLs by a single ACL at the root of the tree, on the **dc=example,dc=com** node. This ACL reads as follows:

```
aci: (target="ldap:///ou=Groups,($dn),dc=example,dc=com")
 (targetattr="*)(targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read,search)
 groupdn="ldap:///cn=DomainAdmins,ou=Groups,[$dn],dc=example,dc=com";)
```

The **target** keyword, which was not previously used, is utilized in the new ACL.

In this example, the number of ACLs is reduced from four to one. The real benefit is a factor of how many repeating patterns you have down and across your directory tree.

### 13.10.2. Macro ACL Syntax

Macro ACLs include the following types of expressions to replace a DN or part of a DN:

- `($dn)`
- `[$dn]`
- `($attr.attrName)`, where *attrName* represents an attribute contained in the target entry

In this section, the ACL keywords used to provide bind credentials, such as **userdn**, **roledn**, **groupdn**, and **userattr**, are collectively called the *subject*, as opposed to the *target*, of the ACL. Macro ACLs can be used in the target part or the subject part of an ACL.

[Table 13.9, “Macros in ACL Keywords”](#) shows in what parts of the ACL you can use DN macros:

**Table 13.9. Macros in ACL Keywords**

| Macro                          | ACL Keyword                                             |
|--------------------------------|---------------------------------------------------------|
| <code>(\$dn)</code>            | target, targetfilter, userdn, roledn, groupdn, userattr |
| <code>[\$dn]</code>            | targetfilter, userdn, roledn, groupdn, userattr         |
| <code>(\$attr.attrName)</code> | userdn, roledn, groupdn, userattr                       |

The following restrictions apply:

- If you use **(\$dn)** in **targetfilter**, **userdn**, **roledn**, **groupdn**, **userattr**, you *must* define a target that contains **(\$dn)**.
- If you use **[\$dn]** in **targetfilter**, **userdn**, **roledn**, **groupdn**, **userattr**, you *must* define a target that contains **(\$dn)**.



## NOTE

When using any macro, you *always* need a target definition that contains the **(\$dn)** macro.

You can combine the **(\$dn)** macro and the **(\$attr.attrName)** macro.

### 13.10.2.1. Macro Matching for (\$dn)

The **(\$dn)** macro is replaced by the matching part of the resource targeted in an LDAP request. For example, you have an LDAP request targeted at the **cn=all, ou=groups, dc=subdomain1, dc=hostedCompany1, dc=example, dc=com** entry and an ACI that defines the target as follows:

```
(target="ldap:///ou=Groups, ($dn), dc=example, dc=com")
```

The **(\$dn)** macro matches with **dc=subdomain1, dc=hostedCompany1**.

When the subject of the ACI also uses **(\$dn)**, the substring that matches the target is used to expand the subject. For example:

```
aci: (target="ldap:///ou=*, ($dn), dc=example, dc=com")
 (targetattr = "") (version 3.0; acl "Domain access"; allow
(read, search)
 groupdn="ldap:///cn=DomainAdmins, ou=Groups, ($dn), dc=example, dc=com";)
```

In this case, if the string matching **(\$dn)** in the target is **dc=subdomain1, dc=hostedCompany1**, then the same string is used in the subject. The ACI is then expanded as follows:

```
aci: (target="ldap:///ou=Groups, dc=subdomain1, dc=hostedCompany1,
 dc=example, dc=com") (targetattr = "") (version 3.0; acl "Domain
 access"; allow (read, search)
groupdn="ldap:///cn=DomainAdmins, ou=Groups,
 dc=subdomain1, dc=hostedCompany1, dc=example, dc=com";)
```

Once the macro has been expanded, Directory Server evaluates the ACI following the normal process to determine whether access is granted.

### 13.10.2.2. Macro Matching for [\$dn]

The matching mechanism for **[\$dn]** is slightly different than for **(\$dn)**. The DN of the targeted resource is examined several times, each time dropping the left-most RDN component, until a match is found.

For example, you have an LDAP request targeted at the **cn=all, ou=groups, dc=subdomain1, dc=hostedCompany1, dc=example, dc=com** subtree and the following ACI:

■

```
aci: (target="ldap:///ou=Groups, ($dn), dc=example, dc=com")
 (targetattr = "*") (version 3.0; acl "Domain access"; allow
(read, search)
 groupdn="ldap:///cn=DomainAdmins, ou=Groups, [$dn], dc=example, dc=com";)
```

The steps for expanding this ACL are as follows:

1. **(\$dn)** in the target matches **dc=subdomain1, dc=hostedCompany1**.
2. **[\$dn]** in the subject is replaced with **dc=subdomain1, dc=hostedCompany1**.

The result is

**groupdn="ldap:///cn=DomainAdmins, ou=Groups, dc=subdomain1, dc=hostedCompany1, dc=example, dc=com"**. If the bind DN is a member of that group, the matching process stops, and the ACL is evaluated. If it does not match, the process continues.

3. **[\$dn]** in the subject is replaced with **dc=hostedCompany1**.

The result is

**groupdn="ldap:///cn=DomainAdmins, ou=Groups, dc=hostedCompany1, dc=example, dc=com"**. In this case, if the bind DN is not a member of that group, the ACL is not evaluated. If it is a member, the ACL is evaluated.

The advantage of the **[\$dn]** macro is that it provides a flexible way of granting access to domain-level administrators to *all* the subdomains in the directory tree. Therefore, it is useful for expressing a hierarchical relationship between domains.

For example, consider the following ACL:

```
aci: (target="ldap:///ou=*, ($dn), dc=example, dc=com")
 (targetattr="*")(targetfilter=(objectClass=nsManagedDomain))
 (version 3.0; acl "Domain access"; allow (read, search)
 groupdn="ldap:///cn=DomainAdmins, ou=Groups, [$dn], dc=example, dc=com";)
```

It grants access to the members of

**cn=DomainAdmins, ou=Groups, dc=hostedCompany1, dc=example, dc=com** to all of the subdomains under **dc=hostedCompany1**, so an administrator belonging to that group could access a subtree like **ou=people, dc=subdomain1.1, dc=subdomain1**.

However, at the same time, members of **cn=DomainAdmins, ou=Groups, dc=subdomain1.1** would be denied access to the **ou=people, dc=hostedCompany1** and **ou=people, dc=hostedCompany1** nodes.

### 13.10.2.3. Macro Matching for (\$attr.attrName)

The **(\$attr.attrName)** macro is always used in the subject part of a DN. For example, define the following **roledn**:

```
roledn = "ldap:///cn=DomainAdmins, ($attr.ou)"
```

Now, assume the server receives an LDAP operation targeted at the following entry:

```
dn: cn=Jane Doe, ou=People, dc=HostedCompany1, dc=example, dc=com
cn: Jane Doe
```

```
sn: Doe
ou: Engineering,dc=HostedCompany1,dc=example,dc=com
...
```

In order to evaluate the **roledn** part of the ACL, the server looks at the **ou** attribute stored in the targeted entry and uses the value of this attribute to expand the macro. Therefore, in the example, the **roledn** is expanded as follows:

```
roledn =
"ldap:///cn=DomainAdmins,ou=Engineering,dc=HostedCompany1,dc=example,dc=com"
```

The Directory Server then evaluates the ACL according to the normal ACL evaluation algorithm.

When an attribute is multi-valued, each value is used to expand the macro, and the first one that provides a successful match is used. For example:

```
dn: cn=Jane Doe,ou=People,dc=HostedCompany1,dc=example,dc=com
cn: Jane Doe
sn: Doe
ou: Engineering,dc=HostedCompany1,dc=example,dc=com
ou: People,dc=HostedCompany1,dc=example,dc=com...
```

In this case, when the Directory Server evaluates the ACL, it performs a logical OR on the following expanded expressions:

```
roledn =
"ldap:///cn=DomainAdmins,ou=Engineering,dc=HostedCompany1,dc=example,dc=com"

roledn =
"ldap:///cn=DomainAdmins,ou=People,dc=HostedCompany1,dc=example,dc=com"
```

## 13.11. SETTING ACCESS CONTROLS ON DIRECTORY MANAGER

Having an unconstrained administrative user makes sense from a maintenance perspective. The Directory Manager requires a high level of access in order to perform maintenance tasks and to response to incidents.

However, because of the power of the Directory Manager user, a certain level of access control may be advisable to prevent unauthorized access or attacks from being performed as the root user.

Regular access control rules are applied to the directory tree, the Directory Manager is not a regular user entry, so no (regular) ACLs can be applied to the Directory Manager user. ACLs are applied through a special plug-in configuration entry.

### 13.11.1. About Access Controls on the Directory Manager Account

Normal access control rules do not apply to the Directory Manager user. The Directory Manager is defined in the **dse.ldif** file, not in the regular user database, and so ACL targets ([Section 13.3.2, “Defining Targets”](#)) which are based on an entry within a subtree do not include the Directory Manager.

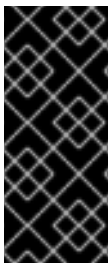
Access controls for Directory Manager are implemented through the *RootDN Access Control Plug-in*. This plug-in applies to the Directory Server configuration, and therefore can apply some access control rules to the Directory Manager entry.

The plug-in does not define a standard ACL. Some information is already implied, including the target (the Directory Manager entry) and the allowed rights (all of them). The purpose of the RootDN Access Control Plug-in is not to restrict *what* the Directory Manager can do; the purpose is to provide a level of security by limiting who can log in as Directory Manager (even with valid credentials) based on their location or time.

For this reason, the ACI for the Directory Manager only sets bind rules:

- Time-based access controls for time ranges, such as 8a.m. to 5p.m. (0800 to 1700), and day-of-week access controls, so access is only allowed on explicitly defined days. This is analogous to [Section 13.4.9, “Defining Access at a Specific Time of Day or Day of Week”](#).
- IP address rules, where only specified IP addresses, domains, or subnets are explicitly allowed or denied. This is analogous to [Section 13.4.6, “Defining Access from a Specific IP Address”](#).
- Host access rules, where only specified host names, domain names, or subdomains are explicitly allowed or denied. This is analogous to [Section 13.4.7, “Defining Access from a Specific Domain”](#).

As with other access control rules, deny rules supercede allow rules.



### IMPORTANT

Make sure that the Directory Manager always has the appropriate level of access allowed. The Directory Manager may need to perform maintenance operations in off-hours (when user load is light) or to respond to failures. In that case, setting stringent time or day-based access control rules could prevent the Directory Manager from being able to adequately manage the directory.

## 13.11.2. Configuring the RootDN Access Control Plug-in

Root DN access control rules are disabled by default. The RootDN Access Control Plug-in must be enabled, and then the appropriate access control rules can be set.



### NOTE

There is only *one* access control rule set for the Directory Manager, in the plug-in entry, and it applies to all access to the entire directory.

1. Enable the RootDN Access Control Plug-in by setting the ***nsslapd-pluginEnabled*** attribute to **on**. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=RootDN Access Control Plug-in,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
```

2. Set the bind rules for the access control instruction.

- ***rootdn-open-time*** and ***rootdn-close-time*** for time-based access controls.
- ***rootdn-days-allowed*** for day-based access controls.
- ***rootdn-allow-host***, ***rootdn-deny-host***, ***rootdn-allow-ip***, and ***rootdn-deny-ip*** for host-based access controls. These are all multi-valued attributes.

Deny rules supercede allow rules. For example, if ***rootdn-allow-host*** attribute is set to ***\*.example.com***, and the ***rootdn-deny-host*** attribute is set to ***\*.front-office.example.com***, anything in the ***front-office.example.com*** subdomain is prevented from logging in as Directory Manager, even though the larger ***example.com*** domain is allowed.

Wild cards can be used to allow IP ranges or full domains.

For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x

dn: cn=RootDN Access Control Plug-in,cn=plugins,cn=config
changetype: modify
add: rootdn-open-time
rootdn-open-time: 0600
-
add: rootdn-close-time
rootdn-close-time: 2100
-
add: rootdn-allow-host
rootdn-allow-host: *.example.com
-
add: rootdn-deny-host
rootdn-allow-host: *.remote.example.com
```

3. Restart the Directory Server to load the new plug-in configuration.

```
service dirsrv restart instance_name
```

## 13.12. ACCESS CONTROL AND REPLICATION

ACIs are stored as attributes of entries; therefore, if an entry containing ACIs is part of a replicated database, the ACIs are replicated like any other attribute.

ACIs are always evaluated on the Directory Server that services the incoming LDAP requests. This means that when a consumer server receives an update request, it returns a referral to the supplier server before evaluating whether the request can be serviced on the supplier.

## 13.13. COMPATIBILITY WITH EARLIER RELEASES

Some ACI keywords that were used in earlier releases of Directory Server have been deprecated. However, for reasons of backward compatibility, the following keywords are still supported:



- userdnattr
- groupdnattr

Therefore, if you have set up a replication agreement between a legacy supplier server and a version 9.0 consumer, there should not be any problems in the replication of ACIs.

## CHAPTER 14. MANAGING USER AUTHENTICATION

When a user connects to the Red Hat Directory Server, first the user is authenticated. Then, the directory grants access rights and resource limits to the user depending upon the identity established during authentication.

This chapter describes tasks for managing users, including configuring the password and account lockout policy for the directory, denying groups of users access to the directory, and limiting system resources available to users depending upon their bind DNs.

### 14.1. MANAGING THE PASSWORD POLICY

A password policy minimizes the risks of using passwords by enforcing a certain level of security. For example, a password policy can define that:

- Users must change their passwords according to a schedule.
- Users must provide non-trivial passwords.
- The password syntax must meet certain complexity requirements.

For an overview on password policy, see "Designing a Password Policy" in the "Designing a Secure Directory" chapter in the *Deployment Guide*.



#### WARNING

When using a password administrator account or the **Directory Manager** (root DN) to set a password, password policies are bypassed and not verified. Do not use these accounts for regular user password management. Use them only to perform password administration tasks that require bypassing the password policies.

Directory Server supports fine-grained password policy, so password policies can be applied to the entire directory (*global* password policy), a particular subtree (*subtree-level* or *local* password policy), or a particular user (*user-level* or *local* password policy).

The complete password policy applied to a user account is comprised of the following elements:

- *The type or level of password policy checks.* This information indicates whether the server should check for and enforce a global password policy or local (subtree/user-level) password policies.

Password policies work in an inverted pyramid, from general to specific. A global password policy is superceded by a subtree-level password policy, which is superceded by a user-level password policy. Only one password policy is enforced for the entry; password policies are not additive. This means that if a particular attribute is configured in the global or subtree-level policy, but not in the user-level password policy, the attribute is not used for the user when a login is attempted because the active, applied policy is the user-level policy.

- *Password add and modify information.* The password information includes password syntax and password history details.

- *Bind information.* The bind information includes the number of grace logins permitted, password aging attributes, and tracking bind failures.



## NOTE

After establishing a password policy, user passwords can be protected from potential threats by configuring an account lockout policy. Account lockout protects against hackers who try to break into the directory by repeatedly guessing a user's password.

### 14.1.1. Configuring the Global Password Policy



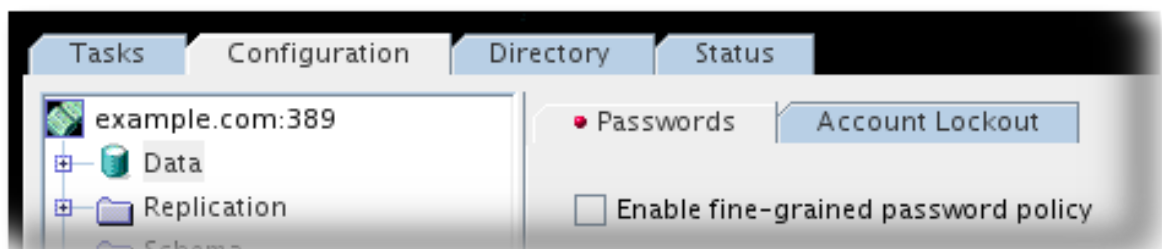
## NOTE

After configuring the password policy, configure an account lockout policy. For details, see [Section 14.4, “Configuring a Password-Based Account Lockout Policy”](#).

#### 14.1.1.1. Configuring a Global Password Policy Using the Console

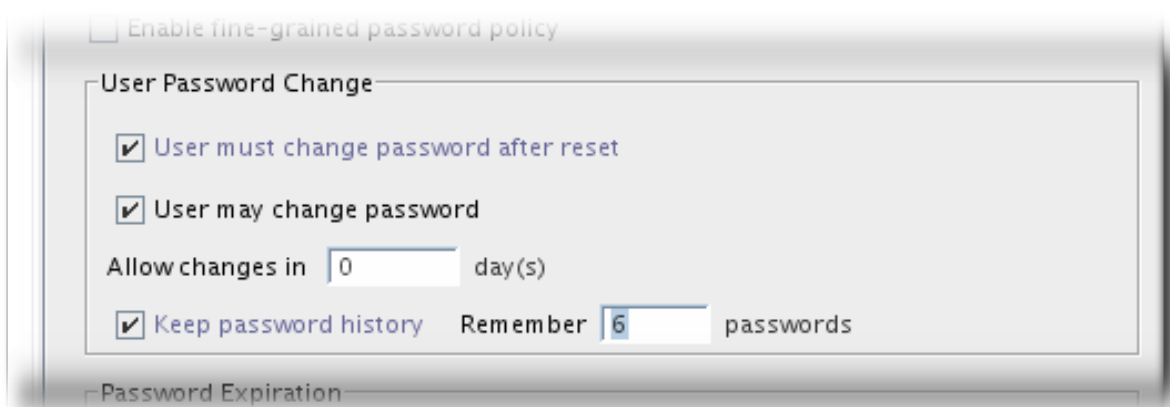
A global password policy applies to every entry in the entire directory.

1. Select the **Configuration** tab and then the **Data** node.
2. In the right pane, select the **Passwords** tab.



This tab contains the password policy for the entire Directory Server.

3. Set the password policies for how users can change their own passwords.



- To require users to change their password the first time they log on, select the **User must change password after reset** check box.

**NOTE**

If users are required to reset their password, only the Directory Manager is authorized to reset the user's password. A regular administrative user cannot force the users to update their password.

- o To allow users to change their own passwords, select the **User may change password** check box.
  - o To prevent users from changing their password for a specific duration, enter the number of days in the **Allow changes in X day(s)** text box. This keeps users from quickly cycling through passwords to reuse a password in their password history.
  - o For the server to maintain a history list of passwords used by each user, select the **Keep password history** check box. Enter the number of passwords for the server to keep for each user in the **Remember X passwords** text box.
4. Set the policies for when passwords expire.

- o If user passwords should not expire, select the **Password never expires** radio button.
- o To require users to change their passwords periodically, select the **Password expires after X days** radio button, and then enter the number of days that a user password is valid.

The maximum value for the password age is derived by subtracting January 18, 2038, from today's date. The entered value must not be set to the maximum value or too close to the maximum value. Setting the value to the maximum value can cause the Directory Server to fail to start because the number of seconds will go past the epoch date. In such an event, the error log will indicate that the password maximum age is invalid. To resolve this problem, correct the **passwordMaxAge** attribute value in the **dse.ldif** file.

A common policy is to have passwords expire every 30 to 90 days. By default, the password maximum age is set to **8640000** seconds (100 days).

- o If the **Password expire after X days** radio button is selected, specify how long before the password expires to send a warning to the user. In the **Send Warning X Days Before Password Expires** text enter the number of days before password expiration to send a warning.

**NOTE**

It is not necessary to configure the Directory Server to send a warning to users. The Directory Server automatically issues a warning the next time the user attempts to log into the Directory Server Console that the password will soon expire or has expired. This is analogous to an operating system warning that reads **"Warning: password will expire in 7 days"** when a user logs in.

5. For the server to check the syntax of a user password to make sure it meets the minimum requirements set by the password policy, select the **Check Password Syntax** check box. Then, specify required password complexity, such as the minimum length and required number of numeric and special characters. The password syntax requirements are described more in [Table 14.1, "Password Policy Attributes"](#).

Allow up to 1 login attempt(s) after password expires

**Password Syntax**

☒ Check password syntax

Password minimum length 8

Minimum required digit characters 1

Minimum required alpha characters 0

Minimum required uppercase characters 1

Minimum required lowercase characters 0

Minimum required special characters 1

Minimum required 8-bit characters 0

Maximum number of repeated characters 0

Minimum required character categories 3

Minimum token length 3

Password encryption: Salted Secure Hashing Algorithm (SSHA)

6. From the **Password Encryption** pull-down menu, select the encryption method for the server to use when storing passwords.

Minimum token length 3

Password encryption: Salted Secure Hashing Algorithm (SSHA)

Save Reset Help

For detailed information about the encryption methods, see the *passwordStorageScheme* attribute in [Table 14.1, "Password Policy Attributes"](#).

The **Password Encryption** menu might contain other encryption methods, as the directory dynamically creates the menu depending upon the existing encryption methods it finds in the directory.

7. Click **Save**.

#### 14.1.1.2. Configuring a Global Password Policy Using the Command Line

To set up the password policy for a subtree or user, add the required entries and attributes at the subtree- or user-level, set the appropriate values to the password policy attributes, and enable fine-grained password policy checking.

No password policy attributes are set by default. Each password policy attribute must be added manually to the **cn=config** entry to create a global policy. These can be passed all together by passing an LDIF file with **ldapmodify**.

1. Create the LDIF file. Each statement is the same as inputting the changes through stdin, with separate update statements separated by a dash (-).

```
dn: cn=config
changetype: modify
add: passwordChange
passwordChange: on
-
add: passwordExp
passwordExp: on
-
add: passwordMaxAge
passwordMaxAge: 8640000
-
add: passwordCheckSyntax
passwordCheckSyntax: on
-
add: passwordMinCategories
passwordMinCategories: 3
-
add: passwordStorageScheme
passwordStorageScheme: SSHA512
^D
```

2. Pass the LDIF file to the server using the **-f** option with the **ldapmodify** command.

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -f user-
pwdpolicy.ldif
```

Table 14.1, “Password Policy Attributes” describes the attributes available to configure the password policy.

**Table 14.1. Password Policy Attributes**

| Attribute Name | Definition |
|----------------|------------|
|----------------|------------|

| Attribute Name          | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| passwordTrackUpdateTime | This attribute sets whether to record a separate timestamp specifically for the most recent update to the password attribute. This can be useful to help coordinate synchronization updates or changes between other LDAP servers, like Active Directory. By default, this is turned off.                                                                                                                                                                                                               |
| passwordGraceLimit      | This attribute indicates the number of grace logins permitted when a user's password is expired. When set to a positive number, the user will be allowed to bind with the expired password for that many times. For the global password policy, the attribute is defined under <b>cn=config</b> . By default, this attribute is set to <b>0</b> , which means grace logins are not permitted.                                                                                                           |
| passwordMustChange      | When <b>on</b> , this attribute requires users to change their passwords when they first login to the directory or after the password is reset by the Directory Manager. The user is required to change their password even if user-defined passwords are disabled. If this attribute is set to <b>off</b> , passwords assigned by the Directory Manager should not follow any obvious convention and should be difficult to discover. This attribute is <b>off</b> by default.                         |
| passwordChange          | When <b>on</b> , this attribute indicates that users may change their own password. Allowing users to set their own passwords runs the risk of users choosing passwords that are easy to remember. However, setting good passwords for the user requires a significant administrative effort. In addition, providing passwords to users that are not meaningful to them runs the risk that users will write the password down somewhere that can be discovered. This attribute is <b>on</b> by default. |
| passwordExp             | When <b>on</b> , this attribute indicates that the user's password will expire after an interval given by the <b>passwordMaxAge</b> attribute. Making passwords expire helps protect the directory data because the longer a password is in use, the more likely it is to be discovered. This attribute is <b>off</b> by default.                                                                                                                                                                       |

| Attribute Name  | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| passwordMaxAge  | <p>This attribute indicates the number of seconds after which user passwords expire. To use this attribute, enable password expiration using the <b><i>passwordExp</i></b> attribute. This attribute is a dynamic parameter in that its maximum value is derived by subtracting January 18, 2038, from today's date. The attribute value must not be set to the maximum value or too close to the maximum value. If the value is set to the maximum value, Directory Server may fail to start because the number of seconds will go past the epoch date. In such an event, the error log will indicate that the password maximum age is invalid. To resolve this problem, correct the <b><i>passwordMaxAge</i></b> attribute value in the <b><i>dse.ldif</i></b> file. A common policy is to have passwords expire every 30 to 90 days. By default, the password maximum age is set to <b>8640000</b> seconds (100 days).</p> |
| passwordWarning | <p>This attribute indicates the number of seconds before a warning message is sent to users whose password is about to expire. Depending on the LDAP client application, users may be prompted to change their password when the warning is sent. By default, the directory sends the warning <b>86400</b> seconds (1 day) before the password is about to expire. However, a password never expires until the warning message has been sent. Therefore, if users do not bind to the Directory Server for longer than the <b><i>passwordMaxAge</i></b>, they will still get the warning message in time to change their password.</p>                                                                                                                                                                                                                                                                                         |
| passwordMinAge  | <p>This attribute indicates the number of seconds that must pass before a user can change their password. Use this attribute in conjunction with the <b><i>passwordInHistory</i></b> attribute to discourage users from reusing old passwords. For example, setting the minimum password age to 2 days prevents users from repeatedly changing their passwords during a single session to cycle through the password history and reuse an old password once it has been removed from the history list. The minimum age can be from <b>0</b> to <b>2147472000</b> seconds (24,855 days). A value of zero indicates that the user can change the password immediately. The default value of this attribute is <b>0</b>.</p>                                                                                                                                                                                                     |



| Attribute Name      | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| passwordHistory     | This attribute indicates whether the directory stores a password history. When set to <b>on</b> , the directory stores the number of passwords specified in the <b>passwordInHistory</b> attribute in a history. If a user attempts to reuse one of the passwords, the password will be rejected. When this attribute is set to <b>off</b> , any passwords stored in the history remain there. When this attribute is set back to <b>on</b> , users will not be able to reuse the passwords recorded in the history before the attribute was disabled. This attribute is <b>off</b> by default, meaning users can reuse old passwords. |
| passwordInHistory   | This attribute indicates the number of passwords the directory stores in the history. There can be <b>2</b> to <b>24</b> passwords stored in the history. This feature is not enabled unless the <b>passwordHistory</b> attribute is set to <b>on</b> . This attribute is set to <b>6</b> by default.                                                                                                                                                                                                                                                                                                                                  |
| passwordCheckSyntax | When <b>on</b> , this attribute indicates that the password syntax is checked by the server before the password is saved. Password syntax checking ensures that the password string meets or exceeds the length and complexity requirements and that the string does not contain any <i>trivial</i> words. A trivial word is any value stored in the <b>uid</b> , <b>cn</b> , <b>sn</b> , <b>givenname</b> , <b>ou</b> , or <b>mail</b> attributes of the user's entry. This attribute is <b>off</b> by default.                                                                                                                       |
| passwordMinLength   | This attribute specifies the minimum number of characters that must be used in passwords. Shorter passwords are easier to crack. Passwords can be two (2) to 512 characters long. Generally, a length of eight characters is long enough to be difficult to crack but short enough for users to remember without writing it down. This attribute is set to <b>8</b> by default.                                                                                                                                                                                                                                                        |
| passwordMaxRepeats  | This attribute set the maximum number of times that the same character can be used in row, such as <b>aaaaa</b> . Setting the attribute to <b>0</b> means that there is no limit on how many time a character can be repeated. This attribute is set to <b>0</b> by default.                                                                                                                                                                                                                                                                                                                                                           |
| passwordMinAlphas   | This attribute sets the minimum number of alphabetic characters that must be used in the password. This setting does not set any requirements for the letter case; requirements for the minimum number of lowercase and uppercase letters are set in the <b>passwordMinLower</b> and <b>passwordMinUpper</b> attributes, respectively. By default, this attribute is set to <b>0</b> , meaning there is no required minimum.                                                                                                                                                                                                           |

| Attribute Name                              | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                        |                            |                            |                       |                                             |                  |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|----------------------------|-----------------------|---------------------------------------------|------------------|
| passwordMinDigits                           | This attribute sets the minimum number of numeric characters (0 through 9) which must be used in the password. By default, this attribute is set to <b>0</b> , meaning there is no required minimum.                                                                                                                                                                                                                                              |                            |                            |                       |                                             |                  |
| passwordMinSpecials                         | This attribute sets the minimum number of special ASCII characters, such as <b>!@#\$. ,</b> which must be used in the password. By default, this attribute is set to <b>0</b> , meaning there is no required minimum.                                                                                                                                                                                                                             |                            |                            |                       |                                             |                  |
| passwordMinLowers                           | This attribute sets the minimum number of lower case alphabetic characters, a to z, which must be used in the password. By default, this attribute is set to <b>0</b> , meaning there is no required minimum.                                                                                                                                                                                                                                     |                            |                            |                       |                                             |                  |
| passwordMinCategories                       | <div>This attribute sets the minimum number of categories which must be used in the password. There are several categories available:<table><tr><td>Uppercase letters (A to Z)</td></tr><tr><td>Lowercase letters (a to z)</td></tr><tr><td>Numbers (0 through 9)</td></tr><tr><td>Special ASCII characters, such as <b>\$</b></td></tr><tr><td>8-bit characters</td></tr></table></div> <div>This attribute is set to <b>3</b> by default.</div> | Uppercase letters (A to Z) | Lowercase letters (a to z) | Numbers (0 through 9) | Special ASCII characters, such as <b>\$</b> | 8-bit characters |
| Uppercase letters (A to Z)                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                            |                            |                       |                                             |                  |
| Lowercase letters (a to z)                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                            |                            |                       |                                             |                  |
| Numbers (0 through 9)                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                            |                            |                       |                                             |                  |
| Special ASCII characters, such as <b>\$</b> |                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                            |                            |                       |                                             |                  |
| 8-bit characters                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                            |                            |                       |                                             |                  |
| passwordMinUppers                           | This attribute sets the minimum number of upper case alphabetic characters, A to Z, which must be used in the password. By default, this attribute is set to <b>0</b> , meaning there is no required minimum.                                                                                                                                                                                                                                     |                            |                            |                       |                                             |                  |

| Attribute Name         | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| passwordMinTokenLength | <p>A trivial word check identifies dictionary or common words used in a password string. This attribute sets the minimum length for tokens or strings which are checked by the <b><i>passwordSyntaxCheck</i></b> attributes. This can be from <b>1</b> to <b>64</b> characters. This attribute is set to <b>3</b> by default, meaning that the password cannot contain a three-character (or longer) sequential string taken from their user name, common name, first or last name, mail, or other relevant attributes.</p> <p>For example, for a token length of three, if the user name is <b>jsmith</b>, then the password <b>123smi!</b> would be rejected because it has three sequential characters which match the user name.</p> |
| passwordMin8bit        | <p>This attribute sets the minimum number of 8-bit characters used in the password. The default number is <b>0</b>, meaning none are required.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

| Attribute Name        | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| passwordStorageScheme | <p>This attribute specifies the type of encryption used to store Directory Server passwords. The following encryption types are supported by Directory Server:</p> <div data-bbox="820 322 1428 1368"> <p><i>SSHA (Salted Secure Hash Algorithm).</i> This method is recommended as it is the most secure. The Directory Server supports <b>SSHA</b>, <b>SSHA256</b>, <b>SSHA384</b>, and <b>SSHA512</b>. SSHA is the default method.</p> <p><i>SHA (Secure Hash Algorithm).</i> A one-way hash algorithm; it is supported only for backwards compatibility with Directory Server 4.x and should not be used otherwise. This includes support for <b>SHA</b>, <b>SHA256</b>, <b>SHA384</b>, and <b>SHA512</b> algorithms, which protects against some insecurities in the SHA1 algorithm.</p> <p><i>MD5.</i> MD5 is not as secure as SSHA but is available for legacy applications require it.</p> <p><i>Salted MD5.</i> This storage scheme is more secure than plain MD5 hash, but still less secure than SSHA. This storage scheme is not included for use with new passwords but to help with migrating user accounts from directories which support salted MD5.</p> <p><i>crypt.</i> The UNIX crypt algorithm, provided for compatibility with UNIX passwords.</p> <p><i>clear.</i> This encryption type indicates that the password will appear in plain text.</p> </div> <p>The only password storage scheme that can be used with SASL DIGEST-MD5 is <b>CLEAR</b>. Passwords stored using <b>crypt</b>, <b>SHA</b>, or <b>SSHA</b> formats cannot be used for secure login through SASL DIGEST-MD5. To provide a customized storage scheme, consult Red Hat professional services.</p> |

## 14.1.2. Configuring a Local Password Policy

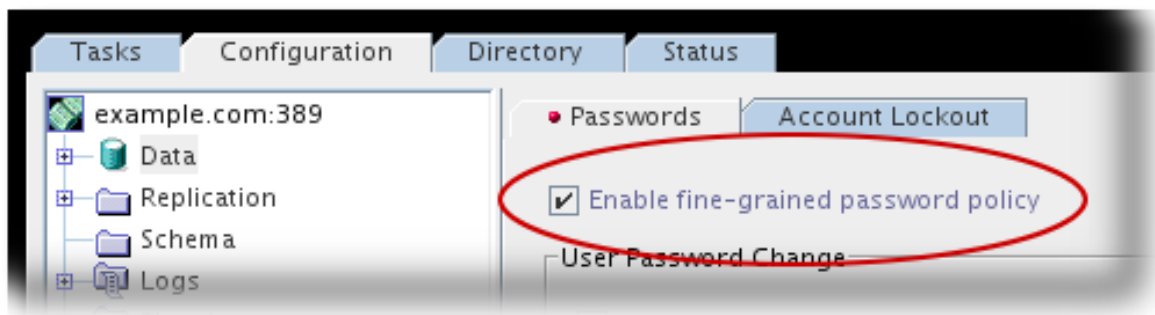


### NOTE

After configuring the password policy, configure an account lockout policy. For details, see [Section 14.4, “Configuring a Password-Based Account Lockout Policy”](#).

### 14.1.2.1. Configuring a Subtree/User Password Policy Using the Console

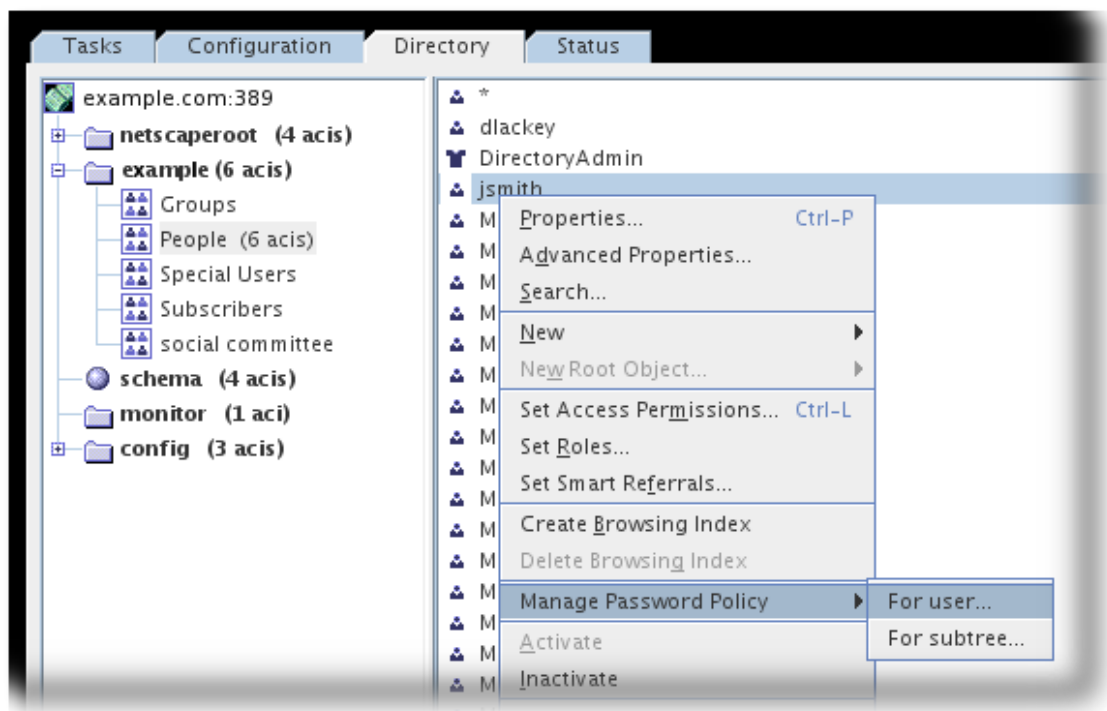
1. Enable a fine-grained password policy globally, as described in [Section 14.1.1.1, “Configuring a Global Password Policy Using the Console”](#). Be sure to check the **Enable fine-grained password policy** check box to allow user-level password policies.



## NOTE

The password policy *must* be enabled globally before it will be applied locally. No other global password policy features must be set, and the global password policy will not override the local policy if they differ.

2. Create the local password policy for the subtree or user.
  1. Select the **Directory** tab.
  2. In the navigation pane, select the subtree or user entry for which to set up the password policy.
  3. From the **Object** menu, select the **Manage Password Policy** option, and then select the **For user** or **For subtree**.



4. In the **Passwords** tab, select the **Create subtree/user level password policy** check box to add the required attributes. The password policy settings — resetting,

expiration, syntax, and encryption — are the same as for the global policy in [Section 14.1.1.1, “Configuring a Global Password Policy Using the Console”](#).



5. In the **Account Lockout** tab, specify the appropriate information, and click **Save**.



#### 14.1.2.2. Configuring Subtree/User Password Policy Using the Command Line

1. Add the required attributes to the subtree or user entries by running the **ns-newpwpolicy.pl** script.

The command syntax for the script is as follows:

```
ns-newpwpolicy.pl [-D rootDN] -w password | -w - | -j filename [-p
port] [-h host] -U userDN -S suffixDN
```

For updating a subtree entry, use the **-S** option. For updating a user entry, use the **-U** option. The **ns-newpwpolicy.pl** script accepts only one user or subtree entry at a time. It can, however, accept both user and suffix entries at the same time. For details about the script, see the *Directory Server Configuration and Command-Line Tool Reference*.

2. The script adds the required attributes depending on whether the target entry is a subtree or user entry.

For a subtree (for example, **ou=people,dc=example,dc=com**), the following entries are added:

- o A container entry (**nsPwPolicyContainer**) at the subtree level for holding various password policy-related entries for the subtree and all its children. For example:

```
dn: cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
objectClass: top
objectClass: nsContainer
cn: nsPwPolicyContainer
```

- o The actual password policy specification entry (**nsPwPolicyEntry**) for holding all the password policy attributes that are specific to the subtree. For example:

```
dn: cn=cn=nsPwPolicyEntry\,ou=people\,dc=example\,dc=com,
 cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: ldapsubentry
objectclass: passwordpolicy
```

- o The CoS template entry (**nsPwTemplateEntry**) that has the **pwdpolicysubentry** value pointing to the above (**nsPwPolicyEntry**) entry. For example:

```
dn: cn=cn=nsPwTemplateEntry\,ou=people\,dc=example\,dc=com,
 cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: costemplate
objectclass: ldapsubentry
cosPriority: 1
pwdpolicysubentry:
cn=cn=nsPwPolicyEntry\,ou=people\,dc=example\,dc=com,
 cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
```

- o The CoS specification entry at the subtree level. For example:

```
dn: cn=newpwdpolicy_cos,ou=people,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn:
cn=cn=nsPwTemplateEntry\,ou=people\,dc=example,dc=com,
 cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
cosAttribute: pwdpolicysubentry default operational
```

For a user (for example, **uid=jdoe,ou=people,dc=example,dc=com**), the following entries are added:

- o A container entry (***nsPwPolicyContainer***) at the parent level for holding various password policy related entries for the user and all its children. For example:

```
dn: cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
objectClass: top
objectClass: nsContainer
cn: nsPwPolicyContainer
```

- o The actual password policy specification entry (***nsPwPolicyEntry***) for holding the password policy attributes that are specific to the user. For example:

```
dn:
cn=cn=nsPwPolicyEntry\,uid=jdoe\,ou=people\,dc=example\,dc=com,
cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: ldapsubentry
objectclass: passwordpolicy
```

3. Assign the value of the above entry DN to the ***pwdpolicysubentry*** attribute of the target entry. For example, this assigns the password policy to the user entry:

```
dn: uid=jdoe,ou=people,dc=example,dc=com
changetype: modify
replace: pwdpolicysubentry
pwdpolicysubentry:
cn=cn=nsPwPolicyEntry\,uid=jdoe\,ou=people\,dc=example\,dc=com,
cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
```

4. Set the password policy attributes for the subtree or user entry with the appropriate values.

[Table 14.1, “Password Policy Attributes”](#) describes the attributes available to configure the password policy. The ***ldapmodify*** utility can be used to change these attributes in the subtree or user entry which contains the ***nsPwPolicyEntry*** object class.



## NOTE

The ***nsslapd-pwpolicy-local*** attribute of the ***cn=config*** entry controls the type of password policy the server enforces. By default, this attribute is disabled (***off***). When the attribute is disabled, the server only checks for and enforces the global password policy; the subtree and user-level password policies are ignored. When the ***ns-newpwpolicy.pl*** script runs, it first checks for the specified subtree and user entries and, if they exist, modifies them. After updating the entries successfully, the script sets the ***nsslapd-pwpolicy-local*** configuration parameter to on. If the subtree and user-level password policy should not be enabled, be sure to set ***nsslapd-pwpolicy-local*** to ***off*** after running the script.

To turn off user- and subtree-level password policy checks, set the ***nsslapd-pwpolicy-local*** attribute to ***off*** by modifying the ***cn=config*** entry. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
```



```
dn: cn=config
changetype: modify
replace: nsslapd-pwpolicy-local
nsslapd-pwpolicy-local: off
```

This attribute can also be disabled by modifying it directly in the configuration file (**dse.ldif**).

1. Stop the server.

```
service dirsrv stop instance
```

2. Open the **dse.ldif** file in a text editor.
3. Set the value of **nsslapd-pwpolicy-local** to **off**, and save.

```
nsslapd-pwpolicy-local: off
```

4. Start the server.

```
service dirsrv start instance
```

### 14.1.2.3. Manually Setting Default Password Syntax Checking for Local Password Policies

The settings for the global password policy should be in effect for any local password policies, if those attributes are not explicitly set. If the password policy attribute is not set in either the global or the local policy, then the default values should be assumed.

However, there is a bug in Directory Server, so that if a password policy attribute is set in the global password policy but not in the local password policy, then neither the global setting nor the default settings is enforced by the local password policy. To work around this, set the password attributes explicitly in the local password policy.

1. Enable global syntax checking, as in [Section 14.1.1.1, “Configuring a Global Password Policy Using the Console”](#), and save the policy.
2. Edit the local password policy to contain all password syntax
3. Enable fine-grained password checking, as in [Section 14.1.2.1, “Configuring a Subtree/User Password Policy Using the Console”](#), and save the policy.
4. Edit the local password policy to contain all password syntax attributes. Set the values to something other than the default settings, as listed in the *Configuration and Command-Line Tool Reference*.
5. Re-edit the local password policy with the desired values, even if they are the defaults.

### 14.1.3. Setting User Passwords

An entry can be used to bind to the directory only if it has a **userPassword** attribute and if it has not been inactivated. Because user passwords are stored in the directory, the user passwords can be set or reset with any LDAP operation, like **ldapmodify**.

For information on creating and modifying directory entries, see [Chapter 3, \*Creating Directory Entries\*](#). For information on inactivating user accounts, see [Section 14.11, “Manually Inactivating Users and Roles”](#).

Passwords can also be set and reset in the **Users and Groups** area of the Red Hat Admin Server or Directory Server Console. For information on how to use the **Users and Groups** area in the Admin Server Console, see the online help that is available in the Red Hat Admin Server.

Only password administrators, described in [Section 14.1.4, “Setting Password Administrators”](#), and the root DN can add pre-hashed passwords. These users can also violate password policy.



### WARNING

When using a password administrator account or the **Directory Manager** (root DN) to set a password, password policies are bypassed and not verified. Do not use these accounts for regular user password management. Use them only to perform password administration tasks that require bypassing the password policies.

#### 14.1.4. Setting Password Administrators

The Directory Manager can add the *password administrator* role to a user or a group of users. Since access control instructions (ACI) need to be set, it is recommended that a group is used to allow just a single ACI set to manage all password administrators. A password administrator can perform any user password operations, including the following:

- forcing the user to change their password,
- changing a user's password to a different storage scheme defined in the password policy,
- bypassing the password syntax checks,
- and adding already hashed passwords.

As explained in [Section 14.1.3, “Setting User Passwords”](#), it is recommended that ordinary password updates are done by an existing role in the database with permissions to update only the **userPassword** attribute. We recommend not to use the password administrator account for these ordinary tasks.

To specify a user or a group of users as password administrator in a local policy, use **ldapmodify** to set the **passwordAdminDN** attribute in the main configuration entry.

```
[root@server ~]# ldapmodify -h localhost -p 389 -D "cn=directory manager"
-W
dn:
cn=cn\3DnsPwPolicyEntry\2Cou\3DPeople\2Cdc\3Dexample\2Cdc\3Dcom,cn=nsPwPol
icyContainer,ou=People,dc=example,dc=com
changetype: modify
replace: passwordAdminDN
passwordAdminDN: cn=Passwd Admins,ou=groups,dc=example,dc=com
```

For setting in the global policy:

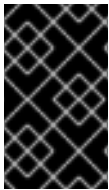
```
[root@server ~]# ldapmodify -h localhost -p 389 -D "cn=directory manager"
-W
dn: cn=config
changetype: modify
replace: passwordAdminDN
passwordAdminDN: cn=Passwd Admins,ou=groups,dc=example,dc=com
```

### 14.1.5. Changing Passwords Stored Externally

While most passwords can be changed through the Console and other Directory Server features or through the **ldapmodify** operation, there are some passwords that cannot be changed through regular LDAP operations. These passwords may be stored outside the Directory Server, such as passwords stored in a SASL application. These passwords can be modified through the *password change extended operation*.

Directory Server supports the password change extended operation as defined in RFC 3062, so users can change their passwords, using a suitable client, in a standards-compliant way. The **ldappasswd** utility passes the changes for the password for the specified user:

```
ldappasswd -x -D bind_dn -w password -p server_port -h server_hostname [-a
oldPassword] [-s newPassword] [user]
```



#### IMPORTANT

Password operations must be performed over a secure connection, meaning SASL, TLS/SSL, or Start TLS. For information on using secure connections with LDAP client tools, see [Section A.2, “Using SSL/TLS and Start TLS with LDAP Client Tools”](#).

**Table 14.2. ldappasswd Options**

| Parameter | Description                                                                                                                                                                                                                                          |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -h        | Gives the host name of the Directory Server.                                                                                                                                                                                                         |
| -p        | Gives the port number of the Directory Server. Since SSL is required for password change operations, this is usually give the TLS/SSL port of the Directory Server. With the <b>-ZZ</b> or <b>-ZZZ</b> for Start TLS, this can be the standard port. |
| -D        | Gives the bind DN.                                                                                                                                                                                                                                   |
| -w        | Gives the password for the bind DN.                                                                                                                                                                                                                  |
| -x        | Disables SASL to allow a simple bind over an SSL/TLS connection.                                                                                                                                                                                     |
| -a        | <i>Optional.</i> Gives the old password, which is being changed.                                                                                                                                                                                     |

| Parameter         | Description                                                                       |
|-------------------|-----------------------------------------------------------------------------------|
| <code>-s</code>   | <i>Optional.</i> Sets the new password.                                           |
| <code>user</code> | <i>Optional.</i> Gives the DN of the user entry for which to change the password. |

To use Start TLS, which runs the command on a non-secure port, run **ldappasswd** with the **-ZZ** option and the standard LDAP port number. The password extended change operation has the following format:

```
ldappasswd -x -D bind_dn -w password -p server_port -h server_hostname -Z
[-a oldPassword] [-s newPassword] [user]
```



## NOTE

For Start TLS connections to work, the SSL/TLS environment variables must be configured as described in [Section A.2, “Using SSL/TLS and Start TLS with LDAP Client Tools”](#).

Use the **-ZZ** to force the connection to be successful.

To modify an entry's password, run **ldappasswd** like any other LDAP operation. It is not necessary to specify a `user` if the account is the same as that given in the bind DN. For example:

```
ldappasswd -x -h ldap.example.com -p 389 -ZZ -D
"uid=jsmith,ou=People,dc=example,dc=com" -W -s newpassword
```

To change the password on an entry other than the one specified in the bind credentials, run **ldappasswd** as shown below, adding the `user` DN to the operation and providing separate credentials, as follows:

```
ldappasswd -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -ZZ -W -s newpassword
"uid=jsmith,ou=People,dc=example,dc=com"
```

Access control is enforced for the password change operation. If the bind DN does not have rights to change the specified password, the operation will fail with an **Insufficient rights** error.

## 14.2. MANAGING THE DIRECTORY MANAGER PASSWORD

The Directory Manager is the privileged database administrator, comparable to the **root** user in UNIX. The Directory Manager entry and its associated password are created during installation. The default DN is **cn=Directory Manager**.

### 14.2.1. Changing the Directory Manager Password

The password for the Directory Manager superuser is defined in the **nsslapd-rootpw** attribute.

**NOTE**

The password can be changed using **ldapmodify** by sending the password in plaintext. Changing the password through the Directory Server Console ensures that the password is immediately hashed when it is saved in the **dse.ldif** file, so it is never saved in clear text.

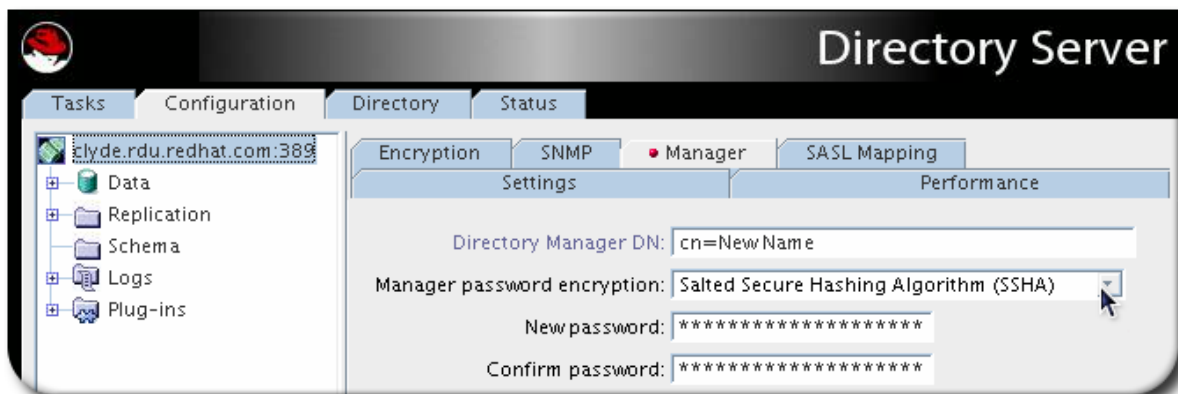
**IMPORTANT**

When resetting the Directory Manager's password from the command line, *do not* use curly braces (**{}**) in the password. The root password is stored in the format *{password-storage-scheme}hashed\_password*. Any characters in curly braces are interpreted by the server as the password storage scheme for the root password. If that text is not a valid storage scheme or if the password that follows is not properly hashed, then the Directory Manager cannot bind to the server.

**NOTE**

Both the Directory Manager password and its password storage scheme can be changed in the Directory Server Console. The password by itself can be changed. However, if the password storage scheme is changed then the password must also be changed, so that it can be rehashed and stored with the new scheme.

1. Log into the Directory Server Console as Directory Manager.
2. Select the **Configuration** tab, and then select the top entry in the navigation tree in the left pane.
3. Select the **Manager** tab in the right pane.



4. Enter a new password, and confirm it.

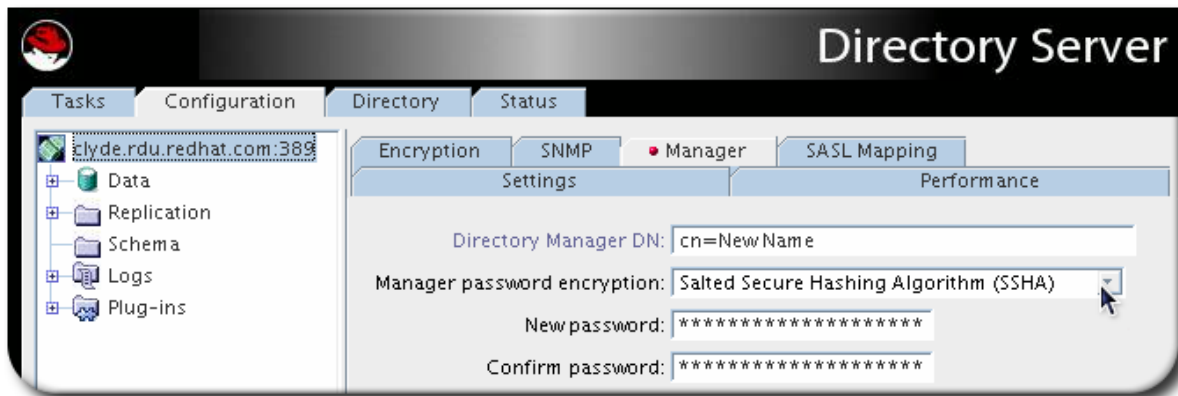
### 14.2.2. Changing the Directory Manager Password Storage Scheme

The **nsslapd-rootpw** attribute contains a hash of the password and an indication of the password storage scheme in braces, such as **{SSHA}**. If the password is in clear text, then the password storage scheme is **CLEAR**. The default password storage scheme is **SSHA**.

```
nsslapd-rootpw: {SSHA}od1V7JmQlMdldxr0lp3XSnMuXZVsXi8/YUVM7Q==
nsslapd-rootpwstorage: SSHA
```

To change the password storage scheme:

1. Log into the Directory Server Console as Directory Manager. For information on changing the bind DN, see [Section 1.4.3, “Changing the Login Identity”](#).
2. Select the **Configuration** tab, and then select the top entry in the navigation tree in the left pane.
3. Select the **Manager** tab in the right pane.



4. Set the storage scheme for the server to use to store the password for Directory Manager in the **Manager Password Encryption** pull-down menu.
5. Enter a new password, and confirm it.



#### NOTE

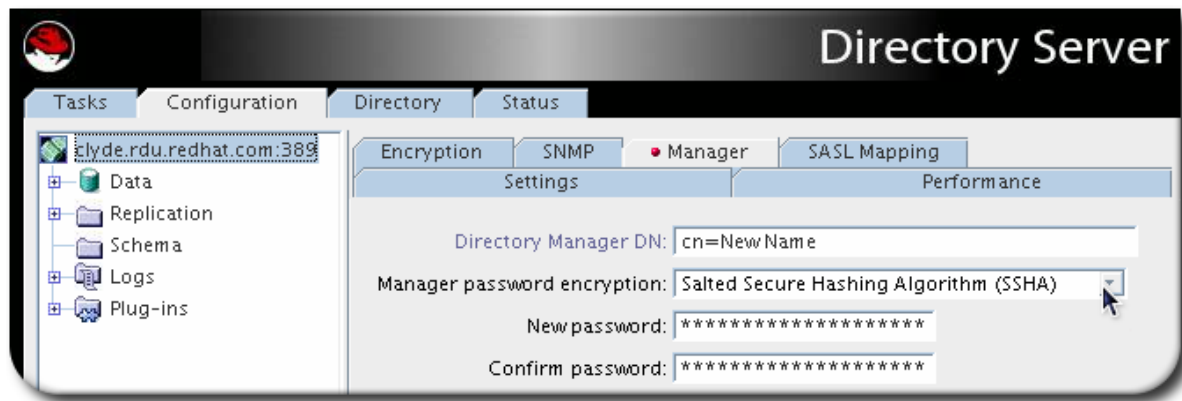
If the password storage scheme is changed, then the password must also be changed so that it can be rehashed and stored with the new scheme.

As always, *do not* use curly braces (`{}`) in the password. The root password is stored in the format `{password-storage-scheme}hashed_password`. Any characters in curly braces are interpreted by the server as the password storage scheme for the root password. If that text is not a valid storage scheme or if the password that follows is not properly hashed, then the Directory Manager cannot bind to the server.

### 14.2.3. Changing the Directory Manager DN

The default DN for the Directory Manager is **cn=Directory Manager**, which is created when the Directory Server is installed. This DN can be changed to a unique DN.

1. Log into the Directory Server Console as Directory Manager.
2. Select the **Configuration** tab, and then select the top entry in the navigation tree in the left pane.
3. Select the **Manager** tab in the right pane.



4. Change the distinguished name for the Directory Manager in the **Root DN** field. The default value is **cn=Directory Manager**.

### 14.3. CHECKING ACCOUNT AVAILABILITY FOR PASSWORDLESS ACCESS

Most of the time, for the Directory Server to return authentication information about a user account, a client actually binds (or attempts to bind) as that user. And a bind attempt requires some sort of user credentials, usually a password or a certificate. While the Directory Server allows unauthenticated binds and anonymous binds, neither of those binds returns any user account information.

There are some situations where a client requires information about a user account — specifically whether an account should be allowed to authenticate — in order to perform some other operation, but the client either does not have or does use any credentials for the user account in Directory Server. Essentially, the client needs to perform a credential-less yet authenticated bind operation to retrieve the user account information (including password expiration information, if the account has a password).

This can be done through an **ldapsearch** by passing the *Account Usability Extension Control*. This control acts as if it performs an authenticated bind operation for a given user and returns the account status for that user — but without actually binding to the server. This allows a client to determine whether that account can be used to log in and then to pass that account information to another application, like PAM.

For example, using the Account Usability Extension Control can allow a system to use the Directory Server as its identity back end to store user data but to employ password-less authentication methods, like smart cards or SSH keys, where the authentication operation is performed outside Directory Server.

#### 14.3.1. Searching for Entries Using the Account Usability Extension Control

The Account Usability Extension Control is an extension for an **ldapsearch**. It returns an extra line for each returned entry that gives the account status and some information about the password policy for that account. A client or application can then use that status to evaluate authentication attempts made outside Directory Server for that user account. Basically, this control signals whether a user should be allowed to authenticate without having to perform an authentication operation.



#### NOTE

The OpenLDAP tools used by Directory Server do not support the Account Usability Extension Control. Other LDAP utilities, like OpenDS, can be used or other clients which do support the control.



For example, using the OpenDS tools, the control can be specified using the **-J** with the control OID (1.3.6.1.4.1.42.2.27.9.5.8) or with the **accountusability:true** flag:

```
[jsmith@server ~]$ pathToOpenDsLdapTools/bin/ldapsearch -D "cn=directory
manager" -W -p 389 -h server.example.com -b "dc=example,dc=com" -s sub -J
"accountusability:true" "(objectclass=*)"
Account Usability Response Control
The account is usable
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
...
```

This can also be run for a specific entry:

```
[jsmith@server ~]$ pathToOpenDsLdapTools/bin/ldapsearch -D "cn=directory
manager" -W -p 389 -h server.example.com -b
"uid=bjensen,ou=people,dc=example,dc=com" -s base -J
"accountusability:true" "(objectclass=*)"
Account Usability Response Control
The account is usable
dn: uid=bjensen,ou=people,dc=example,dc=com
...
```



**NOTE**

By default, only the Directory Manager can use the Account Usability Extension Control. To allow other users to use the Account Usability Extension Control, set on ACL on the supported control entry under **cn=features**. See [Section 14.3.2, “Changing What Users Can Perform an Account Usability Search”](#).

The control returns different messages, depending on the actual status of the account and (if the user has a password) the password policy settings for the user account.

**Table 14.3. Possible Account Usability Control Result Messages**

| Account Status                                         | Control Result Message                                       |
|--------------------------------------------------------|--------------------------------------------------------------|
| Active account with a valid password                   | The account is usable                                        |
| Active account with no password set                    | The account is usable                                        |
| Expired password                                       | Password expired                                             |
| The password policy for the account is modified        | Password expired                                             |
| The account is locked and there is no lockout duration | Password reset                                               |
| The account is locked and there is a lockout duration  | <i>Time</i> (in seconds) for automatic unlock of the account |



| Account Status                                                       | Control Result Message                        |
|----------------------------------------------------------------------|-----------------------------------------------|
| The password for the account should be reset at the first login      | Password reset                                |
| The password has expired and grace logins are allowed                | Password expired and X grace login is allowed |
| The password has expired and the number of grace logins is exhausted | Password expired                              |
| The password will expire (expiration warning)                        | Password will expire in X number of seconds   |

### 14.3.2. Changing What Users Can Perform an Account Usability Search

By default, only the Directory Manager can use the Account Usability Extension Control. Other users can use the Account Usability Extension Control by setting the appropriate ACI on the supported control entry. The control entry is named for the Account Usability Extension Control OID, 1.3.6.1.4.1.42.2.27.9.5.8.

For example:

```
[jsmith@server ~]$ ldapmodify -D "cn=directory manager" -W -x
dn: oid=1.3.6.1.4.1.42.2.27.9.5.8,cn=features,cn=config
changetype: modify
add: aci
aci: (targetattr != "aci")(version 3.0; acl "Account Usable"; allow (read,
search, compare, proxy)(groupdn =
"ldap:///cn=Administrators,ou=groups,dc=example,dc=com"));
```

### 14.3.3. Using pam\_ldap Account Management with the Account Usability Control

The **pam\_ldap** module allows a local system to use passwordless authentication mechanisms (like SSH keys, rlogin, and rsh) while still using an LDAP directory to store user information. The **pam\_ldap** module connects to the Directory Server to verify that an account is active, is not locked, and has a valid password (if applicable). When a user uses a passwordless authentication method, then the **pam\_ldap** module must retrieve the account status without authenticating to the LDAP server as the actual user.

PAM can be configured first for system account management, and then to use the **pam\_ldap** module as one of its authentication sources. On Red Hat Enterprise Linux, the PAM account management is configured in the **/etc/pam.d/other** file.

The default account configuration does not use PAM for account management:

```
account required pam_deny.so
```

This can be changed to use first system accounts and then LDAP-stored accounts for system authentication. For example:

```

account requisite pam_roles.so.1
account binding pam_unix_account.so.1 server_policy
account required pam_ldap.so.1

```

If a user is found in a system account, that account is used first (**pam\_unix\_account**). If the user is not found there, then PAM checks the LDAP server. The LDAP configuration is set in the `/etc/pam_ldap.conf` file. That configuration file needs to set only the connection information for the Directory Server instance, such as its base DN, host, and port. Because the Account Usability Plug-in is enabled by default, the Directory Server is already configured to allow Account Usability Control searches for clients.

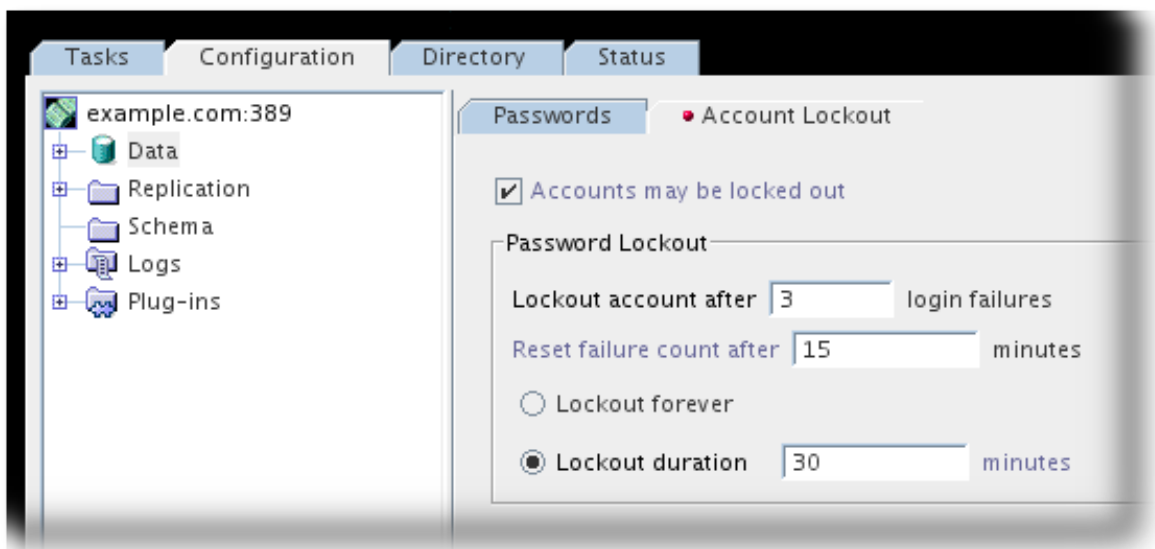
## 14.4. CONFIGURING A PASSWORD-BASED ACCOUNT LOCKOUT POLICY

A password-based account lockout policy protects against hackers who try to break into the directory by repeatedly trying to guess a user's password. The password policy can be set so that a specific user is locked out of the directory after a given number of failed attempts to bind.

### 14.4.1. Configuring the Account Lockout Policy Using the Console

To set up or modify the account lockout policy for the Directory Server:

1. Select the **Configuration** tab and then the **Data** node.
2. In the right pane, select the **Account Lockout** tab.



3. To enable account lockout, select the **Accounts may be locked out** check box.
4. Enter the maximum number of allowed bind failures in the **Lockout account after X login failures** text box. The server locks out users who exceed the limit specified here.
5. In the **Reset failure counter after X minutes** text box, enter the number of minutes for the server to wait before resetting the bind failure counter to zero.
6. Set the interval for users to be locked out of the directory.
  - Select the **Lockout Forever** radio button to lock users out until their passwords have

been reset by the administrator.

- o Set a specific lockout period by selecting the **Lockout Duration** radio button and entering the time (in minutes) in the text box.

7. Click **Save**.

### 14.4.2. Configuring the Account Lockout Policy Using the Command Line

Table 14.4, “Account Lockout Policy Attributes” describes the attributes available to configure the account lockout policy. Use **ldapmodify** to change these attributes in the **cn=config** entry. For example:

```
[jsmith@server ~]$ ldapmodify -D "cn=directory manager" -W -x -p 389 -h
server.example.com -x
dn: cn=config
changetype: modify
replace: passwordLockout
passwordLockout: on
-
add: passwordMaxFailure
passwordMaxFailure: 4
-
add: passwordLockoutDuration
passwordLockoutDuration: 600
-
```

**Table 14.4. Account Lockout Policy Attributes**

| Attribute Name     | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| passwordLockout    | This attribute indicates whether users are locked out of the directory after a given number of failed bind attempts. Set the number of failed bind attempts after which the user will be locked out using the <b>passwordMaxFailure</b> attribute. Users can be locked out for a specific time or until an administrator resets the password. This attribute is set to <b>off</b> by default, meaning that users will not be locked out of the directory. |
| passwordMaxFailure | This attribute indicates the number of failed bind attempts after which a user will be locked out of the directory. This attribute takes affect only if the <b>passwordLockout</b> attribute is set to <b>on</b> . This attribute is set to <b>3</b> bind failures by default.                                                                                                                                                                            |

| Attribute Name            | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| passwordUnlock            | This attribute sets whether a user can log back into the server without administrator intervention. The default is for this attribute to be on, meaning that the user <i>can</i> log back into the server after a certain lockout period has passed. If this attribute is turned off, then the user cannot log back in using that account until it is manually unlocked by an administrator.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| passwordLockoutDuration   | This attribute indicates the time, in seconds, that users will be locked out of the directory. The <b>passwordUnlock</b> attribute specifies if a user is locked out until the password is reset by an administrator (which means that the user is locked out indefinitely). If the <b>passwordUnlock</b> attribute is set to <b>on</b> , then the use can log in again as soon as the lockout duration time is reached. By default, the user is locked out for 3600 seconds.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| passwordResetFailureCount | This attribute specifies the time, in seconds, after which the password failure counter will be reset. Each time an invalid password is sent from the user's account, the password failure counter is incremented. If the <b>passwordLockout</b> attribute is set to <b>on</b> , users will be locked out of the directory when the counter reaches the number of failures specified by the <b>passwordMaxFailure</b> attribute. The account is locked out for the interval specified in the <b>passwordLockoutDuration</b> attribute, after which time the failure counter is reset to zero (0). Because the counter's purpose is to gage when a hacker is trying to gain access to the system, the counter must continue for a period long enough to detect a hacker. However, if the counter were to increment indefinitely over days and weeks, valid users might be locked out inadvertently. The reset password failure count attribute is set <b>600</b> seconds by default. |

### 14.4.3. Disabling Legacy Password Lockout Behavior

There are different ways of interpreting when the maximum password failure (**passwordMaxFailure**) has been reached. It depends on how the server counts the last failed attempt in the overall failure count.

The traditional behavior for LDAP clients is to assume that the failure occurs *after* the limit has been reached. So, if the failure limit is set to three, then the lockout happens at the fourth failed attempt. This also means that if the fourth attempt is successful, then the user can authenticate successfully, even though the user technically hit the failure limit. This is  $n+1$  on the count.

LDAP clients increasingly expect the maximum failure limit to look at the last failed attempt in the count as the final attempt. So, if the failure limit is set to three, then at the third failure, the account is locked. A fourth attempt, even with the correct credentials, fails. This is  $n$  on the count.

The first scenario — where an account is locked only if the attempt count is exceeded — is the historical behavior, so this is considered a legacy password policy behavior. In Directory Server, this policy is enabled by default, so an account is only locked when the failure count is  $n+1$ . This legacy behavior can be disabled so that newer LDAP clients receive the error (LDAP\_CONSTRAINT\_VIOLATION) when they expect it. This is set in the ***passwordLegacyPolicy*** parameter.

For example:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -D
"cn=directory manager" -w secret -p 389 -h server.example.com -x
dn: cn=config
replace: passwordLegacyPolicy
passwordLegacyPolicy: off
```

## 14.5. CONFIGURING TIME-BASED ACCOUNT LOCKOUT POLICIES

Aside from locking accounts for failed authentication attempts, another method of defining an account lockout policy is to base it on account inactivity or an account age. The Account Policy Plug-in uses a *relative* time setting to determine whether an account should be locked.



### NOTE

Roles or classes of service can be used to inactivate accounts based on *absolute* account times. For example, a CoS can be created that inactivates every account created before a certain date.

The Account Policy Plug-in requires three configuration entries:

- A configuration entry for the plug-in itself. This sets global values that are used for all account policies configured on that server.
- An account policy configuration entry. This entry is within the user directory and is essentially a template which is referenced and applied to user account entries.
- An entry which applies the account policy entry. A user account can reference an account policy directly or a CoS or role can be used to apply account policies to sets of user accounts automatically.



### NOTE

An account policy is applied through the ***acctPolicySubentry*** attribute. While this attribute can be added directly to user accounts, this attribute is single-valued — which means that only one account policy can be applied to that account.

That may be fine in most cases. However, an organization could realistically create two account policies, one for account inactivity and then another for account expiration based on age.

Using a CoS to apply account policies allows multiple account policies to be used for an account.

### 14.5.1. Account Policy Plug-in Syntax

The Account Policy Plug-in itself only has two configuration attributes:

- *nsslapd-pluginEnabled*, which sets whether the plug-in is enabled or disabled. This attribute is **off** by default.
- *nsslapd-pluginarg0*, which points to the DN of the plug-in configuration directory. The configuration entry is usually a child entry of the plug-in itself, such as **cn=config,cn=Account Policy Plugin,cn=plugins,cn=config**.

Past that, account policies are defined in two parts:

- The plug-in configuration entry identified in the *nsslapd-pluginarg0* attribute. This sets global configuration for the plug-in to use to identify account policy configuration entries and to manage user account entries. These settings apply across the server.

The configuration entry attributes are listed in [Table 14.5, “Account Policy Plug-in Attributes”](#).

- The account policy configuration entry. This is much like a template entry, which sets specific values for the account policies. User accounts — either directly or through CoS entries — reference this account policy entry.

The account policy and user entry attributes are listed in [Table 14.6, “Account Policy Entry and User Entry Attributes”](#).

**Table 14.5. Account Policy Plug-in Attributes**

| Attribute         | Definition                                                                                                                                                                                                                                                                                                                                                   |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| altstateattrname  | Sets a fallback attribute for the plug-in to use to calculate the expiration time, if the <b>stateattrname</b> attribute does not exist in the user account.                                                                                                                                                                                                 |
| alwaysRecordLogin | Sets whether to record the last login time for every user account, regardless of whether that account has an active account policy applied to it. By default, only entries with the <b>acctPolicySubentry</b> attribute on the entry have a login time recorded. Setting this to yes allows account policies to be applied indirectly, through roles or CoS. |
| limitattrname     | Sets the attribute in the account policy which is used to evaluate the account status. For example, if the <b>accountInactivityLimit</b> attribute is used, then the account policy is evaluated based on how long the account has been inactive.                                                                                                            |
| specattrname      | Sets what attribute <i>on a user account</i> (or CoS or role) is used to flag that that entry has an account policy applied to it.                                                                                                                                                                                                                           |

| Attribute     | Definition                                                                                                                                                                                                |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| stateattrname | Sets the attribute for the plug-in to use to calculate the expiration time. For example, for a policy based on account inactivity, this is generally the last login time ( <b><i>lastLoginTime</i></b> ). |

**Table 14.6. Account Policy Entry and User Entry Attributes**

| Attribute                               | Definition                                                                                                                                                                                           | Configuration or User Entry |
|-----------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------|
| accountpolicy (object class)            | Defines a template entry for account inactivation or expiration policies.                                                                                                                            | Configuration               |
| accountInactivityLimit (attribute)      | Sets the time period, in seconds, from the last login time of an account before that account is locked for inactivity.                                                                               | Configuration               |
| acctPolicySubentry (attribute)          | Identifies any entry which belongs to an account policy (specifically, an account lockout policy). The value of this attribute points to the DN of the account policy which is applied to the entry. | User                        |
| createTimestamp (operational attribute) | Contains the date and time that the entry was initially created.                                                                                                                                     | User                        |
| lastLoginTime (operational attribute)   | Contains a timestamp of the last time that the given account authenticated to the directory.                                                                                                         | User                        |

### 14.5.2. Configuring Time-Based Account Lockout Policies

1. Enable the Account Policy Plug-in.

```
[user@server ~]$ ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

```
dn: cn=Account Policy Plugin,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
```

2. Set the *nsslapd-pluginarg0* attribute to point to the plug-in configuration entry.

```
[user@server ~]$ ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x
```

```
dn: cn=Account Policy Plugin,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginarg0
nsslapd-pluginarg0: cn=config,cn=Account Policy
Plugin,cn=plugins,cn=config
```

### 3. Create the plug-in configuration entry.

- To use CoS or roles with account policies, set the ***alwaysRecordLogin*** value to **yes**. This means every entry has a login time recorded, even if it does not have the ***acctPolicySubentry*** attribute.
- Set the primary attribute to use for the account policy evaluation as value for ***stateAttrName***. For account inactivity, use the ***lastLoginTime*** attribute. For a simple account expiration time, use ***createTimestamp*** attribute.
- You can set a secondary attribute in ***altStateAttrName***, that is checked if the primary one defined in ***stateAttrName*** does not exist. If no attribute is specified as alternative the default value ***createTimestamp*** is used.



#### WARNING

If the value for the primary attribute is set to ***lastLoginTime*** and ***altStateAttrName*** to ***createTimestamp***, users in existing environments are automatically locked out when their accounts do not have the ***lastLoginTime*** attribute and the ***createTimestamp*** is older than the configured inactivity period.

To avert this situation, set the alternative attribute to **1.1**. This explicitly states to use no attribute as alternative. The ***lastLoginTime*** attribute will be created automatically after the user logs in the next time. Alternatively you can write a script to add the ***lastLoginTime*** attribute to each user account.

- Set the attribute to use to show which entries have an account policy applied to them (***acctPolicySubentry***).
- Set the attribute in the account policy which is used to set the actual timeout period, in seconds (***accountInactivityLimit***).

```
[user@server ~]$ ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x
```

```
dn: cn=config,cn=Account Policy Plugin,cn=plugins,cn=config
objectClass: top
objectClass: extensibleObject
cn: config
alwaysRecordLogin: yes
```



```
stateAttrName: lastLoginTime
altStateAttrName: 1.1
specattrname: acctPolicySubentry
limitattrname: accountInactivityLimit
```

4. Restart the server to load the new plug-in configuration.

```
[user@server ~]$ service dirsrv start
```

5. Define an account policy.

```
[user@server ~]$ ldapmodify -a -D "cn=directory manager" -W -p 389 -
h server.example.com -x

dn: cn=Account Inactivation Policy,dc=example,dc=com

objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: accountpolicy
accountInactivityLimit: 2592000
cn: Account Inactivation Policy
```

6. Create the class of service template entry.

```
[user@server ~]$ ldapmodify -a -D "cn=directory manager" -W -p 389 -
h server.example.com -x

dn: cn=TempltCoS,dc=example,dc=com

objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: cosTemplate
acctPolicySubentry: cn=Account Inactivation Policy,dc=example,dc=com
```

Account policies can be defined directly on user entries, instead of using a CoS. However, using a CoS allows an account policy to be applied and updated reliably for multiple entries and it allows multiple policies to be applied to an entry.

7. Create the class of service definition entry. The managed entry for the CoS is the account policy attribute, **acctPolicySubentry**. This example applies the CoS to the entire directory tree.

```
[user@server ~]$ ldapmodify -a -D "cn=directory manager" -W -p 389 -
h server.example.com -x

dn: cn=DefnCoS,dc=example,dc=com

objectClass: top
objectClass: ldapsubentry
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=TempltCoS,dc=example,dc=com
cosAttribute: acctPolicySubentry default operational-default
```

### 14.5.3. Tracking Login Times without Setting Lockout Policies

It is also possible to use the Account Policy Plug-in to track user login times *without* setting an expiration time or inactivity period. In this case, the Account Policy Plug-in is used to add the ***lastLoginTime*** attribute to user entries, but no other policy rules need to be set.

In that case, set up the Account Policy Plug-in as normal, to track login times. However, do not create a CoS to act on the login information that is being tracked.

1. Enable the Account Policy Plug-in.

```
[user@server ~]$ ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=Account Policy Plugin,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
```

2. Set the *nsslapd-pluginarg0* attribute to point to the plug-in configuration entry.

```
[user@server ~]$ ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=Account Policy Plugin,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginarg0
nsslapd-pluginarg0: cn=config,cn=Account Policy
Plugin,cn=plugins,cn=config
```

3. Create the plug-in configuration entry to record login times.

- Set the ***alwaysRecordLogin*** value to yes so that every entry has a login time recorded.
- Set the ***lastLoginTime*** attribute as the attribute to use for the account policy (***stateattrname***).
- Set the attribute to use to show which entries have an account policy applied to them (***acctPolicySubentry***).
- Set the attribute in the account policy which is used to set the actual timeout period, in seconds (***accountInactivityLimit***).

```
[user@server ~]$ ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=config,cn=Account Policy Plugin,cn=plugins,cn=config
objectClass: top
objectClass: extensibleObject
cn: config
alwaysRecordLogin: yes
stateattrname: lastLoginTime
```

```
altstateattrname: createTimeStamp
specattrname: acctPolicySubentry
limitattrname: accountInactivityLimit
```

4. Restart the server to load the new plug-in configuration.

```
[user@server ~]$ service dirsrv start
```

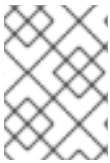
#### 14.5.4. Unlocking Inactive Accounts

Accounts which are inactivated through the Account Policy Plug-in cannot be managed with the tools that are used to manage lockouts that are set manually by the administrator (**ns-activate.pl**) or through the password policy.

If an account is locked because it reached the inactivity limit, it can be reactivated by resetting the **lastLoginTime** attribute. For example:

```
[user@server ~]$ ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: modify
replace: lastLoginTime
lastLoginTime: 20160610080000Z
```



#### NOTE

The **lastLoginTime** is set in GMT/UTC time (Zulu time zone) indicated by the appended **Z** to the time stamp.

### 14.6. REPLICATING ACCOUNT LOCKOUT ATTRIBUTES

Account lockout policies will block a user ID from being able to access the Directory Server if the login attempt fails a set number of times. This prevents hackers or other malicious people from illegitimately accessing the Directory Server by guessing a password. Password policies are set locally, and generally account lockout attributes are local to each replica. This means that a person can attempt to log in to one replica until the account lockout count is reached, then try again immediately on another replica. The way to prevent that is to replicate the attributes related to the account lockout counts for an entry, so that the malicious user is locked out of every supplier and consumer replica in the configuration if a login attempt fails on a single master.

By default, three password policy attributes are not replicated, even if other password attributes are. These attributes are related to of login failures and lockout periods:

- **passwordRetryCount**
- **retryCountResetTime**
- **accountUnlockTime**

#### 14.6.1. Managing the Account Lockouts and Replication

Password and account lockout policies are enforced in a replicated environment slightly differently:

- Password policies are enforced on the data master.
- Account lockout is enforced on all servers participating in replication.

Some of the password policy information in the directory is replicated automatically:

- ***passwordMinAge*** and ***passwordMaxAge***
- ***passwordExp***
- ***passwordWarning***

However, the configuration information is kept locally and is not replicated. This information includes the password syntax and the history of password modifications. Account lockout counters and tiers are not replicated, either, unless specifically configured for replication.

When configuring a password policy in a replicated environment, make sure that these elements are in place, so password policies and account lockout settings are enforced consistently:

- Warnings from the server of an impending password expiration are issued by all replicas. This information is kept locally on each server, so if a user binds to several replicas in turn, they will be issued the same warning several times. In addition, if the user changes the password, it may take time for this information to filter to the replicas. If a user changes a password and then immediately rebinds, he may find that the bind fails until the replica registers the changes.
- The same bind behavior should occur on all servers, including suppliers and replicas. Make sure to create the same password policy configuration information on each server.
- Account lockout counters may not work as expected in a multi-mastered environment. Account lockout counters are not replicated by default (although this can be configured). If account lockout attributes are not replicated at all, then a user could be locked out from one server but could successfully bind to another server (or, conversely, a user may be unlocked on one server and still blocked on another). If account lockout attributes are replicated, then there could be lags between an account lockout change on one server and when that change is propagated to the other servers. It depends on the replication schedule.
- Entries that are created for replication (for example, the server identities) need to have passwords that never expire. To make sure that these special users have passwords that do not expire, add the ***passwordExpirationTime*** attribute to the entry, and give it a value of **20380119031407Z** (the top of the valid range).



#### NOTE

If the password policy is enabled and the ***alwaysRecordLogin*** parameter set to **yes**, the value of the ***lastLoginTime*** attribute can be different on masters and read-only replicas. For example, if a user logs in to a read-only replica, the ***lastLoginTime*** attribute is updated locally but the value is not replicated to the master servers.

### 14.6.2. Configuring Directory Server to Replicate Password Policy Attributes

A special core configuration attribute controls whether password policy operational attributes are replicated. This is the ***passwordIsGlobalPolicy*** attribute, which is enabled in the consumer Directory Server configuration to allow the consumer to accept password policy operational attributes.

By default, this attribute is set to **off**.

To enable these attributes to be replicated, change the ***passwordIsGlobalPolicy*** configuration attribute on the consumer:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x -h
consumer1.example.com

dn: cn=config
changetype: modify
replace: passwordIsGlobalPolicy
passwordIsGlobalPolicy: on
```

Changing that value to **on** allows the ***passwordRetryCount***, ***retryCountResetTime***, and ***accountUnlockTime*** to be replicated. No other configuration is necessary for the attributes to be included with the replicated attributes.

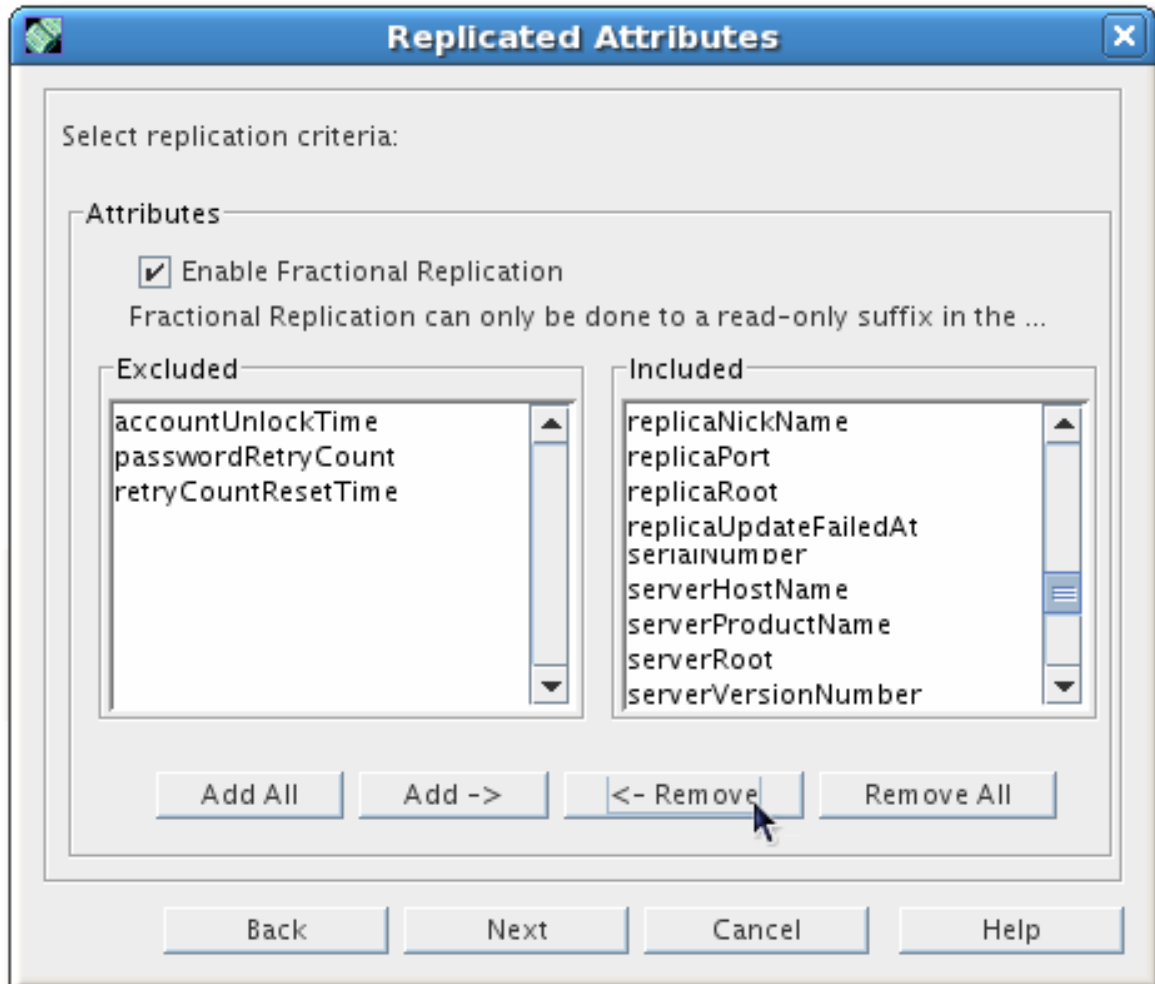
### 14.6.3. Configuring Fractional Replication for Password Policy Attributes

Setting the ***passwordIsGlobalPolicy*** attribute affects the consumer in replication, in that it allows the consumer to receive updates to those attributes. To control whether the password policy attributes are actually replicated by the supplier, use fractional replication, which controls what specific entry attributes are replicated.

If the password policy attributes should be replicated, then make sure these attributes are included in the fractional replication agreement (as they are by default).

If the ***passwordIsGlobalPolicy*** attribute is set to **off** on the consumer, so no password policy attributes should be replicated, use fractional replication (described in [Section 11.1.7, “Replicating a Subset of Attributes with Fractional Replication”](#)) to enforce that on the supplier and specifically exclude those attributes from the replication agreement.

1. When configuring the replication agreement on the supplier, as described (for example) in [Section 11.4.3, “Creating the Replication Agreement”](#), select the **Enable Fractional Replication** check box.
2. By default, every attribute is listed in the **Replicated Attributes** box. Select the ***passwordRetryCount***, ***retryCountResetTime***, and ***accountUnlockTime*** parameters and click the arrow button to move them into the **Do Not Replicate** box.



3. Finish configuring the replication agreement.

## 14.7. SYNCHRONIZING PASSWORDS

Password changes in a Directory Server entry can be synchronized to password attributes in Active Directory entries by using the **Password Sync** utility.

When passwords are synchronized, password policies are enforced on each sync peer locally. The syntax or minimum length requirements on the Directory Server apply when the password is changed in the Directory Server. When the changed password is synced over to the Windows server, the Windows password policy is enforced. The password policies themselves are not synchronized.

Configuration information is kept locally and cannot be synchronized, including the password change history and the account lockout counters.

When configuring a password policy for synchronization, consider the following points:

- The **Password Sync** utility must be installed locally on the Windows machine that will be synchronized with a Directory Server.
- **Password Sync** can only link the Windows machine to a single Directory Server; to sync changes with multiple Directory Server instances, configure the Directory Server for multi-master replication.
- Password expiration warnings and times, failed bind attempts, and other password-related information is enforced locally per server and is not synchronized between sync peer servers.

- The same bind behavior should occur on all servers. Make sure to create the same or similar password policies on both Directory Server and Active Directory servers.
- Entries that are created for synchronization (for example, the server identities) need to have passwords that never expire. To make sure that these special users have passwords that do not expire, add the ***passwordExpirationTime*** attribute to the Directory Server entry, and give it a value of **20380119031407Z** (the top of the valid range).

See [Chapter 12, \*Synchronizing Red Hat Directory Server with Microsoft Active Directory\*](#) for more information on synchronizing Directory Server and Windows users and passwords.

## 14.8. ENABLING DIFFERENT TYPES OF BINDS

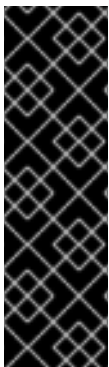
Whenever an entity logs into or accesses the Directory Server, it *binds* to the directory. There are many different types of bind operation, sometimes depending on the method of binding (such as simple binds or autobind) and some depending on the identity of user binding to the directory (anonymous and unauthenticated binds).

The following sections contain configuration parameters that can increase the security of binds (as in [Section 14.8.1, “Requiring Secure Binds”](#)) or streamline bind operations (such as [Section 14.8.4, “Configuring Autobind”](#)).

### 14.8.1. Requiring Secure Binds

A simple bind is when an entity uses a simple bind DN-password combination to authenticate to the Directory Server. Although it is possible to use a password file rather than sending a password directly through the command line, both methods still require sending or accessing a plaintext password over the wire. That makes the password vulnerable to anyone sniffing the connection.

It is possible to require simple binds to occur over a secure connection (SSL/TLS or Start TLS), which effectively encrypts the plaintext password as it is sent with the bind operation. (It is also possible to use alternatives to simple binds, such as SASL authentication and certificate-based authentication.)



#### IMPORTANT

Along with regular users logging into the server and LDAP operations, server-to-server connections are affected by requiring secure connections for simple binds. Replication, synchronization, and database chaining can all use simple binds between servers, for instance.

Make sure that replication and sync agreements and chaining configuration specify secure connections if the ***nsslapd-require-secure-binds*** attribute is turned on. Otherwise, these operations will fail.



#### NOTE

Requiring a secure connection for bind operations only applies to *authenticated binds*. Bind operations without a password (anonymous and unauthenticated binds) can proceed over standard connections.

1. Add the ***nsslapd-require-secure-binds*** attribute to the **cn=config** entry:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
```

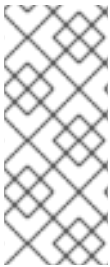
```
dn: cn=config
changetype: modify
replace: nsslapd-require-secure-binds
nsslapd-require-secure-binds: on
```

2. Restart the server.

```
service dirsrv restart
```

### 14.8.2. Disabling Anonymous Binds

If a user attempts to connect to the Directory Server without supplying any user name or password, this is an *anonymous bind*. Anonymous binds simplify common search and read operations, like checking the directory for a phone number or email address, by not requiring users to authenticate to the directory first.



#### NOTE

By default, anonymous binds are allowed (on) for search and read operations. This allows access to *regular directory entries*, which includes user and group entries as well as configuration entries like the root DSE. A different option, **rootdse**, allows anonymous search and read access to search the root DSE itself, but restricts access to all other directory entries.

However, there are risks with anonymous binds. Adequate ACIs must be in place to restrict access to sensitive information and to disallow actions like modifies and deletes. Additionally, anonymous binds can be used for denial of service attacks or for malicious people to gain access to the server.

[Section 13.9.1, “Granting Anonymous Access”](#) has examples on setting ACIs to control what anonymous users can access, and [Section 10.1.5, “Setting Resource Limits on Anonymous Binds”](#) has information on placing resource limits for anonymous users.

If those options do not offer a sufficient level of security, then anonymous binds can be disabled entirely.

1. Add the **nsslapd-allow-anonymous-access** attribute to the **cn=config** entry:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x

dn: cn=config
changetype: modify
replace: nsslapd-allow-anonymous-access
nsslapd-allow-anonymous-access: off
```

2. Restart the server.

```
service dirsrv restart
```



#### NOTE

When anonymous binds are disabled, unauthenticated binds are also disabled automatically.



### 14.8.3. Allowing Unauthenticated Binds

An *unauthenticated bind* is a bind where the user supplies a user name but not a password. For example, running an **ldapsearch** without supplying a password option:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -b "dc=example,dc=com" -s sub -x "(objectclass=*)"
```

When unauthenticated binds are allowed, the bind attempt goes through as an anonymous bind.

Unauthenticated binds are less secure than authenticated binds, and in some directories can be used to circumvent ACLs or performs denial-of-service attacks. This is why in Directory Server unauthenticated binds are disabled by default. If a user tries to bind without a password, the attempt fails:

```
ldap_simple_bind: DSA is unwilling to perform
ldap_simple_bind: additional info: Unauthenticated binds are not allowed
```

Unauthenticated binds only apply to bind attempts where a password is not given but a bind identity is. If the wrong password is given, the operation fails with an invalid credentials error:

```
ldap_simple_bind: Invalid credentials
```

If no bind ID or password is given, then the directory returns whatever information is allowed for an anonymous bind.

The ***nsslapd-allow-unauthenticated-binds*** attribute sets whether to allow an unauthenticated bind to succeed as an anonymous bind. Setting this parameter to **on** allows unauthenticated binds. By default, this parameter is **off**.

1. To configure unauthenticated binds, edit the Directory Server **cn=config** entry:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=config
changetype: replace
replace: nsslapd-allow-unauthenticated-binds
nsslapd-allow-unauthenticated-binds: on
```

2. Restart the server.

```
service dirsrv restart
```

### 14.8.4. Configuring Autobind

*Autobind* is a way to connect to the Directory Server based on local UNIX credentials, which are mapped to an identity stored in the directory itself. Autobind is configured in two parts:

Before configuring autobind, first make sure that LDAPv3 is enabled (in [Section 1.5, “Enabling LDAPv3”](#)). Then, configure the autobind mappings (in [Section 14.8.4.2, “Configuring Autobind”](#)).

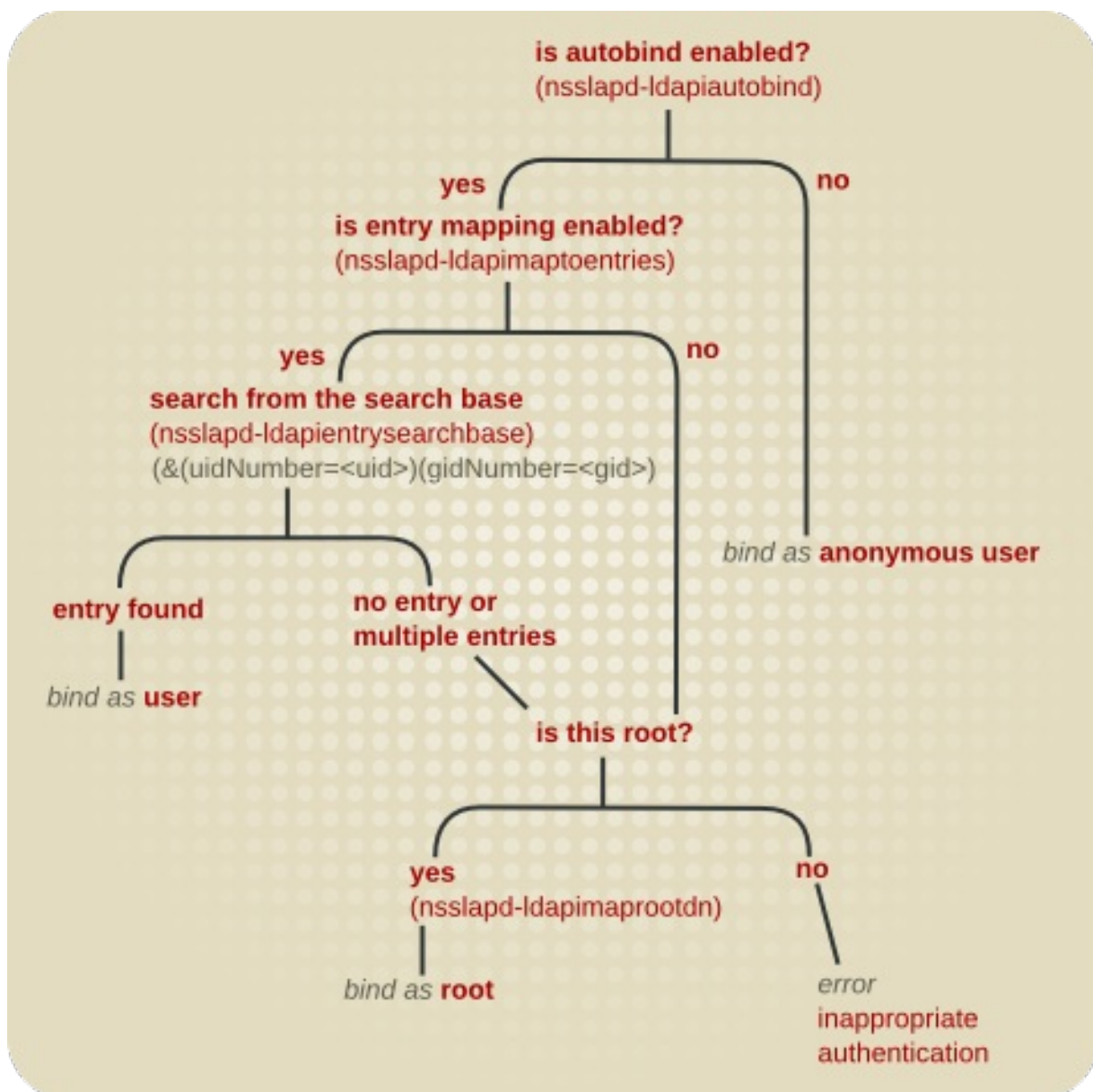
#### 14.8.4.1. Overview of Autobind and LDAPv3

Inter-process communication (IPC) is a way for separate processes on a Unix machine or a network to communicate directly with each other. *LDAPI* is a way to run LDAP connections over these IPC connections, meaning that LDAP operations can run over Unix sockets. These connections are much faster and more secure than regular LDAP connections.

The Directory Server uses these LDAPAPI connections to allow users to bind immediately to the Directory Server or to access the Directory Server using tools which support connections over Unix sockets. Autobind uses the *uid:gid* of the Unix user and maps that user to an entry in the Directory Server, then allows access for that user.

Autobind allows mappings to three directory entries:

- User entries, if the Unix user matches one user entry
- Directory Manager (or the super user defined in *nsslapd-ldapimaprootdn*), if the Unix user is **root**



**Figure 14.1. Autobind Connection Path**

The special autobind users are entries beneath a special autobind suffix (outside the regular user subtree). The entries underneath are identified by their user and group ID numbers:

-

```
gidNumber=gid+uidNumberuid, autobindsuffix
```

If autobind is not enabled but LDAPi is, then Unix users are anonymously bound to the Directory Server, unless they provide other bind credentials.



## NOTE

Autobind allows a client to send a request to the Directory Server without supplying a bind user name and password or using other SASL authentication mechanism. According to the LDAP standard, if bind information is not given with the request, the server processes the request as an anonymous bind. To be compliant with the standard, which requires some kind of bind information, any client that uses autobind should send the request with SASL/EXTERNAL.

For more information on configuring SASL, see [Section 7.11, “Setting up SASL Identity Mapping”](#).

### 14.8.4.2. Configuring Autobind

Configuring autobind alone allows anonymous access to the Directory Server. It is possible to enable mapping Unix users to entries and also to map **root** to Directory Manager.

1. Run **ldapmodify** to update the Directory Server configuration.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x
dn: cn=config
changetype: modify
```

2. Enable autobind.

```
replace: nsslapd-ldapiautobind
nsslapd-ldapiautobind: on
```

3. To map user entries, add four attributes:

- **nsslapd-ldapimaptoentries** to enable entry mapping
- **nsslapd-ldapiuidnumbertype** to set the Directory Server attribute to map to the Unix UID number
- **nsslapd-ldapigidnumbertype** to set the Directory Server attribute to map to the Unix group ID number
- **nsslapd-ldapientrysearchbase** to set the search base to use to find Directory Server user entries

```
add: nsslapd-ldapimaptoentries
nsslapd-ldapimaptoentries: on
-
add: nsslapd-ldapiuidnumbertype
nsslapd-ldapiuidnumbertype: uidNumber
-
```

```
add: nsslapd-ldapigidnumbertype
nsslapd-ldapigidnumbertype: gidNumber
-
add: nsslapd-ldapientrysearchbase
nsslapd-ldapientrysearchbase: ou=people,dc=example,dc=com
```

4. To map the **root** entry to Directory Manager, add the ***nsslapd-ldapimaprootdn*** attribute:

```
add: nsslapd-ldapimaprootdn
nsslapd-ldapimaprootdn: cn=Directory Manager
```

5. Restart the server to apply the new configuration.

```
service dirsrv restart example
```

## 14.9. USING PASS-THROUGH AUTHENTICATION

Pass-through authentication (PTA) is a mechanism which allows one Red Hat Directory Server instance to consult another to authenticate bind requests. Pass-through authentication is implemented through the PTA Plug-in; when enabled, the plug-in lets a Directory Server instance accept simple bind operations (password-based) for entries not stored in its local database.

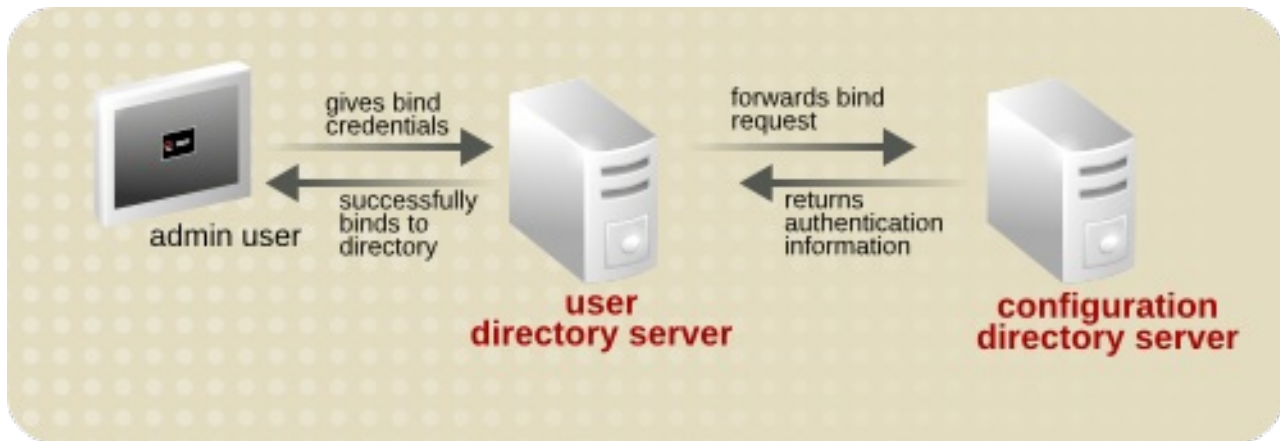
Directory Server uses PTA to administer the user and configuration directories on separate instances of Directory Server.

If the configuration directory and the user directory are installed on separate instances of Directory Server, the setup program automatically sets up PTA to allow the Configuration Administrator user (usually **admin**) to perform administrative duties.

PTA is required in this case because the **admin** user entry is stored under **o=NetScapeRoot** suffix in the configuration directory. Therefore, attempts to bind to the user directory as **admin** would normally fail. PTA allows the user directory to transmit the credentials to the configuration directory, which verifies them. The user directory then allows the **admin** user to bind.

The user directory in this example acts as the *PTA Directory Server*, the server that passes through bind requests to another Directory Server. The configuration directory acts as the *authenticating directory*, the server that contains the entry and verifies the bind credentials of the requesting client.

The *pass-through subtree* is the subtree *not* present on the PTA directory. When a user's bind DN contains this subtree, the user's credentials are passed on to the authenticating directory.



**Figure 14.2. Simple Pass-Through Authentication Process**



#### NOTE

The PTA Plug-in may not be listed in the Directory Server Console if the same server instance is used for the user directory and the configuration directory.

Here's how pass-through authentication works:

1. The configuration Directory Server (authenticating directory) is installed on machine A. The configuration directory always contains the configuration database and suffix, **o=NetscapeRoot**. In this example, the server name is **configdir.example.com**.
2. The user Directory Server (PTA directory) is then installed on machine B. The user directory stores the root suffix, such as **dc=example, dc=com**. In this example, the server name is **userdir.example.com**.
3. When the user directory is set up on machine B, the setup script prompts for the LDAP URL of the configuration directory on machine A.
4. The setup program enables the PTA Plug-in and configures it to use the configuration directory LDAP URL.

This entry contains the LDAP URL for the configuration directory. For example:

```
dn: cn=Pass Through Authentication,cn=plugins,
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: ldap://configdir.example.com/o=NetscapeRoot
...
```

The user directory is now configured to send all bind requests for entries with a DN containing **o=NetscapeRoot** to the configuration directory **configdir.example.com**.

5. When installation is complete, the **admin** user attempts to connect to the user directory to begin adding users.
6. The setup program adds the **admin** user's entry to the directory as **uid=admin, ou=TopologyManagement, o=NetscapeRoot**. So the user directory passes the bind request through to the configuration directory as defined by the PTA Plug-in configuration.

7. The configuration directory authenticates the user's credentials and sends the information back to the user directory.
8. The user directory allows the **admin** user to bind.

### 14.9.1. PTA Plug-in Syntax

PTA Plug-in configuration information is specified in the **cn=Pass Through Authentication, cn=plugins, cn=config** entry on the PTA directory (the user directory configured to pass through bind requests to the authenticating directory) using the required PTA syntax. There are only two attributes in this entry that are significant:

- *nsslapd-pluginEnabled*, which sets whether the plug-in is enabled or disabled. The value for this attribute can be **on** or **off**.
- *nsslapd-pluginarg0*, which points to the configuration directory. The value for this attribute is the LDAP URL of the server and suffix to which to pass the bind requests, along with the optional parameters, *maxconns*, *maxops*, *timeout*, *ldver*, *connlifetime*, *startTLS*.

The variable components of the PTA plug-in syntax are described in [Table 14.7, “PTA Plug-in Parameters”](#).

#### NOTE

The LDAP URL (**ldap|ldaps://authDS/subtree**) must be separated from the optional parameters (*maxconns*, *maxops*, *timeout*, *ldver*, *connlifetime*, *startTLS*) by a single space. If any of the optional parameters are defined, all of them must be defined, even if only the default values are used.

Several authenticating directories or subtrees can be specified by incrementing the ***nsslapd-pluginarg*** attribute suffix by one each time, as in [Section 14.9.3.2, “Specifying Multiple Authenticating Directory Servers”](#). For example:

```
nsslapd-pluginarg0: LDAP URL for the first server
nsslapd-pluginarg1: LDAP URL for the second server
nsslapd-pluginarg2: LDAP URL for the third server
...
```

The optional parameters are described in the following table in the order in which they appear in the syntax.

**Table 14.7. PTA Plug-in Parameters**

| Variable   | Definition                                                                                                                                                                                        |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| state      | Defines whether the plug-in is enabled or disabled. Acceptable values are <b>on</b> or <b>off</b> .                                                                                               |
| ldap ldaps | Defines whether SSL is used for communication between the two Directory Servers. See <a href="#">Section 14.9.2.1, “Configuring the Servers to Use a Secure Connection”</a> for more information. |

| Variable | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| authDS   | <p>The authenticating directory host name. The port number of the Directory Server can be given by adding a colon and then the port number. For example,</p> <p><b>ldap://dirserver.example.com:389/</b>. If the port number is not specified, the PTA server attempts to connect using either of the standard ports:</p> <div><p>Port 389 if <b>ldap://</b> is specified in the URL.</p><p>Port 636 if <b>ldaps://</b> is specified in the URL.</p></div> <p>See <a href="#">Section 14.9.2.2, “Specifying the Authenticating Directory Server”</a> for more information.</p> |
| subtree  | <p>The <i>pass-through subtree</i>. The PTA Directory Server passes through bind requests to the authenticating Directory Server from all clients whose DN is in this subtree. See <a href="#">Section 14.9.2.3, “Specifying the Pass-Through Subtree”</a> for more information. This subtree must not exist on this server. To pass the bind requests for <b>o=NetscapeRoot</b> to the configuration directory, the subtree <b>o=NetscapeRoot</b> must not exist on the server.</p>                                                                                           |
| maxconns | <p><i>Optional.</i> The maximum number of connections the PTA directory can simultaneously open to the authenticating directory. The default is <b>3</b>. See <a href="#">Section 14.9.2.4, “Configuring the Optional Parameters”</a> for more information.</p>                                                                                                                                                                                                                                                                                                                |
| maxops   | <p><i>Optional.</i> The maximum number of simultaneous operations (usually bind requests) the PTA directory can send to the authenticating directory within a single connection. The default is <b>5</b>. See <a href="#">Section 14.9.2.4, “Configuring the Optional Parameters”</a> for more information.</p>                                                                                                                                                                                                                                                                |
| timeout  | <p><i>Optional.</i> The time limit, in seconds, that the PTA directory waits for a response from the authenticating Directory Server. If this timeout is exceeded, the server returns an error to the client. The default is <b>300</b> seconds (five minutes). Specify zero (<b>0</b>) to indicate no time limit should be enforced. See <a href="#">Section 14.9.2.4, “Configuring the Optional Parameters”</a> for more information.</p>                                                                                                                                    |



| Variable     | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ldver        | <i>Optional.</i> The version of the LDAP protocol used to connect to the authenticating directory. Directory Server supports LDAP version 2 and 3. The default is version 3, and Red Hat strongly recommends <i>against</i> using LDAPv2, which is old and will be deprecated. See <a href="#">Section 14.9.2.4, “Configuring the Optional Parameters”</a> for more information.                                                                                                                                                                                                                                                                                                                           |
| connlifetime | <i>Optional.</i> The time limit, in seconds, within which a connection may be used. If a bind request is initiated by a client after this time has expired, the server closes the connection and opens a new connection to the authenticating directory. The server will not close the connection unless a bind request is initiated and the directory determines the connection lifetime has been exceeded. If this option is not specified, or if only one host is listed, no connection lifetime will be enforced. If two or more hosts are listed, the default is <b>300</b> seconds (five minutes). See <a href="#">Section 14.9.2.4, “Configuring the Optional Parameters”</a> for more information. |
| startTLS     | <p><i>Optional.</i> A flag of whether to use Start TLS for the connection to the authenticating directory. Start TLS establishes a secure connection over the standard port, so it is useful for connecting using LDAP instead of LDAPS. The SSL server and CA certificates need to be available on both of the servers.</p> <p>The default is <b>0</b>, which is off. To enable Start TLS, set it to <b>1</b>. To use Start TLS, the LDAP URL must use <b>ldap:</b>, not <b>ldaps:</b>.</p> <p>See <a href="#">Section 14.9.2.4, “Configuring the Optional Parameters”</a> for more information.</p>                                                                                                      |

## 14.9.2. Configuring the PTA Plug-in

The only method for configuring the PTA plug-in is to modify the entry **cn=Pass Through Authentication,cn=plugins,cn=config**. To modify the PTA configuration:

1. Use the **ldapmodify** command to modify **cn=Pass Through Authentication,cn=plugins,cn=config**.
2. Restart Directory Server.

Before configuring any of the PTA Plug-in parameters, the PTA Plug-in entry must be present in the Directory Server. If this entry does not exist, create it with the appropriate syntax, as described in [Section 14.9.1, “PTA Plug-in Syntax”](#).





## NOTE

If the user and configuration directories are installed on different instances of the directory, the PTA Plug-in entry is automatically added to the user directory's configuration and enabled.

This section provides information about configuring the plug-in in the following sections:

- [Section 14.9.2.1, “Configuring the Servers to Use a Secure Connection”](#)
- [Section 14.9.2.2, “Specifying the Authenticating Directory Server”](#)
- [Section 14.9.2.3, “Specifying the Pass-Through Subtree”](#)
- [Section 14.9.2.4, “Configuring the Optional Parameters”](#)

### 14.9.2.1. Configuring the Servers to Use a Secure Connection

The PTA directory can be configured to communicate with the authenticating directory over SSL by specifying LDAPS in the LDAP URL of the PTA directory. For example:

```
nsslapd-pluginarg0: ldaps://ldap.example.com:636/o=NetscapeRoot
```

### 14.9.2.2. Specifying the Authenticating Directory Server

The authenticating directory contains the bind credentials for the entry with which the client is attempting to bind. The PTA directory passes the bind request to the host defines as the authenticating directory. To specify the authenticating Directory Server, replace *authDS* in the LDAP URL of the PTA directory with the authenticating directory's host name, as described in [Table 14.7, “PTA Plug-in Parameters”](#).

1. Use **ldapmodify** edit the PTA Plug-in entry.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=Pass Through Authentication,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginarg0
nsslapd-pluginarg0: ldap://dirserver.example.com/o=NetscapeRoot
```

Optionally, include the port number. If the port number is not given, the PTA Directory Server attempts to connect using either the standard port (389) for **ldap://** or the secure port (636) for **ldaps://**.

If the connection between the PTA Directory Server and the authenticating Directory Server is broken or the connection cannot be opened, the PTA Directory Server sends the request to the next server specified, if any. There can be multiple authenticating Directory Servers specified, as required, to provide failover if the first Directory Server is unavailable. All of the authentication Directory Server are set in the **nsslapd-pluginarg0** attribute.

Multiple authenticating Directory Servers are listed in a space-separate list of *host:port* pairs, with this format:

```
ldap|ldaps://host1:port1 host2:port2/subtree
```

2. Restart the server.

```
service dirsrv restart instance_name
```

### 14.9.2.3. Specifying the Pass-Through Subtree

The PTA directory passes through bind requests to the authenticating directory from all clients with a DN defined in the pass-through subtree. The subtree is specified by replacing the *subtree* parameter in the LDAP URL of the PTA directory.

The pass-through subtree must not exist in the PTA directory. If it does, the PTA directory attempts to resolve bind requests using its own directory contents and the binds fail.

1. Use the **ldapmodify** command to import the LDIF file into the directory.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=Pass Through Authentication,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginarg0
nsslapd-pluginarg0: ldap://dirserver.example.com/o=NetscapeRoot
```

For information on the variable components in this syntax, see [Table 14.7, “PTA Plug-in Parameters”](#).

2. Restart the server.

```
service dirsrv restart instance_name
```

### 14.9.2.4. Configuring the Optional Parameters

Additional parameters that control the PTA connection can be set with the LDAP URL.

```
ldap|ldaps://authDS/subtree maxconns, maxops, timeout, ldver,
connlifetime, startTLS
```

- The maximum number of connections the PTA Directory Server can open simultaneously to the authenticating directory, represented by *maxconns* in the PTA syntax. The default value is **3**.
- The maximum number of bind requests the PTA Directory Server can send simultaneously to the authenticating Directory Server within a single connection. In the PTA syntax, this parameter is *maxops*. The default value is **5**.
- The time limit for the PTA Directory Server to wait for a response from the authenticating Directory Server. In the PTA syntax, this parameter is *timeout*. The default value is **300** seconds (five minutes).
- The version of the LDAP protocol for the PTA Directory Server to use to connect to the authenticating Directory Server. In the PTA syntax, this parameter is *ldver*. The default is **LDAPv3**.
- The time limit in seconds within which a connection may be used. If a bind request is initiated by a client after this time has expired, the server closes the connection and opens a new connection.

to the authenticating Directory Server. The server will not close the connection unless a bind request is initiated and the server determines the timeout has been exceeded. If this option is not specified or if only one authenticating Directory Server is listed in the *authDS* parameter, no time limit will be enforced. If two or more hosts are listed, the default is **300** seconds (five minutes). In the PTA syntax, this parameter is *connlifetime*.

- Whether to use Start TLS for the connection. Start TLS creates a secure connection over a standard LDAP port. For Start TLS, the servers must have their server and CA certificates installed, but they do not need to be running in SSL.

The default is **0**, which means Start TLS is off. To enable Start TLS, set it to **1**. To use Start TLS, the LDAP URL must use **ldaps:**, not **ldap:**.

1. Use **ldapmodify** to edit the plug-in entry.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=Pass Through Authentication,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginarg0
nsslapd-pluginarg0: ldap://dirserver.example.com/o=NetscapeRoot
3,5,300,3,300,0
```

(In this example, each of the optional parameters is set to its default value.) Make sure there is a space between the *subtree* parameter, and the optional parameters.



#### NOTE

Although these parameters are optional, if any one of them is defined, they all must be defined, even if they use the default values.

2. Restart the server.

```
service dirsrv restart instance_name
```

### 14.9.3. PTA Plug-in Syntax Examples

This section contains the following examples of PTA Plug-in syntax in the **dse.ldif** file:

- [Section 14.9.3.1, “Specifying One Authenticating Directory Server and One Subtree”](#)
- [Section 14.9.3.2, “Specifying Multiple Authenticating Directory Servers”](#)
- [Section 14.9.3.3, “Specifying One Authenticating Directory Server and Multiple Subtrees”](#)
- [Section 14.9.3.4, “Using Non-Default Parameter Values”](#)
- [Section 14.9.3.5, “Specifying Different Optional Parameters and Subtrees for Different Authenticating Directory Servers”](#)

#### 14.9.3.1. Specifying One Authenticating Directory Server and One Subtree

This example configures the PTA Plug-in to accept all defaults for the optional variables. This

configuration causes the PTA Directory Server to connect to the authenticating Directory Server for all bind requests to the **o=NetscapeRoot** subtree. The host name of the authenticating Directory Server is **configdir.example.com**.

```
dn: cn=Pass Through Authentication,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: ldap://configdir.example.com/o=NetscapeRoot
...
```

#### 14.9.3.2. Specifying Multiple Authenticating Directory Servers

If the connection between the PTA Directory Server and the authenticating Directory Server is broken or the connection cannot be opened, the PTA Directory Server sends the request to the next server specified, if any. There can be multiple authenticating Directory Servers specified, as required, to provide failover if the first Directory Server is unavailable. All of the authentication Directory Server are set in the **nsslapd-pluginarg0** attribute. Multiple authenticating Directory Servers are listed in a space-separated list of *host:port* pairs. For example:

```
dn: cn=Pass Through Authentication,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: ldap://configdir.example.com:389
config2dir.example.com:1389/o=NetscapeRoot
...
```



#### NOTE

The **nsslapd-pluginarg0** attribute sets the authentication Directory Server; additional **nsslapd-pluginargN** attributes can set additional *suffixes* for the PTA Plug-in to use, but not additional *hosts*.

#### 14.9.3.3. Specifying One Authenticating Directory Server and Multiple Subtrees

The following example configures the PTA Directory Server to pass through bind requests for more than one subtree (using parameter defaults):

```
dn: cn=Pass Through Authentication,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: ldap://configdir.example.com/o=NetscapeRoot
nsslapd-pluginarg1: ldap://configdir.example.com/dc=example,dc=com
...
```

#### 14.9.3.4. Using Non-Default Parameter Values

This example uses a non-default value (**10**) only for the maximum number of connections parameter **maxconns**. Each of the other parameters is set to its default value. However, because one parameter is specified, all parameters must be defined explicitly in the syntax.

```
dn: cn=Pass Through Authentication,cn=plugins,cn=config
...
```

```
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: ldap://configdir.example.com/o=NetscapeRoot
10,5,300,3,300,1
...
```

### 14.9.3.5. Specifying Different Optional Parameters and Subtrees for Different Authenticating Directory Servers

To specify a different pass-through subtree and optional parameter values for each authenticating Directory Server, set more than one LDAP URL/optional parameters pair. Separate the LDAP URL/optional parameter pairs with a single space as follows.

```
dn: cn=Pass Through Authentication,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0:ldap://configdir.example.com/o=NetscapeRoot
10,15,30,3,600,0
nsslapd-pluginarg1:ldap://config2dir.example.com/dc=example,dc=com
7,7,300,3,300,1
...
```

## 14.10. USING PAM FOR PASS-THROUGH AUTHENTICATION

*Pass-through authentication* is when any authentication request is forwarded from one server to another service.

Many systems already have authentication mechanisms in place for Unix and Linux users. One of the most common authentication frameworks is *Pluggable Authentication Modules* (PAM). Since many networks already existing authentication services available, administrators may want to continue using those services. A PAM module can be configured to tell Directory Server to use an existing authentication store for LDAP clients.

PAM pass-through authentication in Red Hat Directory Server uses the PAM Pass-Through Authentication Plug-in, which enables the Directory Server to talk to the PAM service to authenticate LDAP clients.

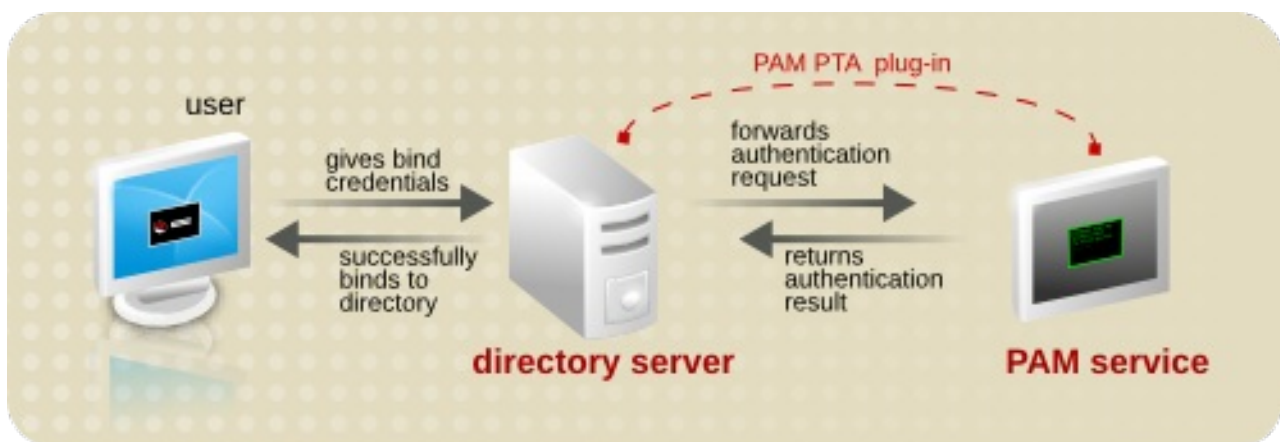


Figure 14.3. PAM Pass-Through Authentication Process

**NOTE**

PAM pass-through authentication works together with account inactivation when authenticating users, assuming that the appropriate mapping method (ENTRY) is used. However, PAM pass-through authentication does not validate passwords against password policies set either globally or locally, because the passwords are set and stored in the PAM module, not in the Directory Server.

### 14.10.1. PAM Pass-Through Authentication Configuration Options

PAM pass-through authentication is configured in child entries beneath the PAM Pass-Through Authentication plug-in container entry. There can be multiple PAM pass-through authentication policies, applied to different suffixes or to different entries within suffixes.

There are several different areas that can be configured for PAM pass-through:

- The suffixes that are controlled by the PAM pass-through authentication plug-in. This covers suffixes to exclude, suffixes to include, and how to handle a missing suffix.
- Individual entries within the configured suffixes which are the target of the authentication configuration. By default, all entries within a suffix are included in the authentication scope, but it is possible to configure multiple, different PAM Pass-Through Auth plug-in instances and then apply different plug-in configuration to different users.
- The PAM attribute mapping. The credentials that are offered to the Directory Server have to be mapped in some way to an LDAP entry and then, back to the credentials in the PAM service. This is done by defining a mapping method and then, optionally, which LDAP attribute to use to match the credentials.
- General configuration such as using SSL connections, the PAM service to use, and whether to fallback to LDAP authentication if PAM authentication fails.

**NOTE**

There can be multiple configuration instances of the PAM Pass-Through Authentication plug-in. An instance of the PAM Pass-Through Authentication plug-in can be applied to a subset of user entries by using the ***pamFilter*** attribute to set an LDAP filter to search for the specific entries to use with the plug-in.

**Table 14.8. PAM Pass-Through Auth Plug-in Attributes**

| Attribute        | Definition                                              |
|------------------|---------------------------------------------------------|
| pamExcludeSuffix | Identifies suffixes to exclude from PAM authentication. |
| pamIncludeSuffix | Identifies suffixes to include for PAM authentication.  |

| Attribute        | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| pamMissingSuffix | Identifies how to handle missing include or exclude suffixes. The options are ERROR (which causes the bind operation to fail); ALLOW, which logs an error but allows the operation to proceed; and IGNORE, which allows the operation and does not log any errors.                                                                                                                                                                                                                                                                                                              |
| pamFilter        | Sets an LDAP filter to use to identify specific entries within the included suffixes for which to use PAM pass-through authentication. If not set, all entries within the suffix are targeted by the configuration entry.                                                                                                                                                                                                                                                                                                                                                       |
| pamIDAttr        | Sets the name of the attribute holding the PAM ID.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| pamIDMapMethod   | <p>Gives the method to use to map the LDAP bind DN to a PAM identity.</p> <div data-bbox="815 909 922 1173" data-label="Image"> </div> <p><b>NOTE</b></p> <p>Directory Server user account inactivation is only validated using the ENTRY mapping method. With RDN or DN, a Directory Server user whose account is inactivated can still bind to the server successfully.</p>                                                                                                                                                                                                   |
| pamFallback      | Sets whether to fallback to regular LDAP authentication if PAM authentication fails.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| pamSecure        | Requires secure (TLS/SSL) connection for PAM authentication.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| pamService       | <p>Contains the service name to pass to PAM. This assumes that the service specified has a configuration file in <b>/etc/pam.d</b>.</p> <div data-bbox="815 1648 922 2040" data-label="Image"> </div> <p><b>IMPORTANT</b></p> <p>The <b>pam_fprintd.so</b> module cannot be in the configuration file referenced by the <b>pamService</b> attribute of the PAM Pass-Through Authentication Plug-in configuration. Using the PAM <b>fprintd</b> module causes the Directory Server to hit the max file descriptor limit and can cause the Directory Server process to abort.</p> |

| Attribute                | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|--------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| nsslapd-pluginConfigArea | <p>Specifies a different container entry to use as the parent for PAM pass-through authentication child entries. By default, the PAM Pass-Through Auth Plug-in entry is used as the parent entry for the configuration entries. This is set in the PAM Pass-Through Auth Plug-in entry.</p> <p>If a different container entry is used, then all PAM pass-through authentication child entries must be located beneath that container entry.</p> <p>All child entries in the specified location must belong to the <b>pamConfig</b> object class, but neither the container entry nor the PAM Pass-Through Auth Plug-in entry must belong to the <b>pamConfig</b> object class in that case.</p> |

#### 14.10.1.1. Specifying the Suffixes to Target for PAM PTA

The PAM PTA plug-in is applied globally, to all suffixes, by default unless they are explicitly excluded. Excluding and including suffixes can help target what areas in the directory use PAM authentication instead of LDAP authentication.



#### NOTE

The target of a PAM pass-through authentication entry must be a suffix, not an arbitrary subtree. As described in [Section 2.1, “Creating and Maintaining Suffixes”](#), a suffix is a subtree which is associated with a specific back end database, such as **cn=config** which is associated with **NetscapeRoot** or the root suffix **dc=example, dc=com** which is associated with **userRoot**.

The **pamExcludeSuffix** attribute excludes a suffix. By default, only the configuration subtree (**cn=config**) is excluded. Alternatively, the PAM PTA plug-in can be applied to a suffix with the **pamIncludeSuffix** attribute. Both of these attributes are multi-valued.

If the include attribute is set, for example, all other suffixes are automatically excluded. Likewise, if an exclude attribute is set, all other suffixes are automatically included.

```
pamExcludeSuffix: cn=config
pamExcludeSuffix: o=NetscapeRoot
```

With **pamIncludeSuffix**, only the given suffix is included and all others are automatically excluded. Since this attribute is multi-valued, more than one suffix can be included in the PAM evaluation by explicitly listing the suffixes.

```
pamIncludeSuffix: ou=Engineering, dc=example, dc=com
pamIncludeSuffix: ou=QE, dc=example, dc=com
```

The **pamMissingSuffix** attribute tells the server how to handle a failure if the specified suffix (include or exclude) does not exist. If it is set to **IGNORE**, then if the suffix does not exist, the plug-in simply skips that suffix and tries the next.

■



```
pamMissingSuffix: IGNORE
pamIncludeSuffix: ou=Engineering,dc=example,dc=com
pamIncludeSuffix: ou=Not Real,dc=example,dc=com
```

### 14.10.1.2. Applying Different PAM Pass-Through Authentication Configurations to Different Entries

By default, a PAM pass-through authentication policy applies to all entries within the designated suffixes. However, it is possible to specify an LDAP filter in the ***pamFilter*** attribute which identifies specific entries within the suffix to which to apply the PAM pass-through authentication policy.

This is useful for applying different PAM configurations or mapping methods to different user types, using multiple PAM pass-through authentication policies.

### 14.10.1.3. Setting PAM PTA Mappings

There has to be a way to connect the LDAP identity to the PAM identity. The first thing to define is the *method* to use to map the entries. There are three options: DN, RDN, and ENTRY. ENTRY uses a user-defined attribute in the entry.

Multiple mapping methods can be supplied in an ordered, space-separated list. The plug-in attempts to use each mapping method in the order listed until authentication succeeds or until it reaches the end of the list.

For example, this mapping method first maps the RDN method, then ENTRY, then DN, in the order the methods are listed:

```
pamIDMapMethod: RDN ENTRY DN
```

The different mapping methods are listed in [Table 14.9, “Mapping Methods for PAM Authentication”](#).



#### NOTE

Directory Server user account inactivation is only validated using the ENTRY mapping method. With RDN or DN, a Directory Server user whose account is inactivated can still bind to the server successfully.

**Table 14.9. Mapping Methods for PAM Authentication**

| Mapping | Description                                                                                                                                                                                                               |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RDN     | This method uses the value from the leftmost RDN in the bind DN. The mapping for this method is defined by Directory Server. This is the default mapping method, if none is given.                                        |
| ENTRY   | This method pulls the value of the PAM identity from a user-defined attribute in the bind DN entry. The identity attribute is defined in the <b><i>pamIDAttr</i></b> attribute.<br><br><pre>pamIDAttr: customPamUid</pre> |

| Mapping | Description                                                                                                                |
|---------|----------------------------------------------------------------------------------------------------------------------------|
| DN      | This method uses the full distinguished name from the bind DN. The mapping for this method is defined by Directory Server. |

#### 14.10.1.4. Configuring General PAM PTA Settings

Three general configuration settings can be set for PAM authentication:

- The service name to send to PAM (**pamService**); this is the name of the configuration file to use in **/etc/pam.d**
- Whether to require a secure connection (**pamSecure**)
- Whether to fall back to LDAP authentication if PAM authentication fails (**pamFallback**)

```
pamFallback: false
pamSecure: false
pamService: ldapserver
```

#### 14.10.2. Configuring PAM Pass-Through Authentication



##### NOTE

There can be multiple configuration instances of the PAM Pass-Through Authentication plug-in. An instance of the PAM Pass-Through Authentication plug-in can be applied to a subset of user entries by using the **pamFilter** attribute to set an LDAP filter to search for the specific entries to use with the plug-in.

PAM pass-through authentication is configured through the command line.

1. Make sure the PAM service is fully configured.
2. Remove the **pam\_fprintd.so** module from the PAM configuration file.



##### IMPORTANT

The **pam\_fprintd.so** module cannot be in the configuration file referenced by the **pamService** attribute of the PAM Pass-Through Authentication Plug-in configuration. Using the PAM **fprintd** module causes the Directory Server to hit the max file descriptor limit and can cause the Directory Server process to abort.

3. Enable the plug-in; this is disabled by default.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
```

```
dn: cn=PAM Pass-Through Auth Plugin,cn=plugins,cn=config
```

```
changetype: modify
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
```

4. Create the PAM Pass-Through Auth plug-in configuration entry.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=Admin PAM PTA Config,cn=PAM Pass-Through Auth
Plugin,cn=plugins,cn=config
cn: AD PAM PTA Config
```

5. Add the attributes available for the PAM plug-in. The available attributes are listed in [Table 14.8](#), “PAM Pass-Through Auth Plug-in Attributes”, and [Example 14.1](#), “Example PAM Pass-Through Authentication Configuration Entry” has an example entry.
6. Restart the server to load the new plug-in configuration.

```
service dirsrv restart
```

#### Example 14.1. Example PAM Pass-Through Authentication Configuration Entry

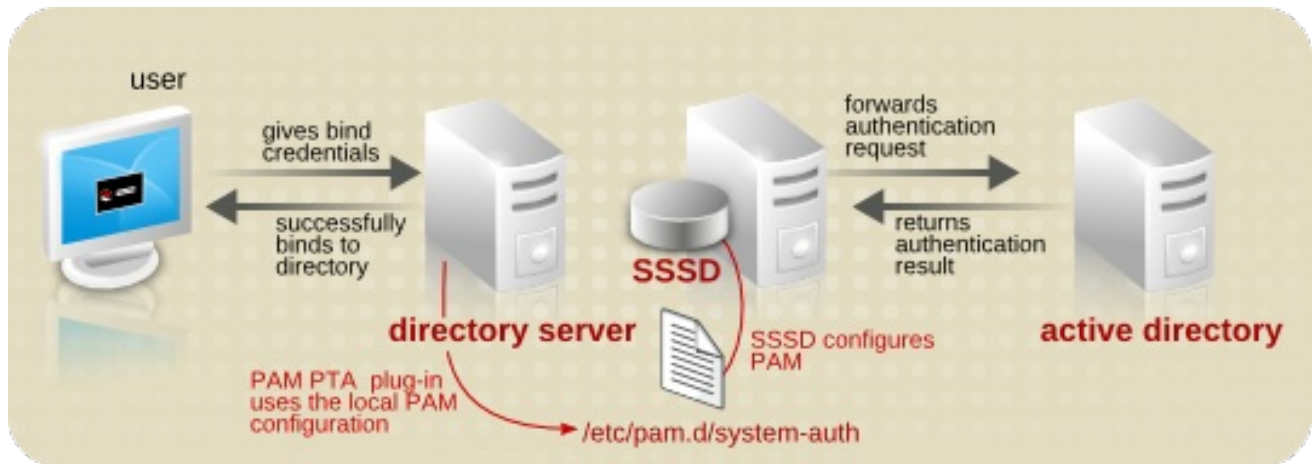
```
dn: cn=Admin PAM PTA Config,cn=PAM Pass Through
Auth,cn=plugins,cn=config
objectclass: top
objectclass: pamConfig
objectClass: nsSlapdPlugin
objectClass: extensibleObject
cn: Admin PAM PTA Config
pamMissingSuffix: ALLOW
pamExcludeSuffix: cn=config
pamExcludeSuffix: o=NetscapeRoot
pamIDMapMethod: RDN ENTRY
pamIDAttr: customPamUid
pamFilter: (manager=uid=bjensen,ou=people,dc=example,dc=com)
pamFallback: FALSE
pamSecure: TRUE
pamService: ldapserver
```

### 14.10.3. Using PAM Pass-Through Authentication with Active Directory as the Backend

PAM pass-through authentication forwards the credentials from the Directory Server to the PAM service. One option is to set up and configure PAM modules specifically for Directory Server. Another option — and one which may be more repeatable and more convenient in some infrastructures — is to use the System Security Services Daemon (SSSD) to configure PAM. Because SSSD can use a variety of different identity stores, a lot of different servers or services can be used to provide credentials, including Active Directory.

Using pass-through authentication through SSSD is a daisy chain of services. The PAM PTA Plug-in is configured as normal. It points to the given PAM service file to use. This service file is managed by

SSSD, and SSSD is configured to connect with whatever identity provider is required, even multiple providers.



**Figure 14.4. PAM Pass-Through Authentication with SSSD**

For more information on SSSD, see the [Red Hat Enterprise Linux 6 Deployment Guide](#).

To configure PAM pass-through authentication with Active Directory:

1. Configure SSSD to use the Active Directory server as one of its identity providers.

This configuration is covered in the [Red Hat Enterprise Linux 6 Deployment Guide](#).

2. Enable the PAM Pass-Through Auth plug-in; this is disabled by default.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x

dn: cn=PAM Pass-Through Auth Plugin,cn=plugins,cn=config
changetype: modify
replace: nsslapd-pluginEnabled
nsslapd-pluginEnabled: on
```

3. Create the PAM Pass-Through Auth plug-in configuration entry.

```
ldapmodify -a -D "cn=directory manager" -W -p 389 -h server.example.com -x

dn: cn=AD PAM PTA Config,cn=PAM Pass-Through Auth Plugin,cn=plugins,cn=config
cn: AD PAM PTA Config
```

4. Set the **pamService** attribute to point to the PAM configuration file managed by SSSD. By default, this is **/etc/pam.d/system-auth**.

```
pamService: system-auth
```



## IMPORTANT

The **pam\_fprintd.so** module cannot be in the configuration file referenced by the **pamService** attribute of the PAM Pass-Through Authentication Plug-in configuration. Using the PAM **fprintd** module causes the Directory Server to hit the max file descriptor limit and can cause the Directory Server process to abort.

5. Configure the ID map method and attribute. There are several options for how this can be done, depending on the Directory Server environment.

The simplest is to use the RDN map method, which automatically uses the **uid** attribute (or the correct naming attribute) to map Directory Server users back to Active Directory users (since Active Directory is the identity provider).

```
pamIDMapMethod: RDN
```

Similarly, this can be accomplished with the ENTRY map method by using the **samAccountName** attribute. If the user accounts in Directory Server are created with **uids** that match the **samAccountName** value for the user account in Active Directory, then the mapping is successful.

```
pamIDMapMethod: ENTRY
pamIDAttr: samAccountName
```

If Windows synchronization is configured, then the ENTRY method can be used with the **ntUserDomainId** attribute. The Directory Server and Active Directory user accounts are already synced, based on that attribute value, so the PAM mapping is successful.

```
pamIDMapMethod: ENTRY
pamIDAttr: ntUserDomainId
```

6. Restart the server to load the plug-in configuration.

```
service dirsrv restart
```

## 14.11. MANUALLY INACTIVATING USERS AND ROLES

A single user account or set of accounts can be temporarily inactivated. Once an account is inactivated, a user cannot bind to the directory. The authentication operation will fail.

Users and roles are inactivated using the operational attribute **nsAccountLock**. When an entry contains the **nsAccountLock** attribute with a value of **true**, the server rejects the bind.

The same procedures are used to inactivate users and roles. However, when a role is inactivated, the *members of the role* are inactivated, not the role entry itself. For more information about roles in general and how roles interact with access control in particular, see [Chapter 6, Organizing and Grouping Entries](#).



## WARNING

The root entry (the entry corresponding to the root or sub suffix) on a database cannot be inactivated. [Chapter 3, \*Creating Directory Entries\*](#) has information on creating the entry for a root or sub suffix, and [Chapter 2, \*Configuring Directory Databases\*](#) has information on creating root and sub suffixes.

### 14.11.1. Activating and Inactivating Users and Roles Using the Console

All user and role entries are active by default. They must be manually marked inactive and, once inactivated, must be manually re-activated.

1. Select the **Directory** tab.
2. Browse the navigation tree in the left navigation pane, and double-click the entry to inactivate.

The **Edit Entry** dialog box appears.

3. Click **Account** in the left pane. The right pane states that the role or user is activate. Click the **Inactivate** button to inactivate the user or role (or the **Activate** button, to re-enable the entry).

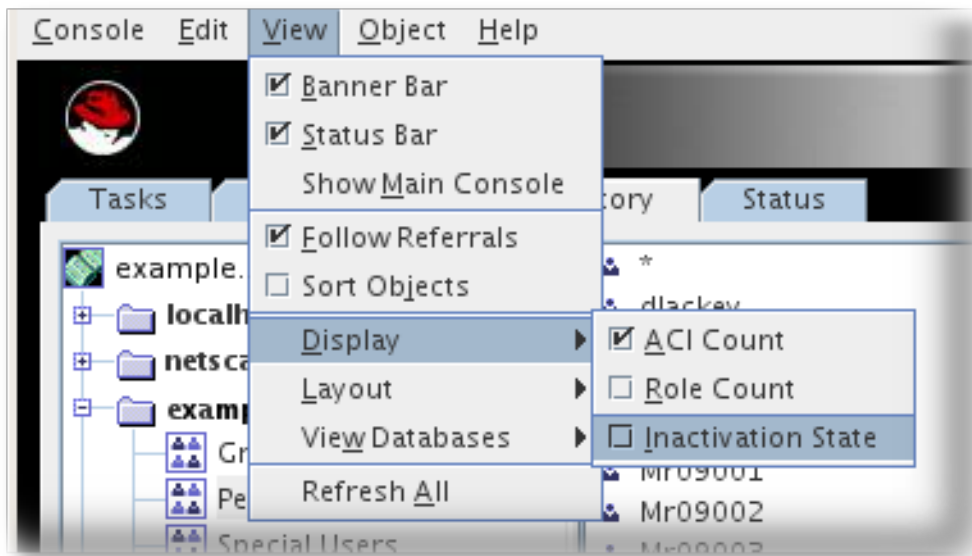


4. Click **OK**.

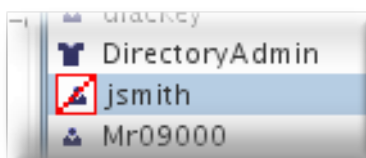
Alternatively, highlight the entry and select **Inactivate** (or **Activate**, if appropriate) from the **Object** menu.

### 14.11.2. Viewing Inactive Users and Roles

1. Select the **View** menu, and select the **Display** item.
2. Select the **Inactivation State** item.



When the inactivation state is visible, any inactive object is listed in the right pane of the Console with a red slash through it.



### 14.11.3. Inactivating and Activating Users and Roles Using the Command Line

The Directory Server uses dual scripts to inactivate or activate entries through the command line. The **ns-inactivate.pl** and **ns-activate.pl** script share similar options to identify the entry to modify, as listed in [Table 14.10, “ns-inactivate.pl and ns-activate.pl Options”](#).

For example, to inactivate a user account:

```
[root@server ~]# /usr/lib[64]/dirsrv/slapd-example/ns-inactivate.pl -D
Directory Manager -w secret -p 389 -h example.com -I
"uid=jfrasier,ou=people,dc=example,dc=com"
```

Then, the account can be re-activated:

```
[root@server ~]# /usr/lib[64]/dirsrv/slapd-example/ns-activate.pl -D
Directory Manager -w secret -p 389 -h example.com -I
"uid=jfrasier,ou=people,dc=example,dc=com"
```

**Table 14.10. ns-inactivate.pl and ns-activate.pl Options**

| Option Name | Description                                  |
|-------------|----------------------------------------------|
| -D          | The DN of the directory administrator.       |
| -w          | The password of the directory administrator. |

| Option Name | Description                                                                        |
|-------------|------------------------------------------------------------------------------------|
| -p          | Port used by the server.                                                           |
| -h          | Name of the server on which the directory resides.                                 |
| -l          | DN of the user account or role to inactivate or activate, depending on the script. |

For more information about running the **ns-inactivate.pl** and **ns-activate.pl** scripts, see the *Directory Server Configuration and Command-Line Tool Reference*.



## CHAPTER 15. MONITORING SERVER AND DATABASE ACTIVITY

This chapter describes monitoring database and Red Hat Directory Server logs. For information on using SNMP to monitor the Directory Server, see [Chapter 16, \*Monitoring Directory Server Using SNMP\*](#).

### 15.1. TYPES OF DIRECTORY SERVER LOG FILES

Directory Server provides three types of logs to help better manage the directory and tune performance:

- The access contains information on client connections and connection attempts to the Directory Server instance.
- The error log contains detailed messages of errors and events the directory experiences during normal operations.



#### WARNING

If the Directory Server fails to write to the errors log, the server sends the message to **syslog** and exits.

- The audit log records changes made to each database as well as to server configuration. This log is not enabled by default.

### 15.2. VIEWING LOG FILES

The access and error logs are enabled by default and can be viewed immediately. Before the audit log can be viewed, audit logging must be enabled for the directory, or the audit log will not be kept.



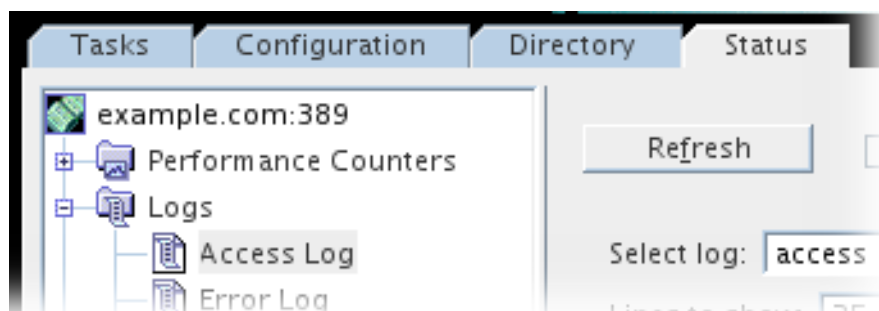
#### NOTE

When the server is not running, the log files cannot be viewed in the Directory Server Console, but they can be viewed in Admin Express. Open the Admin Server URL in a browser:

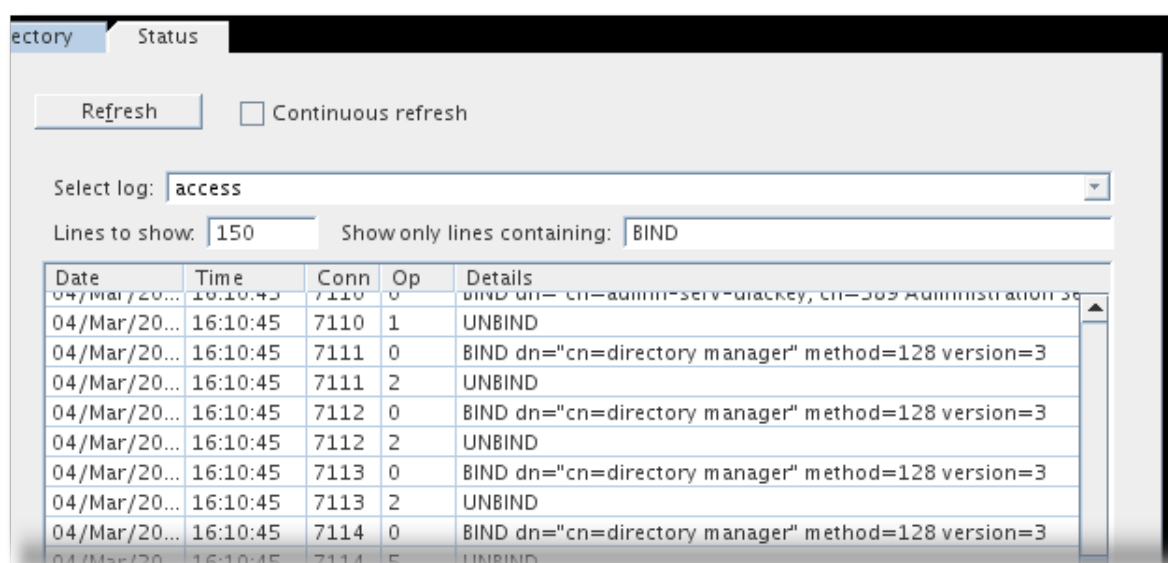
```
http://hostname:admin_server_port
```

Then log in with the admin login ID and password, and click the link for **Administration Express**.

1. In the Directory Server Console, select the **Status** tab.
2. In the navigation tree, expand the **Logs** folder. There are three folders available, for the access, error, and audit logs.



3. When you select the log type to view, a table displays a list of the last 25 entries in the selected log.
4. Optionally, change the settings of the log display and click **Refresh** to update the display.



- The **Select Log** pull-down menu allows you to select an archived (rotated) log rather than the currently active log.
- The **Lines to show** text box changes the number of log entries to display in the window.
- The **Show only lines containing** text box sets a filter, including regular expressions, to use to display only certain matching log entries.



#### NOTE

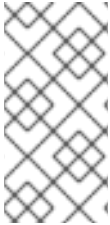
Selecting the **Continuous** check box refreshes the log display automatically every ten seconds. Continuous log refresh does not work well with log files over 10 megabytes.

## 15.3. CONFIGURING LOG FILES

For all types of log files, the log *creation* and log *deletion* policies have to be configured. The log creation policy sets when a new log file is started, and the log deletion policy sets when an old log file is deleted.

### 15.3.1. Enabling or Disabling Logs

The access and error logging is enabled by default. However, audit logging is disabled by default.

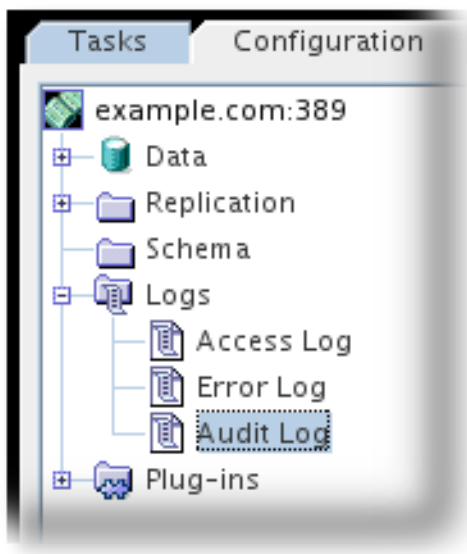


## NOTE

Disabling the access logging can be useful in some scenarios, because every 2000 accesses to the directory increases the log file by approximately 1 megabyte. However, before turning off access logging, consider that this information can help troubleshooting problems.

### Enabling or Disabling Logging in the Directory Server Console

1. Log in to the Directory Server Console.
2. Select the **Configuration** tab.
3. In the navigation tree, expand the **Logs** folder, and select the folder for the log to enable or disable.



4. To enable or disable logging, select the **Enable Logging** check box.
5. If the log is being enabled, enter the full path and file name for the Directory Server to use for logging in the field provided. The default path is `/var/log/dirsrv/slaped-instance/log_type`, such as `/var/log/dirsrv/slaped-instance/access`.
6. Click **Save**.

### Enabling or Disabling Logging Using the Command Line

You can use the `ldapmodify` utility to modify the parameters in the `cn=config` subtree that control the Directory Server logging feature:

- Access log: `nsslapd-accesslog-logging-enabled`
- Error log: `nsslapd-errorlog-logging-enabled`
- Audit log: `nsslapd-auditlog-logging-enabled`

For further details, see the corresponding section in the [Red Hat Directory Server Configuration, Command, and File Reference](#).

For example, to enable audit logging, enter:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=config
changetype: modify
replace: nsslapd-auditlog-logging-enabled
nsslapd-auditlog-logging-enabled: on
```

### 15.3.2. Defining a Log File Rotation Policy

To periodically archive the current log file and create a new one, set a log file rotation policy. You can update the settings in the **cn=config** subtree using the Directory Server Console or command line.

You can set the following configuration parameters to control the log file rotation policy:

#### Access mode

The access mode sets the file permissions on newly created log files.

- Access log: ***nsslapd-accesslog-mode***
- Error log: ***nsslapd-errorlog-mode***
- Audit log: ***nsslapd-auditlog-mode***

#### Maximum number of logs

Sets the maximum number of log files to keep. When the number of files is reached, Directory Server deletes the oldest log file before creating the new one.

- Access log: ***nsslapd-accesslog-maxlogspendir***
- Error log: ***nsslapd-errorlog-maxlogspendir***
- Audit log: ***nsslapd-auditlog-maxlogspendir***

#### File size for each log

Sets the maximum size of a log file in megabytes before it is rotated.

- Access log: ***nsslapd-accesslog-maxlogsize***
- Error log: ***nsslapd-errorlog-maxlogsize***
- Audit log: ***nsslapd-auditlog-maxlogsize***

#### Create a log every

Sets the maximum age of a log file.

- ***nsslapd-accesslog-logrotationtime*** and ***nsslapd-auditlog-logrotationtimeunit***
- ***nsslapd-errorlog-logrotationtime*** and ***nsslapd-errorlog-logrotationtimeunit***
- ***nsslapd-auditlog-logrotationtime*** and ***nsslapd-auditlog-logrotationtimeunit***

Additionally, you can set the time when the log file is rotated using the following parameters:

- ***nsslapd-accesslog-logrotationsynchour*** and ***nsslapd-accesslog-logrotationsyncmin***
- ***nsslapd-errorlog-logrotationsynchour*** and ***nsslapd-errorlog-logrotationsyncmin***
- ***nsslapd-auditlog-logrotationsynchour*** and ***nsslapd-auditlog-logrotationsyncmin***

For details, see the parameter descriptions in the corresponding section in the [Red Hat Directory Server Configuration, Command, and File Reference](#).

Each log file starts with a title, which identifies the server version, host name, and port, for ease of archiving or exchanging log files. For example:

```
389-Directory/1.2.11.15 B2016.312.1929
server.example.com:389 (/etc/dirsrv/slapd-instance)
```

## Configuring Log File Rotation in the Directory Server Console

1. Log in to the Directory Server Console.
2. Select the **Configuration** tab.
3. In the navigation tree, expand the **Logs** folder, and select the folder for the log you want to update the settings.
4. Set the logging settings in the **Creation policy** area. For example:

The screenshot shows the 'Creation Policy' configuration window for a log file. At the top, there is a checkbox labeled 'Enable logging' which is checked, and a 'View Log' button. Below this is the 'Log File' section, which contains a text field with the path '/var/log/dirsrv/slapd-example/errors' and a 'Browse...' button. The 'Creation Policy' section contains several settings: 'Access mode' is set to '600', 'Maximum number of logs' is set to '2', 'File size for each log' is set to '100' MB, and 'Create a new log every' is set to '1' with a dropdown menu showing 'Weeks'. There is also a checkbox for 'at' followed by two time input fields set to '0'.

5. Click **Save**.

## Configuring Log File Rotation Using the Command Line

You can use the **ldapmodify** utility to modify the parameters controlling the Directory Server logging features. For example for the error log, to set access mode **600**, to keep maximum **2**, and to rotate log files at a size of **100** MB or every **5** **days**, run:

■

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=config
changetype: modify
replace: nsslapd-errorlog-mode
nsslapd-errorlog-mode: 600
-
replace: nsslapd-errorlog-maxlogspersdir
nsslapd-errorlog-maxlogspersdir: 2
-
replace: nsslapd-errorlog-maxlogsize
nsslapd-errorlog-maxlogsize: 100
-
replace: nsslapd-errorlog-logrotationtime
nsslapd-errorlog-logrotationtime: 5
-
replace: nsslapd-errorlog-logrotationtimeunit
nsslapd-errorlog-logrotationtimeunit: day
```

### 15.3.3. Defining a Log File Deletion Policy

Directory Server automatically deletes old archived log files, if you set a **Deletion Policy**.



#### NOTE

You can only set a log file deletion policy if you have a log file rotation policy set. Directory Server applies the deletion policy at the time of log rotation.

You can set the following configuration parameters to control the log file deletion policy:

#### Total log size

If the size of all access, error, or audit log files increases the configured value, the oldest log file is automatically deleted.

- Access log: ***nsslapd-accesslog-logmaxdiskspace***
- Error log: ***nsslapd-errorlog-logmaxdiskspace***
- Audit log: ***nsslapd-auditlog-logmaxdiskspace***

#### Free disk space is less than

When the free disk space reaches this value, the oldest archived log file is automatically deleted.

- Access log: ***nsslapd-accesslog-logminfreediskspace***
- Error log: ***nsslapd-errorlog-logminfreediskspace***
- Audit log: ***nsslapd-auditlog-logminfreediskspace***

#### When a file is older than a specified time

When a log file is older than the configured time, it is automatically deleted.

- Access log: ***nsslapd-accesslog-logexpirationtime*** and ***nsslapd-accesslog-logexpirationtimeunit***

- Error log: *nsslapd-errorlog-logminfreediskspace* and *nsslapd-errorlog-logexpirationtimeunit*
- Audit log: *nsslapd-auditlog-logminfreediskspace* and *nsslapd-auditlog-logexpirationtimeunit*

For further details, see the corresponding section in the [Red Hat Directory Server Configuration, Command, and File Reference](#).

## Configuring a Log Deletion Policy in the Directory Server Console

1. Log in to the Directory Server Console.
2. Select the **Configuration** tab.
3. In the navigation tree, expand the **Logs** folder, and select the folder for the log you want to update the settings.
4. Set the logging settings in the **Deletion Policy** area. For example:

5. Click **Save**.

## Configuring Log Deletion Policy Using the Command Line

You can use the `ldapmodify` utility modify the parameters controlling the Directory Server logging features. For example, to auto-delete the oldest access log file if the total size of all access log files increases **500** MB, run:

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=config
changetype: modify
replace: nsslapd-accesslog-logmaxdiskspace
nsslapd-accesslog-logmaxdiskspace: 500
```

### 15.3.4. Manual Log File Rotation

The Directory Server supports automatic log file rotation for all three logs. However, it is possible to rotate log files manually if there are no automatic log file creation or deletion policies configured. By default, access, error, and audit log files can be found in the following location:

```
/var/log/dirsrv/slapd-instance
```

To rotate log files manually:

1. Shut down the server.

```
systemctl stop dirsrv.target instance
```

2. Move or rename the log file being rotated so that the old log file is available for future reference.
3. Restart the server.

```
systemctl restart dirsrv.target instance
```

### 15.3.5. Configuring Log Levels

Both the access and the error log can record different amounts of information, depending on the log level that is set.

You can set the following configuration parameters to control the log levels for the:

- Access log: *nsslapd-accesslog-level*
- Error log: *nsslapd-errorlog-level*

For further details and a list of the supported log levels, see the corresponding section in the [Red Hat Directory Server Configuration, Command, and File Reference](#).



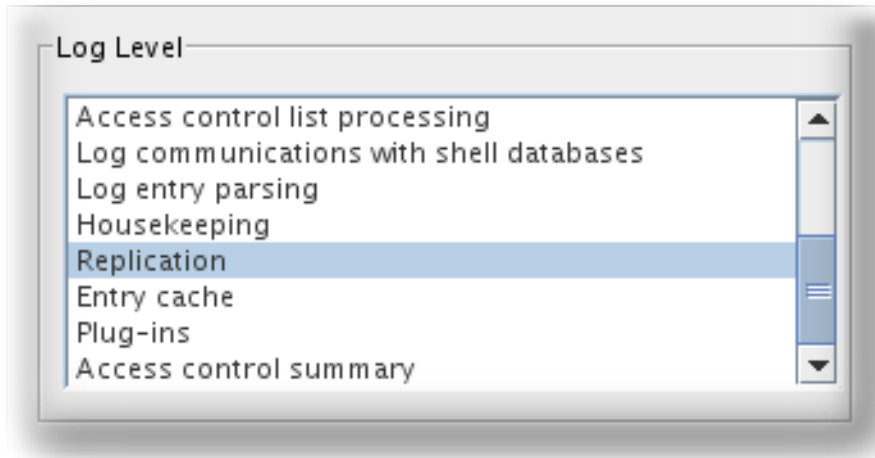
#### NOTE

Changing the log level from the default can cause the log file to grow very rapidly. Red Hat recommends not to change the default values without being asked to do so by the Red Hat technical support.

### Configuring the Log Level in the Directory Server Console

1. Log in to the Directory Server Console.
2. Select the **Configuration** tab.
3. In the navigation tree, expand the **Logs** folder, and select the folder for the log you want to update the settings.
4. Set the log level in the **Log Level** area. For example, for the error log file





5. Click **Save**.

### Configuring the Log Level Using the Command Line

You can use the **ldapmodify** utility to set the log level. For example, to enable search filter logging (32) and config file processing (64), set the **nsslapd-errorlog-level** parameter to **96** (32 + 64):

```
[root@server ~]# ldapmodify -D "cn=directory manager" -W -x
dn: cn=config
changetype: modify
replace: nsslapd-errorlog-level
nsslapd-errorlog-level: 96
```

## 15.4. GETTING ACCESS LOG STATISTICS

The **logconv.pl** script parses the access log and returns summary information on different users and operations that have been run on the server.

At its simplest, the script simply parses the access log (or logs):

```
logconv.pl /relative/path/to/accessLog
```

The script can accept wildcards to parse multiple access logs, which is useful if log rotation is used.

```
logconv.pl /var/log/dirsrv/slapd-instance/access*
```

The different options for **logconv.pl** are covered in the manpage and in the *Configuration and Command-Line Tool Reference*.

There are several different ways that **logconv.pl** can be used to pull general usage information from the access logs.

At its simplest, **logconv.pl** prints a list of total operations, total number of connections, counts per each operation type, counts for some extended operations like persistent searches, and bind information.

```
logconv.pl /var/log/dirsrv/slapd-instance/access
Access Log Analyzer 8.1
Command: logconv.pl /var/log/dirsrv/slapd-instance/access
Processing 1 Access Log(s)...
[001] /var/log/dirsrv/slapd-instance/access size (bytes): 141640
```

Total Log Lines Analysed: 1056

----- Access Log Output -----

Start of Logs: 18/Nov/2016:10:43:55

End of Logs: 18/Nov/2016:12:25:02

Processed Log Time: 1 Hours, 41 Minutes, 7 Seconds

Restarts: 3

Total Connections: 29

- LDAP Connections: 29

- LDAPi Connections: 0

- LDAPS Connections: 0

- StartTLS Extended Ops: 0

Peak Concurrent Connections: 5

Total Operations: 481

Total Results: 479

Overall Performance: 99.6%

Searches: 326 (0.05/sec) (3.22/min)

Modifications: 15 (0.00/sec) (0.15/min)

Adds: 107 (0.02/sec) (1.06/min)

Deletes: 0 (0.00/sec) (0.00/min)

Mod RDNs: 0 (0.00/sec) (0.00/min)

Compares: 0 (0.00/sec) (0.00/min)

Binds: 31 (0.01/sec) (0.31/min)

Proxied Auth Operations: 0

Persistent Searches: 0

Internal Operations: 0

Entry Operations: 0

Extended Operations: 0

Abandoned Requests: 0

Smart Referrals Received: 0

VLV Operations: 2

VLV Unindexed Searches: 0

VLV Unindexed Components: 2

SORT Operations: 14

Entire Search Base Queries: 42

Paged Searches: 0

Unindexed Searches: 0

Unindexed Components: 41

FDs Taken: 29

FDs Returned: 29

Highest FD Taken: 69

Broken Pipes: 0

Connections Reset By Peer: 0

Resource Unavailable: 0

Max BER Size Exceeded: 0

```

Binds: 31
Unbinds: 24
- LDAP v2 Binds: 0
- LDAP v3 Binds: 31
- AUTOBINDs: 0
- SSL Client Binds: 0
- Failed SSL Client Binds: 0
- SASL Binds: 0
- Directory Manager Binds: 20
- Anonymous Binds: 8
- Other Binds: 3

```

```

Cleaning up temp files...
Done.

```

In addition to the summary information for operations and connections, more detailed summary information for all of the connections to the server. This information includes things like most common IP addresses used to connect to the server, DN's with the most failed login attempts, total bind DN's used to access the server, and the most common error or return codes.

Additional connection summaries are passed as a single option. For example, listing the number of DN's used to connect to the server (**b**) and the total connection codes returned by the server (**c**) are passed as **-bc**.

```

logconv.pl -bc /var/log/dirsrv/slapd-instance/access
...
----- Total Connection Codes -----

U1 3 Cleanly Closed Connections
B1 1 Bad Ber Tag Encountered

----- Top 20 Bind DN's -----

Number of Unique Bind DN's: 212

1801 cn=directory manager
1297 Anonymous Binds
311 uid=jsmith,ou=people...
87 uid=bjensen,ou=peopl...
85 uid=mreynolds,ou=peo...
69 uid=jrockford,ou=peo...
55 uid=sspencer,ou=peop...
...

```

The data can be limited to entries after a certain start time (**-S**), before a certain end time (**-E**), or within a range. When start and end times are set, the **logconv.pl** first prints the time range given, then the summary for that period.

```

logconv.pl -S "[01/Jul/2016:16:11:47.0000000000 -0400]" -E "
[01/Jul/2016:17:23:08.9999999999 -0400]"
/var/log/dirsrv/slapd-instance/access
...
----- Access Log Output -----

```

```
Start of Logs: 01/Jul/2016:16:11:47
End of Logs: 01/Jul/2016:17:23:08
...
```

The start and end period only sets time limits for the data used to generate the total summary counts. It still shows aggregated, or total, counts. To get a view of the patterns in connections and operations to the Directory Server, it is possible to output data with counts per minute (**-M**) or per second (**-m**). In this case, the data are printed, in time unit increments, to a specified CSV output file.

```
logconv.pl -m|-M outputFile accessLogFile
```

For example:

```
logconv.pl -M /home/output/statsPerMin.txt
/var/log/dirsrv/slapd-instance/access*
```

The **-M** | **-m** options can also be used with the **-S** and **-E** arguments, to get per-minute or per-second counts within a specific time period.

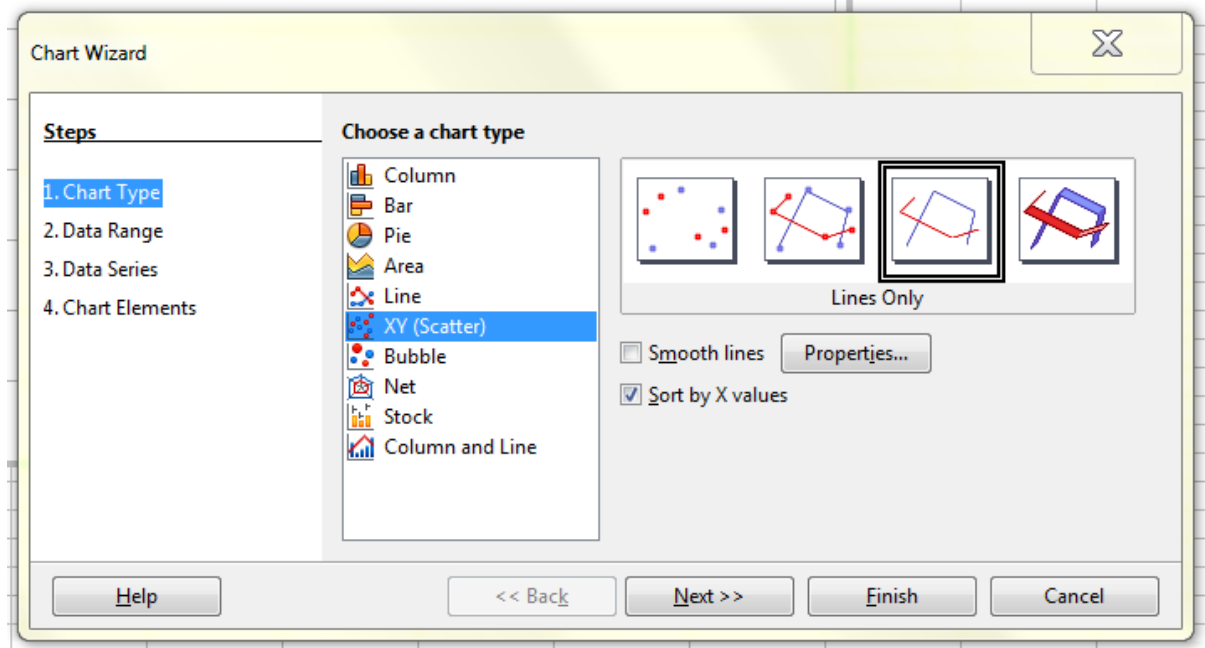
Each row in the file represents one unit of time, either minute or second, with total counts for that time period. The CSV file (for both per-minute and per-second statistics) contains the following columns, in order:

```
Time,time_t,Results,Search,Add,Mod,Modrdn,Delete,Abandon,Connections,SSL
Conns,Bind,Anon Bind,Unbind,Unindexed
```

The CSV file can be manipulated in any spreadsheet program, like OpenOffice Calc, and in many other business applications. The procedures for importing the CSV data and generating charts or other metrics depends on the application itself.

For example, to create a chart in OpenOffice Calc:

1. Open the CSV file.
2. Click the **Insert** menu, and select **Chart**.
3. In the **Chart Type** area, set the chart type to **XY (Scatter)**.
  1. Set the subtype to lines only.
  2. Select the option to sort by X values.



4. Accept the defaults in the other screens (particularly, to use the data series in columns and to set the first row and first column as labels), and create the chart.

## 15.5. REPLACING LOG FILES WITH A NAMED PIPE

The named pipe log script enables administrators to replace a log file with a named pipe to automatically process the log data. This provides advanced logging features, such as:

- Log only certain events, such as failed binds or connections from certain IP addresses.
- Log only lines that match a regular expression.
- Log only a defined number of lines.
- Send an email or other notification when a defined event is logged.

You can configure a named pipe for logging:

- For testing purposes, see [Section 15.5.1, “Temporarily Replacing a Log File with a Named Pipe”](#)
- For permanent usage, see [Section 15.5.2, “Creating a New Named Pipe for Logging”](#)

### 15.5.1. Temporarily Replacing a Log File with a Named Pipe

If you replace a log file with a named pipe, no server modifications are required. With this configuration, you cannot use log viewers, such as in the Admin Console, because they require to read the content a from a file.

To replace a log file with a named pipe:

1. Stop the Directory Server instance:

```
systemctl stop dirsrv.target
```

2. Remove the log file. For example:

```
rm -f /var/log/dirsrv/errors
```

3. Configure the named pipe to start with the Directory Server. For details, see [Section 15.5.3, “Starting and Shutting Down the Named Pipe with the Directory Server Service”](#).
4. Start the Directory Server instance:

```
systemctl start dirsrv.target
```



### IMPORTANT

When the log files are rotated, the named pipe is replaced with a regular file. Use this procedure only as a temporary solution. For a permanent solution, see [Section 15.5.2, “Creating a New Named Pipe for Logging”](#)

## 15.5.2. Creating a New Named Pipe for Logging

To log to a named pipe and additionally be able to use log viewers, such as in the Admin Console, configure the named pipe to use a different name than the name of your log file:

1. Configure the named pipe to start with Directory Server. To enable log viewers, additionally redirect the output of the **ds-logpipe.py** command to a file. For example:

```
python /usr/bin/ds-logpipe.py ... >
/var/log/dirsrv/slapd-instance/errors &
```

For details, see [Section 15.5.3, “Starting and Shutting Down the Named Pipe with the Directory Server Service”](#).

2. Update the Directory Server configuration to log to the named pipe. For example, to send the access log to the **/var/log/dirsrv/slapd-instance/access.pipe** named pipe:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x
dn: cn=config
changetype: modify
replace: nsslapd-accesslog
nsslapd-accesslog: /var/log/dirsrv/slapd-instance/access.pipe
```

Optionally, you can also set:

- The **nsslapd-errorlog** parameter for error events.
- The **nsslapd-auditlog** parameter for audit events. Note that audit logging is disabled by default. To enable it, additionally set the **nsslapd-accesslog-logging-enabled** parameter to **on**.



### NOTE

Updating the parameters takes effect immediately. However, you must start the named pipe manually or restart the Directory Server instance.

3. Disable buffering and log rotation for the event you configured the named pipe for. For example, to disable the features for the access log:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x
dn: cn=config
changetype: modify
replace: nsslapd-accesslog-logbuffering
nsslapd-accesslog-logbuffering: off
-
replace: nsslapd-accesslog-maxlogspersdir
nsslapd-accesslog-maxlogspersdir: 1
-
replace: nsslapd-accesslog-logexpirationtime
nsslapd-accesslog-logexpirationtime: -1
-
replace: nsslapd-accesslog-logrotationtime
nsslapd-accesslog-logrotationtime: -1
```

To disable the parameters for the error log, update:

- ***nsslapd-errorlog-logbuffering***
- ***nsslapd-errorlog-maxlogspersdir***
- ***nsslapd-errorlog-logexpirationtime***
- ***nsslapd-errorlog-logrotationtime***

To disable the parameters for the audit log, update:

- ***nsslapd-auditlog-logbuffering***
- ***nsslapd-auditlog-maxlogspersdir***
- ***nsslapd-auditlog-logexpirationtime***
- ***nsslapd-auditlog-logrotationtime***

4. Restart the Directory Server instance to start the pipe.

```
systemctl restart dirsrv.target
```

### 15.5.3. Starting and Shutting Down the Named Pipe with the Directory Server Service

To start and shut down the named pipe with the Directory Server instance:

1. Open the `/etc/sysconfig/dirsrv-instance` instance configuration file.

**WARNING**

Do not edit the `/etc/sysconfig/dirsrv` file.

2. Append the **ds-logpipe.py** commands at the end of the file. For example:

```
Only keep the last 1000 lines of the error log and
additionally redirect all log data to the
/var/log/dirsrv/slapd-instance/errors file
python /usr/bin/ds-logpipe.py
/var/log/dirsrv/slapd-instance/errors.pipe -m 1000 -u dirsrv -s
/var/run/dirsrv/slapd-instance.pid >
/var/log/dirsrv/slapd-instance/errors &

Only log failed binds
python /usr/bin/ds-logpipe.py
/var/log/dirsrv/slapd-instance/access.pipe -u dirsrv -s
/var/run/dirsrv/slapd-instance.pid --
plugin=/usr/share/dirsrv/data/failedbinds.py
failedbinds.logfile=/var/log/dirsrv/slapd-instance/access.failedbind
s &
```

For details, see the `ds-logpipe.py(1)` man page.

**IMPORTANT**

Make sure that each named pipe command ends with an `&` sign to send the **ds-logpipe.py** process to the background.

### 15.5.4. Using Plug-ins with the Named Pipe Log

You can call a plug-in to read the log data from the named pipe to perform operations on the log data. When using plug-ins with the named pipe log script, consider the following:

- The plug-in function is called for every line read from the named pipe.
- The plug-in function must be a Python script and use the `.py` suffix.
- Any plug-in arguments are passed in the command line to the **ds-logpipe.py** named pipe log script.
- A **pre** operation function can be called for when the plug-in is loaded.
- A **post** operation function can be called for when the plug-in exits.

#### 15.5.4.1. Loading Plug-ins with the Named Pipe Log Script

There are two options for the **ds-logpipe.py** command to use with plug-ins:



- The **--plugin** option gives the path to the plug-in file.
- The *plugin.arg* option passes plug-in arguments to the named pipe log script.
  - **plugin**: The file name without the **.py** suffix.
  - **arg**: Any argument allowed in the plug-in.

For example:

```
ds-logpipe.py /var/log/dirsrv/slapd-example/errors.pipe --
plugin=/usr/share/dirsrv/data/example-funct.py example-
funct.regex="warning" > /var/log/dirsrv/warnings.txt
```

If there are more than one value passed to the same argument, they are converted into a list of values in the plug-in. For example, this script sets two values for the **arg1** argument:

```
--plugin=/path/to/plugin_name.py plugin_name.arg1=example1
plugin_name.arg1=example2 plugin_name.arg2=demo
```

In the plug-in, this is converted to:

```
{'arg1': ['example1', 'example2'], 'arg2': 'demo'}
```

This is a Python **dictionary** object with two keys. The first key is the string **arg1**, and its value is a Python list object with two elements, the strings **foo** and **bar**. The second key is the string **arg2**, and its value is the string **baz**. If an argument has only a single value, it is left as a simple string. Multiple values for a single argument name are converted into a list of strings.

#### 15.5.4.2. Writing Plug-ins to Use with the Named Pipe Log Script

The **ds-logpipe.py** command supports the following functions in a plug-in:

- **plugin()**: Mandatory. Code in this function is applied to every line of log data received.
- **pre()**: Optional. Code is run when the plug-in is started.
- **post()**: Optional. Code is run when the plug-in exits.

Each function can have any arguments defined for it, and these arguments can then be passed to the script using the *plugin.arg* option. Additionally, each function can have its own return values and actions defined for it.

##### Example 15.1. Simple Named Pipe Log Plug-in

```
def pre(myargs):
 retval = True
 myarg = myargs['argname']
 if isinstance(myarg, list): # handle list of values
 else: # handle single value
 if bad_problem:
 retval = False
 return retval
```

```
def plugin(line):
 retval = True
 # do something with line
 if something_is_bogus:
 retval = False
 return retval

def post(): # no arguments
 # do something
 # no return value
```

## 15.5.5. Troubleshooting the Named Pipe

### 15.5.5.1. Directory Server Hangs When Writing to the Named Pipe

If the **ds-logpipe.py** command terminates unexpectedly, the Directory Server hangs while writing to the named pipe. To fix the problem:

Restart the named pipe manually or if configured in the **/etc/sysconfig/dirsrv-instance** file, run:

```
(. /etc/sysconfig/dirsrv-instance)
```

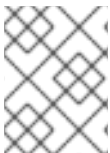
## 15.6. MONITORING THE LOCAL DISK FOR GRACEFUL SHUTDOWN

When the disk space available on a system becomes too small, the Directory Server process (**slapd**) crashes. Any abrupt shutdown runs the risk of corrupting the database or losing directory data.

It is possible to monitor the disk space available to the **slapd** process. A disk monitoring thread is enabled using the **nsslapd-disk-monitoring** configuration attribute. This creates a monitoring thread that wakes every ten (10) seconds to check for available disk space in certain areas.

If the disk space approaches a defined threshold, then the **slapd** begins a series of steps (by default) to reduce the amount of disk space it is consuming:

- Verbose logging is disabled.
- Access logging and error logging are disabled.
- Rotated (archived) logs are deleted.



### NOTE

Error log messages are always recorded, even when other changes are made to the logging configuration.

If the available disk space continues to drop to half of the configured threshold, then the **slapd** begins a graceful shut down process (within a grace period); and if the available disk space ever drops to 4KB, then the **slapd** process shuts down immediately. If the disk space is freed up, then the shutdown process is aborted, and all of the previously disabled log settings are re-enabled.

By default, the monitoring thread checks the configuration, transaction log, and database directories. An additional attribute (***nsslapd-disk-monitoring-logging-critical***) can be set to include the logs directory when evaluating disk space.

Disk monitoring is disabled by default, but it can be enabled and configured by adding the appropriate configuration attributes to the **cn=config** entry. Table 15.1, “Disk Monitoring Configuration Attributes” lists all of the configuration options.

1. Using **ldapmodify**, add the disk monitoring attributes. At a minimum, turn on the ***nsslapd-disk-monitoring*** attribute to enable disk monitoring. The default threshold is 2MB; this can be configured (optionally) in the ***nsslapd-disk-monitoring-threshold*** attribute.

For example:

```
[jsmith@server ~]$ ldapmodify -D "cn=directory manager" -W -x
dn: cn=config
changetype: modify
add: nsslapd-disk-monitoring
nsslapd-disk-monitoring: on
-
add: nsslapd-disk-monitoring-threshold
nsslapd-disk-monitoring-threshold: 3000000
-
add: nsslapd-disk-monitoring-grace-period
nsslapd-disk-monitoring-grace-period: 20
```

2. Restart the Directory Server to load the new configuration.

```
[root@server ~]# service dirsrv restart
```

**Table 15.1. Disk Monitoring Configuration Attributes**

| Configuration Attribute                  | Description                                                                                                                                                                                                                   |
|------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| nsslapd-disk-monitoring                  | Enabled disk monitoring. This is the only required attribute, since the other configuration options have usable defaults.                                                                                                     |
| nsslapd-disk-monitoring-grace-period     | Sets a grace period to wait before shutting down the server after it hits half of the disk space limit. This gives an administrator time to address the situation. The default value is 60 (minutes).                         |
| nsslapd-disk-monitoring-logging-critical | Sets whether to shut down the server if the log directories pass the halfway point set in the disk space limit. This prevents the monitoring thread from disabling audit or access logging or from deleting rotated logfiles. |

| Configuration Attribute           | Description                                                                                                                                                                                                                                    |
|-----------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| nsslapd-disk-monitoring-threshold | Sets the amount of disk space, in bytes, to use to evaluate whether the server has enough available disk space. Once the space reaches half of this threshold, then the server begins a shut down process. The default value is 2000000 (2MB). |

## 15.7. MONITORING SERVER ACTIVITY

The Directory Server's current activities can be monitored from either the Directory Server Console or the command line. It is also possible to monitor the activity of the caches for all of the database.

Many performance counters in Directory Server use 32-bit numbers. This includes many kind of performance monitoring, such as the number of active connections (*nsslapd-db-active-txns*), the number of operations (*threads*), and the number of search requests (*nsslapd-db-cache-try*). However, under heavy loads, these 32-bit counter numbers can wrap too quickly. The *nsslapd-counters* attribute enabled the Directory Server to use 64-bit values for counter numbers, even on 32-bit systems. This enables long term statistics gathering for high-traffic systems.

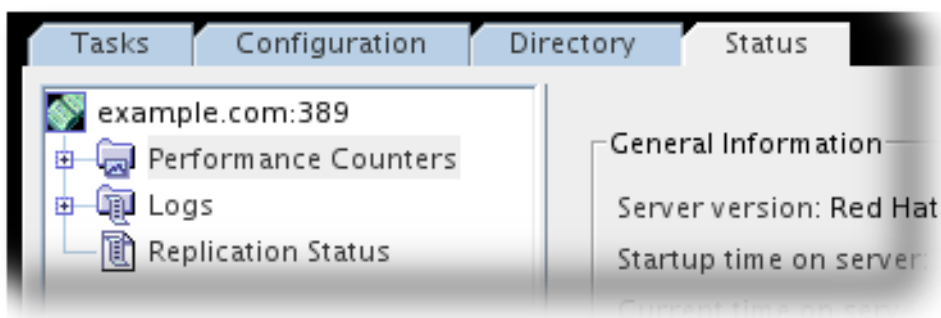
Some of the counters for Directory Server database attributes monitored by server use 64-bit integers, even on 32-bit systems:

- opsinitiated
- opscompleted
- entriessent
- bytessent
- totalconnections

The counters which use 64-bit integers are not configurable.

### 15.7.1. Monitoring the Server from the Directory Server Console

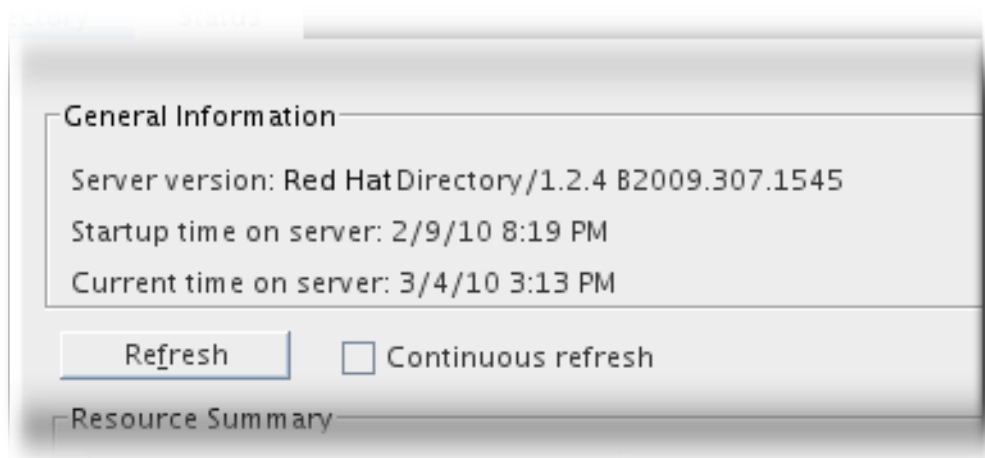
1. Select the **Status** tab.
2. In the navigation tree, select **Performance Counters**.



The **Status** tab in the right pane displays current information about server activity. If the server is currently not running, this tab will not provide performance monitoring information.

- Click **Refresh** to refresh the current display. For the server to continuously update the displayed information, select the **Continuous** check box.

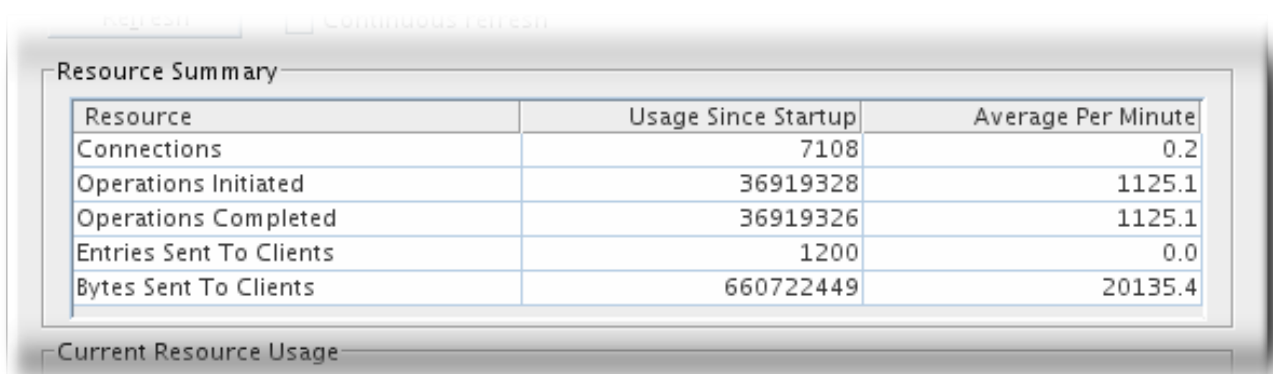
The **General Information** table shows basic information about the server, which helps set a baseline about the statistics that have been gathered.



**Table 15.2. General Information (Server)**

| Field                  | Description                               |
|------------------------|-------------------------------------------|
| Server Version         | Identifies the current server version.    |
| Startup Time on Server | The date and time the server was started. |
| Current Time on Server | The current date and time on the server.  |

The **Resource Summary** table shows the totals of all operations performed by that instance.



**Table 15.3. Resource Summary**

| Resource    | Usage Since Startup                                                  | Average Per Minute                                             |
|-------------|----------------------------------------------------------------------|----------------------------------------------------------------|
| Connections | The total number of connections to this server since server startup. | Average number of connections per minute since server startup. |

| Resource                | Usage Since Startup                                                                                                                                                                                                          | Average Per Minute                                                         |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------|
| Operations Initiated    | The total number of operations initiated since server startup. Operations include any client requests for server action, such as searches, adds, and modifies. Often, multiple operations are initiated for each connection. | Average number of operations per minute since server startup.              |
| Operations Completed    | The total number of operations completed by the server since server startup.                                                                                                                                                 | Average number of operations per minute since server startup.              |
| Entries Sent to Clients | The total number of entries sent to clients since server startup. Entries are sent to clients as the result of search requests.                                                                                              | Average number of entries sent to clients per minute since server startup. |
| Bytes Sent to Clients   | The total number of bytes sent to clients since server startup.                                                                                                                                                              | Average number of bytes sent to clients per minute since server startup.   |

The **Current Resource Usage** table shows the current demands on the server.

bytes sent to clients 600,224,492 2013-4

Current Resource Usage

| Resource                            | Current Total |
|-------------------------------------|---------------|
| Active Threads                      | 30            |
| Open Connections                    | 3             |
| Remaining Available Connections     | 957           |
| Threads Waiting To Read From Client | 2             |
| Databases In Use                    | 2             |

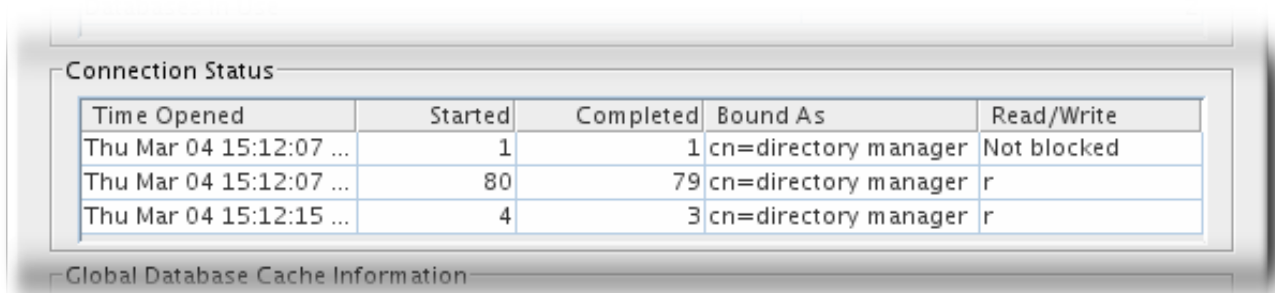
Connection Status

**Table 15.4. Current Resource Usage**

| Resource         | Current Total                                                                                                                                                 |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Active Threads   | The current number of active threads used for handling requests. Additional threads may be created by internal server tasks, such as replication or chaining. |
| Open Connections | The total number of open connections. Each connection can account for multiple operations, and therefore multiple threads.                                    |

| Resource                            | Current Total                                                                                                                                                                                                                                                                                                                                                                      |
|-------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Remaining Available Connections     | The total number of remaining connections that the server can concurrently open. This number is based on the number of currently open connections and the total number of concurrent connections that the server is allowed to open. In most cases, the latter value is determined by the operating system and is expressed as the number of file descriptors available to a task. |
| Threads Waiting to Write to Client  | The total number of threads waiting to write to the client. Threads may not be immediately written when the server must pause while sending data to a client. Reasons for a pause include a slow network, a slow client, or an extremely large amount of information being sent to the client.                                                                                     |
| Threads Waiting to Read from Client | The total number of threads waiting to read from the client. Threads may not be immediately read if the server starts to receive a request from the client, and then the transmission of that request is halted for some reason. Generally, threads waiting to read are an indication of a slow network or client.                                                                 |
| Databases in Use                    | The total number of databases being serviced by the server.                                                                                                                                                                                                                                                                                                                        |

The **Connection Status** table simply lists the current active connections, with related connection information.



| Time Opened             | Started | Completed | Bound As             | Read/Write  |
|-------------------------|---------|-----------|----------------------|-------------|
| Thu Mar 04 15:12:07 ... | 1       | 1         | cn=directory manager | Not blocked |
| Thu Mar 04 15:12:07 ... | 80      | 79        | cn=directory manager | r           |
| Thu Mar 04 15:12:15 ... | 4       | 3         | cn=directory manager | r           |

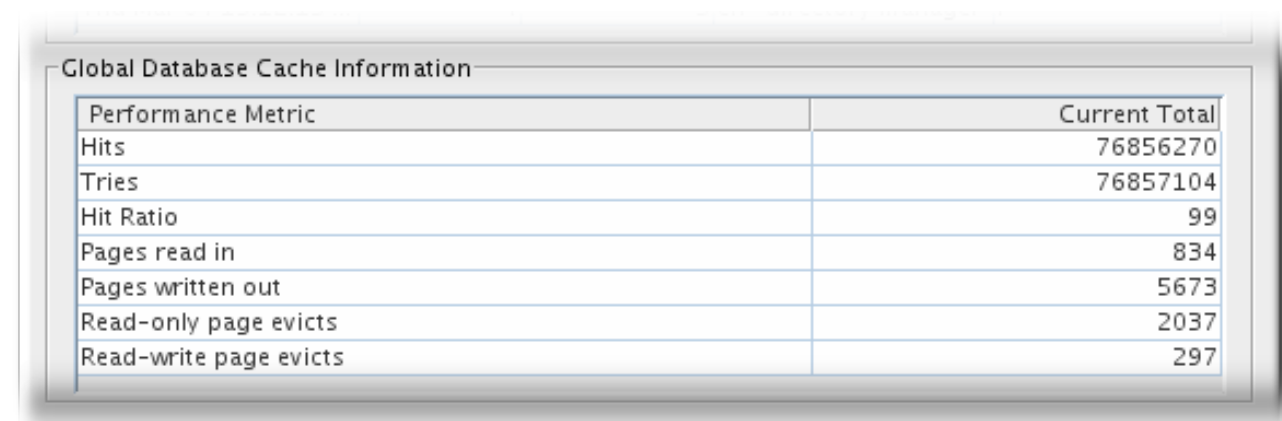
Global Database Cache Information

**Table 15.5. Connection Status**

| Table Header | Description                                                      |
|--------------|------------------------------------------------------------------|
| Time Opened  | The time on the server when the connection was initially opened. |
| Started      | The number of operations initiated by this connection.           |

| Table Header | Description                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Completed    | The number of operations completed by the server for this connection.                                                                                                                                                                                                                                                                                                                                                                            |
| Bound as     | The distinguished name used by the client to bind to the server. If the client has not authenticated to the server, the server displays <b>not bound</b> in this field.                                                                                                                                                                                                                                                                          |
| Read/Write   | <p>Indicates whether the server is currently blocked for read or write access to the client. There are two possible values:</p> <div> <p>Not blocked means that the server is idle, actively sending data to the client, or actively reading data from the client.</p> <p>Blocked means that the server is trying to send data to the client or read data from the client but cannot. The probable cause is a slow network or client.</p> </div> |

The **Global Database Cache** table lists the cache information for all databases within the Directory Server instance.



| Performance Metric     | Current Total |
|------------------------|---------------|
| Hits                   | 76856270      |
| Tries                  | 76857104      |
| Hit Ratio              | 99            |
| Pages read in          | 834           |
| Pages written out      | 5673          |
| Read-only page evicts  | 2037          |
| Read-write page evicts | 297           |

## NOTE

Although the performance counter for the global database cache is listed with the other server performance counters in the Directory Server Console, the actual database cache entries are located and monitored in **cn=monitor, cn=database\_instance, cn=ldbm database, cn=plugins, cn=config**, as are the other database activities. Monitoring these entries through the command line is covered in [Section 15.8.2, “Monitoring Databases from the Command Line”](#).

**Table 15.6. Global Database Cache Information**



| Table Header           | Description                                                                                                                                                                                                                                                                                                                                                       |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Hits                   | The number of times the server could process a request by obtaining data from the cache rather than by going to the disk.                                                                                                                                                                                                                                         |
| Tries                  | The total number of database accesses since server startup.                                                                                                                                                                                                                                                                                                       |
| Hit Ratio              | The ratio of cache tries to successful cache hits. The closer this number is to 100%, the better.                                                                                                                                                                                                                                                                 |
| Pages Read In          | The number of pages read from disk into the cache.                                                                                                                                                                                                                                                                                                                |
| Pages Written Out      | The number of pages written from the cache back to disk.                                                                                                                                                                                                                                                                                                          |
| Read-Only Page Evicts  | The number of read-only pages discarded from the cache to make room for new pages. Pages discarded from the cache have to be written to disk, possibly affecting server performance. The lower the number of page evicts the better.                                                                                                                              |
| Read-Write Page Evicts | The number of read-write pages discarded from the cache to make room for new pages. This value differs from <b>Pages Written Out</b> in that these are discarded read-write pages that have not been modified. Pages discarded from the cache have to be written to disk, possibly affecting server performance. The lower the number of page evicts, the better. |

### 15.7.2. Monitoring the Directory Server from the Command Line

The Directory Server's current activities can be monitored using LDAP tools such as **ldapsearch**, with the following characteristics:

- Search with the attribute filter **objectClass=\***.
- Use the search base **cn=monitor**; the monitoring attributes for the server are found in the **cn=monitor** entry.
- Use the search scope **base**.

For example:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -s base -b "cn=monitor" "(objectclass=*)"
```

The monitoring attributes for the Directory Server are found in the **cn=monitor** entry. For information on searching the Directory Server, see [Section 10.3, “Using ldapsearch”](#).

Table 15.7. Server Monitoring Attributes

| Attribute                                                           | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| version                                                             | Identifies the directory's current version number.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| threads                                                             | The current number of active threads used for handling requests. Additional threads may be created by internal server tasks, such as replication or chaining.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <i>connection:fd:opentime:opsinitiated:opscompleted:binddn:[rw]</i> | <p>Provides the following summary information for each open connection (only available if you bind to the directory as Directory Manager):</p> <div> <p><i>fd</i> — The file descriptor used for this connection.</p> <p><i>opentime</i> — The time this connection was opened.</p> <p><i>opsinitiated</i> — The number of operations initiated by this connection.</p> <p><i>opscompleted</i> — The number of operations completed.</p> <p><i>binddn</i> — The distinguished name used by this connection to connect to the directory.</p> <p><i>rw</i> — The field shown if the connection is blocked for read or write.</p> </div> <p>By default, this information is available to Directory Manager. However, the ACL associated with this information can be edited to allow others to access the information.</p> |
| currentconnections                                                  | Identifies the number of connections currently in service by the directory.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| totalconnections                                                    | Identifies the number of connections handled by the directory since it started.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| dtablesizes                                                         | Shows the number of file descriptors available to the directory. Each connection requires one file descriptor: one for every open index, one for log file management, and one for <b>ns-slapd</b> itself. Essentially, this value shows how many additional concurrent connections can be serviced by the directory. For more information on file descriptors, see the operating system documentation.                                                                                                                                                                                                                                                                                                                                                                                                                  |
| readwaiters                                                         | Identifies the number of threads waiting to read data from a client.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

| Attribute        | Description                                                                                                                       |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| opsinitiated     | Identifies the number of operations the server has initiated since it started.                                                    |
| opscompleted     | Identifies the number of operations the server has completed since it started.                                                    |
| entriessent      | Identifies the number of entries sent to clients since the server started.                                                        |
| bytessent        | Identifies the number of bytes sent to clients since the server started.                                                          |
| currenttime      | Identifies the time when this snapshot of the server was taken. The time is displayed in Greenwich Mean Time (GMT) in UTC format. |
| starttime        | Identifies the time when the server started. The time is displayed in Greenwich Mean Time (GMT) in UTC format.                    |
| nbackends        | Identifies the number of back ends (databases) the server services.                                                               |
| backendmonitorDN | Identifies the DN of each directory database.                                                                                     |

## 15.8. MONITORING DATABASE ACTIVITY

The database's current activities can be monitored through Directory Server Console or from the command line.

Many performance counters in Directory Server use 32-bit numbers. However, under heavy loads, these 32-bit counter numbers can wrap too quickly. The ***nsslapd-counters*** attribute enabled the Directory Server to use 64-bit values for counter numbers, even on 32-bit systems. This enables long term statistics gathering for high-traffic systems.

Some of the counters for Directory Server database attributes monitored by server use 64-bit integers, even on 32-bit systems:

- entrycachehits
- entrycachetries
- currententrycachesize
- maxentrycachesize

The counters which use 64-bit integers are not configurable.



## NOTE

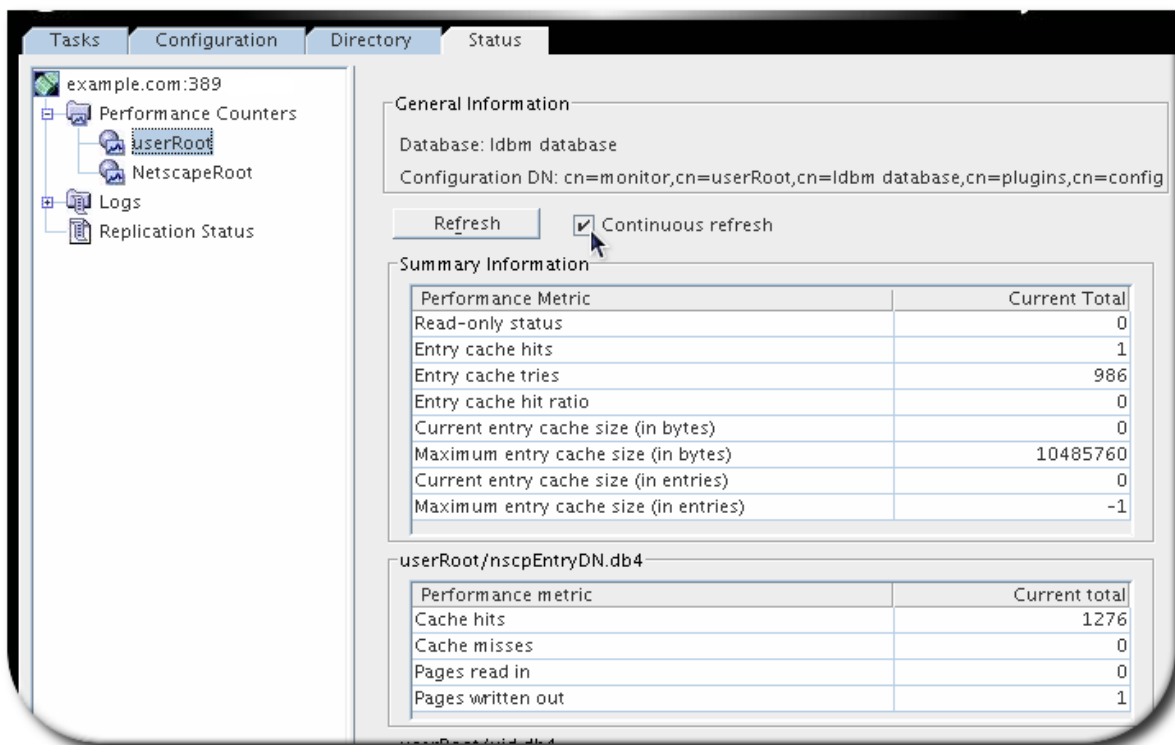
Tips for tuning the entry and database caches to improve server performance are in the *Tuning Red Hat Directory Server Performance*.

### 15.8.1. Monitoring Database Activity from the Directory Server Console

To monitor the database's activities:

1. In the Directory Server Console, select the **Status** tab.
2. In the navigation tree, expand the **Performance Counters** folder, and select the database to monitor.

The tab displays current information about database activity. If the server is currently not running, this tab will not provide performance monitoring information.



3. Click **Refresh** to refresh the currently displayed information. For the directory to continuously update the displayed information, select the **Continuous** check box, and then click **Refresh**.

**Table 15.8. General Information (Database)**

| Field            | Description                                                                                                                                    |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| Database         | Identifies the type of database being monitored.                                                                                               |
| Configuration DN | Identifies the distinguished name that must be used as a search base to obtain these results using the <b>ldapsearch</b> command-line utility. |

The **Summary Information** section shows the cumulative information for all of the databases being monitored and some cache-related configuration settings which are applied to all databases.

**Table 15.9. Summary Information**

| Performance Metric                    | Current Total                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Read-Only Status                      | Shows whether the database is currently in read-only mode. The database is in read-only mode when the <b><i>nsslapd-readonly</i></b> attribute is set to <b>on</b> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Entry Cache Hits                      | The total number of successful entry cache lookups. That is, the total number of times the server could process a search request by obtaining data from the cache rather than by going to disk.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Entry Cache Tries                     | The total number of entry cache lookups since the directory was last started. That is, the total number of entries requested since server startup.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Entry Cache Hit Ratio                 | <p>Ratio that indicates the number of entry cache tries to successful entry cache lookups. This number is based on the total lookups and hits since the directory was last started. The closer this value is to 100%, the better. Whenever an operation attempts to find an entry that is not present in the entry cache, the directory has to perform a disk access to obtain the entry. Thus, as this ratio drops towards zero, the number of disk accesses increases, and directory search performance drops.</p> <p>To improve this ratio, increase the size of the entry cache by increasing the value of the <b><i>nsslapd-cachememsize</i></b> attribute in the <b><i>cn=database_name, cn=ldbm database, cn=plugins, cn=config</i></b> entry for the database. In the Directory Server Console, this is set in the <b>Memory available for cache</b> field in the database settings.</p> |
| Current Entry Cache Size (in Bytes)   | The total size of directory entries currently present in the entry cache.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Maximum Entry Cache Size (in Bytes)   | <p>The size of the entry cache maintained by the directory.</p> <p>This value is managed by the <b><i>nsslapd-cachememsize</i></b> attribute in the <b><i>cn=database_name, cn=ldbm database, cn=plugins, cn=config</i></b> entry for the database. This is set in the <b>Memory available for cache</b> field in the database settings in the Directory Server Console.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Current Entry Cache Size (in Entries) | The number of directory entries currently present in the entry cache.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

| Performance Metric                    | Current Total                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Maximum Entry Cache Size (in Entries) | <p><i>DEPRECATED.</i></p> <p>The maximum number of directory entries that can be maintained in the entry cache.</p> <p>Do not attempt to manage the cache size by setting a maximum number of allowed entries. This can make it difficult for the host to allocate RAM effectively. Manage the cache size by setting the amount of RAM available to the cache, using the <b><i>nsslapd-cachememsize</i></b> attribute.</p> |

There are many different databases listed for the database monitoring page, by default, because databases are maintained for both entries and indexed attributes. All databases, though, have the same kind of cache information monitored in the counters.

**Table 15.10. Database Cache Information**

| Performance Metric | Current Total                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Hits               | The number of times the database cache successfully supplied a requested page.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Tries              | The number of times the database cache was asked for a page.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Hit Ratio          | <p>The ratio of database cache hits to database cache tries. The closer this value is to 100%, the better. Whenever a directory operation attempts to find a portion of the database that is not present in the database cache, the directory has to perform a disk access to obtain the appropriate database page. Thus, as this ratio drops towards zero, the number of disk accesses increases, and directory performance drops.</p> <p>To improve this ratio, increase the amount of data that the directory maintains in the database cache by increasing the value of the <b><i>nsslapd-dbcachesize</i></b> attribute. This is the <b>Maximum Cache Size</b> database setting in the Directory Server Console.</p> |
| Pages Read In      | The number of pages read from disk into the database cache.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Pages Written Out  | The number of pages written from the cache back to disk. A database page is written to disk whenever a read-write page has been modified and then subsequently deleted from the cache. Pages are deleted from the database cache when the cache is full and a directory operation requires a database page that is not currently stored in cache.                                                                                                                                                                                                                                                                                                                                                                        |

| Performance Metric     | Current Total                                                                                                                                                                                                  |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Read-Only Page Evicts  | The number of read-only pages discarded from the cache to make room for new pages.                                                                                                                             |
| Read-Write Page Evicts | The number of read-write pages discarded from the cache to make room for new pages. This value differs from <b>Pages Written Out</b> in that these are discarded read-write pages that have not been modified. |

**Table 15.11. Database File-Specific**

| Performance Metric | Current Total                                                                                                                                                                                                              |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Cache Hits         | The number of times that a search result resulted in a cache hit on this specific file. That is, a client performs a search that requires data from this file, and the directory obtains the required data from the cache. |
| Cache Misses       | The number of times that a search result failed to hit the cache on this specific file. That is, a search that required data from this file was performed, and the required data could not be found in the cache.          |
| Pages Read In      | The number of pages brought to the cache from this file.                                                                                                                                                                   |
| Pages Written Out  | The number of pages for this file written from cache to disk.                                                                                                                                                              |

### 15.8.2. Monitoring Databases from the Command Line

A database's current activities can be monitored using LDAP tools such as **ldapsearch**. The search targets the monitoring subtree of the LDBM database entry, **cn=monitor,cn=database\_name,cn=ldbm database,cn=plugins,cn=config**. This contains all of the monitoring attributes for the that specific database instance.

For example:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -s base -b "cn=monitor,cn=database_name,cn=ldbm
database,cn=plugins,cn=config" "(objectclass=*)"
```

**Table 15.12. Database Monitoring Attributes**

| Attribute             | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| database              | Identifies the type of database currently being monitored.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| readonly              | Indicates whether the database is in read-only mode; <b>0</b> means that the server is not in read-only mode, <b>1</b> means that it is in read-only mode.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| entrycachehits        | The total number of successful entry cache lookups. That is, the total number of times the server could process a search request by obtaining data from the cache rather than by going to disk.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| entrycachetries       | The total number of entry cache lookups since the directory was last started. That is, the total number of search operations performed against the server since server startup.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| entrycachehitratio    | <p>Ratio that indicates the number of entry cache tries to successful entry cache lookups. This number is based on the total lookups and hits since the directory was last started. The closer this value is to 100%, the better. Whenever a search operation attempts to find an entry that is not present in the entry cache, the directory has to perform a disk access to obtain the entry. Thus, as this ratio drops towards zero, the number of disk accesses increases, and directory search performance drops.</p> <p>To improve this ratio, increase the size of the entry cache by increasing the value of the <b>nsslapd-cachememsize</b> attribute in the <b>cn=database_name, cn=ldbm database, cn=plugins, cn=config</b> entry for the database. In the Directory Server Console, this is set in the <b>Memory available for cache</b> field in the database settings.</p> |
| currententrycachesize | <p>The total size, in bytes, of directory entries currently present in the entry cache.</p> <p>To increase the size of the entries which can be present in the cache, increase the value of the <b>nsslapd-cachememsize</b> attribute in the <b>cn=database_name, cn=ldbm database, cn=plugins, cn=config</b> entry for the database. In the Directory Server Console, this is set in the <b>Memory available for cache</b> field in the database settings.</p>                                                                                                                                                                                                                                                                                                                                                                                                                          |



| Attribute                     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| maxentrycachesize             | <p>The maximum size, in bytes, of directory entries that can be maintained in the entry cache.</p> <p>To increase the size of the entries which can be present in the cache, increase the value of the <b>nsslapd-cachememsize</b> attribute in the <b>cn=database_name, cn=ldbm database, cn=plugins, cn=config</b> entry for the database. In the Directory Server Console, this is set in the <b>Memory available for cache</b> field in the database settings.</p> |
| dbcachehits                   | The number of times the server could process a request by obtaining data from the cache rather than by going to the disk.                                                                                                                                                                                                                                                                                                                                              |
| dbcachetries                  | The total number of database accesses since server startup.                                                                                                                                                                                                                                                                                                                                                                                                            |
| dbcachehitratio               | The ratio of cache tries to successful cache hits. The closer this number is to 100%, the better.                                                                                                                                                                                                                                                                                                                                                                      |
| dbcachepagein                 | The number of pages read from disk into the cache.                                                                                                                                                                                                                                                                                                                                                                                                                     |
| dbcachepageout                | The number of pages written from the cache back to disk.                                                                                                                                                                                                                                                                                                                                                                                                               |
| dbcacheroevict                | The number of read-only pages discarded from the cache to make room for new pages. Pages discarded from the cache have to be written to disk, possibly affecting server performance. The lower the number of page evicts the better.                                                                                                                                                                                                                                   |
| dbcacherwevict                | The number of read-write pages discarded from the cache to make room for new pages. This value differs from <b>Pages Written Out</b> in that these are discarded read-write pages that have not been modified. Pages discarded from the cache have to be written to disk, possibly affecting server performance. The lower the number of page evicts the better.                                                                                                       |
| dbfilename- <i>number</i>     | The name of the file. <i>number</i> provides a sequential integer identifier (starting at 0) for the file. All associated statistics for the file are given this same numerical identifier.                                                                                                                                                                                                                                                                            |
| dbfilecachehit- <i>number</i> | The number of times that a search result resulted in a cache hit on this specific file. That is, a client performs a search that requires data from this file, and the directory obtains the required data from the cache.                                                                                                                                                                                                                                             |

| Attribute                     | Description                                                                                                                                                                                                                                                                                                                                     |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>dbfilecachemiss-number</i> | The number of times that a search result failed to hit the cache on this specific file. That is, a search that required data from this file was performed, and the required data could not be found in the cache.                                                                                                                               |
| <i>dbfilepagein-number</i>    | The number of pages brought to the cache from this file.                                                                                                                                                                                                                                                                                        |
| <i>dbfilepageout-number</i>   | The number of pages for this file written from cache to disk.                                                                                                                                                                                                                                                                                   |
| <i>currentdncachesize</i>     | <p>The total size, in bytes, of DNs currently present in the DN cache.</p> <p>To increase the size of the entries which can be present in the DN cache, increase the value of the <b><i>nsslapd-dncachememsize</i></b> attribute in the <b><i>cn=database_name, cn=ldbm database, cn=plugins, cn=config</i></b> entry for the database.</p>     |
| <i>maxdncachesize</i>         | <p>The maximum size, in bytes, of DNs that can be maintained in the DN cache.</p> <p>To increase the size of the entries which can be present in the cache, increase the value of the <b><i>nsslapd-dncachememsize</i></b> attribute in the <b><i>cn=database_name, cn=ldbm database, cn=plugins, cn=config</i></b> entry for the database.</p> |
| <i>currentdncachecount</i>    | The number of DNs currently present in the DN cache.                                                                                                                                                                                                                                                                                            |

## 15.9. MONITORING DATABASE LINK ACTIVITY

It is possible to monitor the activity of database links from the command line using the **ldapsearch** command-line utility to return the monitoring attributes that are required. The monitoring attributes are stored in the ***cn=monitor, cn=database\_link\_name, cn=chaining database, cn=plugins, cn=config***.

For example, the **ldapsearch** command-line utility can be used to retrieve the number of add operations received by a particular database link. For example, this command monitors a database link called **DBLink1**:

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -s sub -b "cn=monitor, cn=DBLink1, cn=chaining
database, cn=plugins, cn=config" "(objectclass=*)" nsAddCount
```

Table 15.13, “Database Link Monitoring Attributes” lists the database link monitoring attributes which can be monitored.

**Table 15.13. Database Link Monitoring Attributes**

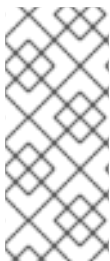
| Attribute Name             | Description                                           |
|----------------------------|-------------------------------------------------------|
| nsAddCount                 | The number of add operations received.                |
| nsDeleteCount              | The number of delete operations received.             |
| nsModifyCount              | The number of modify operations received.             |
| nsRenameCount              | The number of rename operations received.             |
| nsSearchBaseCount          | The number of base-level searches received.           |
| nsSearchOneLevelCount      | The number of one-level searches received.            |
| nsSearchSubtreeCount       | The number of subtree searches received.              |
| nsAbandonCount             | The number of abandon operations received.            |
| nsBindCount                | The number of bind request received.                  |
| nsUnbindCount              | The number of unbinds received.                       |
| nsCompareCount             | The number of compare operations received.            |
| nsOperationConnectionCount | The number of open connections for normal operations. |
| nsBindConnectionCount      | The number of open connections for bind operations.   |

## 15.10. ENABLING AND DISABLING COUNTERS

The **nsslapd-counters** attribute enabled counters to run. However, running counters can affect performance, so it also possible to turn off counters. If counters are off, they all have a value of zero (0).

By default, counters are already enabled. To enable or disable performance counters, use **ldapmodify**:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com -x
dn: cn=config
changetype: modify
replace: nsslapd-counters
nsslapd-counters: off
```

**NOTE**

Regardless of the whether the Directory Server is run on a 32-bit or 64-bit platform, the server itself records many counter values using 64-bit integers. For high-traffic sites, 32-bit counters turn over too quickly, which leads to quirky ratios and counts. Using 64-bit integers (on all platforms) improves performance and enables administrators to gather long-term statistics accurately.

## CHAPTER 16. MONITORING DIRECTORY SERVER USING SNMP

The server and database activity monitoring log setup described in [Chapter 15, \*Monitoring Server and Database Activity\*](#) is specific to Directory Server. You can also monitor your Directory Server using Simple Network Management Protocol (SNMP), which is a management protocol used for monitoring network activity which can be used to monitor a wide range of devices in real time.

Directory Server can be monitored with SNMP through an AgentX subagent. SNMP monitoring collects useful information about the Directory Server, such as bind information, operations performed on the server, and cache information. The Directory Server SNMP subagent supports SNMP traps to send notifications about changes in the running state of your server instances.

### 16.1. ABOUT SNMP

SNMP has become interoperable on account of its widespread popularity. It is this interoperability, combined with the fact that SNMP can take on numerous jobs specific to a whole range of different device classes, that make SNMP the ideal standard mechanism for global network control and monitoring. SNMP allows network administrators to unify all network monitoring activities, with Directory Server monitoring part of the broader picture.

SNMP is used to exchange data about network activity. With SNMP, data travels between a managed device and a network management application (NMS) where users remotely manage the network. A managed device is anything that runs SNMP, such as hosts, routers, and your Directory Server. An NMS is usually a powerful workstation with one or more network management applications installed. A network management application graphically shows information about managed devices, which device is up or down, which and how many error messages were received, and so on.

Information is transferred between the NMS and the managed device through the use of two types of agents: the subagent and the *master agent*. The subagent gathers information about the managed device and passes the information to the master agent. Directory Server has a subagent. The master agent exchanges information between the various subagents and the NMS. The master agent usually runs on the same host machine as the subagents it talks to, although it can run on a remote machine.

Values for SNMP attributes, otherwise known as variables, that can be queried are kept on the managed device and reported to the NMS as necessary. Each variable is known as a *managed object*, which is anything the agent can access and send to the NMS. All managed objects are defined in a management information base (MIB), which is a database with a tree-like hierarchy. The top level of the hierarchy contains the most general information about the network. Each branch underneath is more specific and deals with separate network areas.

SNMP exchanges network information in the form of protocol data units (PDUs). PDUs contain information about variables stored on the managed device. These variables, also known as managed objects, have values and titles that are reported to the NMS as necessary. Communication between an NMS and a managed device takes place either by the NMS sending updates or requesting information or by the managed object sending a notice or warning, called a *trap*, when a server shuts down or starts up.

### 16.2. CONFIGURING THE MASTER AGENT

To use the subagent, you must have a master agent that supports AgentX. A common agent is Net-SNMP master agent, which may be available through your operating system vendor or can be downloaded from the Net-SNMP website, <http://www.net-snmp.org>.

The SNMP subagent included with Directory Server uses the AgentX protocol to communicate with the SNMP master agent running on your system. You must make sure that you enable AgentX support on

your master agent. For Net-SNMP, add a line containing **agentx master** in the master agent's **snmpd.conf** file. For more details on configuring the master agent for AgentX support, see the Net-SNMP website, <http://www.net-snmp.org>.

## 16.3. CONFIGURING THE SUBAGENT

The Directory Server SNMP subagent is installed as a service on the Red Hat Enterprise Linux host machine, the same as the Directory Server and Admin Server processes.

### 16.3.1. Creating the Subagent Configuration File

The subagent is configured in a **.conf** configuration file. This file must be created manually. It can be named and located wherever you want, but it is strongly recommended that you use the default location. The default location is **/etc/dirsrv/config/ldap-agent.conf**. This file, as installed, is a template for the SNMP subagent service. The **ldap-agent.conf** file should be modified to fit the specific network environment.

This configuration file specifies the connection information for the master agent, logfile location, and which Directory Server instances to monitor.

#### 16.3.1.1. agentx-master

The **agentx-master** setting tells the subagent how to communicate with the SNMP master agent. If this setting is not specified, the subagent tries to communicate the master agent through the Unix domain socket **/var/agentx/master**. This is also where the Net-SNMP master agent listens for AgentX communications by default. If you configured your master agent to listen on a different Unix domain socket, you must use the **agentx-master** setting for your subagent to communicate with your master agent by setting the new path for the **agentx-master** parameter. For example:

```
agentx-master /var/snmp/agentx
```

Make sure that the user as whom you are running the subagent has the appropriate permissions to write to this socket.

If the master agent is listening for AgentX communications on a TCP port, the **agentx-master** setting has the host name and port number for the master agent. For example:

```
agentx-master localhost:705
```

#### 16.3.1.2. agent-logdir

The **agent-logdir** setting specifies the directory where the subagent will write its logfile. For example:

```
agent-logdir /var/log
```

If this parameter is not specified, the agent will write its logfile to the same location as your subagent configuration file. The logfile will be named **ldap-agent.log**.

The default log directory as specified in the **ldap-agent.conf** template is **/var/log/dirsrv/**. This is the same directory where instance log files are stored, but the SNMP subagent log files are not instance-specific. It is recommended that you use the default location.

Make sure that the user as whom your subagent is running has write permission to this directory.

### 16.3.1.3. server

The **server** setting specifies a Directory Server instance that you want to monitor. You must use one **server** setting for each Directory Server instance. The subagent requires at least one **server** setting to be specified in its configuration file. The **server** setting should be set to the name of the Directory Server instance you would like to monitor. For example:

```
server slapd-phonebook
```

To monitor multiple Directory Server instances, an additional **server** parameter in the subagent configuration file for each instance.

```
server slapd-phonebook
server slapd-example
server slapd-directory
```

### 16.3.2. Starting the Subagent

Once your master agent is running and you have created your subagent configuration file, start the subagent.



#### NOTE

The Directory Server does not have to be started for the subagent to be started.

The subagent is started and runs as a separate process than the Red Hat Directory Server or Admin Server processes. To start or stop the subagent, use the **service** system tools.

```
service dirsrv-snmp {start|stop|restart}
```

The status of the subagent can also be checked with the **service** tool.

```
service dirsrv-snmp status
dirsrv-snmp is stopped
```

### 16.3.3. Testing the Subagent

To test your subagent, use any SNMP client tools to query the master agent. Net-SNMP contains simple command-line utilities such as **snmpwalk** and **snmpget**. In order for these tools to use variable names for queries, configure them to load the Directory Server's MIB file. The Directory Server's MIB file, **redhat-directory.mib**, is located in **/usr/share/dirsrv/mibs** on Red Hat Enterprise Linux 6 (64-bit). There are some additional common required MIB files in this **mibs** directory if you do not already have them with your MIB tools.

The MIB file is not needed for the subagent to operate; it is only required for any SNMP client application to use variable names instead of numeric OIDs to see the monitored information provided by the subagent.

Each monitored server instance uses its port number as an index to identify that particular Directory Server instance. For example, querying for the **dsEntityName.389** SNMP variable returns the variable value for a server running on port 389, assuming that instance exists and is being monitored by the subagent.

For details on configuring and using the Net-SNMP command-line tools, check out the Net-SNMP website, <http://www.net-snmp.org>.

## 16.4. CONFIGURING SNMP TRAPS

An SNMP trap is essentially a threshold which triggers a notification if it is encountered by the monitored server. To use traps, the master agent must be configured to accept traps and do something with them. For example, a trap can trigger an email notification for an administrator of the Directory Server instance stops.

The subagent is only responsible for sending the traps to the master agent. The master agent and a trap handler must be configured according to the documentation for the SNMP master agent you are using.

Traps are accompanied by information from the **Entity Table**, which contains information specific to the Directory Server instance, such as its name and version number. The **Entity Table** is described in [Section 16.6.3, “Entity Table”](#). This means that the action the master agent takes when it receives a trap is flexible, such as sending an email to an email address defined in the **dsEntityContact** variable for one instance while sending a notification to a pager number in the **dsEntityContact** variable for another instance.

There are two traps supported by the subagent:

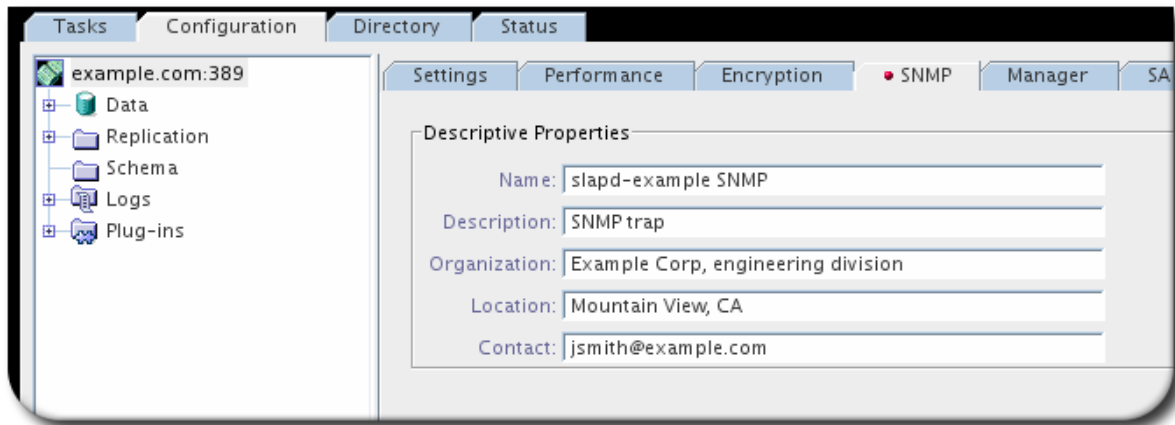
- *DirectoryServerDown*. This trap is generated whenever the subagent detects the Directory Server is potentially not running. This trap will be sent with the Directory Server instance description, version, physical location, and contact information, which are detailed in the **dsEntityDescr**, **dsEntityVers**, **dsEntityLocation**, and **dsEntityContact** variables.
- *DirectoryServerStart*. This trap is generated whenever the subagent detects that the Directory Server has started or restarted. This trap will be sent with the Directory Server instance description, version, physical location, and contact information, which are detailed in the **dsEntityDescr**, **dsEntityVers**, **dsEntityLocation**, and **dsEntityContact** variables.

## 16.5. CONFIGURING THE DIRECTORY SERVER FOR SNMP

By default, the Directory Server is ready to be monitored using SNMP as soon as the subagent is configured. However, there are some useful variables in the Directory Server instances which can be configured to help identify the Directory Server instance with SNMP. To configure these SNMP settings from the Directory Server Console:

1. Select the **Configuration** tab, and then select the topmost entry in the navigation tree in the left pane.
2. Select the **SNMP** tab in the main window.
3. Fill in the information about the SNMP descriptors so that it is easy to identify the Directory Server instance in Net-SNMP.





- A unique name and description for the instance.
- The company or organization to which the directory instance belongs.
- The physical location of the directory instance or the organization which manages the instance.
- The email address or contact number for the person who maintains the Directory Server instance.

4. Click **Save**.

## 16.6. USING THE MANAGEMENT INFORMATION BASE

The Directory Server's MIB is a file called **redhat-directory.mib**. This MIB contains definitions for variables pertaining to network management for the directory. These variables are known as managed objects. Using the directory MIB and Net-SNMP, you can monitor your directory like all other managed devices on your network. For more information on using the MIB, see [Section 16.3.3, "Testing the Subagent"](#).

The client tools need to load the Directory Server MIB to use the variable names listed in the following sections.

Using the directory MIB enables administrators to use SNMP to see administrative information about the directory and monitor the server in real-time. The directory MIB is broken into four distinct tables of managed objects:

- [Section 16.6.1, "Operations Table"](#)
- [Section 16.6.2, "Entries Table"](#)
- [Section 16.6.3, "Entity Table"](#)
- [Section 16.6.4, "Interaction Table"](#)



### NOTE

All of the Directory Server attributes monitored by SNMP use 64-bit integers for the counters, even on 32-bit systems.

### 16.6.1. Operations Table

The **Operations Table** provides statistical information about Directory Server access, operations, and errors. [Table 16.1, “Operations Table: Managed Objects and Descriptions”](#) describes the managed objects stored in the **Operations Table** of the `redhat-directory.mib` file.

**Table 16.1. Operations Table: Managed Objects and Descriptions**

| Managed Object       | Description                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dsAnonymousBinds     | The number of anonymous binds to the directory since server startup.                                                                                                                                       |
| dsUnauthBinds        | The number of unauthenticated binds to the directory since server startup.                                                                                                                                 |
| dsSimpleAuthBinds    | The number of binds to the directory that were established using a simple authentication method (such as password protection) since server startup.                                                        |
| dsStrongAuthBinds    | The number of binds to the directory that were established using a strong authentication method (such as SSL or a SASL mechanism like Kerberos) since server startup.                                      |
| dsBindSecurityErrors | The number of bind requests that have been rejected by the directory due to authentication failures or invalid credentials since server startup.                                                           |
| dsInOps              | The number of operations forwarded to this directory from another directory since server startup.                                                                                                          |
| dsReadOps            | The number of read operations serviced by this directory since application start. The value of this object will always be 0 because LDAP implements read operations indirectly using the search operation. |
| dsCompareOps         | The number of compare operations serviced by this directory since server startup.                                                                                                                          |
| dsAddEntryOps        | The number of add operations serviced by this directory since server startup.                                                                                                                              |
| dsRemoveEntryOps     | The number of delete operations serviced by this directory since server startup.                                                                                                                           |
| dsModifyEntryOps     | The number of modify operations serviced by this directory since server startup.                                                                                                                           |
| dsModifyRDNOps       | The number of modify RDN operations serviced by this directory since server startup.                                                                                                                       |

| Managed Object          | Description                                                                                                                                                                                                                                             |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dsListOps               | The number of list operations serviced by this directory since server startup. The value of this object will always be 0 because LDAP implements list operations indirectly using the search operation.                                                 |
| dsSearchOps             | The total number of search operations serviced by this directory since server startup.                                                                                                                                                                  |
| dsOneLevelSearchOps     | The number of one-level search operations serviced by this directory since server startup.                                                                                                                                                              |
| dsWholeSubtreeSearchOps | The number of whole subtree search operations serviced by this directory since server startup.                                                                                                                                                          |
| dsReferrals             | The number of referrals returned by this directory in response to client requests since server startup.                                                                                                                                                 |
| dsSecurityErrors        | The number of operations forwarded to this directory that did not meet security requirements.                                                                                                                                                           |
| dsErrors                | The number of requests that could not be serviced due to errors (other than security or referral errors). Errors include name errors, update errors, attribute errors, and service errors. Partially serviced requests will not be counted as an error. |

### 16.6.2. Entries Table

The **Entries Table** provides information about the contents of the directory entries. [Table 16.2, “Entries Table: Managed Objects and Descriptions”](#) describes the managed objects stored in the **Entries Table** in the `redhat-directory.mib` file.

**Table 16.2. Entries Table: Managed Objects and Descriptions**

| Managed Object  | Description                                                                                                                                                            |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dsMasterEntries | The number of directory entries for which this directory contains the master entry. The value of this object will always be 0 (as no updates are currently performed). |
| dsCopyEntries   | The number of directory entries for which this directory contains a copy. The value of this object will always be 0 (as no updates are currently performed).           |
| dsCacheEntries  | The number of entries cached in the directory.                                                                                                                         |

| Managed Object | Description                                                                                                                             |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| dsCacheHits    | The number of operations serviced from the locally held cache since application startup.                                                |
| dsSlaveHits    | The number of operations that were serviced from locally held replications (shadow entries). The value of this object will always be 0. |

### 16.6.3. Entity Table

The **Entity Table** contains identifying information about the Directory Server instance. The values for the **Entity Table** are set in the Directory Server Console, as described in [Section 16.5, “Configuring the Directory Server for SNMP”](#).

[Table 16.3, “Entity Table: Managed Objects and Descriptions”](#) describes the managed objects stored in the **Entity Table** of the `redhat-directory.mib` file.

**Table 16.3. Entity Table: Managed Objects and Descriptions**

| Managed Object   | Description                                                                                    |
|------------------|------------------------------------------------------------------------------------------------|
| dsEntityDescr    | The description set for the Directory Server instance.                                         |
| dsEntityVers     | The Directory Server version number of the Directory Server instance.                          |
| dsEntityOrg      | The organization responsible for the Directory Server instance.                                |
| dsEntityLocation | The physical location of the Directory Server instance.                                        |
| dsEntityContact  | The name and contact information for the person responsible for the Directory Server instance. |
| dsEntityName     | The name of the Directory Server instance.                                                     |

### 16.6.4. Interaction Table



#### NOTE

The **Interaction Table** is *not* supported by the subagent. The subagent can query the table, but it will not ever be updated with valid data.

[Table 16.4, “Interaction Table: Managed Objects and Descriptions”](#) describes the managed objects stored in the **Interaction Table** of the `redhat-directory.mib` file.

**Table 16.4. Interaction Table: Managed Objects and Descriptions**

| Managed Object             | Description                                                                                                                                                                                                                                                                                  |
|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dsIntTable                 | Details, in each row of the table, related to the history of the interaction of the monitored Directory Servers with their respective peer Directory Servers.                                                                                                                                |
| dsIntEntry                 | The entry containing interaction details of a Directory Server with a peer Directory Server.                                                                                                                                                                                                 |
| dsIntIndex                 | Part of the unique key, together with <b>applIndex</b> , to identify the conceptual row which contains useful information on the (attempted) interaction between the Directory Server (referred to by <b>applIndex</b> ) and a peer Directory Server.                                        |
| dsName                     | The distinguished name (DN) of the peer Directory Server to which this entry belongs.                                                                                                                                                                                                        |
| dsTimeOfCreation           | The value of <b>sysUpTime</b> when this row was created. If the entry was created before the network management subsystem was initialized, this object will contain a value of zero.                                                                                                         |
| dsTimeOfLastAttempt        | The value of <b>sysUpTime</b> when the last attempt was made to contact this Directory Server. If the last attempt was made before the network management subsystem was initialized, this object will contain a value of zero.                                                               |
| dsTimeOfLastSuccess        | The value of <b>sysUpTime</b> when the last attempt made to contact this Directory Server was successful. This entry will have a value of zero if there have been no successful attempts or if the last successful attempt was made before the network management subsystem was initialized. |
| dsFailuresSinceLastSuccess | The number of failures since the last time an attempt to contact this Directory Server was successful. If there has been no successful attempts, this counter will contain the number of failures since this entry was created.                                                              |
| dsFailures                 | Cumulative failures since the creation of this entry.                                                                                                                                                                                                                                        |
| dsSuccesses                | Cumulative successes since the creation of this entry.                                                                                                                                                                                                                                       |
| dsURL                      | The URL of the Directory Server application.                                                                                                                                                                                                                                                 |

## CHAPTER 17. PLANNING FOR DISASTER

Part of running a Directory Server deployment efficiently is planning for that worst case scenario. This chapter covers general principles for drafting a disaster recovery plan and highlights features in Directory Server that can be used to aide in disaster recovery.

*Disaster recovery* is a way of planning and implementing a smooth transition from one operating environment to another environment whenever there is some sort of catastrophic failure. A disaster recovery plan for Directory Server may be part of a larger business continuity plan or it could be a standalone plan specifically for an interruption in directory services.



### NOTE

This chapter covers very general concepts for disaster recovery.

Disaster recovery can be a very complex and detail-specific thing. Consider using a professional service to design, maintain, and test any disaster recovery plan for sensitive or mission-critical services, like Red Hat Directory Server.

### 17.1. IDENTIFYING POTENTIAL SCENARIOS

The first step is identifying what potential issues you may encounter, what services will be affected, and what responses you should take. In the *Directory Server Deployment Guide*, administrators made a site survey of their existing and proposed infrastructure to determine what kind of directory to design. Do something similar for disaster planning; as in [Table 17.1, “Disaster Scenarios and Responses”](#), identify where your data infrastructure is, determine what the affect of losing that component is, and look at potential ideal responses.

**Table 17.1. Disaster Scenarios and Responses**

| Scenario                                | Effects on Infrastructure                                                                                                                                                                                                                                                    | Ideal Response                                                                                                                                                                                                                                                                |
|-----------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Data corruption                         | Through software or hardware failure (or through a malicious attack), the data at one site or on one server could be corrupted. If that corrupted server is a supplier in multi-master replication, then the corruption can quickly be propagated throughout the deployment. | An isolated server should be available with access to the most recent backup of uncorrupted data. When a problem is detected, replication can be suspended on the regular infrastructure, and this server can be brought online to reinitialize the suppliers with good data. |
| Natural disasters and other mass events | Natural disasters can take an entire office or data center offline, even through something as simple as a long-term power outage.                                                                                                                                            | Directory operations can be transferred to a mirrored site at another physical location, with the same data.                                                                                                                                                                  |
| Server or machine loss                  | A single machine could fail.                                                                                                                                                                                                                                                 | Another machine, with the same data, can assume the lost machine's place.                                                                                                                                                                                                     |

## 17.2. DEFINING THE TYPE OF ROLLOVER

Disaster recovery, as the introduction says, is the process for transitioning from one system to another system with as little interruption of service as possible. That's called a *rollover*, and there are three different ways of doing a rollover:

- A *hot* rollover means that the infrastructure is completely mirrored at another site and that the backup site is always up and current with the primary site. This requires only a few adjustments to switch operations from the primary to the backup.
- A *warm* rollover means that all of the elements for the backup site are in place (adequate network connections, all required applications and hardware) but the system is not actively running or necessarily configured. This can require some extra time to configure the machines and get the system running.
- A *cold* rollover means that a site is available but there are few resources immediately available to set it up.

The obvious difference in the types of rollover is the time and expense necessary to set up the backup site. Hot and warm sites have higher initial expenditures to set up and run.

A mix of rollover types can be used, depending on the specific disaster scenario being planned. For example, a rollover plan for the loss of a single server could use a hot rollover easily and relatively cheaply by creating and keeping a virtual machine copy of the Directory Server instance which can be brought online within minutes. It would not even require keeping the virtual machine in a separate facility or network. On the other hand, a cold rollover could be planned for the loss of an entire data center or office.

Match the rollover process to the severity of the disaster scenario, your budget and available resources, and the likelihood of encountering problems.

## 17.3. IDENTIFYING USEFUL DIRECTORY SERVER FEATURES FOR DISASTER RECOVERY

The hardest part of a recovery is not the hardware; it is getting a reliable copy of the data in the server. There are three Directory Server features that are excellent tools for preparing data copies for disaster recovery:

- Multi-master replication, chaining, backing up databases, and monitoring the server with a named pipe script.
- Chaining
- Backing up databases regularly

Additionally, monitoring the server with a named pipe script and with other Directory Server performance counters can be effective at catching and quickly responding to specific, critical events.

### 17.3.1. Multi-Master Replication for Disaster Recovery

Multi-master replication is the best defense against losing a single server and, possibly, even an entire office or department. While a small number of servers are data masters, multiple servers all hold the same data — potentially dozens of masters and hubs in a single replication environment. This keeps information accessible to clients even if multiple servers go offline.

Replication can be used to copy over data to servers and bring replacements online more quickly.

**NOTE**

To protect against data corruption being propagated through replication, schedule a lag in replication to a backup server or to another server cluster. This way, replication can be stopped before the corrupt data are replicated over to the backup site.

Replication configuration also allows write operations to be referred to failover servers if the primary supplier is inaccessible. This means that write operations can proceed as normal from the client perspective, even when servers go offline.

**Example 17.1. Scenarios for Multi-Master Replication**

Replication is a versatile tool for disaster recovery in several scenarios:

- For a single server failure, all of the data stored on that instance is both accessible and retrievable from other servers.
- For the loss of an entire office or colocation facility, servers can be mirrored at an entirely different physical location (which is aided by Directory Server's wide area replication performance). With minimal effort, traffic can be redirected to the replicated site without having to bring new servers online.

Configuring replication is covered in [Chapter 11, \*Managing Replication\*](#).

**17.3.2. Chaining Databases for Disaster Recovery**

*Chaining* is a configuration where a client sends a request to one server and it automatically forwards that request to another server to process. There can be multiple servers configured in the database link (or chain) to allow for automatic failover if one server is not available.

**Example 17.2. Scenarios for Chaining**

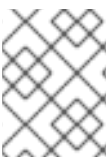
When chaining is combined with a list of failover servers, client traffic can be automatically redirected from a single server (or even group of servers) when they are offline. This does not help in recovery, but it helps manage the transition from primary to backup servers.

Chaining databases is covered in [Section 2.3, “Creating and Maintaining Database Links”](#).

**17.3.3. Backing up Directory Data for Disaster Recovery**

The most useful tool for disaster recovery is to do frequent backups of a directory instance. Archives can be stored on physical media, at different locations than the primary data center or on-site at a cold backup location. Because both backup and restore operations can be done through either shell or perl scripts (such as **db2bak.pl**) backups can be automated to run regularly through simple cron jobs.

```
0 7 * * 1 /usr/lib[64]/dirsrv/slapd-example/db2bak.pl
```

**NOTE**

The **db2bak.pl** Perl script backs up the directory data without having to stop the server first.



Backing up both directory databases and the directory configuration (`dse.ldif` file) are covered in [Section 4.3, “Backing up and Restoring Data”](#).

### 17.3.4. Using a Named Pipe Script for Disaster Recovery

Named pipe scripts can be used to do two very good things: identify specific error messages and work with plug-ins or other scripts to respond to events. For certain kinds of failures and recoveries, this could be used to essentially automate a response, such as rerouting client traffic or bringing a virtual machine online.

Using named pipe scripts is described in [Section 15.5, “Replacing Log Files with a Named Pipe”](#).

## 17.4. DEFINING THE RECOVERY PROCESS

There are a lot of tools that can help with disaster recovery, but an effective recovery process circles back to having a well-defined plan of what to do in every scenario. Two things, at least, need to be clearly identified:

- What signals a disaster? Some things are obvious (a massive power outage, network loss, or fire), but other situations need to be defined. For example, what signals that a backup server needs to be brought online?
- Who responds to a disaster and how? Once a disaster situation occurs, who has the responsibility to act? How are they notified of the event? What are they expected to do?

After a disaster plan is written for each scenario, then test the plan regularly (at least yearly). As the server configuration changes, be sure to update the plan so it reflects the current infrastructure.

## 17.5. BASIC EXAMPLE: PERFORMING A RECOVERY

An administrator, John Smith, has to create a disaster recovery plan for his directory deployment. Example Corp. has three physical offices, in San Francisco, Dallas, and Arlington. Each site has 10 servers which replicate to each other locally, and then one server at each site replicates to another server at the other two sites.

Each site has business-critical customer data stored in its directory, as well as human resources data. Several external applications require access to the data to perform operations like billing.

John Smith's first step is to perform a site survey. He is looking for three things: what his directory usage is (clients that access it and traffic loads across the sites), what his current assets are, and what assets he may need to acquire. This is much like the initial site survey he performed when deploying Red Hat Directory Server.

His next step is identifying potential disaster scenarios. Two of the three sites are highly vulnerable to natural disasters (San Francisco and Dallas). All three sites could face normal interruptions, like outages for power or Internet access. Additionally, since each site masters its own local data, each site is vulnerable to losing a server instance or machine.

John Smith then breaks his disaster recovery plan into three parts:

- Plan A covers losing a single instance of Directory Server
- Plan B covers some kind of data corruption or attack
- Plan C covers losing an entire office

For plans A and B, John Smith decides to use a hot recovery to immediately switch functionality from a single instance to the backup. Each server is backed up daily, using a cron job, and then the archive is copied over and restored on a virtual machine. The virtual machine is kept on a different subnet, but can be switched over immediately if its peer ever does offline. John Smith uses simple SNMP traps to track each Directory Server instance's availability.

Plan C is more extensive. Along with replication between sites and the local backups, he decides to mail a physical copy of each site's backup, for every local instance, once a week to the other two colocation facilities. He also keep a spare server with adequate Internet access and software licenses to restore an entire site, using virtual machines, one of the other different colocation facilities. He designates the Arlington site as the primary recovery location because that is where most of the IT staff is located, then San Francisco and last Dallas, based on the distribution of personnel. For every event, the IT administrator at all three sites will be notified, and the manager assumes the responsibilities of setting up the virtual machines, restoring the Directory Server instances from the physical backups, and rerouting client traffic.

John Smith schedules to review and update the plan quarterly to account for any new hardware or application changes. Once a year, all three sites have to run through the procedure of recovering and deploying the other two sites, according to the procedures in Disaster Plan C.

## APPENDIX A. USING LDAP CLIENT TOOLS

Red Hat Directory Server 9.0 uses the LDAP tools (such as **ldapsearch** and **ldapmodify**) supplied with OpenLDAP. The OpenLDAP tool options are described in the OpenLDAP manpages at <http://www.openldap.org/software/man.cgi>.

This appendix gives some common usage scenarios and examples for using these LDAP tools.

More extensive examples for using **ldapsearch** are given in [Chapter 10, Finding Directory Entries](#). More examples for using **ldapmodify** and **ldapdelete** are given in [Section 3.2, “Managing Entries from the Command Line”](#).

### A.1. ENVIRONMENT VARIABLES USED WITH LDAP CLIENT TOOLS

Some information related to running LDAP client tools can be set through environment variables. This allows certain operation conditions (like SSL/TLS settings) to be set once and then applied consistently to every operation.



#### NOTE

The SSL/TLS parameters can be set as either an environment variable or within the OpenLDAP configuration, meaning set in `/etc/openldap/ldap.conf` or the `$HOME/[.]ldaprc` profiles.

**Table A.1. LDAP Tools Environment Variables**

| Environment Variable | ldap.conf Parameter | Description                                                                                                                                                    |
|----------------------|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LDAP_BASEDN          | none                | Sets the default base DN for <b>ldapsearch</b> to use. This is equivalent to the <b>-b</b> argument and allows that argument to be skipped.                    |
| LDAPTLS_CACERTDIR    | TLS_CACERTDIR       | Gives the directory where the NSS security databases ( <b>cert8.db</b> and <b>key3.db</b> ) are located. For example, <b>/etc/dirsrv/slapd-instance_name</b> . |
| LDAPTLS_CERT         | TLS_CERT            | Gives the nickname for the server certificate in the <b>cert8.db</b> database. For example, <b>Server-Cert</b> .                                               |

| Environment Variable | ldap.conf Parameter | Description                                                                                                                                                                                                                                 |
|----------------------|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LDAPTLS_KEY          | TLS_KEY             | Gives the password and, optionally, the token name which stores the key, in the format <i>[token_name:]password</i> . The default token name (which is assumed) is <b>internal</b> . For example, <b>internal:secret</b> or <b>secret</b> . |

## A.2. USING SSL/TLS AND START TLS WITH LDAP CLIENT TOOLS

There are three things that must be configured for SSL/TLS connections with LDAP command-line tools:

- The appropriate environment variables must be set before running the command.
- The appropriate arguments must be passed with the command to identify the server to which to connect. This requires either **-H** to specify an LDAP or LDAP URL or **-h** and **-p** to give the fully-qualified domain name and port of the server.



### NOTE

The fully-qualified domain name is *always* required with the **-h** option. This prevents man-in-the-middle attacks.

- SSL/TLS must be specified. This can be done by invoking Start TLS with the **-Z** or **-ZZ** (force Start TLS) options. Start TLS is described in [Section 7.5, “Command-Line Functions for Start TLS”](#). For example:

```
ldapsearch -D "cn=directory manager" -W -p 389 -h server.example.com
-b "dc=example,dc=com" -s sub -x -ZZ "(objectclass=*)"
```

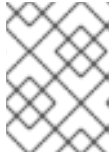
Alternatively, this can be done using the LDAPS protocol with **-H** or using the secure LDAPS port with the **-p** option. Although using the **ldaps** protocol is supported, it is deprecated. The recommended method is to use Start TLS.

When using SSL/TLS with LDAP command-line tools for client connections, the appropriate TLS environment variables ([Section A.1, “Environment Variables Used with LDAP Client Tools”](#)) must be set in order to access the required security databases and certificates.

To set up *system-wide* SSL/TLS configuration for LDAP tools, edit the **ldap.conf** file. To edit *per-user* SSL/TLS configuration for LDAP tools, edit the **\$HOME/.ldaprc** profile for the specific user. Whichever file is edited, the same configuration parameters need to be set.

1. Open the **ldap.conf** file or **\$HOME/.ldaprc** profile. For example:

```
vim /etc/openldap/ldap.conf
```

**NOTE**

These parameters can also be set as environment variables. See [Table A.1, “LDAP Tools Environment Variables”](#) for the variable names.

2. Add a line to define the security databases directory location. This is required.

```
TLS_CACERTDIR /etc/dirsrv/slapd-instance_name
```

3. Optionally, add lines for the client certificate name and token database password for the Directory Server's NSS security databases. This allows certificate-based client authentication.

```
TLS_CERT Server-Cert
TLS_KEY internal:secret
```

Once the security databases parameters are set, then SSL connections can be invoked with the LDAP command-line tools. For example, using **-ZZ** opens a Start TLS connection and forces the use of TLS or the operation fails:

```
ldapsearch -D "cn=directory manager" -W -p 389 -h server.example.com -b
"dc=example,dc=com" -s sub -x -ZZ "(objectclass=*)"
```

The **-x** allows simple (user name/password) binds. Alternatively, use the **-Y EXTERNAL** option to indicate that an authentication method other than SASL is being used. The **-Y EXTERNAL** argument can be used with client authentication:

```
ldapsearch -H ldaps://server.example.com:636 -b "dc=example,dc=com" -Y
EXTERNAL "givenname=Richard"
```

### A.3. USING SASL WITH LDAP CLIENT TOOLS

Directory Server uses SASL for authentication and network security, particularly for environments which are using Kerberos to implement single sign-on. Directory Server allows user to use SASL to authenticate and bind to the server and then to encrypt (secure) the network connection to the server.

SASL authentication and security include LDAP tools like **ldapsearch** and **ldapmodify**. For example:

```
ldapsearch -O noplain,minssf=1,maxbufsize=512 -Y GSSAPI -U
"dn:uid=jsmith,ou=people,dc=example,dc=com" -R EXAMPLE.COM ...
```

The SASL-related LDAP tool parameters are listed in [Table A.2, “LDAP Client Tool SASL Parameters”](#).

**NOTE**

SASL proxy authorization is not supported in Directory Server; therefore, Directory Server ignores any SASL **authzid** value supplied by the client.

Two primary pieces of information are required to use SASL with Directory Server:

- The authentication method, in this example GSS-API

- The user as whom you are authenticating (the authorization ID)

Other information, such as the Kerberos realm, can also be passed with the command.

When a client connects to Directory Server using SASL, the Directory Server takes the identity offered as the SASL ***authid*** and maps that entry back to an entry in the Directory Server. If the ***authid*** is defined as a DN (as in ***authid=dn: DN***), this is done simply by matching the DN. It is also possible to use a user name or a part of a DN, and these can be mapped to the directory entry using SASL identity mappings.

**Table A.2. LDAP Client Tool SASL Parameters**

| Option | Description | Allowed Values |
|--------|-------------|----------------|
|--------|-------------|----------------|

| Option | Description                                                       | Allowed Values                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------|-------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -O     | <i>Optional.</i> Sets the security properties for the connection. | <p>All mechanisms:</p> <ul style="list-style-type: none"> <li>• <i>noplain</i> — Do not permit mechanisms susceptible to simple passive attack.</li> <li>• <i>noanonymous</i> — Do not permit mechanisms that allow anonymous access.</li> <li>• <i>minssf</i> — Require a minimum security strength; this option needs a numeric value specifying bits of encryption. A value of - <b>1</b> means integrity is provided without privacy.</li> <li>• <i>maxssf</i> — Require a maximum security strength; this option needs a numeric value specifying bits of encryption. A value of - <b>1</b> means integrity is provided without privacy.</li> </ul> <p>GRAM-MD mechanism only:</p> <ul style="list-style-type: none"> <li>• <i>noactive</i> — Do not permit mechanisms susceptible to active attacks.</li> <li>• <i>nodict</i> — Do not permit mechanisms susceptible to passive dictionary attacks.</li> <li>• <i>forwardsec</i> — Require forward secrecy.</li> <li>• <i>passcred</i> — Attempt to pass client credentials.</li> <li>• <i>maxbufsize</i> — Set the maximum receive buffer size the client will accept when using integrity or privacy settings.</li> </ul> |
| -R     | Gives the Kerberos realm.                                         | Depends on the mechanism.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

| Option | Description                                      | Allowed Values                                                                                                                                                                                                                                 |
|--------|--------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -U     | Gives the ID used to authenticate to the server. | <ul style="list-style-type: none"> <li>• <i>UID</i>. For example, <b>msmith</b>.</li> <li>• <i>u: uid</i>. For example, <b>u:msmith</b>.</li> <li>• <i>dn: dn_value</i>. For example, <b>dn:uid=msmith,ou=People,o=example.com</b>.</li> </ul> |
| -Y     | Sets the SASL authentication mechanism to use.   | <ul style="list-style-type: none"> <li>• GSSAPI</li> <li>• CRAM-MD5</li> <li>• DIGEST-MD5</li> <li>• EXTERNAL</li> <li>• PLAIN</li> </ul>                                                                                                      |

## A.4. RUNNING EXTENDED OPERATIONS

Red Hat Directory Server supports a variety of extended operations, especially extended search operations. An extended operation passes an additional operation (such as a get effective rights search or server-side sort) along with the LDAP operation. Likewise, LDAP clients have the potential to support a number of extended operations.

The OpenLDAP LDAP tools support extended operations in two ways. All client tools (**ldapmodify**, **ldapsearch**, and the others) use either the **-e** or **-E** options to send an extended operation. The **-e** argument can be used with any OpenLDAP client tool and sends general instructions about the operation, like how to handle password policies. The **-E** is used only with **ldapsearches** and passes more useful controls like GER searches, sort and page information, and information for other, not-explicitly-support extended operations.

Additionally, OpenLDAP has another tool, **ldapexop**, which is used exclusively to perform extended search operations, the same as running **ldapsearch -E**.

The format of an extended operation with **ldapsearch** is generally:

```
-E extended_operation_type=operation_parameters
```

When an extended operation is explicitly handled by the OpenLDAP tools, then the *extended\_operation\_type* can be an alias, like **deref** for a dereference search or **sss** for server-side sorting. A supported extended operation has formatted output. Other extended operations, like GER searches, are passed using their OID rather than an alias, and then the *extended\_operation\_type* is the OID. For those unsupported operations the tool does not recognize the response from the server, so the output is unformatted.



For example, the **pg** extended operation type formats the results in simple pages:

```
ldapsearch -x -D "cn=Directory Manager" -W -b
"ou=Engineers,ou=People,dc=example,dc=com" -E pg=3 "(objectclass=*)" cn

dn: uid=jsmith,ou=Engineers,ou=People,dc=example,dc=com
 cn: John Smith

dn: uid=bjensen,ou=Engineers,ou=People,dc=example,dc=com
 cn: Barbara Jensen

dn: uid=hmartin,ou=Engineers,ou=People,dc=example,dc=com
 cn: Henry Martin

Results are sorted.
next page size (3): 5
```

The same operation with **ldapexop** can be run using only the OID of the simple paged results operation and the operation's settings (3 results per page):

```
ldapexop 1.2.840.113556.1.4.319=3
```

However, **ldapexop** does not accept the same range of search parameters that **ldapsearch** does, making it less flexible.

## A.5. ADDING ENTRIES

There are two ways to add entries using LDAP client tools:

- **ldapadd**
- **ldapmodify -a**

Both **ldapmodify -a** and **ldapadd** are exactly the same; the only difference is that with **ldapmodify**, the **-a** option explicitly states that it is an add operation. With **ldapadd**, the **-a** option is implied.

The example using **ldamodify -a** is given in [Section 3.2.4.1, “Adding Entries Using ldapmodify”](#). To add entries using **ldapadd**, use the same options and format as **ldamodify -a**:

```
ldapadd -D "cn=directory manager" -w secret -p 389 -h server.example.com -
x -f new.ldif

ldapadd -D "cn=directory manager" -w secret -p 389 -h server.example.com -
x

dn: uid=jsmith,ou=people,dc=example,dc=com
objectclass: inetOrgPerson
objectclass: person
objectclass: top
uid: jsmith
sn: Smith
```

```
givenname: John
mail: jsmith@example.com
cn: John Smith
```

## A.6. COMPARING ENTRIES

**ldapcompare** checks entries to see if the specified entry or entries contain an attribute of a specific value. For example, this checks to see if an entry has an **sn** value of Smith:

```
ldapcompare -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x sn:smith uid=bjensen,ou=people,dc=example,dc=com
comparing type: "sn" value: "smith" in entry
"uid=bjensen,ou=people,dc=example,dc=com"
compare FALSE
```

```
ldapcompare -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x sn:smith uid=jsmith,ou=people,dc=example,dc=com
comparing type: "sn" value: "smith" in entry
"uid=jsmith,ou=people,dc=example,dc=com"
compare TRUE
```

The compare attribute can be specified in one of three ways:

- A single *attribute:value* statement passed in the command line directly

```
sn:Smith
```

- A single *attribute::base64value* statement passed in the command line directly, for attributes like **jpegPhoto** or to verify certificates or CRLs

```
jpegPhoto:dkdkPDKCDdKo0eiofk==
```

- An *attribute:file* statement that points to a file containing a list of comparison values for the attribute, and the script iterates through the list

```
postalCode:/tmp/codes.txt
```

The compare operation itself has to be run against a specific entry or group of entries. A single entry DN can be passed through the command line, or a list of DNs to be compared can be given using the **-f** option.

### Example A.1. Comparing One Attribute Value to One Entry

Both the attribute-value comparison and the DN are passed with the script.

```
ldapcompare -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x sn:smith uid=jsmith,ou=people,dc=example,dc=com
comparing type: "sn" value: "smith" in entry
"uid=jsmith,ou=people,dc=example,dc=com"
compare TRUE
```

**Example A.2. Comparing a List Attribute Values from a File**

First, create a file of possible *sn* values.

```
jensen
johnson
johannson
jackson
jorgenson
```

Then, create a list of entries to compare the values to.

```
uid=jen200,ou=people,dc=example,dc=com
uid=dsj,ou=people,dc=example,dc=com
uid=matthewjms,ou=people,dc=example,dc=com
uid=john1234,ou=people,dc=example,dc=com
uid=jack.son.1990,ou=people,dc=example,dc=com
```

Then run the script.

```
ldapcompare -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x sn:/tmp/surnames.txt -f /tmp/names.txt
comparing type: "sn" value: "jensen" in entry
"uid=jen200,ou=people,dc=example,dc=com"
compare TRUE
```

**A.7. CHANGING PASSWORDS**

The **ldappasswd** command can either set a new user-defined password or generate a new password for an account. [Table A.3, “Password Operation-Related Parameters for ldappasswd”](#) lists the most important parameters for setting passwords through the command line. Other settings (for bind information, connection information, or other command settings) may be required and are listed in the OpenLDAP manpages.

```
ldappasswd -x -D bind_dn -w password -p server_port -h server_hostname [-A
| -a oldPassword] [-S | -s newPassword] [user]
```

**IMPORTANT**

Password change operations must be run over a secure connection, such as SSL/TLS, Start TLS, or SASL. For information on how to configure SSL/TLS for LDAP clients, see [Section A.2, “Using SSL/TLS and Start TLS with LDAP Client Tools”](#).

**Table A.3. Password Operation-Related Parameters for ldappasswd**

| Option | Description                                                |
|--------|------------------------------------------------------------|
| -A     | Prompts for the original password, which is being changed. |

| Option      | Description                                                                                                                                                                                                                                                                 |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -a          | Gives the old password, which is being changed.                                                                                                                                                                                                                             |
| -n          | Tells the server not to set a new password. This is mainly used with the <b>-v</b> option (which increases the verbosity of the output) or the <b>-d</b> option (which sets the debug level) by testing the output without actually performing a password change operation. |
| -S          | Prompts for the new password.                                                                                                                                                                                                                                               |
| -s          | Sets the new password.                                                                                                                                                                                                                                                      |
| <i>user</i> | Gives the DN of the user entry for which to change the password.                                                                                                                                                                                                            |

### Example A.3. Directory Manager Changing a User's Password Over SSL

The Directory Manager changes the password of the user **uid=tuser1,ou=People,dc=example,dc=com** to **new\_password** over SSL.

```
ldappasswd -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -s new_password
"uid=tuser1,ou=People,dc=example,dc=com"
```

### Example A.4. Directory Manager Generating a User's Password

The Directory Manager generates the password of the user **uid=tuser2,ou=People,dc=example,dc=com** over SSL.

```
ldappasswd -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x "uid=tuser2,ou=People,dc=example,dc=com"
```

### Example A.5. User Changing His Own Password

A user, **tuser3**, changes the password from **old\_newpassword** to **new\_password** over SSL.

```
ldappasswd -p 389 -h server.example.com -x -D
"uid=tuser3,ou=People,dc=example,dc=com" -W -a old_password -s
new_password
```

### Example A.6. User Authenticating with DIGEST\_MD5 and Changing His Password

A user, **jsmith**, authenticates with GSS-API and changes the password to **new\_password**.

```
ldappasswd -p 389 -h server.example.com -O
noplain,minssf=1,maxbufsize=512 -Y GSSAPI -U
"dn:uid=jsmith,ou=people,dc=example,dc=com" -R EXAMPLE.COM -W -s
new_password
```

### Example A.7. User Already Authenticated by Kerberos Prompts for a New Password

A user, who has already authenticated by Kerberos, prompts for the new password. This is not performed over SSL.

```
ldappasswd -p 389 -h server.example.com -O
noplain,minssf=1,maxbufsize=512 -I
```

## A.8. GENERATING LDAP URLs

LDAP URLs are used in a variety of different configuration areas and operations: referrals and chaining, replication, synchronization, ACIs, and indexing, as a starting list. Constructing accurate LDAP URLs is critical, because incorrect URLs may connect to the wrong server or simply cause operations to fail. Additionally, all OpenLDAP tools allow the **-H** option to pass an LDAP URL instead of other connection information (like the host name, port, subtree, and search base).



### NOTE

LDAP URLs are described in [Appendix C, LDAP URLs](#).

The **ldapurl** command manages URL in two ways:

- Deconstruct a given LDAP URL into its constituent element
- Construct a new, valid LDAP URL from given elements

The parameters for working with URLs are listed in [Table A.4, “Idapurl Parameters”](#); the full list of parameters are in the OpenLDAP manpages.

**Table A.4. Idapurl Parameters**

| Option                          | Description                                                                          |
|---------------------------------|--------------------------------------------------------------------------------------|
| <b>For Deconstructing a URL</b> |                                                                                      |
| <b>-H "URL"</b>                 | Passes the LDAP URL to break down into elements.                                     |
| <b>For Constructing a URL</b>   |                                                                                      |
| <b>-a <i>attributes</i></b>     | Gives a comma-separated attributes that are specifically returned in search results. |
| <b>-b <i>base</i></b>           | Sets the search base or subtree for the URL.                                         |

| Option                           | Description                                                                                  |
|----------------------------------|----------------------------------------------------------------------------------------------|
| <code>-f filter</code>           | Sets the search filter to use.                                                               |
| <code>-h hostname</code>         | Gives the Directory Server's host name.                                                      |
| <code>-p port</code>             | Gives the Directory Server's port.                                                           |
| <code>-S ldap ldaps ldapi</code> | Gives the protocol to use to connect, such as <b>ldap</b> , <b>ldaps</b> , or <b>ldapi</b> . |
| <code>-s scope</code>            | Gives the search scope.                                                                      |

### Example A.8. Deconstructing an LDAP URL

**ldapur1** uses the **-H** option to feed in an existing LDAP URL, and the tool returns the elements of the URL in a neat list:

```
ldapur1 -H "ldap://:389/dc=example,dc=com?cn,sn?sub?
(objectclass=inetorgperson)"
scheme: ldap
port: 389
dn: dc=example,dc=com
selector: cn
selector: sn
scope: sub
filter: (objectclass=inetorgperson)
```

### Example A.9. Constructing an LDAP URL

The most useful application of **ldapur1** is to construct a valid LDAP URL manually. The Directory Server Console has tools to develop valid URLs for areas like ACIs and referrals, but very complex configurations or scripted operations may require administrators to manually construct the URL. Using **ldapur1** ensures that the URL is valid.

**ldapur1** accepts the normal connection parameters of all LDAP client tools and additional **ldapsearch** arguments for search base, scope, and attributes, but this tool never connects to a Directory Server instance, so it does not require any bind information. It accepts the connection and search settings and feeds them in as elements to the URL.

```
ldapur1 -a cn,sn -b dc=example,dc=com -s sub -f "
(objectclass=inetorgperson)"

ldap://:389/dc=example,dc=com?cn,sn?sub?(objectclass=inetorgperson)
```

## APPENDIX B. LDAP DATA INTERCHANGE FORMAT

Red Hat Directory Server (Directory Server) uses the LDAP Data Interchange Format (LDIF) to describe a directory and directory entries in text format. LDIF is commonly used to build the initial directory database or to add large numbers of entries to the directory all at once. In addition, LDIF is also used to describe changes to directory entries. For this reason, most of Directory Server's command-line utilities rely on LDIF for either input or output.

Because LDIF is a text file format, LDIF files can be created using virtually any language. All directory data is stored using the UTF-8 encoding of Unicode. Therefore, the LDIF files created must also be UTF-8 encoded.

For information on using LDIF to modify directory entries, see [Chapter 3, Creating Directory Entries](#).

### B.1. ABOUT THE LDIF FILE FORMAT

LDIF consists of one or more directory entries separated by a blank line. Each LDIF entry consists of an optional entry ID, a required distinguished name, one or more object classes, and multiple attribute definitions.

The LDIF format is defined in RFC 2849, *The LDAP Data Interchange Format (LDIF)*. Directory Server is compliant with this standard.

The basic form of a directory entry represented in LDIF is as follows:

```
dn: distinguished_name
objectClass: object_class
objectClass: object_class
...
attribute_type[;subtype]:attribute_value
...
```

- Every LDIF entry must have a DN and at least one object class definition.
- Include any attributes required by the object classes defined for the entry.
- All other attributes and object classes are optional.
- Object classes and attributes can be specified in any order.
- The space after the colon is optional.

For information on standard object classes and attributes, see the [Red Hat Directory Server 9 Configuration, Command, and File Reference](#).

[Table B.1, “LDIF Fields”](#) describes the LDIF fields shown in the previous definition.

**Table B.1. LDIF Fields**

| Field         | Definition                                                                                                                                                               |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| [ <i>id</i> ] | <i>Optional.</i> A positive decimal number representing the entry ID. The database creation tools generate this ID automatically. Never add or edit this value yourself. |

| Field                            | Definition                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dn: <i>distinguished_name</i>    | Specifies the distinguished name for the entry.                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| objectClass: <i>object_class</i> | Specifies an object class to use with this entry. The object class identifies the types of attributes, or schema, allowed and required for the entry. See the <a href="#">Red Hat Directory Server 9 Configuration, Command, and File Reference</a> for a list of standard object classes and <a href="#">Chapter 8, Managing the Directory Schema</a> for information on customizing the schema.                                                                                       |
| <i>attribute_type</i>            | Specifies a descriptive attribute to use with the entry. The attribute should be defined either in the schema. See the <a href="#">Red Hat Directory Server 9 Configuration, Command, and File Reference</a> for a list of standard attributes and <a href="#">Chapter 8, Managing the Directory Schema</a> for information on customizing the schema.                                                                                                                                  |
| [ <i>subtype</i> ]               | <i>Optional.</i> Specifies subtype, language, binary, or pronunciation. Use this tag to identify the language in which the corresponding attribute value is expressed or whether the attribute value is binary or a pronunciation of an attribute value. For information on attribute subtypes, see <a href="#">Section 3.1.3.5, “Adding an Attribute Subtype”</a> . For a complete list of the supported subtypes tags, see <a href="#">Table D.2, “Supported Language Subtypes”</a> . |
| <i>attribute_value</i>           | Specifies the attribute value to be used with the attribute type.                                                                                                                                                                                                                                                                                                                                                                                                                       |

**NOTE**

The LDIF syntax for representing a change to an entry in the directory is different from the syntax described in [Table B.1, “LDIF Fields”](#). For information on using LDIF to modify directory entries, see [Chapter 3, Creating Directory Entries](#).

## B.2. CONTINUING LINES IN LDIF

In LDIF files, a line can be broken and continued (called *folded*) by indenting the continued portion of the line by exactly one space. For example, the following two statements are identical:

```
dn: cn=Jake Lupinski,dc=example,dc=com

dn: cn=Jake Lup
 inski,dc=exa
 mple,dc=com
```

It is not required to break and continue LDIF lines. However, doing so may improve the readability of the LDIF file. The usual convention is that an LDIF file does not contain more than 78 columns of text.



## B.3. REPRESENTING BINARY DATA

Binary data, such as a JPEG image, is represented in LDIF using one of two methods, standard LDIF notation or base-64 encoding.

### B.3.1. Standard LDIF Notation

Standard LDIF notation uses the lesser than (<) symbol to indicate that the data are binary. For example:

```
jpegphoto: < file:/path/to/photo
```

With this standard notation, it is not necessary to specify the **ldapmodify -b** parameter. However, standard notation requires that the following line be added to the beginning of the LDIF file or the LDIF update statements:

```
version: 1
```

For example:

```
ldapmodify -x -D userDN -W
version: 1
dn: cn=Barney Fife,ou=People,dc=example,dc=com
changetype: modify
add: usercertificate
usercertificate;binary: < file: BarneysCert
```

### B.3.2. Base-64 Encoding

Binary data can be converted to base-64, which can be used in LDIF files, for a variety of data, from images to SSL certificates. Base 64-encoded data are identified by using the `::` symbol. For example:

```
jpegPhoto::encoded_data
```

In addition to binary data, other values that must be base-64 encoded include the following:

- Any value that begins with a colon (:) or a space.
- Any value that contains non-ASCII data, including new lines.

Use the **ldif** command-line utility with the **-b** parameter to convert binary data to LDIF format:

```
ldif -b attribute_name
```

*attribute\_name* is the name of the attribute to which the binary data is supplied. The binary data is read from standard input and the results are written to standard output. Thus, use redirection operators to select input and output files.

The **ldif** command-line utility will take any input and format it with the correct line continuation and appropriate attribute information. The **ldif** utility also assesses whether the input requires base-64 encoding. For example:

```
ldif -b jpegPhoto < mark.jpg > out.ldif
```

This example takes a binary file containing a JPEG-formatted image and converts it into LDIF format for the attribute *jpegPhoto*. The output is saved to **out.ldif**.

The **-b** option specifies that the **ldif** utility should interpret the entire input as a single binary value. If **-b** is not present, each line is considered to be a separate input value.

## B.4. SPECIFYING DIRECTORY ENTRIES USING LDIF

Many types of entries can be stored in the directory. This section concentrates on three of the most common types of entries used in a directory: domain, organizational unit, and organizational person entries.

The object classes defined for an entry are what indicate whether the entry represents a domain or domain component, an organizational unit, an organizational person, or some other type of entry. For a complete list of the object classes that can be used by default in the directory and a list of the most commonly used attributes, see the [Red Hat Directory Server 9 Configuration, Command, and File Reference](#).

### B.4.1. Specifying Domain Entries

Directories often have at least one domain entry. Typically this is the first, or topmost, entry in the directory. The domain entry often corresponds to the DNS host and domain name for your directory. For example, if the Directory Server host is called **ldap.example.com**, then the domain entry for the directory is probably named **dc=ldap,dc=example,dc=com** or simply **dc=example,dc=com**.

The LDIF entry used to define a domain appears as follows:

```
dn: distinguished_name
objectClass: top
objectClass: domain
dc: domain_component_name
 list_of_optional_attributes
...
```

The following is a sample domain entry in LDIF format:

```
dn: dc=example,dc=com
objectclass: top
objectclass: domain
dc: example
description: Fictional example company
```

Each element of the LDIF-formatted domain entry is defined in [Table B.2, “LDIF Elements in Domain Entries”](#).

**Table B.2. LDIF Elements in Domain Entries**

| LDIF Element                  | Description                                                      |
|-------------------------------|------------------------------------------------------------------|
| dn: <i>distinguished_name</i> | <i>Required.</i> Specifies the distinguished name for the entry. |

| LDIF Element                      | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>objectClass: top</code>     | <i>Required.</i> Specifies the <b>top</b> object class.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <code>objectClass: domain</code>  | Specifies the <b>domain</b> object class. This line defines the entry as a domain or domain component. See the <a href="#">Red Hat Directory Server 9 Configuration, Command, and File Reference</a> for a list of the attributes that can be used with this object class.                                                                                                                                                                                                                                                              |
| <code>dc: domain_component</code> | Attribute that specifies the domain's name. The server is typically configured during the initial setup to have a suffix or naming context in the form <b>dc=hostname,dc=domain,dc=toplevel</b> . For example, <b>dc=ldap,dc=example,dc=com</b> . The domain entry should use the leftmost <b>dc</b> value, such as <b>dc: ldap</b> . If the suffix were <b>dc=example,dc=com</b> , the <b>dc</b> value is <b>dc: example</b> . Do not create the entry for <b>dn: dc=com</b> unless the server has been configured to use that suffix. |
| <code>list_of_attributes</code>   | Specifies the list of optional attributes to maintain for the entry. See the <a href="#">Red Hat Directory Server 9 Configuration, Command, and File Reference</a> for a list of the attributes that can be used with this object class.                                                                                                                                                                                                                                                                                                |

### B.4.2. Specifying Organizational Unit Entries

Organizational unit entries are often used to represent major branch points, or subdirectories, in the directory tree. They correspond to major, reasonably static entities within the enterprise, such as a subtree that contains people or a subtree that contains groups.

The organizational unit attribute that is contained in the entry may also represent a major organization within the company, such as marketing or engineering. However, this style is discouraged. Red Hat strongly encourages using a flat directory tree.

There is usually more than one organizational unit, or branch point, within a directory tree.

The LDIF that defines an organizational unit entry must appear as follows:

```
dn: distinguished_name
objectClass: top
objectClass: organizationalUnit
ou: organizational_unit_name
list_of_optional_attributes
...
```

The following is a sample organizational unit entry in LDIF format:

```
dn: ou=people,dc=example,dc=com
objectclass: top
```

```

objectclass: organizationalUnit
ou: people
description: Fictional example organizational unit

```

Table B.3, “LDIF Elements in Organizational Unit Entries” defines each element of the LDIF-formatted organizational unit entry.

**Table B.3. LDIF Elements in Organizational Unit Entries**

| LDIF Element                                     | Description                                                                                                                                                                                                                                                                      |
|--------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>dn: <i>distinguished_name</i></code>       | Specifies the distinguished name for the entry. A DN is required. If there is a comma in the DN, the comma must be escaped with a backslash (\), such as <b>dn: ou=people,dc=example,dc=com</b> .                                                                                |
| <code>objectClass: top</code>                    | <i>Required.</i> Specifies the <b>top</b> object class.                                                                                                                                                                                                                          |
| <code>objectClass: organizationalUnit</code>     | Specifies the <b>organizationalUnit</b> object class. This line defines the entry as an <b>organizational unit</b> . See the <a href="#">Red Hat Directory Server 9 Configuration, Command, and File Reference</a> for a list of the attributes available for this object class. |
| <code>ou: <i>organizational_unit_name</i></code> | Attribute that specifies the organizational unit's name.                                                                                                                                                                                                                         |
| <code><i>list_of_attributes</i></code>           | Specifies the list of optional attributes to maintain for the entry. See the <a href="#">Red Hat Directory Server 9 Configuration, Command, and File Reference</a> for a list of the attributes available for this object class.                                                 |

### B.4.3. Specifying Organizational Person Entries

The majority of the entries in the directory represent organizational people.

In LDIF, the definition of an organizational person is as follows:

```

dn: distinguished_name
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: common_name
sn: surname
list_of_optional_attributes

```

The following is an example organizational person entry in LDIF format:

```

dn: uid=bjensen,ou=people,dc=example,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson

```

```

cn: Babs Jensen
sn: Jensen
givenname: Babs
uid: bjensen
ou: people
description: Fictional example person
telephoneNumber: 555-5557
userPassword: {SSHA}dkfljljk34r2kljdsfk9

```

Table B.4, “LDIF Elements in Person Entries” defines each aspect of the LDIF person entry.

**Table B.4. LDIF Elements in Person Entries**

| LDIF Element                      | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dn: <i>distinguished_name</i>     | <i>Required.</i> Specifies the distinguished name for the entry. For example, <b>dn: uid=bjensen,ou=people,dc=example,dc=com</b> . If there is a comma in the DN, the comma must be escaped with a backslash (\).                                                                                                                                                                                                                                                                                                                                       |
| objectClass: top                  | <i>Required.</i> Specifies the <b>top</b> object class.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| objectClass: person               | Specifies the <b>person</b> object class. This object class specification should be included because many LDAP clients require it during search operations for a person or an organizational person.                                                                                                                                                                                                                                                                                                                                                    |
| objectClass: organizationalPerson | Specifies the <b>organizationalPerson</b> object class. This object class specification should be included because some LDAP clients require it during search operations for an organizational person.                                                                                                                                                                                                                                                                                                                                                  |
| objectClass: inetOrgPerson        | Specifies the <b>inetOrgPerson</b> object class. The <b>inetOrgPerson</b> object class is recommended for the creation of an organizational person entry because this object class includes the widest range of attributes. The <b>uid</b> attribute is required by this object class, and entries that contain this object class are named based on the value of the <b>uid</b> attribute. See the <a href="#">Red Hat Directory Server 9 Configuration, Command, and File Reference</a> for a list of the attributes available for this object class. |
| cn: <i>common_name</i>            | Specifies the person's common name, which is the full name commonly used by the person. For example, <b>cn: Bill Anderson</b> . At least one common name is required.                                                                                                                                                                                                                                                                                                                                                                                   |
| sn: <i>surname</i>                | Specifies the person's surname, or last name. For example, <b>sn: Anderson</b> . A surname is required.                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

| LDIF Element              | Description                                                                                                                                                                                                                      |
|---------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>list_of_attributes</i> | Specifies the list of optional attributes to maintain for the entry. See the <a href="#">Red Hat Directory Server 9 Configuration, Command, and File Reference</a> for a list of the attributes available for this object class. |

## B.5. DEFINING DIRECTORIES USING LDIF

The contents of an entire directory can be defined using LDIF. Using LDIF is an efficient method of directory creation when there are many entries to add to the directory.

To create a directory using LDIF:

1. Create an ASCII file containing the entries to add in LDIF format.

Make sure each entry is separated from the next by an empty line. Use just one line between entries, and make sure the first line of the file is not blank, or else the **ldapmodify** utility will exit. For more information, see [Section B.4, "Specifying Directory Entries Using LDIF"](#).

2. Begin each file with the topmost, or root, entry in the database.

The root entry must represent the suffix or sub-suffix contained by the database. For example, if the database has the suffix **dc=example,dc=com**, the first entry in the directory must be **dn:dc=example,dc=com**.

For information on suffixes, see the "Suffix" parameter described in the *Directory Server Configuration and Command-Line Tool Reference*.

3. Make sure that an entry representing a branch point in the LDIF file is placed before the entries to create under that branch.

For example, to place an entry in a people and a group subtree, create the branch point for those subtrees before creating entries within those subtrees.



### NOTE

The LDIF file is read in order, so parent entries must be listed before the child entries.

4. Create the directory from the LDIF file using one of the following methods:
  - *Initializing the database through the Directory Server Console.* Use this method if there is a small database to import (less than 10,000 entries). See [Section 4.1.4, "Importing a Database from the Console"](#).

**WARNING**

This method is destructive and will erase any existing data in the suffix.

- *ldif2db* or *ldif2db.pl* command-line utility. Use this method if there is a large database to import (more than 10,000 entries). See [Section 4.1.6.1, “Importing Using the ldif2db Command-Line Script”](#).
  - **ldif2db** cannot be used if the server is running.
  - **ldif2db.pl** can only be used if the server is running.

**WARNING**

This method is destructive and will erase any existing data in the suffix.

- *ldapmodify* command-line utility with the *-a* parameter. Use this method if a new subtree is being added to an existing database or there is existing data in the suffix which should not be deleted. Unlike the other methods for creating the directory from an LDIF file, Directory Server must be running before a subtree can be added using **ldapmodify**. See [Section 3.2.4, “Adding and Modifying Entries Using ldapmodify”](#).

**Example B.1. LDIF File Example**

This LDIF file contains one domain, two organizational units, and three organizational person entries:

```
dn: dc=example,dc=com
objectclass: top
objectclass: domain
dc: example
description: Fictional example domain

dn: ou=People,dc=example,dc=com
objectclass: top
objectclass: organizationalUnit
ou: People
description: Fictional example organizational unit
tel: 555-5559

dn: cn=June Rossi,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: June Rossi
```

```
sn: Rossi
givenName: June
mail: rossi@example.com
userPassword: {sha}KDIE3AL9DK
ou: Accounting
ou: people
telephoneNumber: 2616
roomNumber: 220

dn: cn=Marc Chambers,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Marc Chambers
sn: Chambers
givenname: Marc
mail: chambers@example.com
userPassword: {sha}jdl2alem87dlacz1
telephoneNumber: 2652
ou: Manufacturing
ou: People
roomNumber: 167

dn: cn=Robert Wong,ou=People,example.com Corp,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Robert Wong
cn: Bob Wong
sn: Wong
givenname: Robert
givenname: Bob
mail: bwong@example.com
userPassword: {sha}nn2msx761
telephoneNumber: 2881
roomNumber: 211
ou: Manufacturing
ou: people

dn: ou=Groups,dc=example,dc=com
objectclass: top
objectclass: organizationalUnit
ou: groups
description: Fictional example organizational unit
```

## B.6. STORING INFORMATION IN MULTIPLE LANGUAGES

If the directory contains a single language, it is not necessary to do anything special to add a new entry to the directory. However, if an organization is multinational, it may be necessary to store information in multiple languages so that users in different locales can view directory information in their own language.



When information in the directory is represented in multiple languages, the server associates language tags with attribute values. When a new entry is added, the attribute values used in the RDN (relative distinguished name, the naming attribute) must be provided without any language codes.

Multiple languages can be stored for a single attribute. In this case, the attribute types are the same, but each value has a different language code.

For a list of the languages supported by Directory Server and their associated language tags, see [Section D.2, “Supported Locales”](#).



## NOTE

The language tag has no effect on how the string is stored within the directory. All object class and attribute strings are stored using UTF-8. The user is responsible for converting the data used in the LDIF to UTF-8. The **iconv** or **uconv** command provided by most operating systems can be used to convert data from the native charset into UTF-8.

For example, Example Corporation has offices in the United States and France and wants employees to be able to view directory information in their native language. When adding directory entries, the directory administrator chooses to provide attribute values in both English and French. When adding a directory entry for a new employee, Babs Jensen, the administrator does the following:

1. The administrator creates a file, **street.txt**, with the French street address value:

```
1 rue de l'Université
```

2. The file contents are then converted to UTF-8:

```
iconv -t UTF-8 -o output.txt street.txt
```

3. The following LDIF entry is created using the UTF-8 value of the street address value for **streetAddress;lang-fr**.

```
dn: uid=bjensen,ou=people,dc=example,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
name: Babs Jensen
cn: Babs Jensen
sn: Jensen
uid: bjensen
streetAddress: 1 University Street
streetAddress;lang-en: 1 University Street
streetAddress;lang-fr:: Aasljd0aAJASI023909jaASJaonasd0ADS
preferredLanguage: fr
```

The double colons after the attribute name and subtype indicate that the value is binary base-64 encoded.

Users accessing this directory entry with an LDAP client with the preferred language set to English will see the address **1 University Street**. Users accessing the directory with an LDAP client with the preferred language set to French will see the address **1 rue de l'Université**.

## APPENDIX C. LDAP URLS

LDAP URLs identify the Red Hat Directory Server instance, similarly to the way site URLs identify a specific website or web page. There are three common times when the LDAP URL of the Directory Server instance is used:

- The LDAP URL is used to identify the specific Directory Server instance when the Directory Server is accessed using a web-based client.
- LDAP URLs are used to configure Directory Server referrals.
- LDAP URLs are used to configure access control instructions.



### NOTE

The LDAP URL format is described in RFC 4516, which is available at <http://www.ietf.org/rfc/rfc4516.txt>.

### C.1. COMPONENTS OF AN LDAP URL

LDAP URLs have the following syntax:

```
ldap[s]://hostname:port/base_dn?attributes?scope?filter
```

It is also possible to use IPv4 or IPv6 addresses instead of the host name.

The **ldap://** protocol is used to connect to LDAP servers over unsecured connections, and the **ldaps://** protocol is used to connect to LDAP servers over TLS/SSL connections. [Table C.1, “LDAP URL Components”](#) lists the components of an LDAP URL.



### NOTE

The LDAP URL format is described in RFC 4516, which is available at <http://www.ietf.org/rfc/rfc4516.txt>.

**Table C.1. LDAP URL Components**

| Component | Description                                                                                                                                                                                                   |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| host name | Name (or IPv4 or IPv6 address) of the LDAP server. For example, <b>ldap.example.com</b> or <b>192.202.185.90</b> .                                                                                            |
| port      | Port number of the LDAP server (for example, <b>696</b> ). If no port is specified, the standard LDAP port ( <b>389</b> ) or LDAPS port ( <b>636</b> ) is used.                                               |
| base_dn   | Distinguished name (DN) of an entry in the directory. This DN identifies the entry that is the starting point of the search. If no base DN is specified, the search starts at the root of the directory tree. |

| Component  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| attributes | The attributes to be returned. To specify more than one attribute, use commas to separate the attributes; for example, <b>cn,mail,telephoneNumber</b> . If no attributes are specified in the URL, all attributes are returned.                                                                                                                                                                                                                                                                                                                                                                                                                 |
| scope      | <p>The scope of the search, which can be one of these values:</p> <div> <p><b>base</b> retrieves information only about the distinguished name (<i>base_dn</i>) specified in the URL.</p> <p><b>one</b> retrieves information about entries one level below the distinguished name (<i>base_dn</i>) specified in the URL. The base entry is not included in this scope.</p> <p><b>sub</b> retrieves information about entries at all levels below the distinguished name (<i>base_dn</i>) specified in the URL. The base entry is included in this scope.</p> </div> <p>If no scope is specified, the server performs a <b>base</b> search.</p> |
| filter     | Search filter to apply to entries within the specified scope of the search. If no filter is specified, the server uses the filter ( <b>objectClass=*</b> ).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

The attributes, scope, and filter components are identified by their positions in the URL. Even if no attributes are specified, the question marks still must be included to delimit that field.

For example, to specify a subtree search starting from **dc=example,dc=com** that returns all attributes for entries matching (**sn=Jensen**), use the following LDAP URL:

```
ldap://ldap.example.com/dc=example,dc=com??sub?(sn=Jensen)
```

The two consecutive question marks, **??**, indicate that no attributes have been specified. Since no specific attributes are identified in the URL, all attributes are returned in the search.

## C.2. ESCAPING UNSAFE CHARACTERS

Any *unsafe* characters in the URL need to be escaped, or substituted with a special sequence of characters.

For example, a space is an unsafe character that must be represented as **%20** within the URL. Thus, the distinguished name **o=example.com corporation** must be encoded as **o=example.com%20corporation**.

The following table lists the characters that are considered unsafe within URLs and provides the associated escape characters to use in place of the unsafe character:

| Unsafe Character | Escape Characters |
|------------------|-------------------|
| space            | %20               |
| <                | %3c               |
| >                | %3e               |
| "                | %22               |
| #                | %23               |
| %                | %25               |
| {                | %7b               |
| }                | %7d               |
|                  | %7c               |
| \                | %5c               |
| ^                | %5e               |
| ~                | %7e               |
| [                | %5b               |
| ]                | %5d               |
| `                | %60               |

## C.3. EXAMPLES OF LDAP URLS



### NOTE

The LDAP URL format is described in RFC 4516, which is available at <http://www.ietf.org/rfc/rfc4516.txt>.

### Example 1

The following LDAP URL specifies a base search for the entry with the distinguished name **dc=example, dc=com**.

```
ldap://ldap.example.com/dc=example,dc=com
```

- Because no port number is specified, the standard LDAP port number (**389**) is used.

- Because no attributes are specified, the search returns all attributes.
- Because no search scope is specified, the search is restricted to the base entry **dc=example,dc=com**.
- Because no filter is specified, the directory uses the default filter (**objectclass=\***).

### Example 2

The following LDAP URL retrieves the **postalAddress** attribute of the entry with the DN **dc=example,dc=com**:

```
ldap://ldap.example.com/dc=example,dc=com?postalAddress
```

- Because no search scope is specified, the search is restricted to the base entry **dc=example,dc=com**.
- Because no filter is specified, the directory uses the default filter (**objectclass=\***).

### Example 3

The following LDAP URL retrieves the **cn**, **mail**, and **telephoneNumber** attributes of the entry for Barbara Jensen:

```
ldap://ldap.example.com/cn=Barbara%20Jensen,dc=example,dc=com?cn,mail,telephoneNumber
```

- Because no search scope is specified, the search is restricted to the base entry **cn=Barbara Jensen,dc=example,dc=com**.
- Because no filter is specified, the directory uses the default filter (**objectclass=\***).

### Example 4

The following LDAP URL specifies a search for entries that have the surname **Jensen** and are at any level under **dc=example,dc=com**:

```
ldap://ldap.example.com/dc=example,dc=com??sub?(sn=Jensen)
```

- Because no attributes are specified, the search returns all attributes.
- Because the search scope is **sub**, the search encompasses the base entry **dc=example,dc=com** and entries at all levels under the base entry.

### Example 5

The following LDAP URL specifies a search for the object class for all entries one level under **dc=example,dc=com**:

```
ldap://ldap.example.com/dc=example,dc=com?objectClass?one
```

- Because the search scope is **one**, the search encompasses all entries one level under the base entry **dc=example,dc=com**. The search scope does not include the base entry.
- Because no filter is specified, the directory uses the default filter (**objectclass=\***).

**NOTE**

The syntax for LDAP URLs does not include any means for specifying credentials or passwords. Search requests initiated through LDAP URLs are unauthenticated, unless the LDAP client that supports LDAP URLs provides an authentication mechanism.

## APPENDIX D. INTERNATIONALIZATION

Red Hat Directory Server allows users to store, manage, and search for entries and their associated attributes in a number of different languages. An internationalized directory can be an invaluable corporate resource, providing employees and business partners with immediate access to the information they need in languages they understand.

Directory Server supports all international character sets by default because directory data is stored in UTF-8. Further, Directory Server can use specified matching rules and collation orders based on language preferences in search operations.



### NOTE

ASCII characters are required for attribute and object class names.

### D.1. ABOUT LOCALES

Directory Server provides support for multiple languages through the use of *locales*. A locale identifies language-specific information about how users of a specific region, culture, or custom expect data to be presented, including how data of a given language is interpreted and how data is to be sorted, or *collated*.

In addition, the locale information indicates what code page should be used to represent a given language. A code page is an internal table that the operating system uses to relate keyboard keys to character font screen displays.

More specifically, a locale defines four things:

- *Collation order*. The collation order provides language and cultural-specific information about how the characters of a given language are to be sorted. It identifies things like the sequence of the letters in the alphabet, how to compare letters with accents to letters without accents, and if there are any characters that can be ignored when comparing strings. The collation order also takes into account culture-specific information about a language, such as the direction in which the language is read (left to right, right to left, or up and down).
- *Character type*. The character type distinguishes alphabetic characters from numeric or other characters. For example, in some languages, the pipe (|) character is considered punctuation while in others it is considered alphabetic. In addition, it defines the mapping of upper-case to lower-case letters.
- *Monetary format*. The monetary format specifies the monetary symbol used by a specific region, whether the symbol goes before or after its value, and how monetary units are represented.
- *Time/date format*. The time and date format indicates the customary formatting for times and dates in the region. The time and date format indicates whether dates are customarily represented in the *mm/dd/yy* (month, day, year) or *dd/mm/yy* (day, month, year) format and specifies what the days of the week and month are in a given language. For example, the date January 10, 1996, is represented as **10. leden 1996** in Czechoslovakian and **10 janvier 1996** in French.

Because a locale describes cultural, customary, and regional differences in addition to mechanical language differences, the directory data can both be translated into the specific languages understood by users as well as be presented in a way that users in a given region expect.

## D.2. SUPPORTED LOCALES

When performing directory operations that require that a locale be specified, such as a search operation, use a language tag or a collation order object identifier (OID).

A *language tag* is a string that begins with the two-character lowercase language code that identifies the language, as defined in ISO Standard 639. If necessary to distinguish regional differences in language, the language tag may also contain a two-character string for the country code, as defined in ISO Standard 3166. The language code and country code are separated by a hyphen. For example, the language tag used to identify the British English locale is **en-GB**.

An *object identifier* (OID) is a decimal number used to uniquely identify an object, such as an attribute or object class. The OIDs for searching or indexing an internationalized directory identify specific collation orders supported by the Directory Server. For example, the OID **2.16.840.1.113730.3.3.2.17.1** identifies the Finnish collation order.

When performing an international search in the directory, use either the language tag or the OID to identify the collation order to use. However, when setting up an international index, the OIDs must be used. For more information on indexing, see [Chapter 9, Managing Indexes](#).

[Table D.1, “Supported Locales”](#) lists each locale supported by Directory Server and identifies the associated language tags and OIDs.

**Table D.1. Supported Locales**

| Locale                | Language Tag | Collation Order Object Identifiers (OIDs) |
|-----------------------|--------------|-------------------------------------------|
| Albanian              | sq           | 2.16.840.1.113730.3.3.2.44.1              |
| Arabic                | ar           | 2.16.840.1.113730.3.3.2.1.1               |
| Belorussian           | be           | 2.16.840.1.113730.3.3.2.2.1               |
| Bulgarian             | bg           | 2.16.840.1.113730.3.3.2.3.1               |
| Catalan               | ca           | 2.16.840.1.113730.3.3.2.4.1               |
| Chinese (Simplified)  | zh           | 2.16.840.1.113730.3.3.2.49.1              |
| Chinese (Traditional) | zh-TW        | 2.16.840.1.113730.3.3.2.50.1              |
| Croatian              | hr           | 2.16.840.1.113730.3.3.2.22.1              |
| Czechoslovakian       | cs           | 2.16.840.1.113730.3.3.2.5.1               |
| Danish                | da           | 2.16.840.1.113730.3.3.2.6.1               |
| Dutch                 | nl or nl-NL  | 2.16.840.1.113730.3.3.2.33.1              |
| Dutch (Belgian)       | nl-BE        | 2.16.840.1.113730.3.3.2.34.1              |



| Locale            | Language Tag | Collation Order Object Identifiers (OIDs) |
|-------------------|--------------|-------------------------------------------|
| English (US)      | en or en-US  | 2.16.840.1.113730.3.3.2.11.1              |
| Estonian          | et           | 2.16.840.1.113730.3.3.2.16.1              |
| Finnish           | fi           | 2.16.840.1.113730.3.3.2.17.1              |
| French            | fr or fr-FR  | 2.16.840.1.113730.3.3.2.18.1              |
| French (Belgian)  | fr-BE        | 2.16.840.1.113730.3.3.2.19.1              |
| French (Canadian) | fr-CA        | 2.16.840.1.113730.3.3.2.20.1              |
| French (Swiss)    | fr-CH        | 2.16.840.1.113730.3.3.2.21.1              |
| German            | de           | 2.16.840.1.113730.3.3.2.7.1               |
| German (Austrian) | de-AT        | 2.16.840.1.113730.3.3.2.8.1               |
| German (Swiss)    | de-CH        | 2.16.840.1.113730.3.3.2.9.1               |
| Greek             | el           | 2.16.840.1.113730.3.3.2.10.1              |
| Hebrew            | iw           | 2.16.840.1.113730.3.3.2.27.1              |
| Hungarian         | hu           | 2.16.840.1.113730.3.3.2.23.1              |
| Icelandic         | is           | 2.16.840.1.113730.3.3.2.24.1              |
| Italian           | it           | 2.16.840.1.113730.3.3.2.25.1              |
| Italian (Swiss)   | it-CH        | 2.16.840.1.113730.3.3.2.26.1              |
| Japanese          | ja           | 2.16.840.1.113730.3.3.2.28.1              |
| Korean            | ko           | 2.16.840.1.113730.3.3.2.29.1              |
| Latvian, Lettish  | lv           | 2.16.840.1.113730.3.3.2.31.1              |
| Lithuanian        | lt           | 2.16.840.1.113730.3.3.2.30.1              |
| Macedonian        | mk           | 2.16.840.1.113730.3.3.2.32.1              |
| Norwegian         | no           | 2.16.840.1.113730.3.3.2.35.1              |

| Locale              | Language Tag | Collation Order Object Identifiers (OIDs) |
|---------------------|--------------|-------------------------------------------|
| Norwegian (Bokmul)  | nb           | 2.16.840.1.113730.3.3.2.36.1              |
| Norwegian (Nynorsk) | no-NO-NY     | 2.16.840.1.113730.3.3.2.37.1              |
| Polish              | pl           | 2.16.840.1.113730.3.3.2.38.1              |
| Romanian            | ro           | 2.16.840.1.113730.3.3.2.39.1              |
| Russian             | ru           | 2.16.840.1.113730.3.3.2.40.1              |
| Serbian (Cyrillic)  | sr           | 2.16.840.1.113730.3.3.2.45.1              |
| Serbian (Latin)     | sh           | 2.16.840.1.113730.3.3.2.41.1              |
| Slovakian           | sk           | 2.16.840.1.113730.3.3.2.42.1              |
| Slovenian           | sl           | 2.16.840.1.113730.3.3.2.43.1              |
| Spanish             | es or es-ES  | 2.16.840.1.113730.3.3.2.15.1              |
| Swedish             | sv           | 2.16.840.1.113730.3.3.2.46.1              |
| Turkish             | tr           | 2.16.840.1.113730.3.3.2.47.1              |
| Ukrainian           | uk           | 2.16.840.1.113730.3.3.2.48.1              |

## D.3. SUPPORTED LANGUAGE SUBTYPES

Language subtypes can be used by clients to determine specific values for which to search. For more information on using language subtypes, see [Section 3.1.3.5, “Adding an Attribute Subtype”](#). [Table D.2, “Supported Language Subtypes”](#) lists the supported language subtypes for Directory Server.

**Table D.2. Supported Language Subtypes**

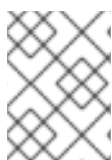
| Language Tag | Language    |
|--------------|-------------|
| af           | Afrikaans   |
| be           | Belorussian |
| bg           | Bulgarian   |
| ca           | Catalan     |

| Language Tag | Language        |
|--------------|-----------------|
| cs           | Czechoslovakian |
| da           | Danish          |
| de           | German          |
| el           | Greek           |
| en           | English         |
| es           | Spanish         |
| eu           | Basque          |
| fi           | Finnish         |
| fo           | Faroese         |
| fr           | French          |
| ga           | Irish           |
| gl           | Galician        |
| hr           | Croatian        |
| hu           | Hungarian       |
| id           | Indonesian      |
| is           | Icelandic       |
| it           | Italian         |
| ja           | Japanese        |
| ko           | Korean          |
| nl           | Dutch           |
| no           | Norwegian       |
| pl           | Polish          |
| pt           | Portuguese      |

| Language Tag | Language  |
|--------------|-----------|
| ro           | Romanian  |
| ru           | Russian   |
| sk           | Slovakian |
| sl           | Slovenian |
| sq           | Albanian  |
| sr           | Serbian   |
| sv           | Swedish   |
| tr           | Turkish   |
| uk           | Ukrainian |
| zh           | Chinese   |

## D.4. SEARCHING AN INTERNATIONALIZED DIRECTORY

When performing search operations, the Directory Server can sort the results based on any language for which the server has a supporting collation order. For a listing of the collation orders supported by the directory, see [Section D.2, “Supported Locales”](#).



### NOTE

An LDAPv3 search is required to perform internationalized searches. Therefore, do not set the LDAPv2 option on the call for **ldapsearch**.

This section focuses using matching rule filters to return international attribute values. For more information on general **ldapsearch** syntax, see [Section 10.4, “LDAP Search Filters”](#). For information on searching internationalized directories using the **Users and Groups** portion of the Red Hat Console, see the online help.

- [Section D.4.1, “Matching Rule Formats”](#)
- [Section D.4.2, “Supported Search Types”](#)
- [Section D.4.3, “International Search Examples”](#)

### D.4.1. Matching Rule Formats

The matching rule filters for internationalized searches can be represented in any several ways, and which one should be used is a matter of preference:

- As the OID of the collation order for the locale on which to base the search.
- As the language tag associated with the collation order on which to base the search.
- As the OID of the collation order and a suffix that represents a relational operator.
- As the language tag associated with the collation order and a suffix that represents a relational operator.

The syntax for each of these options is discussed in the following sections:

- [Section D.4.1.1, “Using an OID for the Matching Rule”](#)
- [Section D.4.1.2, “Using a Language Tag for the Matching Rule”](#)
- [Section D.4.1.3, “Using an OID and Suffix for the Matching Rule”](#)
- [Section D.4.1.4, “Using a Language Tag and Suffix for the Matching Rule”](#)

#### D.4.1.1. Using an OID for the Matching Rule

Each locale supported by the Directory Server has an associated collation order OID. For a list of locales supported by the directory server and their associated OIDs, see [Table D.1, “Supported Locales”](#).

The collation order OID can be used in the matching rule portion of the matching rule filter as follows:

```
attr:OID:=(relational_operator value)
```

The relational operator is included in the value portion of the string, separated from the value by a single space. For example, to search for all **departmentNumber** attributes that are at or after **N4709** in the Swedish collation order, use the following filter:

```
departmentNumber:2.16.840.1.113730.3.3.2.46.1:=>= N4709
```

#### D.4.1.2. Using a Language Tag for the Matching Rule

Each locale supported by the Directory Server has an associated language tag. For a list of locales supported by the directory server and their associated language tags, see [Table D.1, “Supported Locales”](#).

The language tag can be used in the matching rule portion of the matching rule filter as follows:

```
attr:language-tag:=(relational_operator value)
```

The relational operator is included in the value portion of the string, separated from the value by a single space. For example, to search the directory for all description attributes with a value of **estudiante** using the Spanish collation order, use the following filter:

```
cn:es:== estudiante
```

#### D.4.1.3. Using an OID and Suffix for the Matching Rule

As an alternative to using a relational operator-value pair, append a suffix that represents a specific operator to the OID in the matching rule portion of the filter. Combine the OID and suffix as follows:

```
attr: OID+suffix:=value
```

For example, to search for ***businessCategory*** attributes with the value **softwareprodukte** in the German collation order, use the following filter:

```
businessCategory:2.16.840.1.113730.3.3.2.7.1.3:=softwareprodukte
```

The **.3** in the previous example is the equality suffix.

For a list of locales supported by the Directory Server and their associated OIDs, see [Table D.1, “Supported Locales”](#). For a list of relational operators and their equivalent suffixes, see [Table D.3, “Search Types, Operators, and Suffixes”](#).

#### D.4.1.4. Using a Language Tag and Suffix for the Matching Rule

As an alternative to using a relational operator-value pair, append a suffix that represents a specific operator to the language tag in the matching rule portion of the filter. Combine the language tag and suffix as follows:

```
attr: language-tag+suffix:=value
```

For example, to search for all surnames that come at or after **La Salle** in the French collation order, use the following filter:

```
sn:fr.4:=La Salle
```

For a list of locales supported by the Directory Server and their associated language tags, see [Table D.1, “Supported Locales”](#). For a list of relational operators and their equivalent suffixes, see [Table D.3, “Search Types, Operators, and Suffixes”](#).

#### D.4.2. Supported Search Types

The Directory Server supports the following types of international searches:

- equality (=)
- substring (\*)
- greater-than (>)
- greater-than or equal-to (>=)
- less-than (<)
- less-than or equal-to (<=)

Approximate, or phonetic, and presence searches are supported only in English.

As with a regular **ldapsearch** search operation, an international search uses operators to define the type of search. However, when invoking an international search, either use the standard operators (=, >=, >, <, <=) in the value portion of the search string, or use a special type of operator, called a suffix (not to be confused with the directory suffix), in the matching rule portion of the filter. [Table D.3, “Search Types, Operators, and Suffixes”](#) summarizes each type of search, the operator, and the equivalent suffix.

**Table D.3. Search Types, Operators, and Suffixes**

| Search Type              | Operator | Suffix |
|--------------------------|----------|--------|
| Less-than                | <        | .1     |
| Less-than or equal-to    | <=       | .2     |
| Equality                 | =        | .3     |
| Greater-than or equal-to | >=       | .4     |
| Greater-than             | >        | .5     |
| Substring                | *        | .6     |

### D.4.3. International Search Examples

The following sections show examples of how to perform international searches on directory data. Each example gives all the possible matching rule filter formats so that you can become familiar with the formats and select the one that works best.

#### D.4.3.1. Less-Than Example

Performing a locale-specific search using the less-than operator (<), or suffix (.1) searches for all attribute values that come before the given attribute in a specific collation order.

For example, to search for all surnames that come before the surname **Marquez** in the Spanish collation order, any of the following matching rule filters would work:

```
sn:2.16.840.1.113730.3.3.2.15.1:=< Marquez
...
sn:es:=< Marquez
...
sn:2.16.840.1.113730.3.3.2.15.1.1:=Marquez
...
sn:es.1:=Marquez
```

#### D.4.3.2. Less-Than or Equal-to Example

Performing a locale-specific search using the less-than or equal-to operator (<=), or suffix (.2) searches for all attribute values that come at or before the given attribute in a specific collation order.

For example, to search for all room numbers that come at or before room number **CZ422** in the Hungarian collation order, any of the following matching rule filters would work:

```
roomNumber:2.16.840.1.113730.3.3.2.23.1:=<= CZ422
...
roomNumber:hu:=<= CZ422
...
```

```
roomNumber:2.16.840.1.113730.3.3.2.23.1.2:=CZ422
...
roomNumber:hu.2:=CZ422
```

#### D.4.3.3. Equality Example

Performing a locale-specific search using the equal to operator (=), or suffix (.3) searches for all attribute values that match the given attribute in a specific collation order.

For example, to search for all **businessCategory** attributes with the value **softwareprodukte** in the German collation order, any of the following matching rule filters would work:

```
businessCategory:2.16.840.1.113730.3.3.2.7.1:==softwareprodukte
...
businessCategory:de:== softwareprodukte
...
businessCategory:2.16.840.1.113730.3.3.2.7.1.3:=softwareprodukte
...
businessCategory:de.3:=softwareprodukte
```

#### D.4.3.4. Greater-Than or Equal-to Example

Performing a locale-specific search using the greater-than or equal-to operator (>=), or suffix (.4) searches for all attribute values that come at or after the given attribute in a specific collation order.

For example, to search for all localities that come at or after **Québec** in the French collation order, any of the following matching rule filters would work:

```
locality:2.16.840.1.113730.3.3.2.18.1:>= Québec
...
locality:fr:>= Québec
...
locality:2.16.840.1.113730.3.3.2.18.1.4:=Québec
...
locality:fr.4:=Québec
```

#### D.4.3.5. Greater-Than Example

Performing a locale-specific search using the greater-than operator (>), or suffix (.5) searches for all attribute values that come at or before the given attribute in a specific collation order.

For example, to search for all mail hosts that come after host **schranka4** in the Czechoslovakian collation order, any of the following matching rule filters would work:

```
mailHost:2.16.840.1.113730.3.3.2.5.1:=> schranka4
...
mailHost:cs:=> schranka4
...
mailHost:2.16.840.1.113730.3.3.2.5.1.5:=schranka4
...
mailHost:cs.5:=schranka4
```



### D.4.3.6. Substring Example

Performing an international substring search searches for all values that match the given pattern in the specified collation order.

For example, to search for all user IDs that end in **ming** in the Chinese collation order, any of the following matching rule filters would work:

```
uid:2.16.840.1.113730.3.3.2.49.1:=* *ming
...
uid:zh:=* *ming
...
uid:2.16.840.1.113730.3.3.2.49.1.6:=* *ming
..
uid:zh.6:=* *ming
```

Substring search filters that use DN-valued attributes, such as *modifiersName* or *memberOf*, do not always match entries correctly if the filter contains one or more space characters.

To work around this problem, use the entire DN in the filter instead of a substring, or ensure that the DN substring in the filter begins at an RDN boundary; that is, make sure it starts with the *type=* part of the DN. For example, this filter should not be used:

```
(memberOf=*Domain Administrators*)
```

But either one of these will work correctly:

```
(memberOf=cn=Domain Administrators*)
...
(memberOf=cn=Domain Administrators,ou=Groups,dc=example,dc=com)
```

## D.5. TROUBLESHOOTING MATCHING RULES

International collation order matching rules may not behave consistently. Some forms of matching-rule invocation do not work correctly, producing incorrect search results. For example, the following rules do not work:

```
ldapsearch -x -p 389 -D "uid=userID,ou=people,dc=example,dc=com" -w secret
-b "dc=example,dc=com" "sn:2.16.840.1.113730.3.3.2.7.1:=passin"

ldapsearch -x -p 389 -D "uid=userID,ou=people,dc=example,dc=com" -w secret
-b "dc=example,dc=com" "sn:de:=passin"
```

However, the rules listed below will work (note the **.3** before the **passin** value):

```
ldapsearch -x -p 389 -D "uid=userID,ou=people,dc=example,dc=com" -w secret
-b "dc=example,dc=com" "sn:2.16.840.1.113730.3.3.2.7.1.3:=passin"

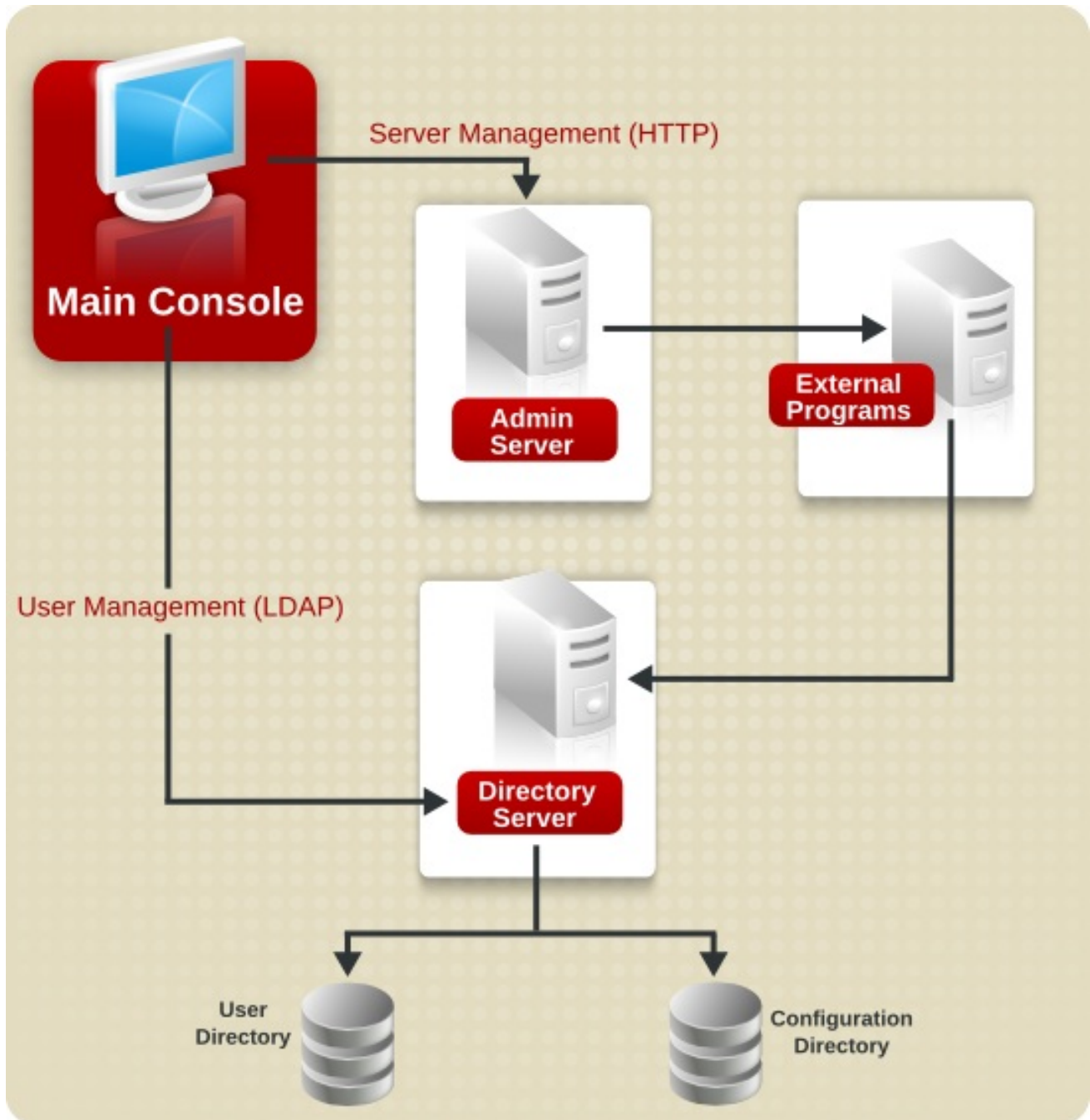
ldapsearch -x -p 389 -D "uid=userID,ou=people,dc=example,dc=com" -w secret
-b "dc=example,dc=com" "sn:de.3:=passin"
```

## APPENDIX E. MANAGING THE ADMIN SERVER

### E.1. INTRODUCTION TO RED HAT ADMIN SERVER

Identity management and directory services with Red Hat Directory Server use three components, working in tandem:

- A Java-based management console
- An administration server which also functions as a web server
- An LDAP directory server



**Figure E.1. Interactions between the Console, Admin Server and Directory Server**

The Admin Server processes configuration requests for Directory Server instances and performs many common server tasks, such as stopping and starting server instances. Directory services are usually

divided into two categories: *configuration* databases which store the Console and Admin Server settings and some Directory Server configuration and *user* databases which contain user and group information. These databases can be kept in the same Directory Server instance, but it is also possible to break these services into separate Directory Server instances. In that case, a Directory Server instance's configuration are stored in a separate Directory Server, called the *Configuration Directory Server*, and user data is stored in the *User Directory Server*. Because the Admin Server processes server configuration requests for Red Hat Directory Server, the Configuration Directory Server and User Directory Server instances are both defined in the Admin Server configuration.

As a web server, the Admin Server provides all of the online functions of the Directory Server, including handling connections to the Console and hosting web applications such as Admin Express. Clients connect to the Admin Server both over secure and standard connections, since the Admin Server supports both HTTP or HTTPS, if SSL/TLS is enabled.

When Red Hat Directory Server or Red Hat Certificate System (which depends on Red Hat Directory Server) is installed, then the Admin Server is automatically installed and configured as well. There can be multiple Directory Server instances and multiple Certificate System subsystems on a single machine, and all use the same instance of Admin Server.

There can be *only one* Admin Server per machine. This single Admin Server instance can handle multiple instances of Directory Server and other clients which can use the Admin Server, like Red Hat Certificate System.

When the Console is opened to manage an instance of Directory Server or Certificate System, even if the Console is on a different machine than the server instance being managed, it contacts the local Admin Server instance to perform the requested tasks. For example, Admin Server can execute programs to modify the server and application settings that are stored in the configuration directory or to change the port number that a server listens to.

The Admin Server itself can be managed through its own Java-based interface, by editing its configuration files, or through command-line tools.

## E.2. ADMIN SERVER CONFIGURATION

The Admin Server is a separate server from Red Hat Directory Server or Red Hat Certificate System, although they work interdependently. The Admin Server processes, file locations, and configuration options are also separate. This chapter covers the Admin Server information, including starting and stopping the Admin Server, enabling SSL, viewing logs, and changing Admin Server configuration properties, such as the server port number.

### E.2.1. Directory Server File Locations

Red Hat Admin Server conforms to the Filesystem Hierarchy Standards. For more information on FHS, see the FHS homepage, <http://www.pathname.com/fhs/>.

There are slight difference in the file locations depending on the platform, so the default Red Hat Enterprise Linux FHS locations (used in the examples) may not match every installation. Some platforms treat the Admin Server as optional software and therefore, under FHS, store Admin Server files in **/opt** directories.

The files and directories installed with Directory Server are listed in the tables below for each supported platform.

**Table E.1. Red Hat Enterprise Linux 4 and 5 (x86 and x86\_64)**

| File or Directory   | Location                                                                                |
|---------------------|-----------------------------------------------------------------------------------------|
| Log files           | <code>/var/log/dirsrv/admin-serv/</code>                                                |
| Configuration files | <code>/etc/dirsrv/admin-serv/</code>                                                    |
| Instance directory  | <code>/usr/lib/dirsrv/admin-serv/</code>                                                |
| Database files      | <code>/var/lib/dirsrv/admin-serv/</code>                                                |
| Runtime files       | <code>/var/lock/dirsrv/admin-serv.*</code><br><code>/var/run/dirsrv/admin-serv.*</code> |
| Init scripts        | <code>/etc/rc.d/init.d/dirsrv-admin/</code><br><code>/etc/sysconfig/dirsrv-admin</code> |
| Tools               | <code>/usr/bin/</code><br><code>/usr/sbin/</code>                                       |

## E.2.2. Starting and Stopping the Admin Server

The Admin Server is running when the `setup-ds-admin.pl` configuration script completes. Avoid stopping and starting the server to prevent interrupting server operations.

- When starting in SSL, the start script prompts for the password for the security (SSL certificate) database. It is possible to restart in SSL without being prompted for a password by using a password file. See [Section E.2.9.4, “Creating a Password File for the Admin Server”](#) for more information.

If there is not password file, then the Admin Server cannot be restarted in SSL through the Console, only the command-line scripts.

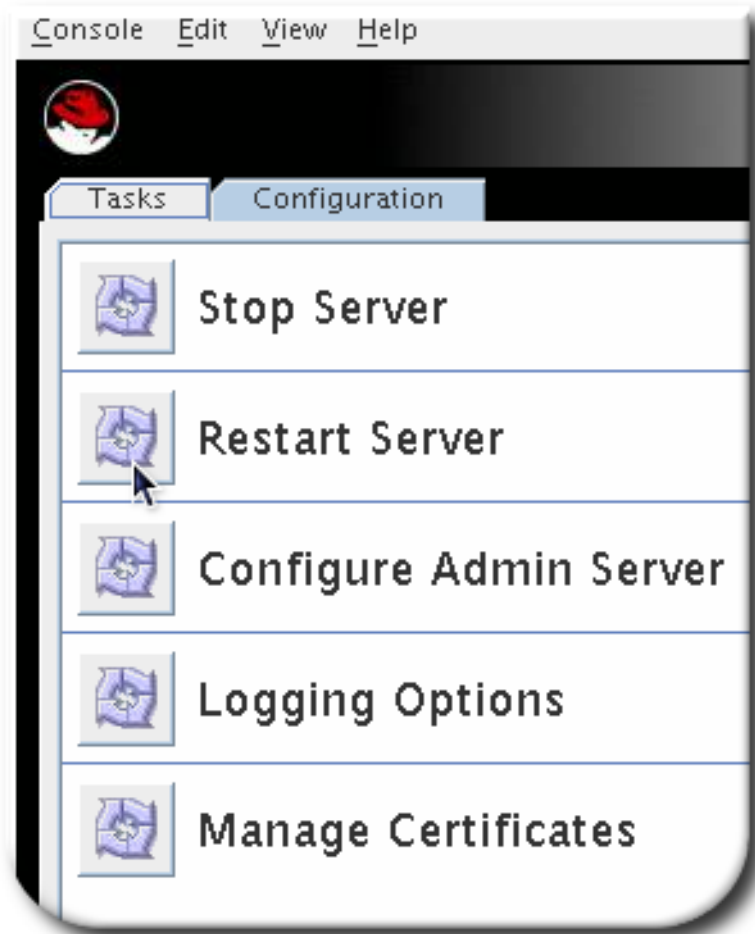
- Rebooting the host system can automatically start the Admin Server's `httpd` process. The directory provides startup or run command (`rc`) scripts. On Red Hat Enterprise Linux, use the `chkconfig` command to enable the Admin Server to start on boot.

### E.2.2.1. Starting and Stopping Admin Server from the Console

1. Start the Console, and open the Admin Console.

```
/usr/bin/redhat-idm-console -a http://localhost:9830
```

2. In the **Tasks** tab, click **Restart Server** or **Stop Server**.



When the Admin Server is successfully started or stopped from the Console, the server displays a message box stating that the server has either started or shut down.

### E.2.2.2. Starting and Stopping Admin Server from the Command Line

There are two ways to start, stop, or restart the Admin Server:

- There are scripts in the **/usr/sbin** directory.

```
/usr/sbin/{start|stop|restart}-ds-admin
```

- The Admin Server service can also be stopped and started using system tools on Red Hat Enterprise Linux 6 (64-bit) using the **service** command. For example:

```
service dirsrv-admin {start|stop|restart}
```



#### NOTE

The service name for the Admin Server process on Red Hat Enterprise Linux 6 (64-bit) is **dirsrv-admin**.

### E.2.3. Opening the Admin Server Console

There is a simple script to launch the main Console. On Red Hat Enterprise Linux, run the following:

```
/usr/bin/redhat-idm-console
```

When the login screen opens, the Admin Server prompts for the user name, password, and Admin Server location. The Admin Server location is a URL; for a standard connection, this has the **http:** prefix for a standard HTTP protocol. If SSL/TLS is enabled, then this uses the **https:** prefix for the secure HTTPS protocol.



Figure E.2. Login Box

#### NOTE

It is possible to send the Admin Server URL and port with the start script. For example:

```
/usr/bin/redhat-idm-console -a http://localhost:9830
```

The **a** option is a convenience, particularly for logging into a Directory Server for the first time. On subsequent logins, the URL is saved. If the Admin Server port number is not passed with the **redhat-idm-console** command, then the server prompts for it at the Console login screen.

This opens the main Console window. To open the Admin Server Console, select the Admin Server instance from the server group on the left, and then click the **Open** at the top right of the window.

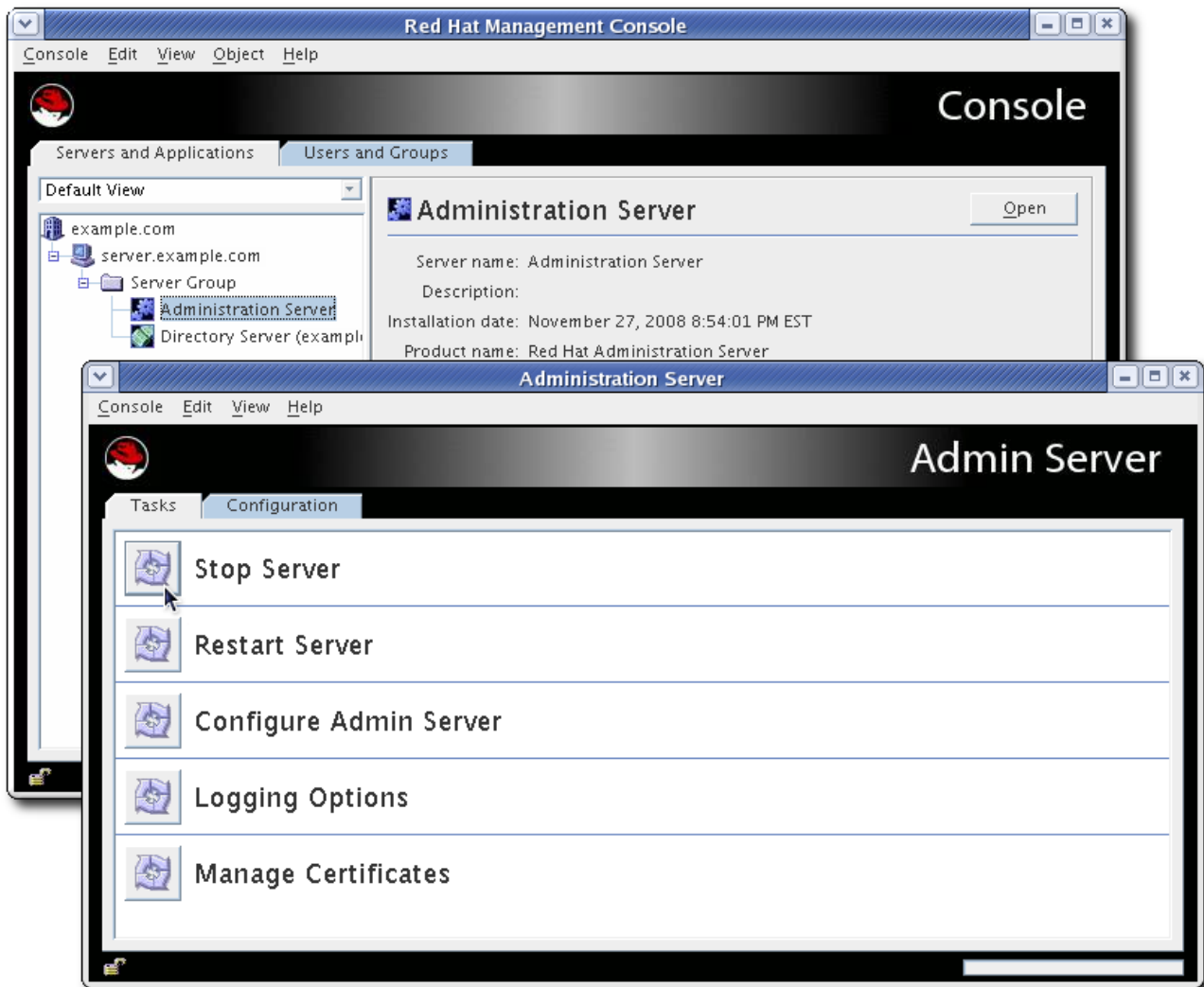


Figure E.3. The Admin Server Console

#### NOTE

Make sure that the Oracle Java Runtime Environment (JRE) or OpenJDK version 1.8.0 is set in the **PATH** before launching the Console. Run the following to see if the Java program is in the **PATH** and to get the version and vendor information:

```
java -version
```

### E.2.4. Viewing Logs

Log files monitor activity for Admin Server and can help troubleshoot server problems. Admin Server logs use the Common Logfile Format, a broadly supported format that provides information about the server.

Admin Server generates two kinds of logs:

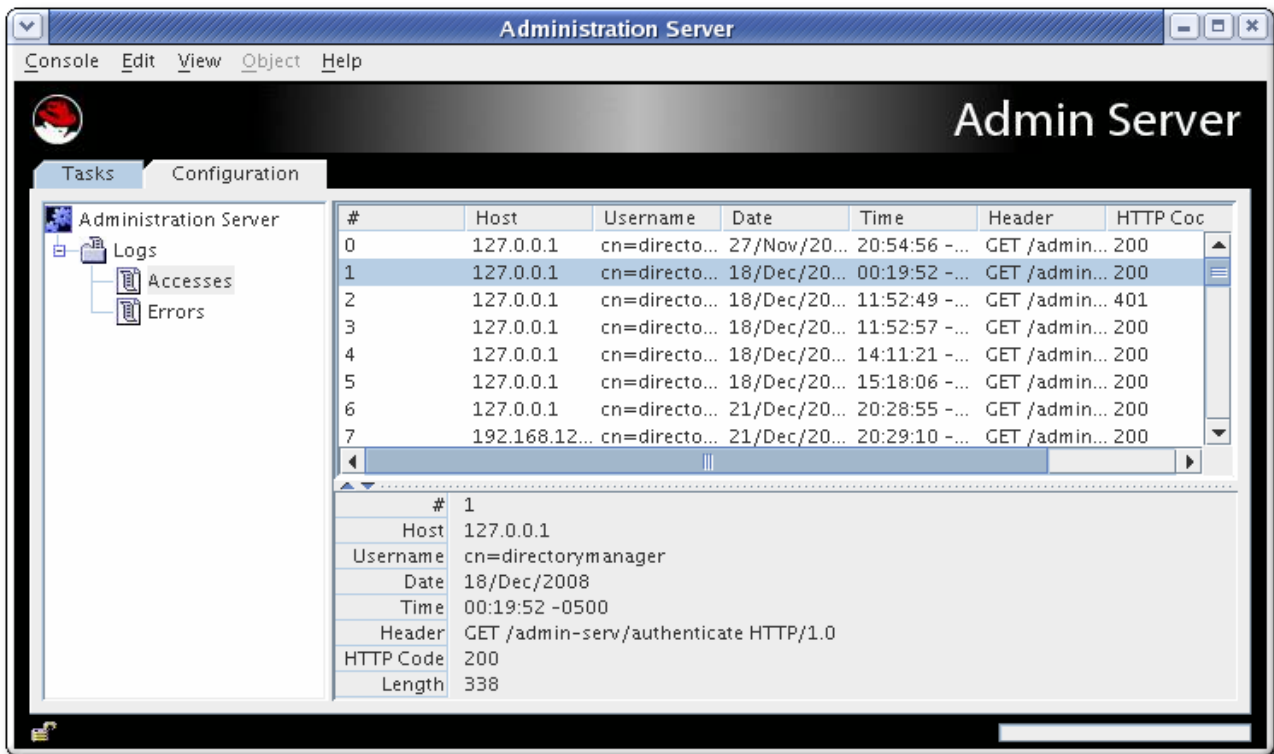
- *Access logs.* Access logs show requests to and responses from the Admin Server. By default, the file is located at **/var/log/dirsrv/admin-servaccess**.
- *Error logs.* Error logs show messages for errors which the server has encountered since the log file was created. It also contains informational messages about the server, such as when the server was started and who tried unsuccessfully to log on to the server. By default, the file is located at **/var/log/dirsrv/admin-servererror**.



The logs can be viewed through Admin Server Console or by opening the log file.

#### E.2.4.1. Viewing the Logs through the Console

1. Open the Admin Server management window.
2. Click the **Configuration** tab.
3. Expand the **Logs** directory, and click the log file name, either **Accesses** or **Error**.



#### E.2.4.2. Viewing Logs in the Command Line

The access log, by default, is at `/var/log/dirsrv/admin-servaccess`. To view the access log, open it in an editor such as `vi`.

Access logs show connections to the Admin Server based on the IP address of the client, the user name, and the method that the request was sent. Each line has the following format:

```
ip_address - bind_DN [timestamp -0500] "GET|POST cgi" HTTP_response bytes
```

Example logs are shown in [Example E.1, "Example Access Logs"](#).

##### Example E.1. Example Access Logs

```
127.0.0.1 - cn=directory manager [23/Dec/2008:19:32:52 -0500] "GET
/admin-serv/authenticate HTTP/1.0" 200 338
192.168.123.121 - cn=directory manager [23/Dec/2008:19:33:14 -0500]
"POST /admin-serv/tasks/Configuration/ServerSetup HTTP/1.0" 200 244
192.168.123.121 - cn=directory manager [23/Dec/2008:19:33:16 -0500] "GET
/admin-serv/tasks/Configuration/ReadLog?op=count&name=access HTTP/1.0"
200 10
```



The error log, by default, is at `/var/log/dirsrv/admin-servererrors`. To view the error log, open it in an editor such as `vi`.

Error logs record any problem response from the Admin Server. Like the access log, error logs also records entries based the client's IP address, along with the type of error message, and the message text:

```
[timestamp] [severity] [client ip_address error_message]
```

The *severity* message indicates whether the error is critical enough for administrator intervention. **[warning]**, **[error]**, and **[critical]** require immediate administrator action. Any other severity means the error is informational or for debugging.

Example logs are shown in [Example E.2, "Example Error Logs"](#).

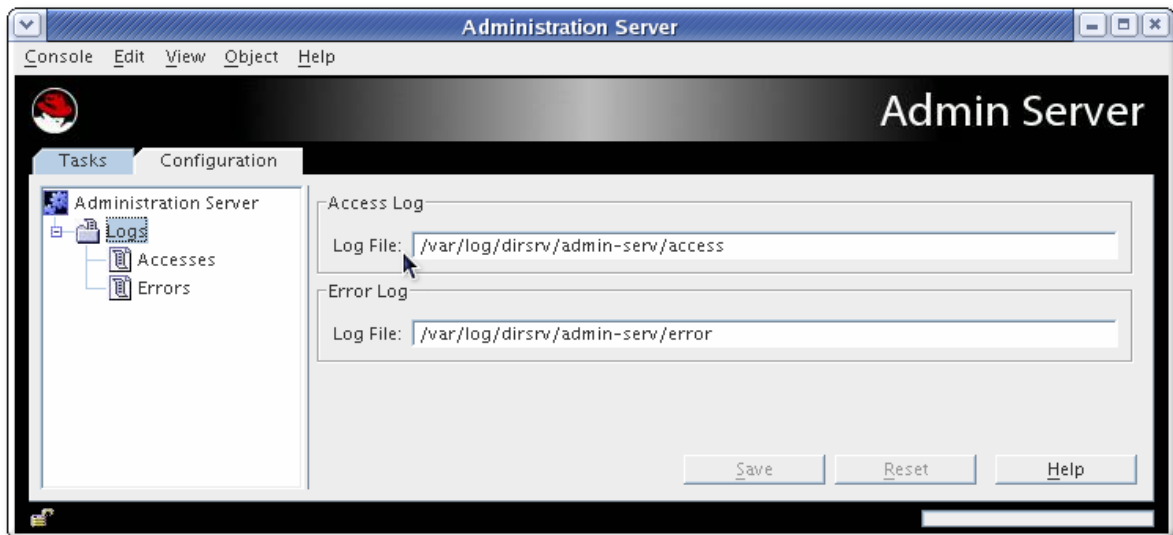
### Example E.2. Example Error Logs

```
[Mon Dec 22 23:44:59 2008] [notice] [client 127.0.0.1]
admserv_host_ip_check: ap_get_remote_host could not resolve 127.0.0.1
[Mon Dec 22 23:44:59 2008] [notice] [client 127.0.0.1]
admserv_host_ip_check: host [localhost.localdomain] did not match
pattern [*.example.com] -will scan aliases
[Mon Dec 22 23:44:59 2008] [notice] [client 127.0.0.1]
admserv_host_ip_check: host alias [localhost] did not match pattern
[*.example.com]
[Mon Dec 22 23:44:59 2008] [notice] [client 127.0.0.1]
admserv_check_authz(): passing [/admin-serv/authenticate] to the
userauth handler
[Mon Dec 22 23:45:16 2008] [notice] [client 192.168.123.121]
admserv_host_ip_check: ap_get_remote_host could not resolve
192.168.123.121
```

#### E.2.4.3. Changing the Log Name in the Console

The access and error log files' names can be changed to rotate the files. This rotation has to be done manually to create new files if the existing log files become too large.

1. Open the Admin Server management window.
2. Click the **Configuration** tab.
3. Click **Logs** in the left panel.
4. In the **Logs** window on the right, enter the new log file name.



### WARNING

The path to the log file is absolute and cannot be changed.

5. Click **OK** to save the changes.
6. Open the **Tasks** tab, and click the **Restart Server** button to restart the server and apply the changes.

#### E.2.4.4. Changing the Log Location in the Command Line

The access and error log files' names and locations can be changed to rotate the files. This rotation has to be done manually to create new files if the existing log files become too large. The location can be changed if the default location in `/var/log/dirsrv/admin-srv` does not meet the application needs.

The Admin Server configuration is stored in two locations. The main entry is an LDAP entry in the Configuration Directory Server's **o=NetscapeRoot** database. The other is the `console.conf` file. Changing the log settings requires changing both settings.

1. Edit the Admin Server configuration entry in the Configuration Directory Server.
  1. Get the name of the Admin Server entry. Since the Admin Server entry has a special object class, **nsAdminConfig**, it is possible to search for the entry using that object class to retrieve the DN.

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -b "o=NetscapeRoot" "(objectclass=nsAdminConfig)" dn
```

```
version:1
```

```
dn: cn=configuration,cn=admin-serv-example,cn=Red Hat
Administration Server,cn=Server
Group,cn=server.example.com,ou=example.com,o=NetscapeRoot
```

2. The Admin Server entry can be edited using **ldapmodify**. The access and error log settings are stored in the **nsAccessLogs** and **nsErrorLogs** attributes, respectively. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=configuration,cn=admin-serv-example,cn=Red Hat
Administration Server,cn=Server
Group,cn=server.example.com,ou=example.com,o=NetscapeRoot
changetype:modify
replace:nsAccessLog
nsAccessLog:/var/log/dirsrv/admin-serv/access_new
```

Hit **Enter** twice to submit the operation, and then **Control+C** to close **ldapmodify**.

2. Open the Admin Server configuration directory.

```
cd /etc/dirsrv/admin-serv
```

3. Edit the **console.conf** file. For the access log, edit the path and filename in the **CustomLog** parameter. For the error log, edit the path and filename in the **ErrorLog** parameter.

```
CustomLog /var/log/dirsrv/admin-serv/access_new common
ErrorLog /var/log/dirsrv/admin-serv/error_new
```

Leave the term **common** after the access log path; this means that the access log is in the Common Log Format.

4. Restart the Admin Server.

```
service dirsrv-admin restart
```

#### E.2.4.5. Setting the Logs to Show Hostnames Instead of IP Addresses

By default, the logs show the IP address of the clients which connect to the Admin Server. This is faster for the Admin Server, since it does not have to do a DNS lookup for every connection. It is possible to set the Admin Server to perform a DNS lookup so that host names are used in the logs. Along with being friendlier to read and search, using host names instead of IP addresses also removes some unnecessary error messages about being unable to resolve host names.

To configure the Admin Server to perform DNS lookups:

1. Edit the **console.conf** file for the Admin Server.

```
cd /etc/dirsrv/admin-serv
vim console.conf
```

2. Set the **HostnameLookups** parameter to **on**. By default, this is turned off, so that IP addresses are recorded in logs instead of host names.

HostnameLookups on

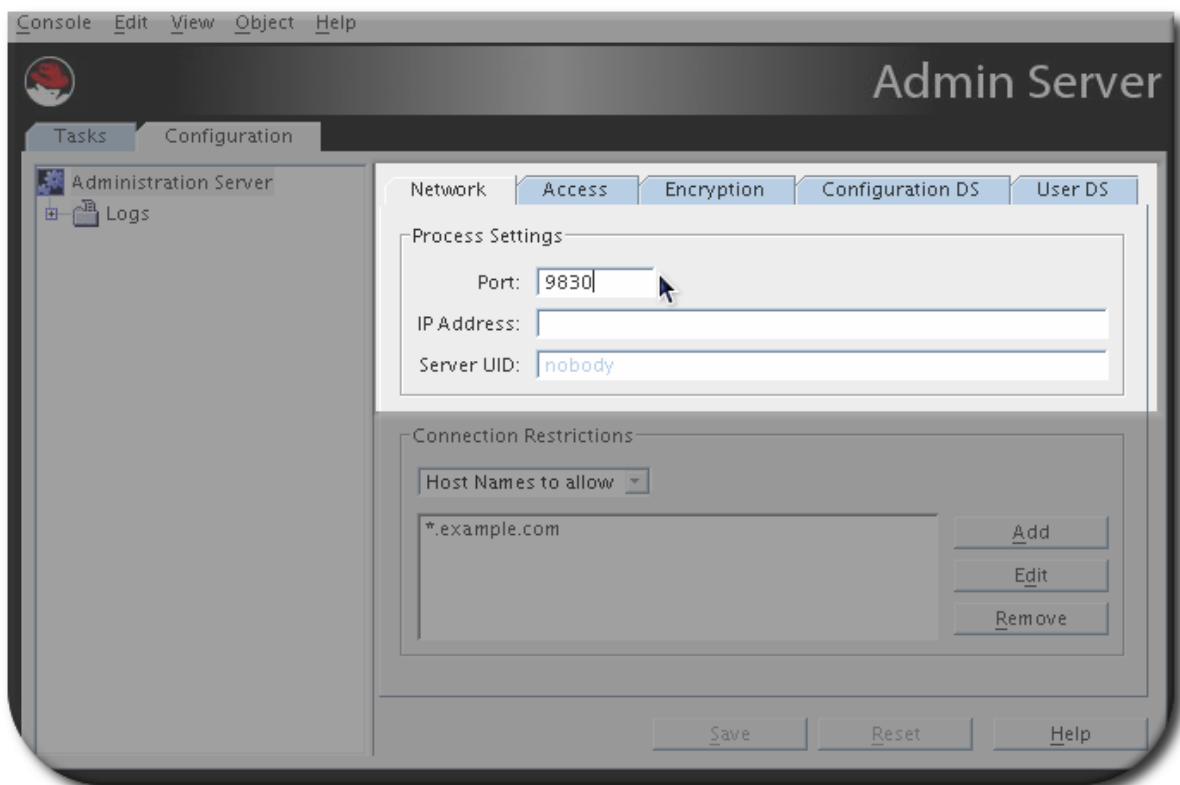
## E.2.5. Changing the Port Number

The *port number* specifies where an instance of Admin Server listens for messages.

The default port number for Admin Server is set when the instance is first installed and the configuration script, such as **setup-ds-admin.pl**, is run. The default port number is **9830**, although if that number is in use, then the setup program will use a randomly-generated number larger than **1024** or one can assign any port number between **1025** and **65535**.

### E.2.5.1. Changing the Port Number in the Console

1. Open the Admin Server management window.
2. Click the **Configuration** tab.
3. Click the **Network** tab.



4. Enter the port number for the Admin Server instance in the **Port** field. The Admin Server port number has a default number of **9830**.
5. Click **OK**.
6. Open the **Tasks** tab, and click the **Restart Server** button to restart the server and apply the changes.

7. Close the Console, and then restart the Console, specifying the new Admin Server port number in the connection URL.

### E.2.5.2. Changing the Port Number in the Command Line

The port number for the Admin Server is **9830** by default.

The Admin Server configuration is stored in two locations. The main entry is an LDAP entry in the Configuration Directory Server's **o=NetscapeRoot** database. The other is the **console.conf** file. Changing the port number requires changing both settings.

1. Edit the Admin Server configuration entry in the Configuration Directory Server.
  1. Get the name of the Admin Server entry. Since the Admin Server entry has a special object class, **nsAdminConfig**, it is possible to search for the entry using that object class to retrieve the DN.

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -b "o=NetscapeRoot" "
(objectclass=nsAdminConfig)" dn

version:1
dn: cn=configuration,cn=admin-serv-example,cn=Red Hat
Administration Server,cn=Server
Group,cn=server.example.com,ou=example.com,o=NetscapeRoot
```

2. The Admin Server entry can be edited using **ldapmodify**. The port number is set in the **nsServerPort** attribute. For example:

```
ldapmodify -D "cn=directory manager" -W -p 389 -h
server.example.com -x

dn: cn=configuration,cn=admin-serv-example,cn=Red Hat
Administration Server,cn=Server
Group,cn=server.example.com,ou=example.com,o=NetscapeRoot
changetype:modify
replace:nsServerPort
nsServerPort:10030
```

Hit **Enter** twice to submit the operation, and then **Control+C** to close **ldapmodify**.

2. Open the Admin Server configuration directory.

```
cd /etc/dirsrv/admin-serv
```

3. Edit the **Listen** parameter in the **console.conf** file.

```
Listen 0.0.0.0:10030
```

4. Restart the Admin Server.

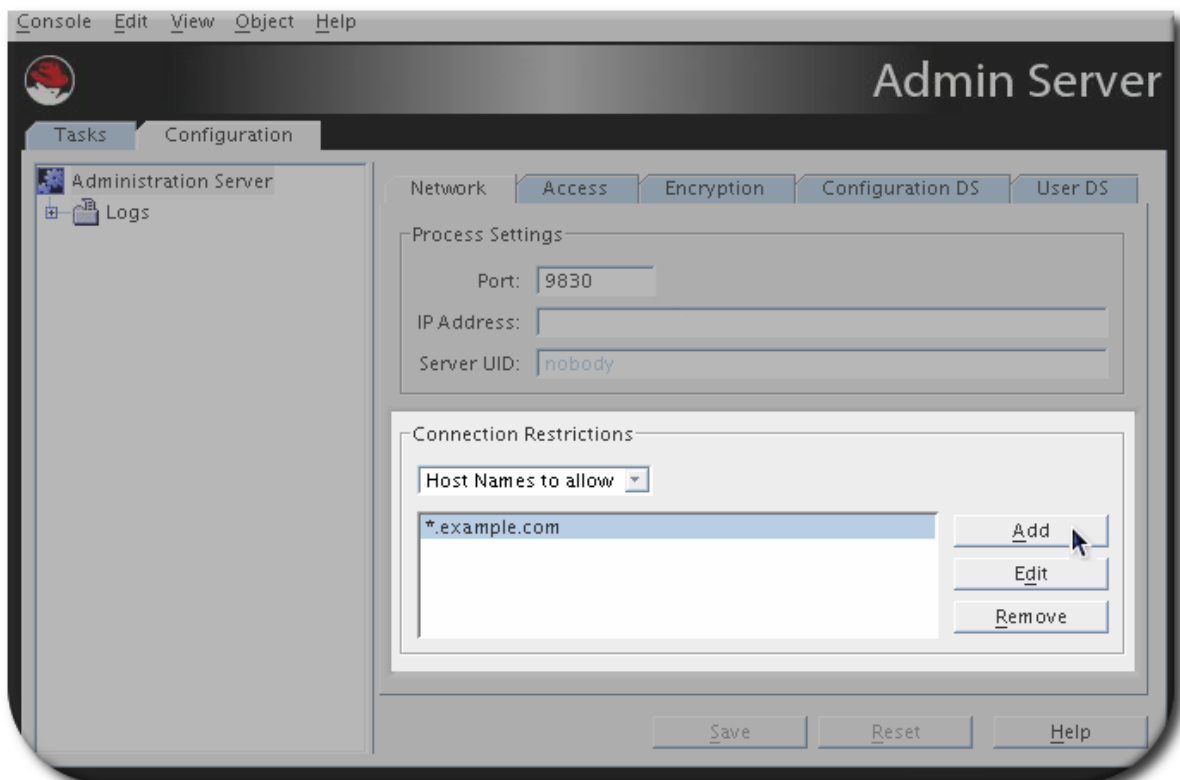
```
service dirsrv-admin restart
```

## E.2.6. Setting Host Restrictions

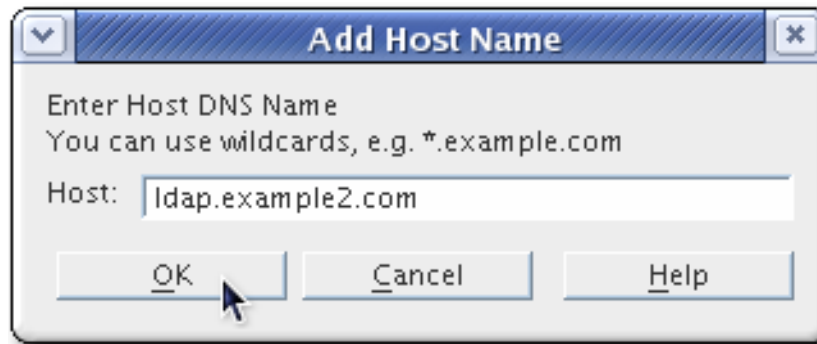
*Connection restrictions* specify which hosts are allowed to connect to the Admin Server. You can list these hosts by DNS name, IP address, or both. Only host machines listed within the connection restriction parameters are allowed to connect to the Admin Server. This setting allows wildcards within a domain or an IP address range to make setting connection restrictions simpler.

### E.2.6.1. Setting Host Restrictions in the Console

1. Open the Admin Server management window.
2. Click the **Configuration** tab.
3. Click the **Network** tab.
4. The **Connection Restrictions** area displays a list of hosts allowed to connect to the Admin Server. The drop-down list specifies whether the list entries are added by DNS name or by IP address. The list is evaluated first by host names, and then by IP addresses.



5. Click the **Add** button to add another host to the list of allowed computers. To add a host name, make sure the drop-down list at the top reads **Host Names to allow**; to add an IP address, select **IP Addresses to allow**.
6. Fill in the host information, either the host name or an IPv4 or IPv6 address.



The `*` wildcard can be used to specify a group of hosts. For instance, `*.example.com` allows all machines in the **example.com** domain to access the instance. Entering `205.12.*` allows all hosts whose IP addresses begin with **205.12** to access the instance.

When specifying IP address restrictions, include all three separating dots. If you do not, the Admin Server returns an error message.

7. Click **OK** to close the **Add . . .** dialog box, and then click the **Save** button to save the new host.
8. Open the **Tasks** tab, and click the **Restart Server** button to restart the server and apply the changes.

To change the information for a host or IP address listed, click the **Edit** button and change the given information. To remove an allowed host or IP address, select the host from the list, and click **Remove**. **Admin Server**.

### E.2.6.2. Setting Host Restrictions in the Command Line

Host restrictions sets rules for what network clients can connect to the Admin Server and, therefore, to services which use the Admin Server. There are two kinds of host restrictions, restrictions based on the host or domain name and restrictions based on the IP address.

The Admin Server host restrictions are set in the main configuration entry in the Configuration Directory Server's **o=NetscapeRoot** database. There are two attributes for setting host restrictions, **nsAdminAccessAddresses** and **nsAdminAccessHosts** for IP addresses and host names, respectively.



#### NOTE

The Admin Server supports both IPv4 and IPv6 addresses.

The Admin Server entry can be edited using **ldapmodify**.

To set host restrictions:

1. Get the name of the Admin Server entry. Since the Admin Server entry has a special object class, **nsAdminConfig**, it is possible to search for the entry using that object class to retrieve the DN.

```
ldapsearch -D "cn=directory manager" -w secret -p 389 -h
server.example.com -x -b "o=NetscapeRoot" "
(objectclass=nsAdminConfig)" dn
```

```
version:1
dn: cn=configuration,cn=admin-serv-example,cn=Red Hat Administration
Server,cn=Server
Group,cn=server.example.com,ou=example.com,o=NetscapeRoot
```

2. To set IP address-based restrictions, edit the ***nsAdminAccessAddresses*** attribute. Either IPv4 or IPv6 addresses can be used.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=configuration,cn=admin-serv-example,cn=Red Hat Administration
Server,cn=Server
Group,cn=server.example.com,ou=example.com,o=NetscapeRoot
changetype:modify
replace:nsAdminAccessAddresses
nsAdminAccessAddresses:72.5.*.*
```

Hit **Enter** twice to submit the operation, and then **Control+C** to close **ldapmodify**.

The ***nsAdminAccessAddresses*** value can use wildcards to allow ranges. Either IPv4 or IPv6 addresses can be used.

For example, to allow all IP addresses:

```
nsAdminAccessAddresses:*
```

To allow only a subset of addresses on a local network:

```
nsAdminAccessAddresses:192.168.123.*
```

3. To set host name or domain-based restrictions, edit the ***nsAdminAccessHosts*** attribute.

```
ldapmodify -D "cn=directory manager" -W -p 389 -h server.example.com
-x

dn: cn=configuration,cn=admin-serv-example,cn=Red Hat Administration
Server,cn=Server
Group,cn=server.example.com,ou=example.com,o=NetscapeRoot
changetype:modify
replace:nsAdminAccessHosts
nsAdminAccessHosts:*.example.com
```

Hit **Enter** twice to submit the operation, and then **Control+C** to close **ldapmodify**.

4. Restart the Admin Server to apply the changes.

```
service dirsrv-admin restart
```

## E.2.7. Changing the Admin User's Name and Password



During installation, you are asked to enter a user name and password for the *Configuration Administrator*, the user authorized to access and modify the entire configuration directory. The Configuration Administrator entry is stored in the directory under the following DN:

```
uid=userID,ou=Administrators,ou=TopologyManagement,o=NetscapeRoot
```

The Configuration Administrator's user name and password are managed through the Directory Server and are represented in an LDAP entry; this is described in the *Directory Server Administrator's Guide*.

During installation, the Configuration Administrator's user name and password are used to automatically create the *Administration Server Administrator*. This user can perform a limited number of administrative tasks, such as starting, stopping, and restarting servers in a local server group. The Administration Server Administrator is created for the purpose of logging into the Console when the Directory Server is not running.

The Administration Server Administrator does not have an LDAP entry; it exists only as an entity in a local configuration file, `/etc/dirsrv/admin-serv/admpw`.

Even though they are created at the same time during installation, and are identical at that time, the Configuration Administrator and Administration Server Administrator are two separate entities. If you change the user name or password for one in the Console, the Console does not automatically make the same changes for the other.

The Administration Server Administrator has full access to all configuration settings in the Admin Server. The information for the admin user is set on the **Access** tab in the Console.

## NOTE

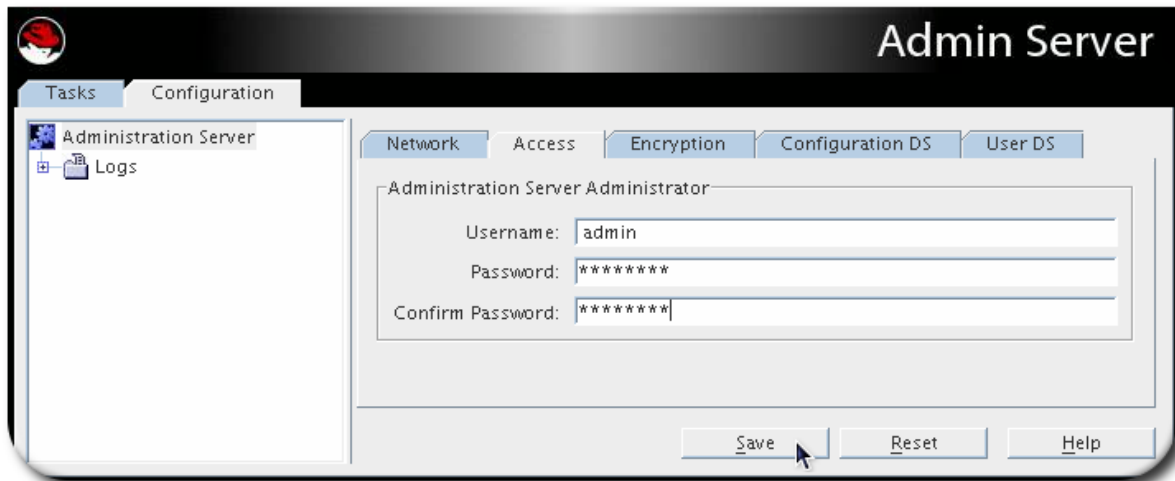
The Admin Server administrator user name and password are stored in the `/etc/dirsrv/admin-serv/admpw` file. For example:

```
admin:{SHA}W6ph5Mm5Pz8GgiULbPgZG37mj9g=
```

The password is encrypted and cannot be changed directly in the `admpw` file. The user name can be changed in this file, but cannot be used to log into the Console unless the password is updated in the Console first. For this reason, it is better to edit the Administration Server Administrator user name and password only through the Admin Server Console.

To change the Administration Server Administrator's ID or password:

1. Open the Admin Server management window.
2. Click the **Configuration** tab.
3. Click the **Access** tab.
4. Change the admin user's name or password. The user name is the ID given for logging into the Admin Server.

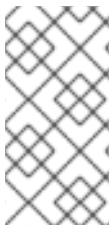


5. Click **Save**.

## E.2.8. Managing SELinux for the Admin Server

SELinux is a security function in Linux that categorizes files, directories, ports, processes, users, and other objects on the server. Each object is placed in an appropriate security context to define how the object is allowed to behave on the server through its role, user, and security level. These roles for objects are grouped in domains, and SELinux rules define how the objects in one domain are allowed to interact with objects in another domain.

SELinux itself is much more complex to manage and implement than what is described here. This section is concerned only with giving the SELinux details for the Administration Server. For more general information about SELinux, see the [Red Hat Enterprise Linux 6 Security-Enhanced Linux User Guide](#).



### NOTE

SELinux is a feature of Red Hat Enterprise Linux and, as such, is covered in the Red Hat Enterprise Linux *Security-Enhanced Guide* at [https://access.redhat.com/site/documentation/en-US/Red\\_Hat\\_Enterprise\\_Linux/6/html/Security-Enhanced\\_Linux/index.html](https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Security-Enhanced_Linux/index.html).

### E.2.8.1. SELinux Definitions for the Admin Server

SELinux has three different levels of enforcement: disabled (no SELinux), permissive (where the rules are lax), and enforcing (where all rules are strictly enforced). Red Hat Directory Server and the Admin Server have defined SELinux policies that allow them to run as normal under strict SELinux enforcing mode.

By default, the Admin Server runs confined by SELinux policies.

There are two SELinux security contexts that apply to the Admin Server. When the server starts, it starts within its own Admin Server-specific SELinux domain, **dirsrvadmin\_t**, where the Admin Server scripts are confined. However, the Admin Server process is simply the Apache web server daemon, **httpd**. So, once the Admin Server process is started, it transitions into the existing **httpd\_t** domain on Red Hat Enterprise Linux.

**NOTE**

For the Admin Server to start with the process properly confined, it must be started or restarted using the **service** command. For example:

```
service dirsrv-admin restart
```

The **start-ds-admin** script is not supported by SELinux.

All CGIs invoked by the Admin Server, such as scripts for Admin Express, run in a special confined security domain, **httpd\_dirsrvadmin\_script\_t**, which is separate from the **dirsrvadmin\_t** or **httpd\_t** domains.

Table E.2, “Summary of Admin Server SELinux Policies” lists the security contexts and domains for the major components of the Admin Server.

**Table E.2. Summary of Admin Server SELinux Policies**

| File Path                                                       | Security Context                | Description                                                                                                                                                                                               |
|-----------------------------------------------------------------|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>dirsrvadmin_t Domain</b>                                     |                                 |                                                                                                                                                                                                           |
| /usr/sbin/[start stop restart]-ds-admin                         | dirsrvadmin_exec_t              | The Admin Server start, stop, and restart scripts                                                                                                                                                         |
| /etc/dirsrv/admin-serv/*                                        | dirsrvadmin_config_t            | Admin Server configuration files, such as <b>adm.conf</b>                                                                                                                                                 |
| <b>httpd_dirsrvadmin_script_t Domain</b>                        |                                 |                                                                                                                                                                                                           |
| /usr/lib[64]/dirsrv/cgi-bin/*                                   | httpd_dirsrvadmin_script_exec_t | The CGI scripts and files used by Admin Server web services, like Admin Express                                                                                                                           |
| <b>httpd_t Domain<sup>[a]</sup></b>                             |                                 |                                                                                                                                                                                                           |
| /var/log/dirsrv/admin-serv/*                                    | httpd_log_t                     | The log files for the Admin Server                                                                                                                                                                        |
| /var/run/dirsrv/admin-serv.*                                    | httpd_var_run_t                 | The PID file for the Admin Server process                                                                                                                                                                 |
| Ports 80, 443, and the Admin Server HTTP port (9830 by default) | http_port_t                     | The ports used by the Apache web server and the Admin Server web services, including the default HTTP and HTTPS Apache ports and whatever the configured HTTP port <sup>[b]</sup> for the Admin Server is |

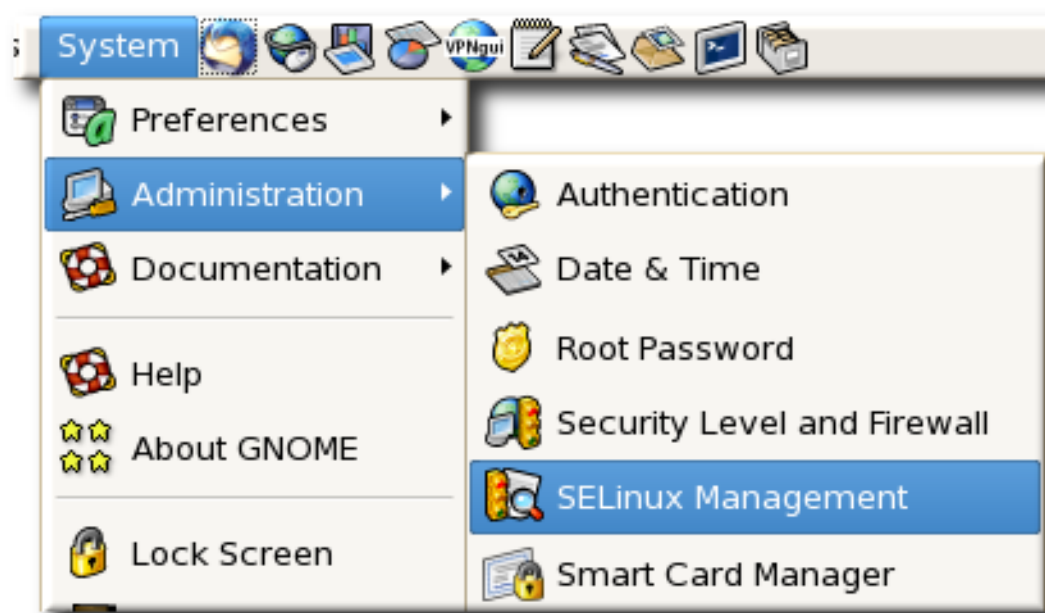
| File Path                                                                                                                                                                                                                                                             | Security Context | Description |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|-------------|
| [a] There are more contexts configured by default within the <code>httpd_t</code> domain, but they are not relevant to the Admin Server SELinux policies.                                                                                                             |                  |             |
| [b] Only the HTTP port is configured for the Admin Server when it is set up, so only this port is added to the SELinux configuration automatically. The HTTPS port must be added manually, as described in <a href="#">Section 1.10.6, “Labeling SSL/TLS Ports”</a> . |                  |             |

The Admin Server SELinux policies are configured when the server is set up (when **setup-ds-admin.pl** or **register-ds-admin.pl** are run). These policies are removed when the Admin Server is uninstalled.

### E.2.8.2. Viewing SELinux Policies for the Admin Server

All Admin Server policies are located in `/usr/share/selinux/strict/dirsrv-admin.pp`. The configured policies can be viewed using the SELinux Administration GUI.

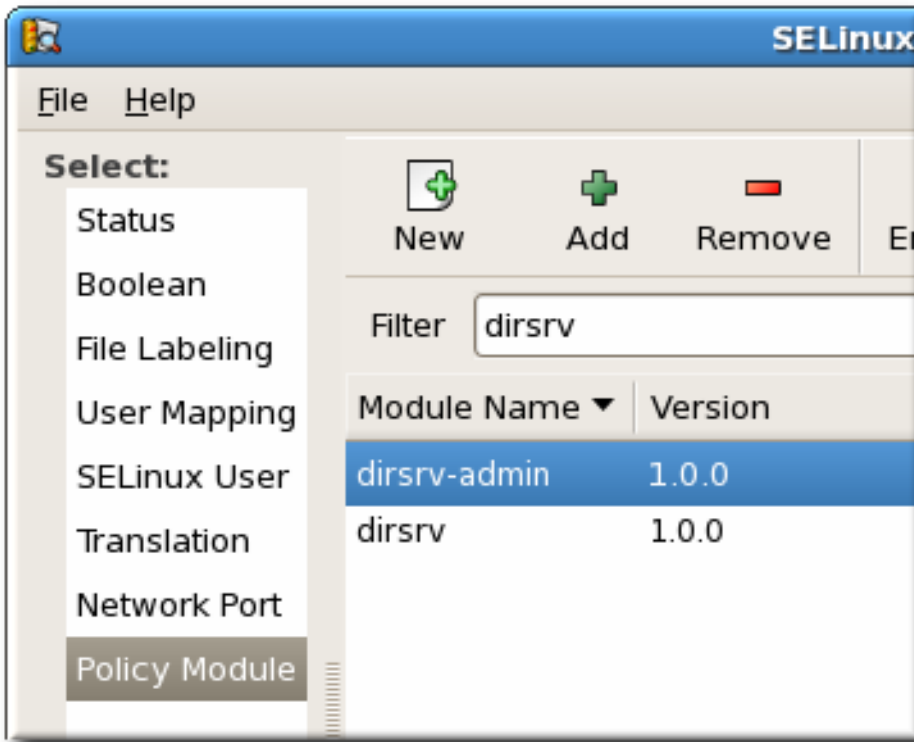
1. Open the **Systems** menu.
2. Open the **Administration** menu, and select the **SELinux Management** item.



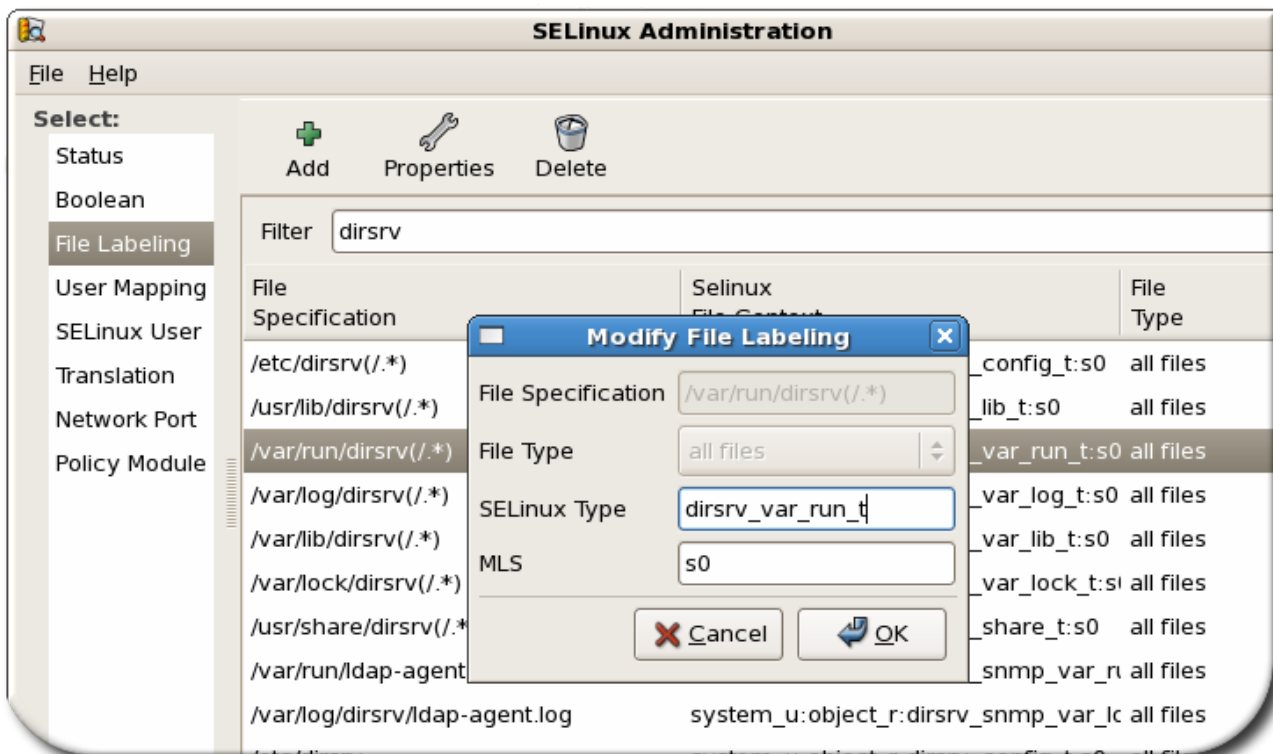
#### NOTE

You can launch the GUI from the command line using **system-config-selinux**.

To check the version of the Admin Server SELinux policy installed, click the **Policy Module** link.



To view the policies set on the individual files and processes, click the **File Labeling** link. To view the policies for the port assignments for the server, click the **Network Port** link.



### E.2.8.3. Labeling SSL/TLS Ports

When the Admin Server is first set up, the given HTTP port is labeled for SELinux (the default is port 9830). However, SSL/TLS is set up separately, after the Admin Server is already configured, so the HTTPS port for the Admin Server is not automatically labeled.

The default HTTPS port, 443, is already labeled as part of the Apache web service policies. If the Admin

Server uses a port other than the default for its SSL/TLS connections, however, then an administrator must label the port manually. This can be done in the SELinux administrative interface show in [Section 1.10.3, “Viewing and Editing SELinux Policies for the Directory Server”](#). It can also be done easily using the **semanage** script.

Use the **port** subcommand, the **-t** option to identify the security context, and the **-p** option to identify the port. The **-a** option adds the port label. For example:

```
semanage port -a -t http_port_t -p tcp 1443
```

To delete a port label, use the **-d** option. For example:

```
semanage port -d -t http_port_t -p tcp 1443
```

#### E.2.8.4. Starting the Admin Server Confined by SELinux

The Admin Server's **httpd** process initially starts in its own **dirsrvadmin\_t**, and then transitions to the **http\_t** domain after starting. This daemon only runs confined in the appropriate SELinux policies when the **service** command is used to run the Admin Server.

```
service dirsrv-admin start|stop|restart
```

### E.2.9. Working with SSL

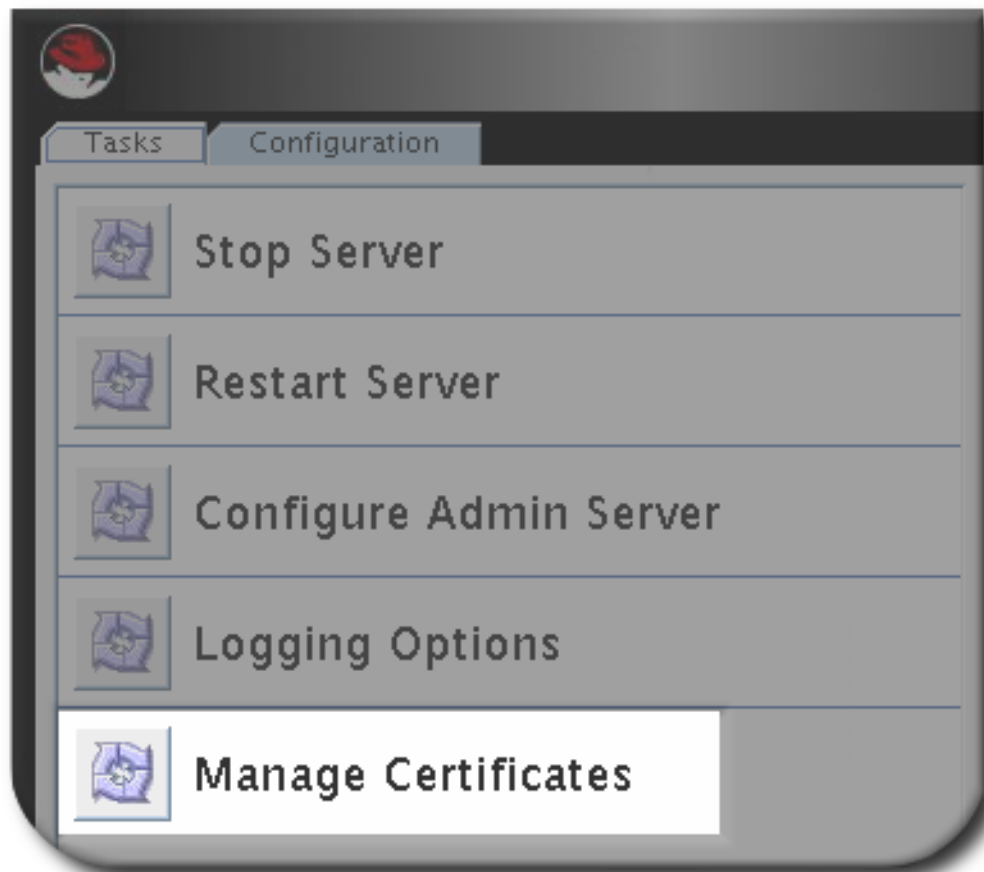
The Admin Server can run over HTTPS (secure HTTP) if SSL is enabled on the server. There are steps to enabling SSL:

1. Generating and submitting a certificate request.
2. Receiving and installing the certificate.
3. Trusting the certificate authority (CA) which issued the certificate.
4. Changing the Admin Server configuration to allow SSL connections.

#### E.2.9.1. Requesting and Installing a Server Certificate

The Admin Server Console has a tool, the **Certificate Request Wizard**, which generates a valid certificate request to submit to any certificate authority (CA).

1. In the Admin Server Console, select the **Tasks** tab, and click **Manage Certificates**.

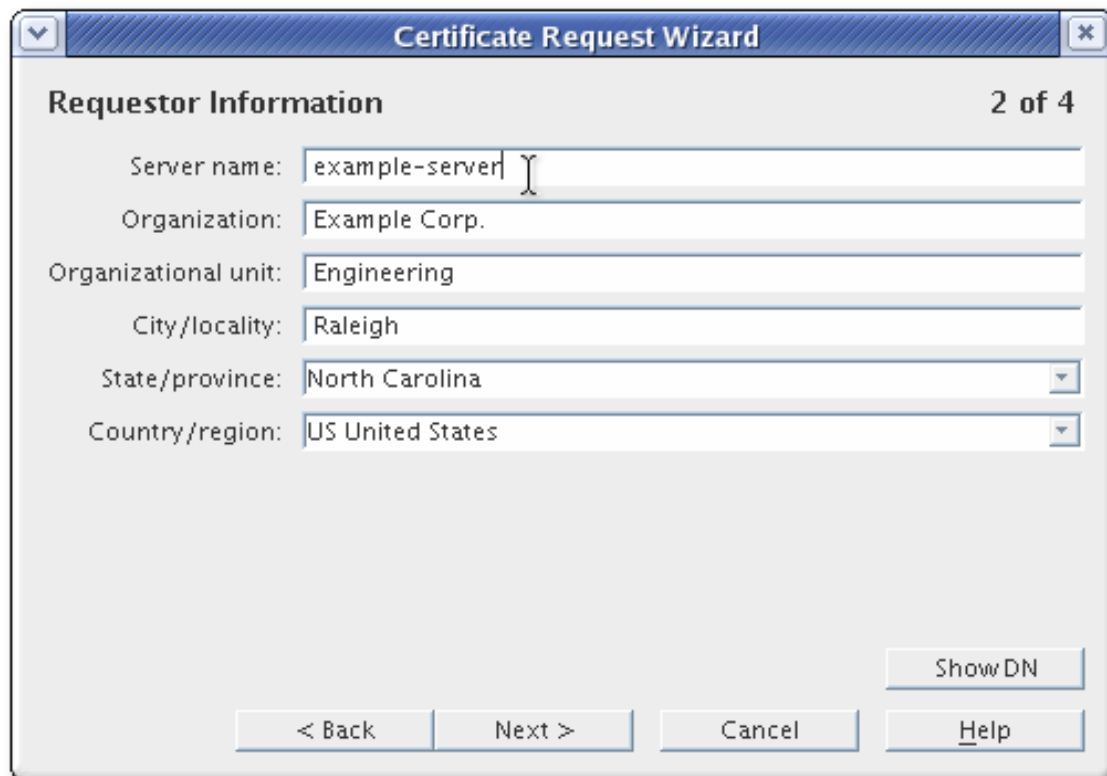


2. Create a certificate request.

1. Select the **Server Certs** tab, and click the **Request** button.

Click **Next**.

2. Enter the **Requester Information** in the blank text fields, then click **Next**.



The image shows a Windows-style dialog box titled "Certificate Request Wizard" with a progress indicator "2 of 4". The "Requestor Information" section contains the following fields:

- Server name: example-server
- Organization: Example Corp.
- Organizational unit: Engineering
- City/locality: Raleigh
- State/province: North Carolina (dropdown menu)
- Country/region: US United States (dropdown menu)

At the bottom right is a "Show DN" button. At the bottom are four buttons: "< Back", "Next >", "Cancel", and "Help".

- **Server Name.** The fully qualified host name of the Directory Server as it is used in DNS and reverse DNS lookups; for example, **server.example.com**. The server name is critical for client-side validation to work, which prevents man-in-the-middle attacks.



### IMPORTANT

This *must* be a valid host name that can be resolved correctly by all Admin Server clients, or TLS/SSL will not work.

- **Organization.** The legal name of the company or institution. Most CAs require this information to be verified with legal documents such as a copy of a business license.
- **Organizational Unit. Optional.** A descriptive name for the organization within the company.
- **Locality. Optional.** The company's city name.
- **State or Province.** The full name of the company's state or province (no abbreviations).
- **Country.** The two-character abbreviation for the country's name (ISO format). The country code for the United States is US.

3. Enter the password that used to protect the private key, and click **Next**.





The **Next** button is grayed out until a password is supplied.

3. The **Request Submission** dialog box provides two ways to submit a request: directly to the CA (if there is one internally) or manually. To submit the request manually, select **Copy to Clipboard** or **Save to File** to save the certificate request which will be submitted to the CA.



To submit the request to a CA manually, either email it or use the web form for the CA, if one is available. Copy the certificate request information and submit it using the appropriate method.

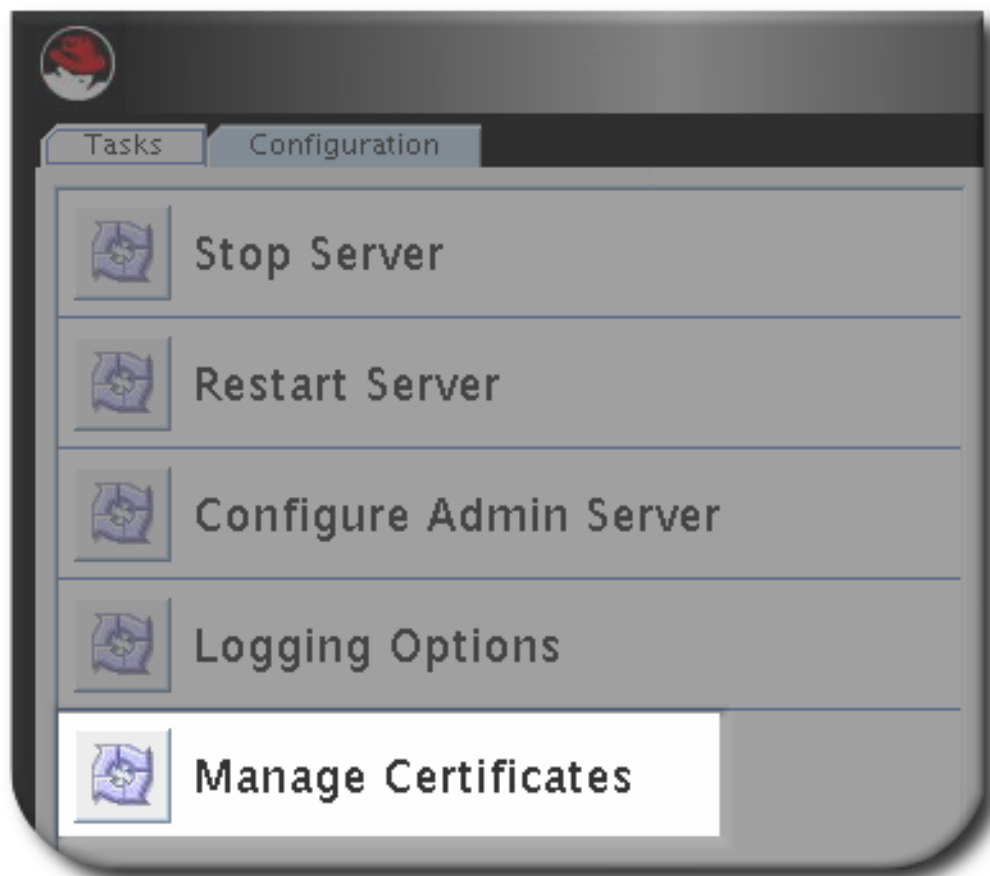
```
-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBrjCCARcCAQAwbjELMAkGA1UEBhMCVXMxEzARBgNVBAgTCkNBTElGT1J
OSUEXLDAQBgVBaoTI25ldHNjYXB1IGNvbW11bm1jYXRpb25zIGNvcnBvcnF
0aW9uMRwwGgYDVQQDEXNtZWxsb24ubmV0c2NhcnGUuY29tMIGfMA0GCSqGSI
b3DQEBAQUAA4GNADCBiQKBgQCwAbskGh6SKY0gHy+UCSLnm3ok3X3u83Us7
ug0EfgSLR0f+K41eNqqRftGR83emqPLD0f0ZLTLjVGJaH4Jn4l1gG+Jdf/n
/zMyahxtV7+mT8G0FFigFfuxaxMjr2j7IvELlxQ4IfZgWwqCm4qQecv3G+N
9YdbjveMVXW0v4XwIDAQABoAAwDQYK
-----END NEW CERTIFICATE REQUEST-----
```

4. Wait for the CA to respond with the server certificate; this can be as short as a few hours for an internal CA or as long as several weeks for a third-party CA.
5. Save the issued certificate to a file.

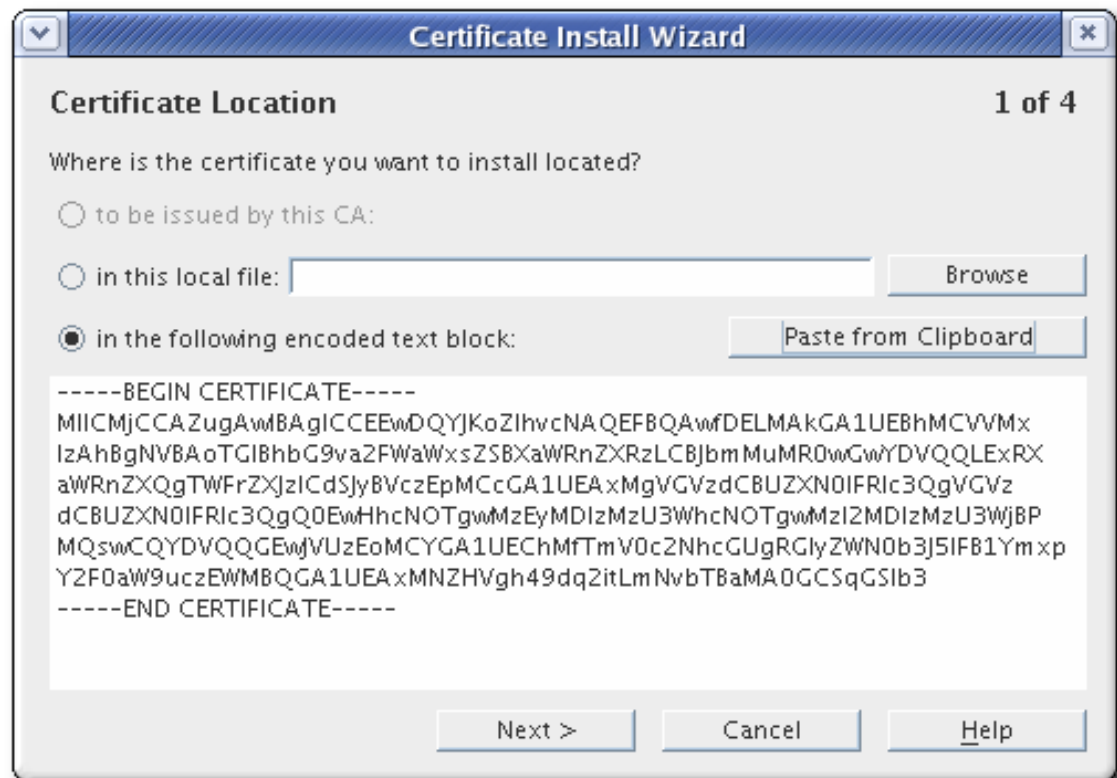
**NOTE**

Keep a backup of the certificate data in a safe location. If the system ever loses the certificate data, the certificate can be reinstalled using the backup file.

6. Install the certificate.
  1. Select the **Tasks** tab, and click **Manage Certificates**.



2. Select the **Server Certs** tab, and click **Install**.
3. Give the absolute path to the certificate (*In this file* radio button) or paste the certificate text in the text box (*In the following encoded text block* radio button), then click **Next**.



4. Check that the certificate information displayed is correct, and click **Next**.
5. Name the certificate, and click **Next**.
6. Provide the password that protects the private key. This password is the same as the one provided in step [c](#).

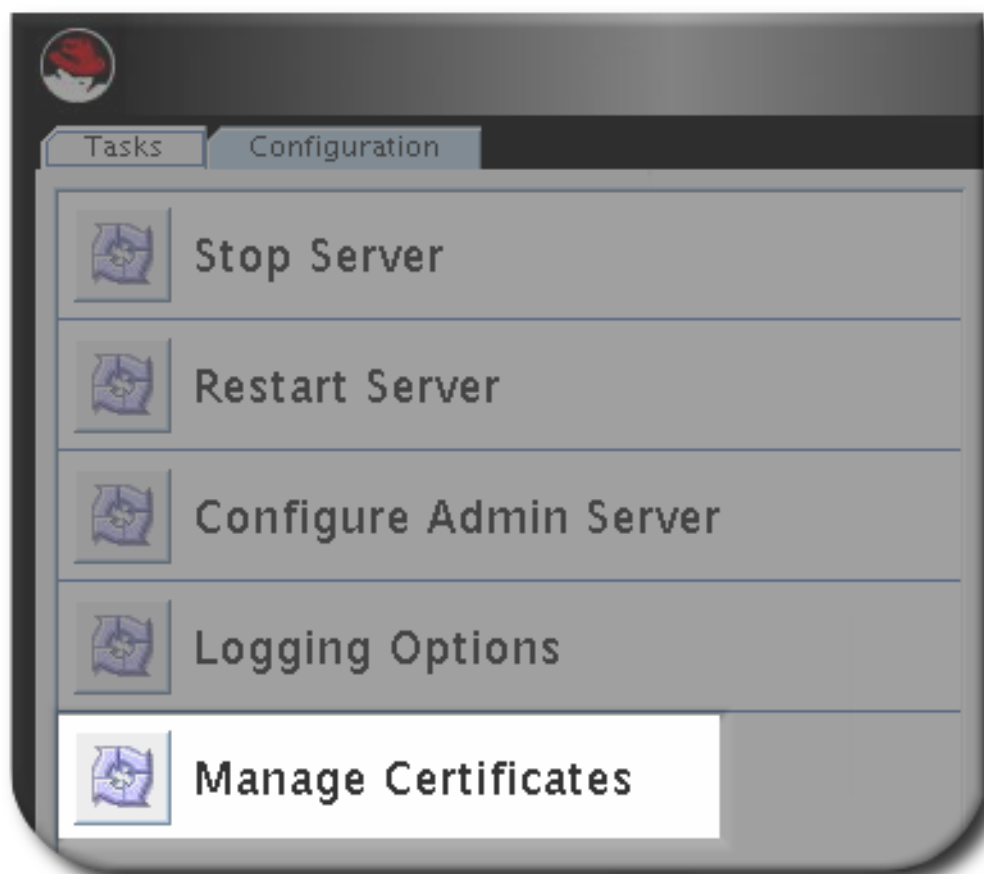
After installing the server certificate, configure the Admin Server to trust the CA which issued the server's certificate.

### E.2.9.2. Installing a CA Certificate

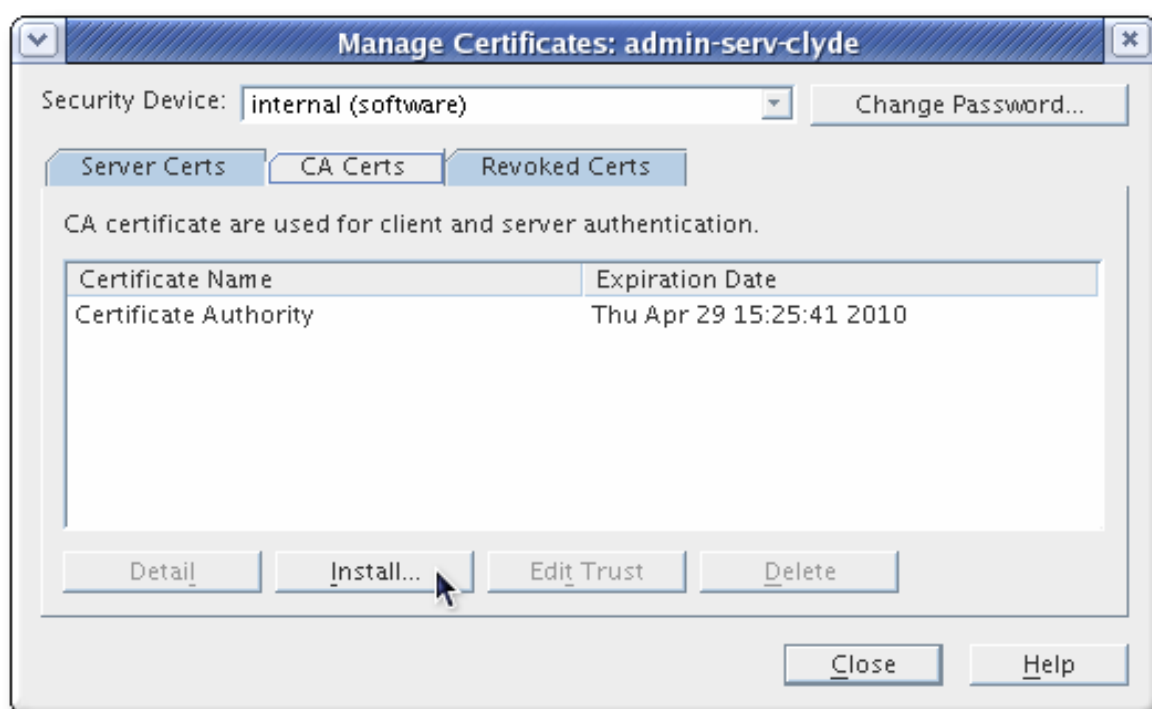
To configure the Admin Server to trust the CA, obtain the CA's certificate and install it into the server's certificate database. Some commercial CAs provide a web site that allow users to automatically download the certificate, while others will email it back to users.

After receiving the CA certificate, use the **Certificate Install Wizard** to configure the Admin Server to trust the CA.

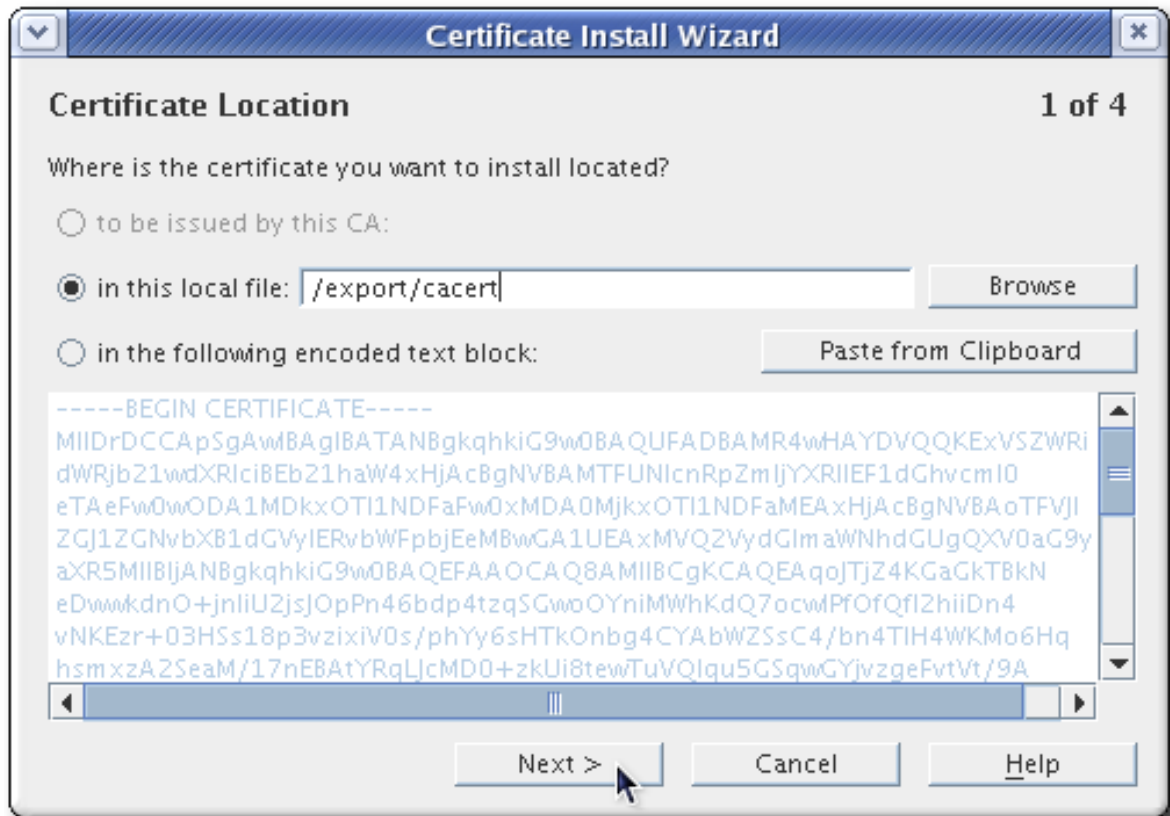
1. In the Admin Server Console, select the **Tasks** tab, and click **Manage Certificates**.



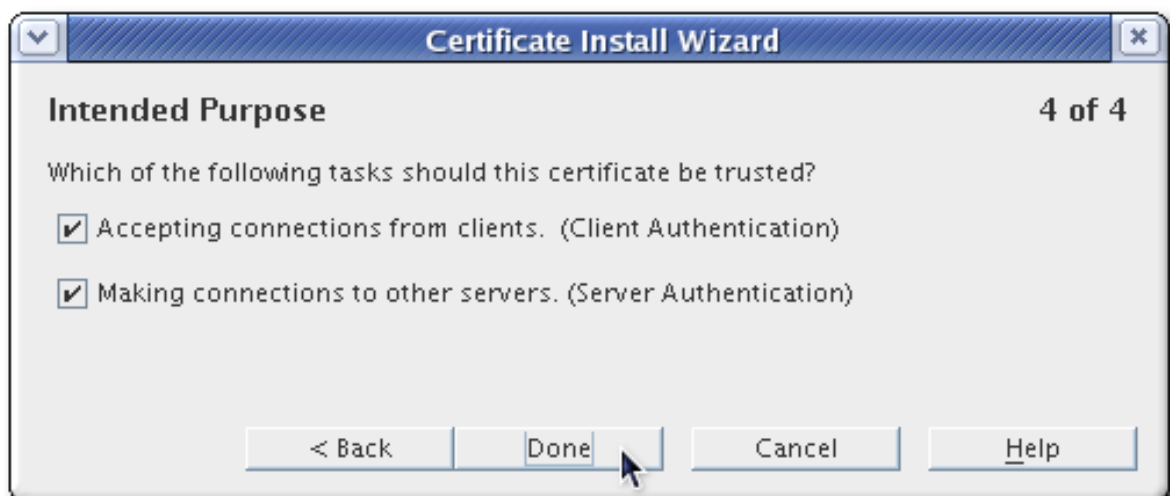
2. Go to the **CA Certs** tab, and click **Install**.



3. If the CA's certificate is saved to a file, enter the path in the field provided. Alternatively, copy and paste the certificate, including the headers, into the text box. Click **Next**.

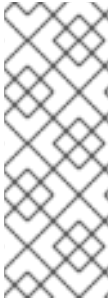


4. Click **Next** to move through the panels that show the CA certificate information and the certificate name.
5. Select the purpose of trusting this certificate authority; it is possible to select both options:
  - *Accepting connections from clients (Client Authentication)*. The server checks that the client's certificate has been issued by a trusted certificate authority.
  - *Accepting connections to other servers (Server Authentication)*. This server checks that the directory to which it is making a connection (for replication updates, for example) has a certificate that has been issued by a trusted certificate authority.



6. Click **Done**.

After installing the CA certificate, it is listed in the **CA Certificates** tab in the Console.



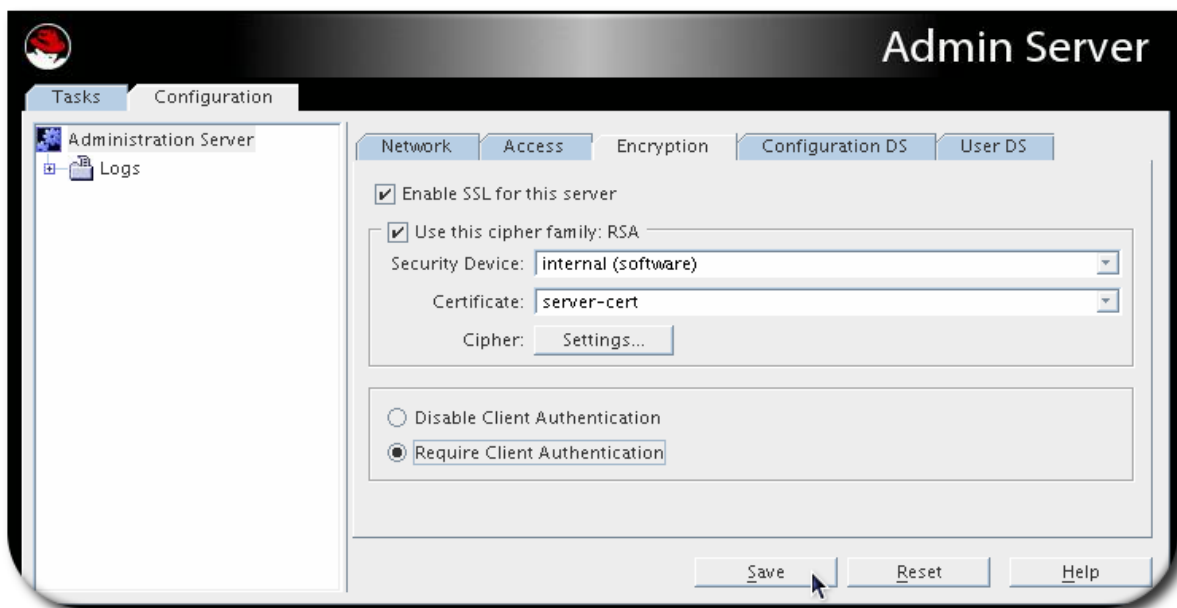
## NOTE

If a CA certificate is incorrectly generated, it is listed in the **Server Certificates** tab in the Console rather than the **CA Certificates** tab. The certificate still works as a CA certificate, even though it is listed in the wrong tab.

Still, request certificates from a real certificate authority to minimize the risk of using an incorrectly generated certificate and breaking SSL/TLS in the Admin Server.

### E.2.9.3. Enabling SSL

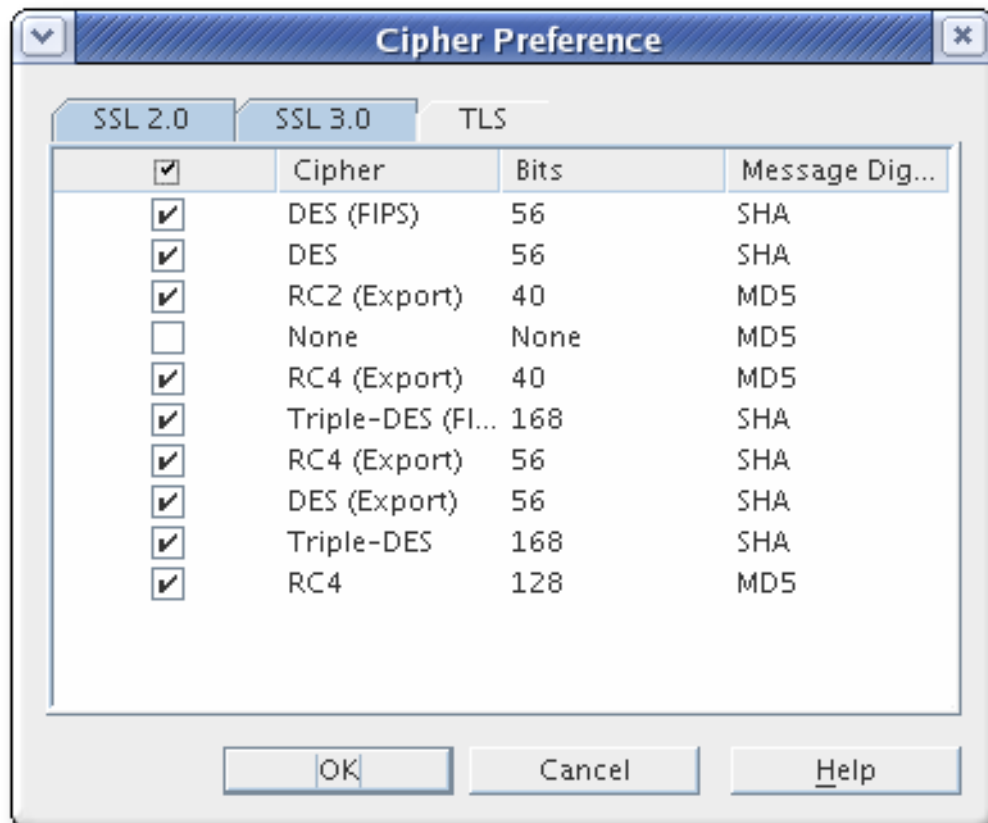
1. Open the Admin Server management window.
2. Click the **Configuration** tab.
3. Click the **Encryption** tab.



4. Select the **Enable SSL for this server** check box.
5. Select the **Use this cipher family: RSA** check box.
6. Choose the security device where the key is stored. By default, the key is stored in the local key database, **Internal (Software-based)**. If the key is stored on an external device (such as a smart card), select that device from the menu.
7. Choose the server certificate to use with SSL.

The certificates available in the token certificate database are listed in the drop-down menu.

8. Click the **Settings** button to set the ciphers that the Admin Server accepts for SSL/TLS connections.



9. Set whether to require client authentication to the Admin Server. Client authentication means that the server checks that the client's certificate has been issued by a trusted CA.

10. Click **Save**.

#### E.2.9.4. Creating a Password File for the Admin Server

Normally, if SSL is enabled, the server prompts for a security password when the Admin Server is restarted:

```
Starting dirsrv-admin:
Please enter password for "internal" token:
```

The Admin Server can use a password file when TLS/SSL is enabled so that the server restarts silently, without prompting for the security password.



#### WARNING

This password is stored in clear text within the password file, so its usage represents a significant security risk. Do not use a password file if the server is running in an unsecured environment.

1. Open the Admin Server configuration directory.

```
cd /etc/dirsrv/admin-serv
```

2. Create a password file named **password.conf**. The file should include a line with the token name and password, in the form *token:password*. For example:

```
internal:secret
```

For the NSS software crypto module (the default software database), the token is always called **internal**.

The password file should be owned by the Admin Server user and set to read-only by the Admin Server user, with no access to any other user (mode **0400**).



#### NOTE

To find out what the Admin Server user ID is, run **grep** in the Admin Server configuration directory:

```
cd /etc/dirsrv/admin-serv
grep ^User console.conf
```

3. In the **/etc/dirsrv/admin-serv** directory, edit the **nss.conf** file to point to the location of the new password file.

```
Pass Phrase Dialog:
Configure the pass phrase gathering process.
The filtering dialog program ('builtin' is a internal
terminal dialog) has to provide the pass phrase on stdout.
NSSPassPhraseDialog file:///etc/dirsrv/admin-serv/password.conf
```

4. Restart the Admin Server. For example:

```
service dirsrv-admin restart
```

After TLS/SSL is enabled, then the Admin Server can only be connected to using HTTPS. All of the previous HTTP (standard) URLs for connecting to the Admin Server and its services no longer work. This is true whether connecting to the Admin Server using the Console or using a web browser.

## E.2.10. Changing Directory Server Settings

The Admin Server stored information about the Directory Server *Configuration Directory* (which stores the instance configuration information) and the Directory Server *User Directory* (which stores the actual directory entries). These can be the same directory instance, but they do not have to be. The settings for both of those databases can be edited in the Admin Server configuration so that it communicates with a different Directory Server instance.

### E.2.10.1. Changing the Configuration Directory Host or Port



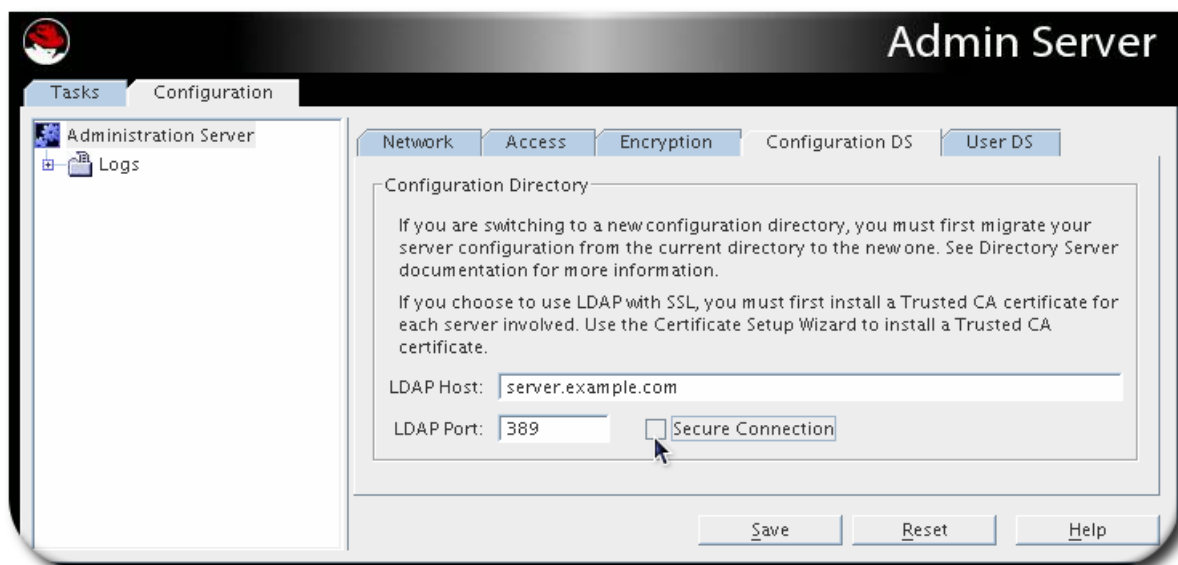
Configuration data are stored under **o=NetScapeRoot** in the Configuration Directory. The configuration database contains server settings such as network topology information and server instance entries. When server configuration changes are stored in the configuration directory subtree.



### WARNING

Changing the Directory Server host name or port number impacts the rest of the servers in the server group. Changing a setting here means the same change must be made for every server in the server group.

1. Open the Admin Server management window.
2. Click the **Configuration** tab.
3. Click the **Configuration DS** tab.
4. Set the Configuration Directory Server connection information.



- The **LDAP Host** is the host name, IPv4, or IPv6 address of the Configuration Directory Server machine.
  - The **LDAP Port** is the port number to use for the Directory Server instance. The regular LDAP port is **389**; the default LDAPS (secure) port number is **636**.
  - Check the **Secure Connection** check box to use the secure port. Before checking this box, make sure that the Configuration Directory Server has enabled SSL.
5. Click **Save**.

### E.2.10.2. Changing the User Directory Host or Port

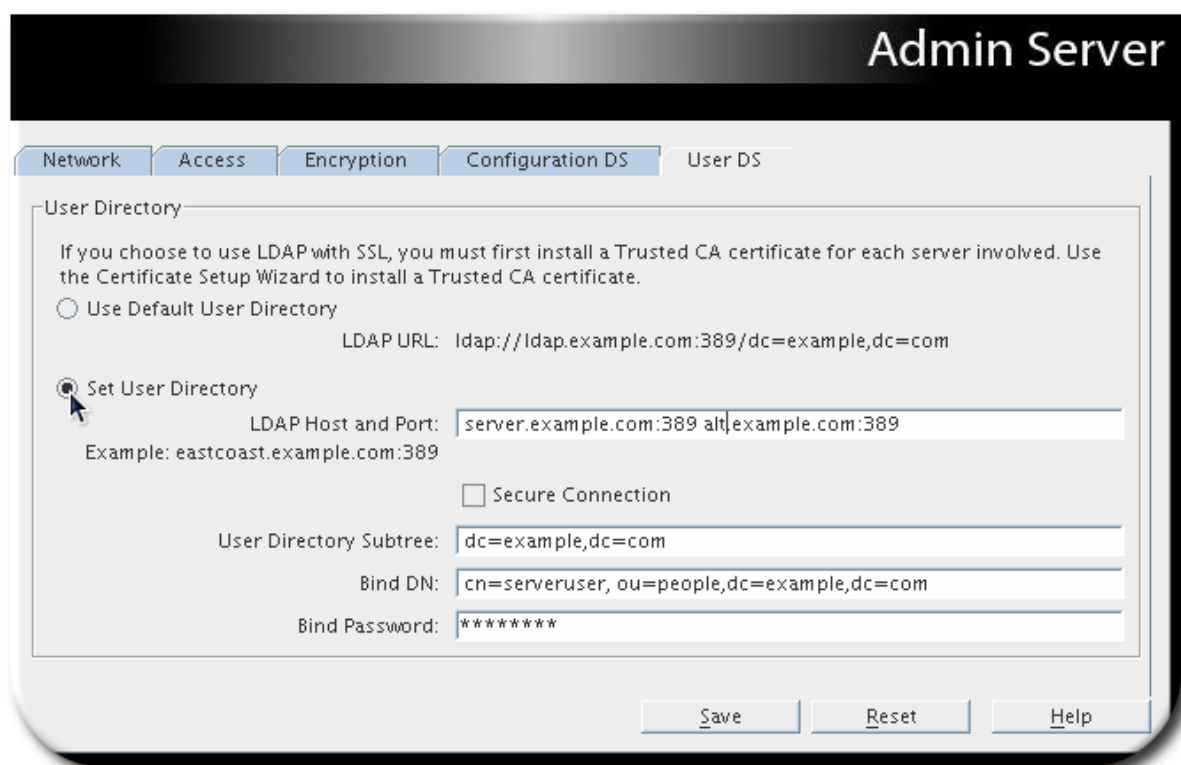
The user directory is used for authentication, user management, and access control. It stores all user and group data, account data, group lists, and access control instructions (ACIs).

There can be multiple user directories in a single deployment because using multiple user directories enhances overall performance for organizations which are geographically spread out, which have high usage, or have discrete divisions which benefit from individual directories.

Admin Server can be configured to authenticate users against multiple user directories.

To change the information for the user directory:

1. Open the Admin Server management window.
2. Click the **Configuration** tab.
3. Click the **User DS** tab.
4. Set the User Directory Server connection information.
5. Edit the user directory information.



The **Use Default User Directory** radio button uses the default user directory associated with the domain. To use multiple Directory Server instances or to use a different instance, select the **Set User Directory** radio button and set the required information:

- o The **LDAP Host and Port** field specifies the location of the user directory instance, using the format *hostname:port* or *ip\_address:port*, with an IPv4 or IPv6 address.

It is possible to configure multiple locations for the user directory for authentication and other directory functions; separate each location with a space. For example:

```
server.example.com:389 alt.example.com:389
```

**NOTE**

If more than one location is given in the **LDAP Host and Port** field, the settings for the remaining fields will apply to all of those instances.

- Check the **Secure Connection** box to use SSL to connect to the user directory. *Only* select this if the Directory Server is already configured to use SSL.
- Give the **User Directory Subtree**. For example:

```
dc=example,dc=com
```

Every location listed in the **LDAP Host and Port** field must contain that subtree and the subtree must contain the user information.

- Optionally, enter the **Bind DN** and **Bind Password** for the user which connects to the user directory.

6. Click **Save**.

## APPENDIX F. USING ADMIN EXPRESS

### F.1. MANAGING SERVERS IN ADMIN EXPRESS

Admin Express provides a quick, simple web-based gateway to do basic management of servers. There are three tasks that can be performed through Admin Express:

- Stopping and starting the server
- Checking the server access, error, and audit logs
- Monitoring the progress and information for replication between Directory Servers

#### F.1.1. Opening Admin Express

The Admin Server services pages URL is the Admin Server host (host name, IPv4 address, or IPv6 address) and port. For example:

```
http://ldap.example.com:9830/
```

The Admin Express page is always available at that URL.



#### NOTE

If SSL/TLS is enabled on the Admin Server, then the URL must use the prefix **https** with the same port number. The standard HTTP URLs will not work.

```
https://ldap.example.com:9830/
```

#### F.1.2. Starting and Stopping Servers

On the main Admin Express page, there are buttons to turn servers off and on.



Figure F.1. Stopping and Starting Servers

## IMPORTANT

If either the Admin Server or the Configuration Directory Server is turned off through the Admin Express page, then it must be restarted through the command line, not through the Admin Express **On/Off** buttons because Admin Express requires access to both the Admin Server and Configuration Directory Server in order to function.

Other Directory Server instances can be safely stopped and restarted through Admin Express.

### F.1.3. Viewing Server Logs

Admin Express can show and search the access and error logs for Directory Server and Admin Server and the audit logs for the Directory Server.

1. In the Admin Express page, click the **Logs** link by the server name.
2. Select which log type to view, how many lines to return, and any string to search for, and click **OK**.

Fedora Administration Express

### admin-serv Logs

Log to view:

Number of entries:

Only show entries with:

Last 25 accesses to access with bin:

```

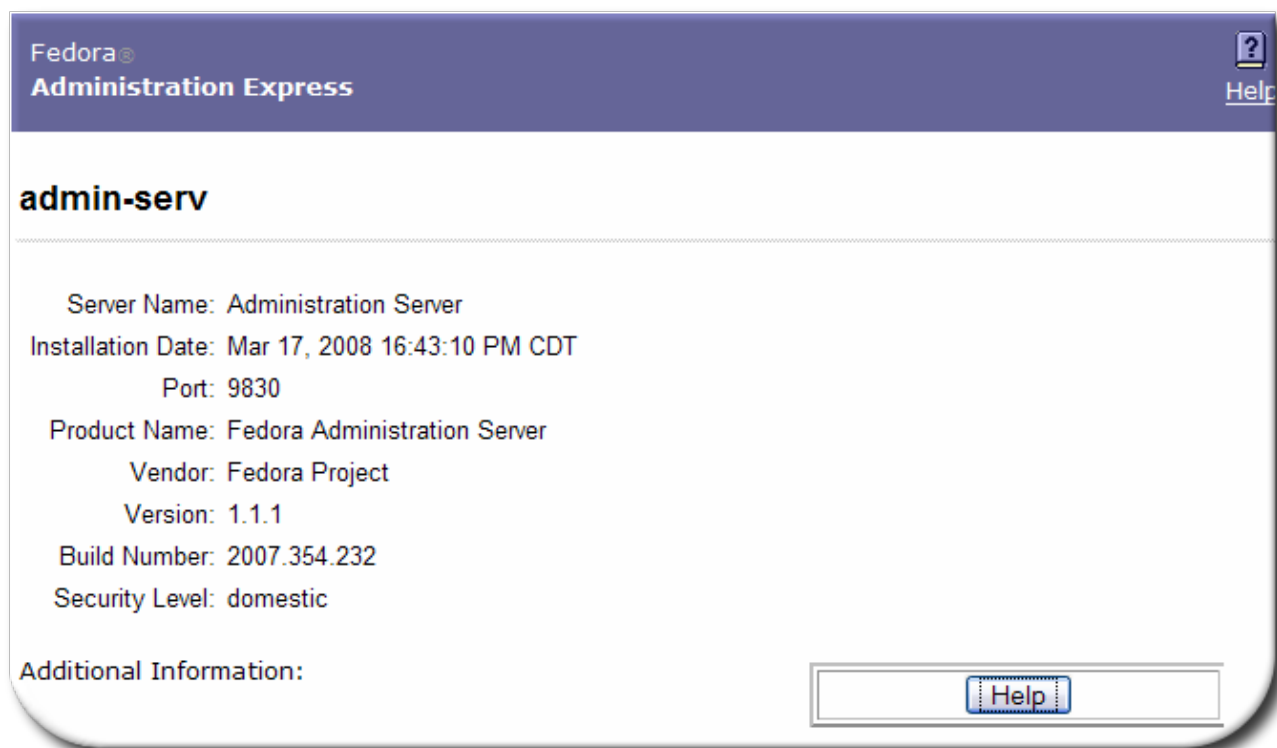
192.168.123.122 - - [17/Apr/2008:12:06:50 -0500] "GET /bin/admin/admin/bin/download HTTP/1.1" 200 6294
192.168.123.122 - - [17/Apr/2008:12:06:51 -0500] "GET /clients/dsgw/bin/lang?context=pb HTTP/1.1" 200 3020
192.168.123.122 - - [17/Apr/2008:12:07:51 -0500] "GET /bin/admin/admin/bin/download HTTP/1.1" 200 6294
192.168.123.122 - - [17/Apr/2008:12:07:52 -0500] "GET /clients/dsgw/bin/lang?context=pb HTTP/1.1" 200 3020
192.168.123.122 - - [17/Apr/2008:12:21:31 -0500] "GET /bin/admin/admin/bin/download HTTP/1.1" 200 6294
192.168.123.122 - - [17/Apr/2008:12:21:33 -0500] "GET /clients/dsgw/bin/lang?context=pb HTTP/1.1" 200 3020
192.168.123.122 - - [17/Apr/2008:12:22:03 -0500] "GET /bin/admin/admin/bin/download HTTP/1.1" 200 6294
192.168.123.122 - - [17/Apr/2008:12:22:05 -0500] "GET /clients/dsgw/bin/lang?context=pb HTTP/1.1" 200 3020
192.168.123.122 - - [17/Apr/2008:12:29:42 -0500] "GET /clients/dsgw/bin/lang?context=dsgw HTTP/1.1" 200 2906
192.168.123.122 - - [17/Apr/2008:12:32:28 -0500] "GET /clients/dsgw/bin/lang?context=pb HTTP/1.1" 200 3020

```

Figure F.2. Checking Logs

### F.1.4. Viewing Server Information

The **Server Info** link on the Admin Express page opens a page with the basic description of the server instance, such as the build number, installation date, and server port number. This is the same information displayed in the Console when an instance is selected.



**Figure F.3. Checking Server Information**

The Directory Server information is located in the `/etc/dirsrv/slaped-instance_name/dse.ldif` file; the Admin Server information is located in `.conf` files in the `/etc/dirsrv/admin-serv` directory.

## F.2. CONFIGURING ADMIN EXPRESS

Admin Express can be edited for the page appearance, but most functionality is controlled through the web server or the Admin Server configuration and should be edited through those servers, not by editing the configuration files directly.

### F.2.1. Admin Express File Locations

The directories for all of the Admin Express configuration files are listed in [Table F.1, “Admin Express File Directories”](#); the specific files are described in each section describing the different Admin Express page configurations.

**Table F.1. Admin Express File Directories**

| Directory                            | Description                                                                                                                                    |
|--------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>/etc/dirsrv/admin-serv</code>  | Contains the <b>local.conf</b> , <b>httpd.conf</b> , and other configuration files which define the Admin Server and configure the web server. |
| <code>/usr/share/dirsrv/html/</code> | Contains the HTML files and graphics used for the Admin Express appearance.                                                                    |

### F.2.2. Admin Express Configuration Files

The behavior for Admin Express is mostly set through the web server configuration and should not be edited. The other Admin Express configuration is set through directives which insert data or form fields.

There is not cascading style sheet (CSS) file to centralize the formatting for pages in Admin Express. All formatting is done inline with the tags or through `<style>` tags in the page head. For information on editing inline tags, see <http://directory.fedoraproject.org/docs/389ds/administration/htmlediting.html>.

### F.2.2.1. Files for the Admin Server Welcome Page

The configuration files for the introductory page for Admin Express is located in the `/etc/dirsrv/admin-serv` directory. One file sets the formatting, copyright text, and some web application text, `admserv.html`.

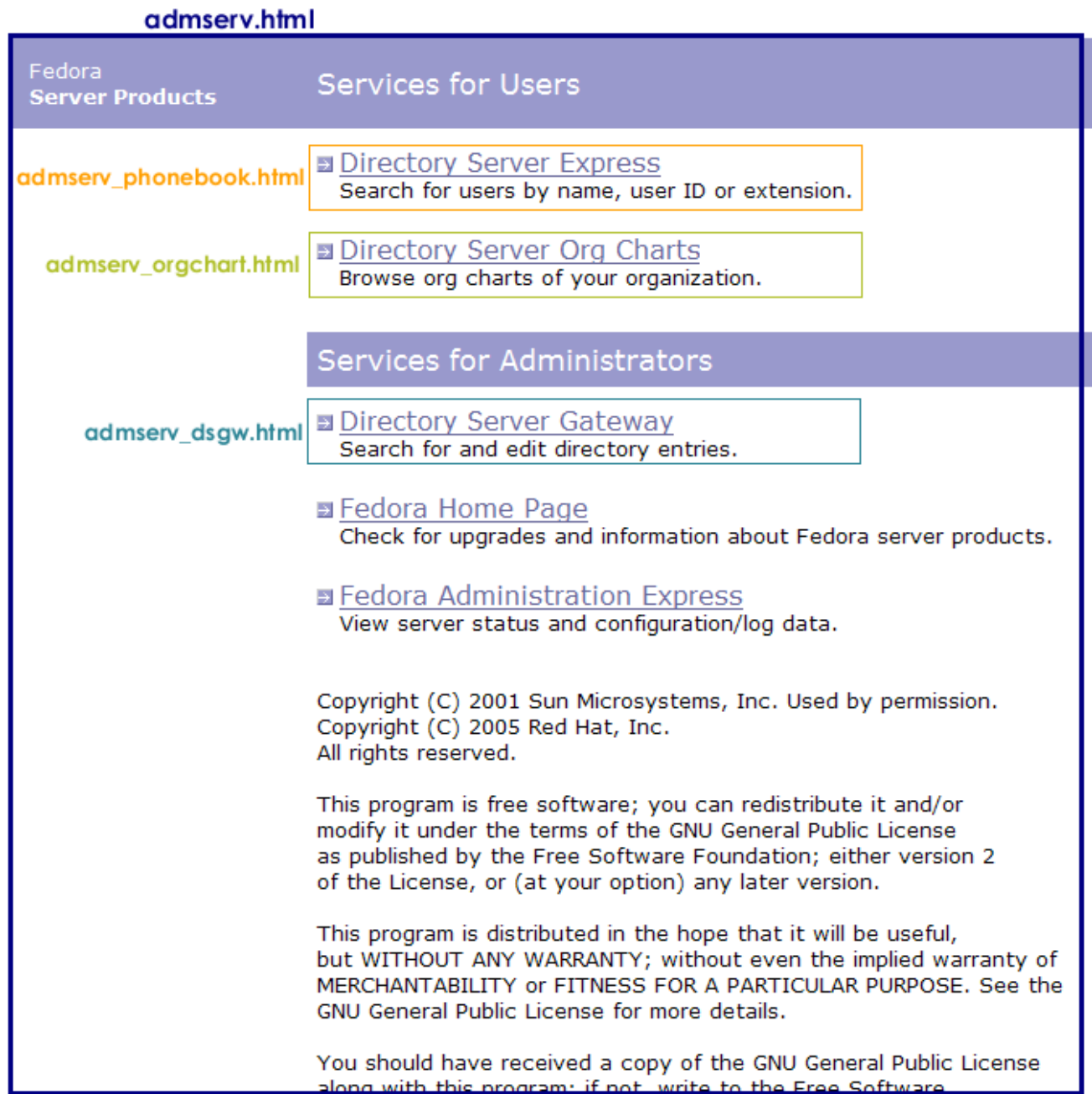


Figure F.4. Intro Page Elements

All of the formatting for the page is set inline. The text files are inserted using the `INCLUDEIFEXISTS` directive.

```

<tr valign="TOP">
 <td> </td>
 <td bgcolor="#9999cc" colspan="4"> Services
 for Administrators</td>
 <td> </td>
</tr>
<tr valign="TOP">
 <td> </td>
 <td colspan="4">
 <table border="0" cellspacing="0" cellpadding="0">
 <tr valign="TOP">
 <td></td>
 <td></td>
 </tr>
 </table>
 </td>
</tr>
<!-- INCLUDEIFEXISTS admserv_dsgw.html -->

```

The text files themselves have inline formatting for the inserted table rows.

### F.2.2.2. Files for the Replication Status Appearance

There are two pages for monitoring the replication status. The first is for the configuration page, which requires two files:

- The body of the page, `/usr/share/dirsrv/html/monreplication.html`
- The heading of the page, `/usr/share/dirsrv/html/htmladmin.html`

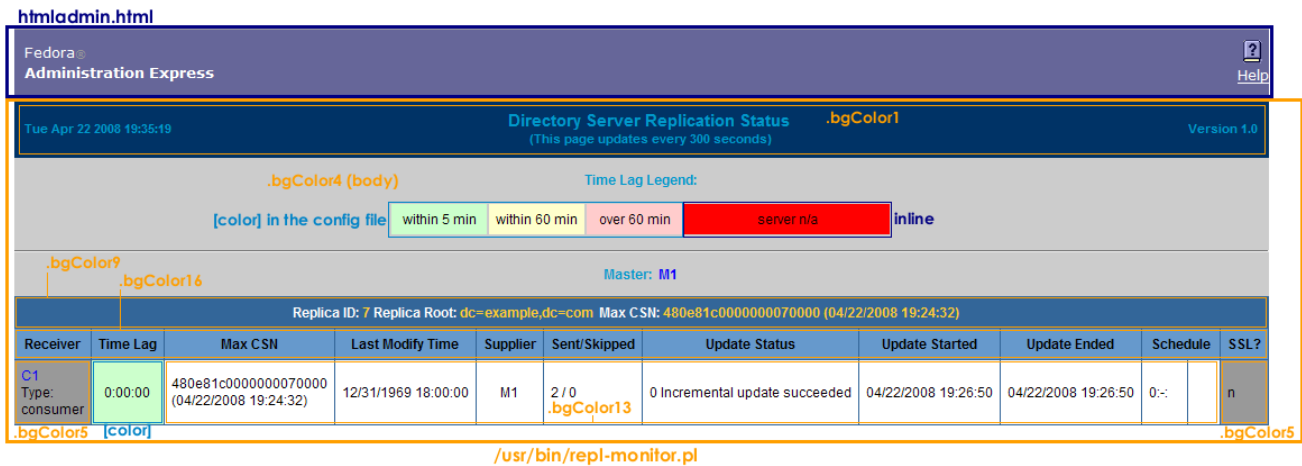


Figure F.5. Monitoring Replication Setup Page Elements

The **Replication Status** page uses two script-related configuration files:

- The body of the page, which is configured in the replication monitoring script, `/usr/bin/repl-monitor.pl`
- Optionally, the configuration file for the replication monitoring, which can configure the time lag colors with the `[colors]` section
- The heading of the page, `/usr/share/dirsrv/html/htmladmin.html`





**Figure F.6. Monitoring Replication View Page Elements**

The text for the table headings, labels, and page sections are set in the Perl script. For example:

```
#Print the header of consumer
print "\n<tr class=bgColor16>\n";
print "<th nowrap>Receiver</th>\n";
print "<th nowrap>Time Lag</th>\n";
print "<th nowrap>Max CSN</th>\n";
....
print "</tr>\n";
```

The styles for the **Replication Status** page are printed in the Perl script in the <style> tag in the HTML header. Many of the classes are the same as those in the **style.css** for the other web applications. These can be edited in the Perl script or by uncommenting the stylesheet reference and supplying a CSS file. For example:

```
print the HTML header

print "Content-type: text/html\n\n";
print "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2//EN\"><html>\n";
print "<head><title>Replication Status</title>\n";
print "<link type=text/css rel=stylesheet href=\"master-
style.css\">\n";
print "<style text/css>\n";
print "Body, p, table, td, ul, li {color: #000000; font-family: Arial,
Helvetica, sans-serif; font-size: 12px;}\n";
print "A {color:blue; text-decoration: none;}\n";
print "BODY {font-family: Arial, Helvetica, sans-serif}\n";
print "P {font-family: Arial, Helvetica, sans-serif}\n";
print "TH {font-weight: bold; font-family: Arial, Helvetica, sans-
serif}\n";
print "TD {font-family: Arial, Helvetica, sans-serif}\n";
print ".bgColor1 {background-color: #003366;}\n";
print ".bgColor4 {background-color: #cccccc;}\n";
print ".bgColor5 {background-color: #999999;}\n";
print ".bgColor9 {background-color: #336699;}\n";
print ".bgColor13 {background-color: #ffffff;}\n";
print ".bgColor16 {background-color: #6699cc;}\n";
print ".text8 {color: #0099cc; font-size: 11px; font-weight:
bold;}\n";
print ".text28 {color: #ffcc33; font-size: 12px; font-weight:
```

```

bold;}\n";
 print ".areatitle {font-weight: bold; color: #ffffff; font-family:
Arial, Helvetica, sans-serif}\n";
 print ".page-title {font-weight: bold; font-size: larger; font-family:
Arial, Helvetica, sans-serif}\n";
 print ".page-subtitle {font-weight: bold; font-family: Arial,
Helvetica, sans-serif}\n";

 print "</style></head>\n<body class=bgColor4>\n";

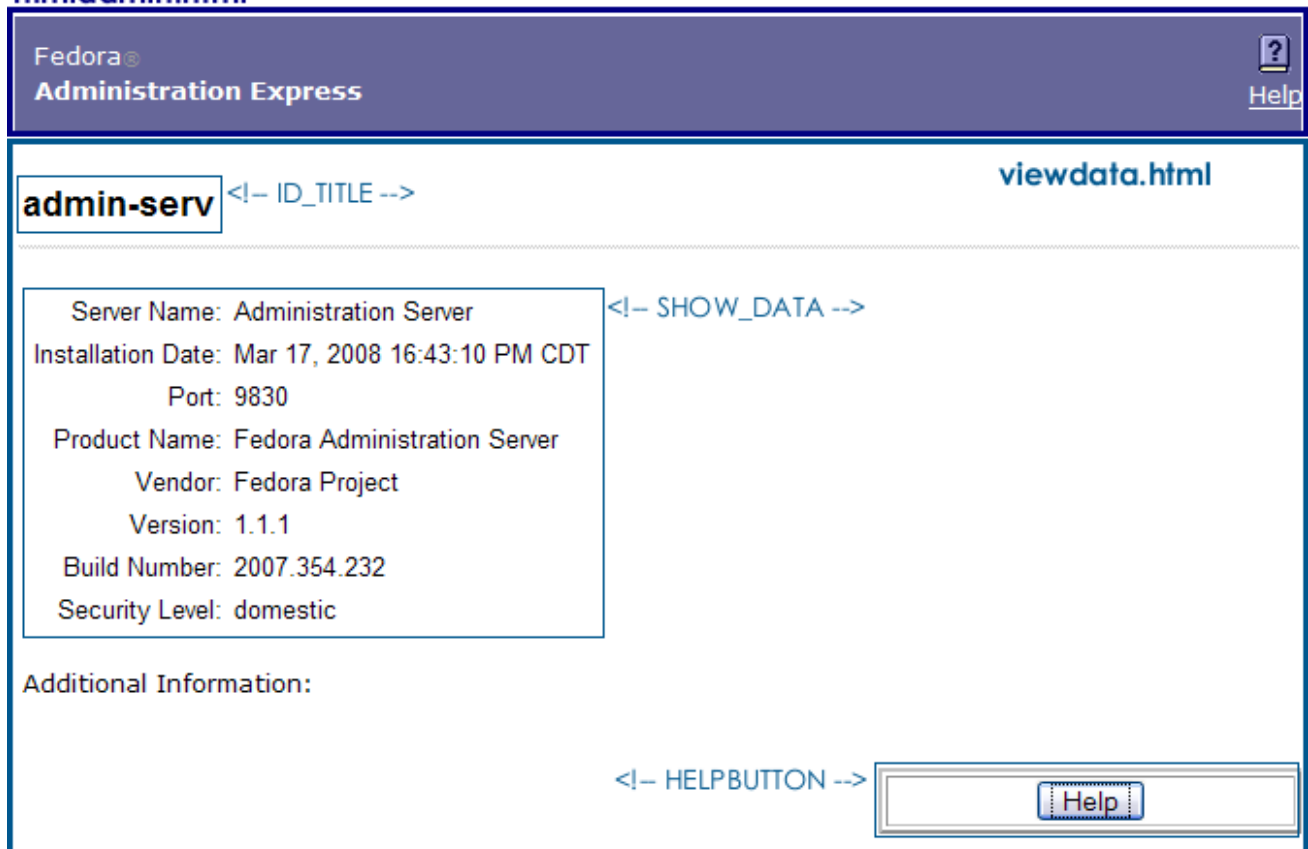
```

### F.2.2.3. Files for the Server Information Page

There are two files formatting the server information page:

- The body of the page, `/usr/share/dirsrv/html/viewdata.html`
- The heading of the page, `/usr/share/dirsrv/html/htmladmin.html`

#### htmladmin.html



**Figure F.7. Server Information Page Elements**

The `viewdata.html` file is very simple, using only the two directives to insert the server data, plus other directives to insert other information. For the Admin Server, the `SHOW_DATA` directive takes the information from the `/etc/dirsrv/admin-serv/local.conf` file. For the Directory Server, it takes the data from the `/etc/dirsrv/slapped-instance_name/dse.ldif` file. The `ID_TITLE` is the name of the server instance.

```

<body text="#000000" bgcolor="#FFFFFF" link="#666699" vlink="#666699"
alink="#333366">


```

```

<table BORDER=0 CELSPACING=2 CELLPADDING=2 WIDTH="100%">
<!-- ID_TITLE -->
<p>
<!-- SHOW_DATA -->
<p>
Additional
Information:
<p>
<!-- CHECK_UPGRADE -->
<p>
<!-- SHOW_URL -->
</table>

<!-- HELPBUTTON -->

</body>

```

### F.2.2.4. Files for the Server Logs Page

There are two files formatting the server logs page:

- The body of the page, `/usr/share/dirsrv/html/viewlog.html`
- The heading of the page, `/usr/share/dirsrv/html/htmladmin.html`

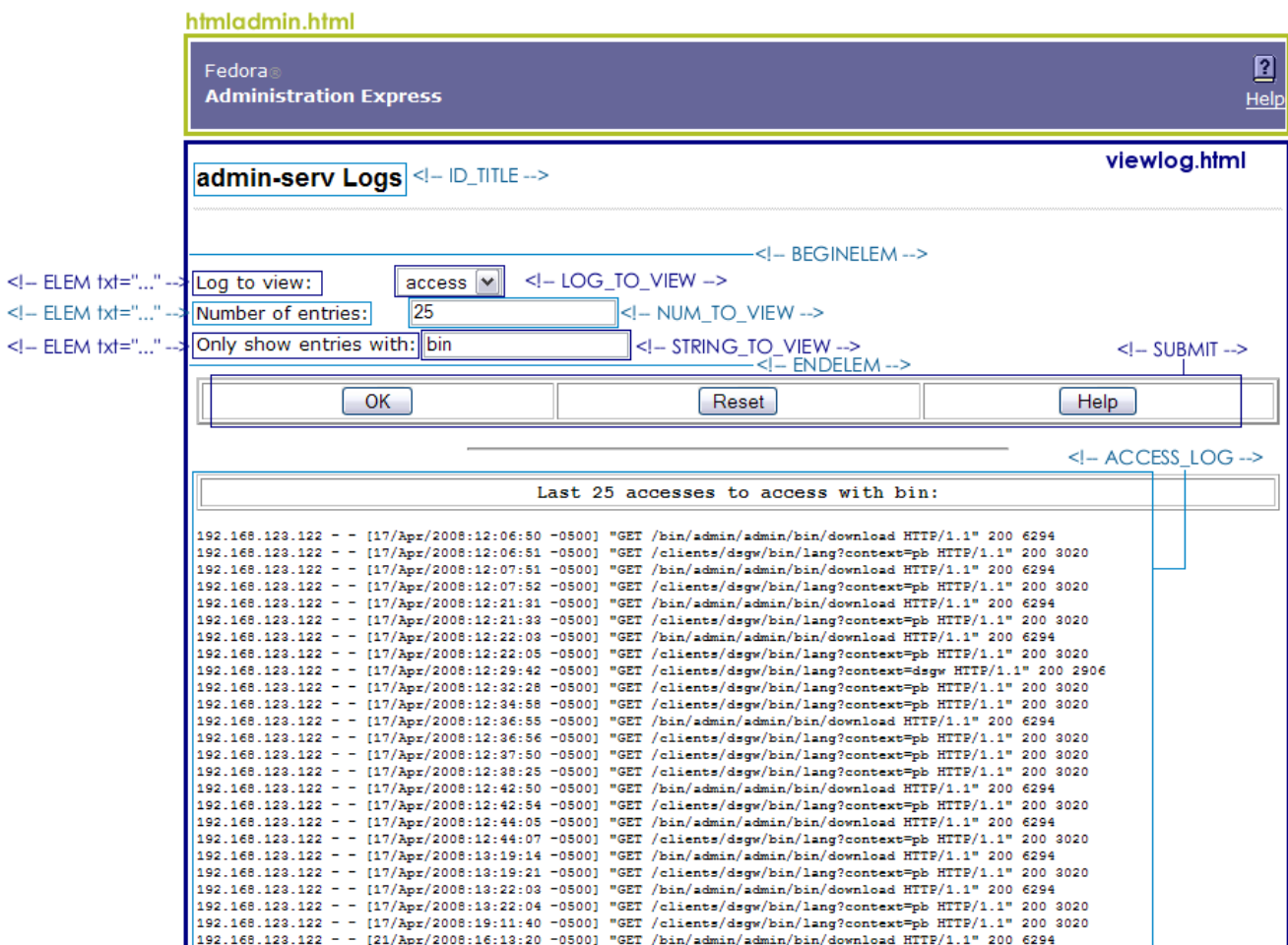


Figure F.8. Log View Page Elements

The page information is set through the inserted directives. The server instance name is set in the

**ID\_TITLE** directive. The log is displayed through the **ACCESS\_LOG** directives. The form at the top is formatted with directive pairs, one which sets the descriptive text and the other inserting the field type. For example, this sets the log type menu:

```
<form method=GET action=ViewLog>

<!-- BEGINLEM -->
<!-- ELEM txt="Log to view: " -->
<!-- LOG_TO_VIEW -->
....
<!-- SUBMIT -->

</form>
```

### F.2.3. Admin Express Directives

The Admin Express directives are HTML comments that are interpreted by the CGI scripts; these directives are used to set form fields and to pull data from the server configuration and log files.

**Table F.2. Admin Express Directives**

Directive	Description	Example
ACCESS_LOG	Inserts the server log file.	<!-- ACCESS_LOG -->
ADMURL		<!-- ADMURL -->
BEGINLEM	Marks the opening of form input elements. This is always paired with <b>ENDELEM</b> .	<!-- BEGINLEM -->
CHECK_UPGRADE		<!-- CHECK_UPGRADE -->
ELEM	Inserts a text element. This has one argument, <b>txt=</b> , which defines the text to use.	<!-- ELEM txt="Field name here: " -->
ELEMADD	Inserts a text element. This has one argument, <b>txt=</b> , which defines the text to use.	<!-- ELEMADD txt="Field name here: " -->
ENDELEM	Marks the ending of form input elements. This is always paired with <b>BEGINLEM</b> .	<!-- ENDELEM -->
HELP_BUTTON	Inserts a button to open context-specific help.	<!-- HELP_BUTTON -->
HELPLINK	Inserts a link to the general Admin Express help file.	<!-- HELPLINK -->

Directive	Description	Example
HIDDEN_ID		<!-- HIDDEN_ID -->
ID_TITLE	Inserts the name of the server instance, such as <b>admin-serv</b> or <b>example</b> (if the Directory Server instance name is <b>slapd-example</b> )	<!-- ID_TITLE -->
INCLUDEIFEXISTS	Inserts the contents of the HTML file. The inserted file should include both the text and any HTML markup.	<!-- INCLUDEIFEXISTS "file.html" -->
LOG_TO_VIEW	Inserts a drop-down menu with the types of logs available to view.	<!-- LOG_TO_VIEW -->
NUM_TO_VIEW	Inserts a form field to set the number of lines to return.	<!-- NUM_TO_VIEW -->
REFRESHINTERVAL	Inserts a form field to set the refresh interval (in seconds) for replication monitoring.	<!-- REFRESHINTERVAL -->
SERVHOST		<!-- SERVHOST -->
SERVPORT		<!-- SERVPORT -->
SHOW_DATA	Inserts the server data from the configuration file, including the port number, installation date, and build number.	<!-- SHOW_DATA -->
SHOW_URL		<!-- SHOW_URL -->
SITEROOT		<!-- SITEROOT -->
STRING_TO_VIEW	Inserts a form field to use to set the search string for the logs.	<!-- STRING_TO_VIEW -->
SUBMIT	Inserts a three-button set: to save or submit the form; to reset the form; and to open a help topic.	<!-- SUBMIT -->

## APPENDIX G. USING THE CONSOLE

### G.1. OVERVIEW OF THE DIRECTORY SERVER CONSOLE

Red Hat Management Console is the user interface to manage Red Hat Directory Server and Admin Server configuration and directory information. There is a single main Console window which administers the servers (collected and identified in *administration domains*). The main Console allows you to open server-specific Consoles to manage the settings and information in individual instances.

This chapter provides an overview of how the Console interacts with the Directory Server and Admin Server and walks through the Console windows and options.

#### G.1.1. How the Console, Directory Server, and Admin Server Work Together

The Red Hat Console is an independent Java application which works in conjunction with instances of Red Hat Directory Server and Admin Server. Most server management functions are carried out in server-specific console windows for the Directory Server and Admin Server. Red Hat Console is part of a system that manages Red Hat Directory Server instances and the Admin Server and, therefore, information in the directory. Although Red Hat Directory Server, Red Hat Management Console, and Red Hat Admin Server work tightly with one another, each plays a specific role in managing servers, applications, and users.

Red Hat Management Console is the front-end management application for Red Hat Directory Server. It finds all servers and applications registered in the configuration directory, displays them in a graphical interface, and can manage and configure them. The Main Console can also search for, create, and edit user and group entries in the user directory.

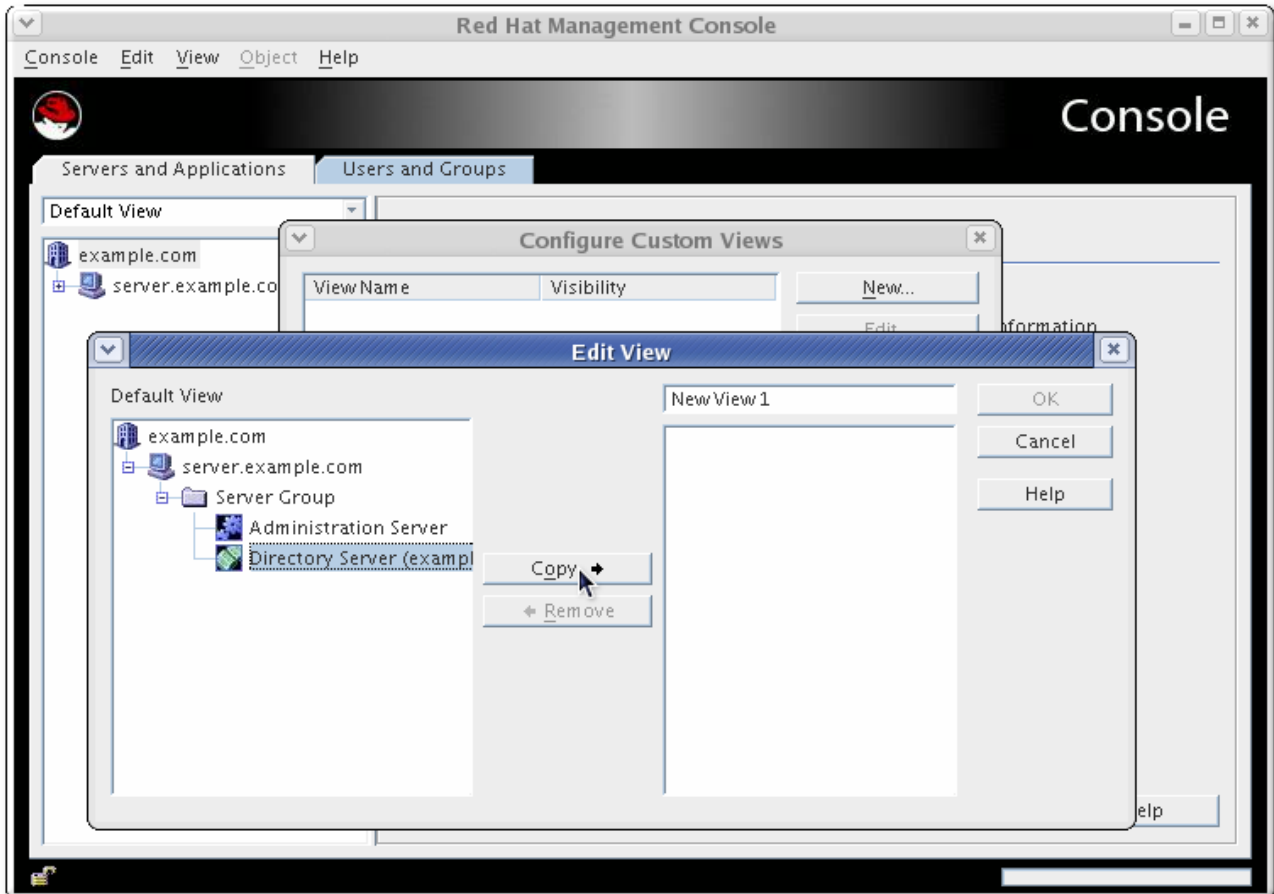
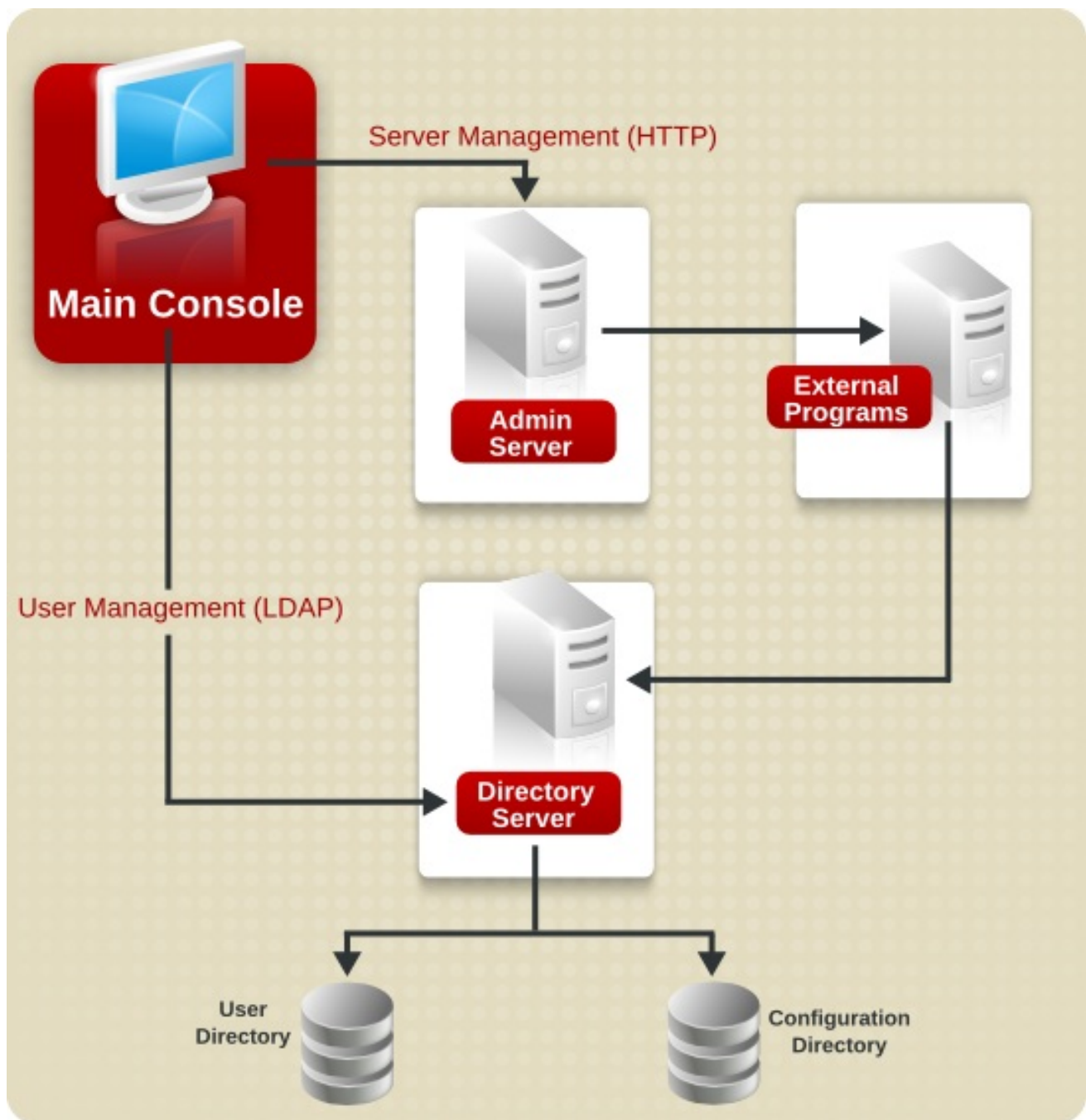


Figure G.1. The Red Hat Management Console Interface

When a user logs into Red Hat Management Console, the Console connects to the Admin Server over Hypertext Transfer Protocol (HTTP). The Admin Server receives requests to administer the different Directory Server instances and performs the changes to the configuration, such as changing a port number. When a request is sent to the Red Hat Management Console to add or edit user entries, the Console sends a Lightweight Directory Access Protocol (LDAP) message directly to Directory Server to update the user directory.



**Figure G.2. Simple System Using Red Hat Management Console**

Red Hat Directory Server stores server and application configuration settings as well as user information. Typically, application and server configuration information is stored in one subtree of Red Hat Directory Server while user and group entries are stored in another subtree. With a large enterprise, however, configuration and user information can be stored in separate *instances* of Directory Server (which can be on the same host machine or on two different host machines). [Figure G.2, “Simple System Using Red Hat Management Console”](#) illustrates a relatively simple Red Hat Directory Server system. As an enterprise grows and needs change, additional hosts and Directory and Admin Servers can be added to the administration domain in the Console, so that a single Console can manage multiple Directory and Admin Servers.



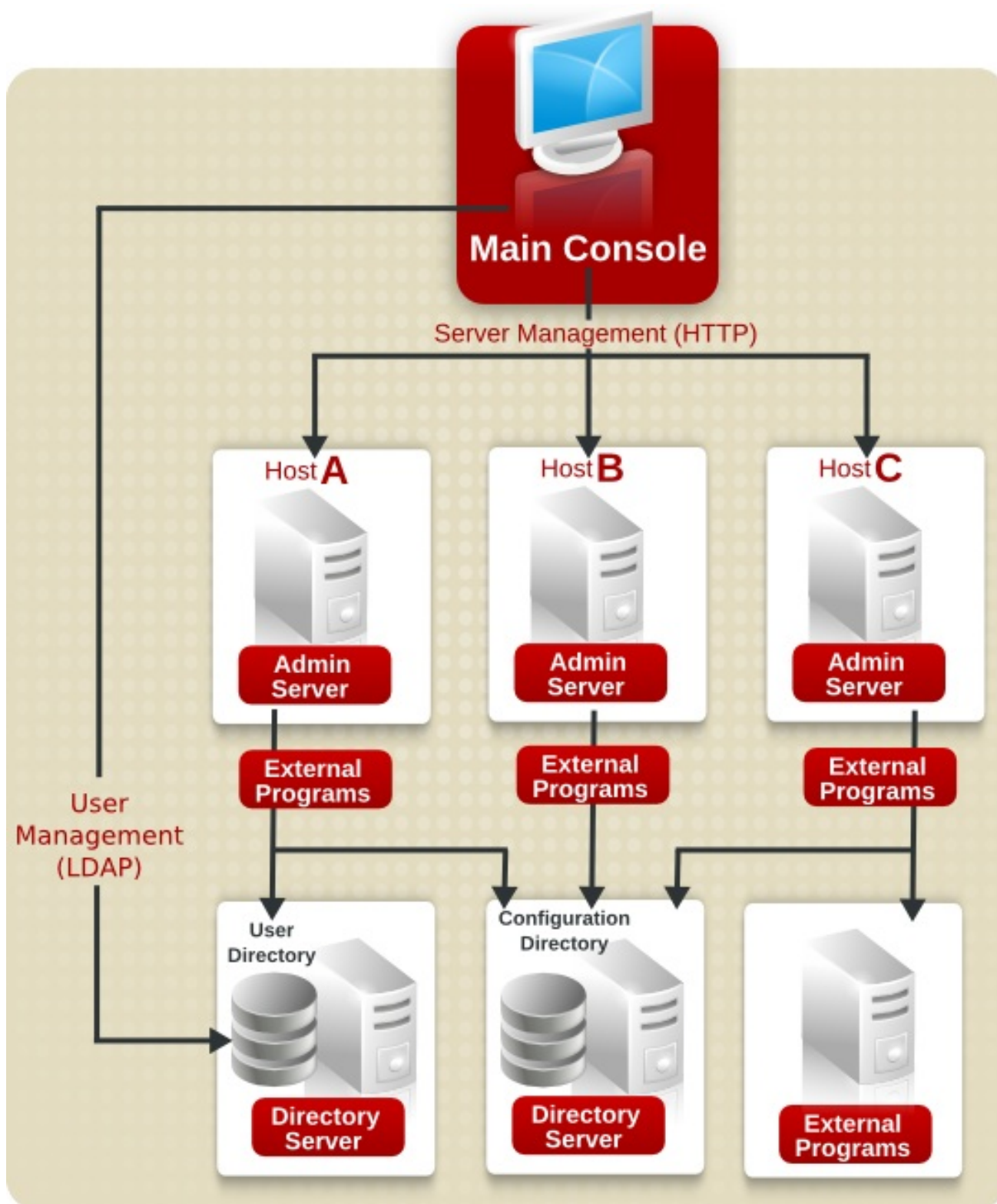


Figure G.3. A More Complex System



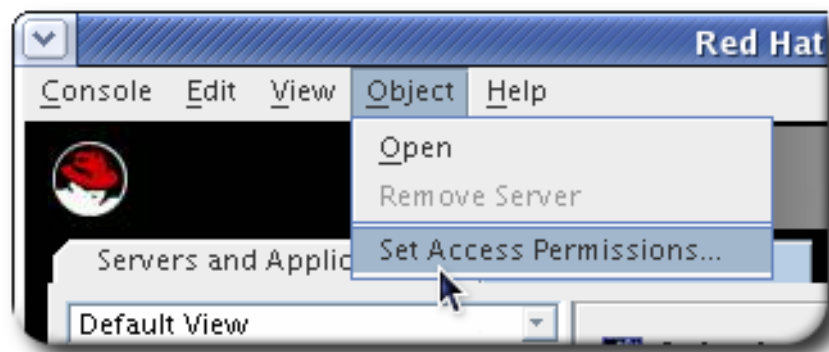
#### NOTE

When the terms *configuration directory* and *user directory* are used in this guide, they define where the configuration information and the user information is stored, regardless of whether that is in the subtrees of a single instance of Directory Server or in two separate instances of Directory Server.

### G.1.2. Red Hat Management Console Menus



There are five menu items in the top menu the Console. The options for each of these menus varies depending on the Console window open (the main Console, Directory Server Console, or Admin Server Console) and the types of objects available in that server area.



**Figure G.4. Main Console Menus**

**Table G.1. Console Menus**

Menu	Description
Console	<p>Manages the Console session, such as closing the window or exiting the session entirely.</p> <ul style="list-style-type: none"> <li>• For the main window, this menu also can be used to add and remove admin domain.</li> <li>• For the Directory Server Console, this allows people to log in as a different user.</li> <li>• For the Admin Server Console, it manages security issues, such as certificates and tokens.</li> </ul>
Edit	<p>Sets display preferences, for all three Consoles. For the Directory Server Console, this also provides ways to copy, paste, and delete directory entries or text.</p>
View	<p>Sets whether to display certain parts of the Console window, such as the top banner, menus, and side navigation panes. This also refreshes the current display. For the Directory Server Console, this menu also sets what parts of the directory or which databases to view.</p>

Menu	Description
Object	<p>Provides available operations for the active object; this is the same as the right-click menu for the active area or entry.</p> <ul style="list-style-type: none"> <li>• For the main window, this menu simply opens or deletes a server instance.</li> <li>• For the Directory Server Console, this provides all of the configuration options for the directory entries, such as advanced property editors or creating new entries.</li> <li>• For the Admin Server Console, this opens a configuration editor, starts, and stops the server.</li> </ul>
Help	Opens context-specific help for the current Console area.

### G.1.3. Red Hat Management Console Tabs

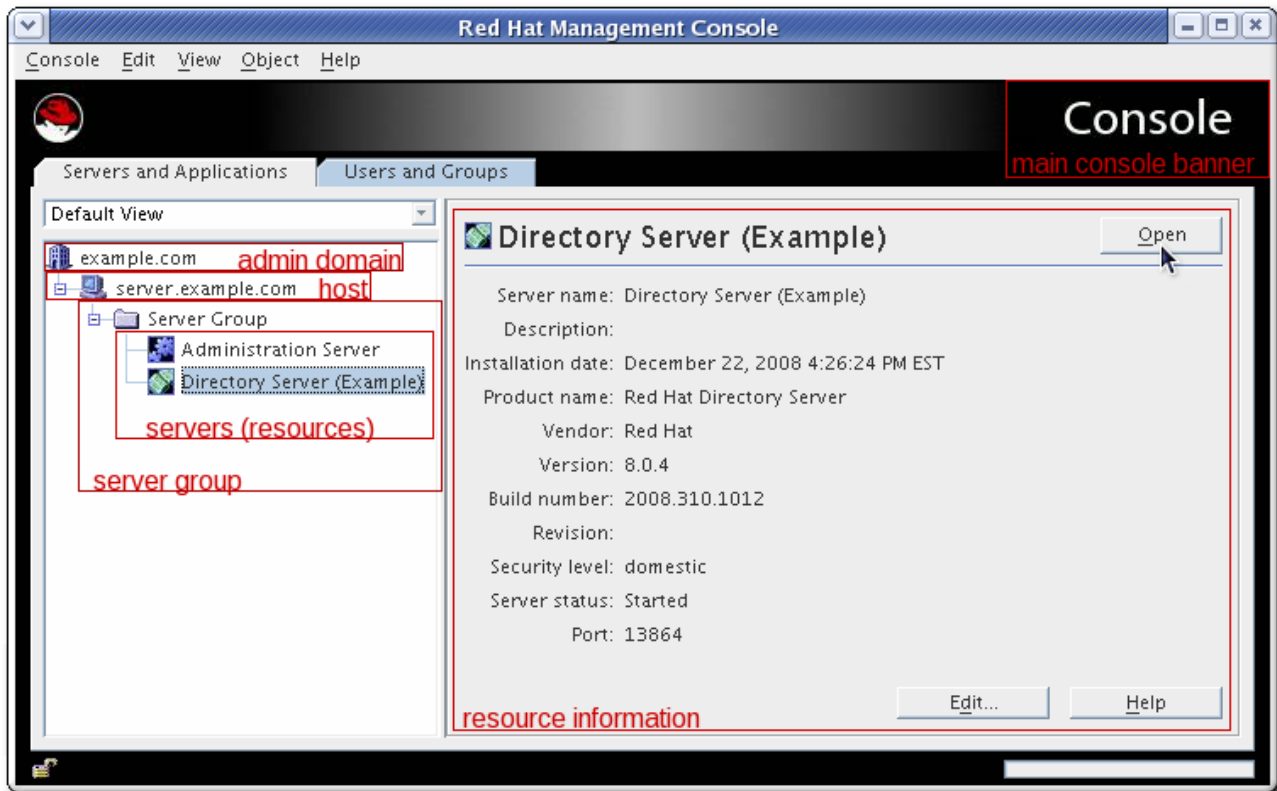
There are two tabs in the main Console window:

- **Servers and Applications**, for managing the Directory Server and Admin Server instances
- **Users and Groups**, for searching for and creating user and group entries within the Directory Server

#### G.1.3.1. The Servers and Applications Tab

The **Servers and Applications** tab, by default, has a navigation tree on the left for viewing hosts and Directory and Admin Servers and a center information panel. To access the Directory Server instance, directory information, or Admin Server, open the server resource listed in the navigation tree. The information for the server instance, such as the build number and port number,

The navigation tree displays the Red Hat Directory Server *topology*, a hierarchical representation of all the resources (such as servers and hosts), that are registered in a configuration directory.



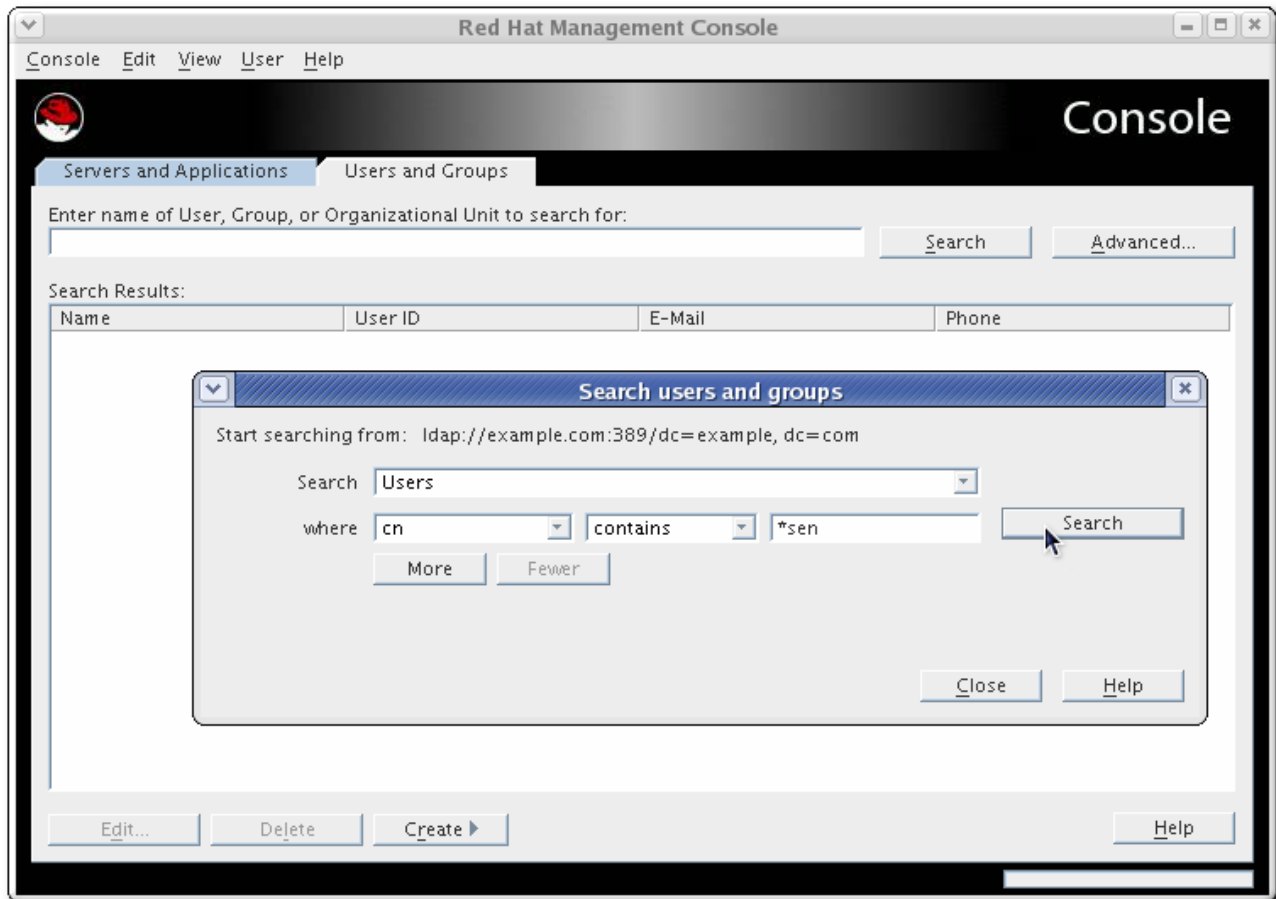
**Figure G.5. The Servers and Applications Tab**

The top of the topology is the *administration domain*. An administration domain is a collection of host systems and servers that share the same user directory. The server which hosts Directory Server or Admin Server instances belongs to the admin domain; that is the *host*.

A *server group* consists of all Directory Servers that are managed by a common Admin Server. A number of server groups can exist within an administration domain.

### G.1.3.2. The Users and Groups Tab

The **Users and Groups** tab can search for user and group entries in any Directory Server administered by the Console. Any of the returned entries can be edited or deleted through this tab, assuming that the user has the proper access permissions. New entries can also be created through the **Users and Groups** tab.



**Figure G.6. The Users and Groups Tab**

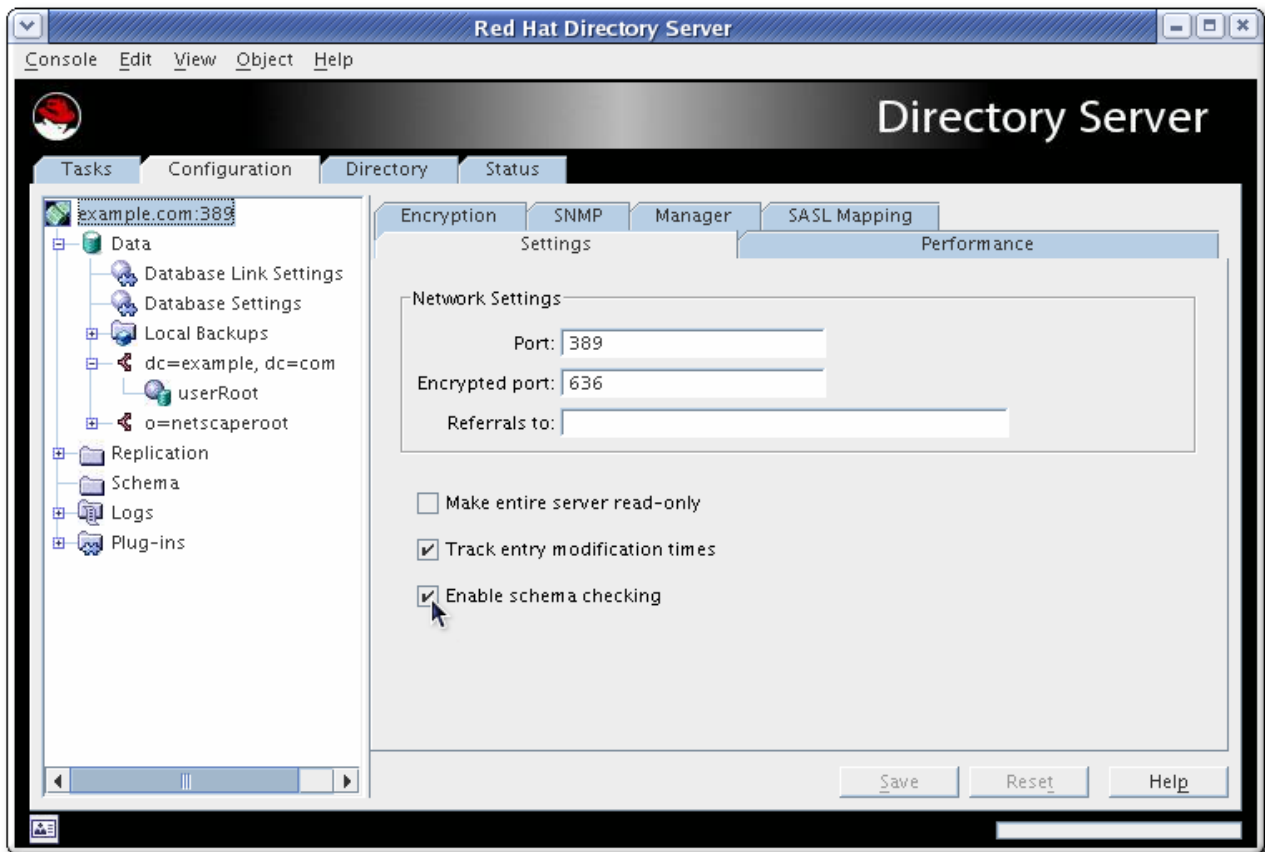
Switch the directory being searched or where the entries are added through the options in the **Users** menu, as described in [Section G.4.1, “Searching for Users and Groups”](#).

### G.1.4. Server-Specific Consoles

The main Console can open into two server-specific windows to manage the Admin Server and Directory Server. These windows are opened by clicking the server name in the navigation area, and then clicking the **Open** button in the resources area.

#### G.1.4.1. The Directory Server Console

The Directory Server Console manages the specific Directory Server instance configuration, including the port number, SSL settings, and logging. The Directory Server Console also manages the directory information (entries) and directory operations like importing and exporting databases, creating suffixes, and extending the schema.



**Figure G.7. The Directory Server Console**

There are four tabs in the Directory Server Console:

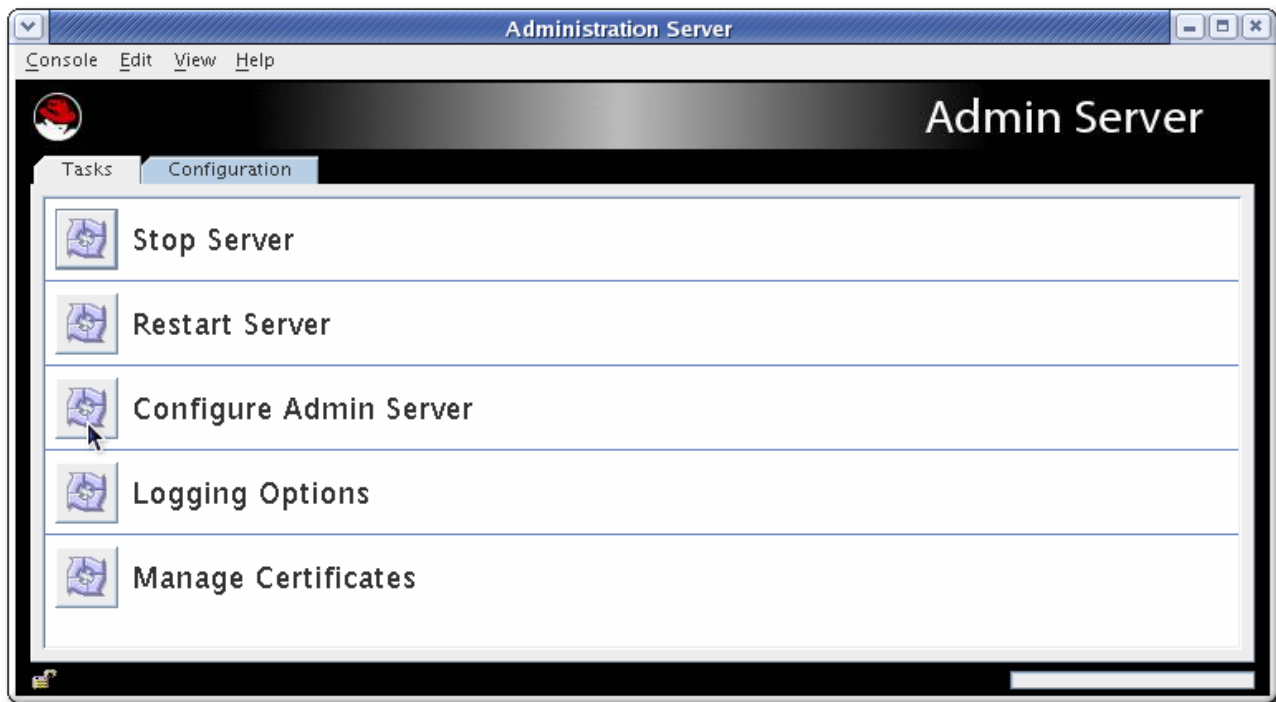
- **Tasks**, which has shortcuts to common server operations, including starting and stopping the Directory Server instance, importing and exporting databases, and managing SSL certificates
- **Configuration**, which defines all of the server configuration settings, including SASL and SSL authentication, port numbers, schema, replication and synchronization, databases and suffixes, logging, and plug-ins
- **Directory**, which access and manages the directory information, including user entries and all group entries, including roles, classes of service, views, and groups
- **Status**, which monitors the server performance and displays the different monitoring and performance counters for the Directory Server and databases

Similar to the main Console, the Directory Server Console tabs have a navigation area on the left and a center panel that displays information about the active setting, entry, or database.

The procedures for using the Directory Server Console to manage the Directory Server configuration and directory entries is covered in the *Directory Server Administrator's Guide*.

#### G.1.4.2. The Admin Server Console

The Admin Server itself administers the configuration of other servers, especially the configuration and user directories for the server group. The Admin Server Console manages the Admin Server settings and the settings for these two Directory Server directories; whenever the settings are changed in the Directory Server configuration, the modifications must be carried into the Admin Server configuration for the server to properly manage those servers.



**Figure G.8. The Admin Server Console**

The Admin Server Console is simpler than the Directory Server Console, with only two tabs:

- **Tasks**, which has shortcuts to common server operations, including starting and stopping the Admin Server instance, setting up logging, and managing SSL certificates
- **Configuration**, which defines all of the Admin Server configuration settings, including SSL authentication, port numbers, and logging, as well as the Configuration Directory Server and User Directory Server settings which the Admin Server uses to connect to the directory services

The procedures for using the Admin Server Console to manage the Admin Server configuration and associated directory services is covered in the *Using the Admin Server* guide.

## G.2. BASIC TASKS IN THE RED HAT CONSOLE

While most server management functions are carried out in server-specific console windows for the Red Hat Directory Server and Admin Server, the main Red Hat Console itself has some basic management functions, such as creating server instances, searching the directory, setting some access controls, and allowing some entry modifications.

This chapter covers basic tasks in the Red Hat Console, including installing the Console, creating and editing server instances, and configuring the Console appearance.

### G.2.1. Installing the Console

The Red Hat Console package, **389-ds-console.noarch**, can be installed on Red Hat Enterprise Linux systems using tools like **yum**. For example:

```
yum install redhat-idm-console
```

The Red Hat Console package can also be downloaded through Red Hat Network and installed using package management tools such as **rpm** and **pkgadd**. For example:

■

```
rpm -ivh redhat-idm-console-1.0.0-22.el4idm.i386.rpm
```

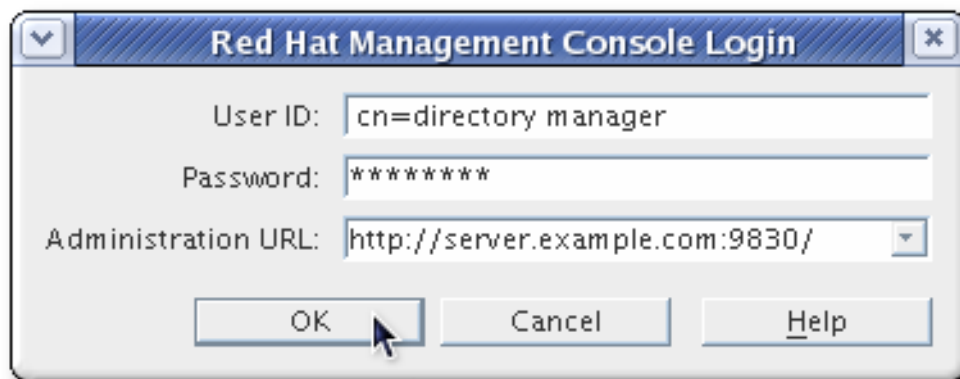
## G.2.2. Launching the Console

1. Run the **redhat-idm-console** command. For example:

```
redhat-idm-console -a http://server.example.com:9830
```

The different options for the **redhat-idm-console** command are listed in [Table G.2, “Arguments for redhat-idm-console”](#).

2. Enter the user name and password.



Also, enter or select the URL for the instance of Admin Server, if one was not passed with the command. The URL can be either the host name or the IP address of the Admin Server host. The Admin Server port number must be given, as well. The five most recent Admin Server URLs accessed are available as a drop-down menu option.

**Table G.2. Arguments for redhat-idm-console**

Argument	Description	Example
<code>-a adminURL</code>	Specifies a base URL for the instance of Admin Server to log into.	<code>-a http://eastcoast.example.com:987</code>
<code>-f fileName</code>	Writes errors and system messages to <i>fileName</i> .	<code>-f system.out</code>
<code>-h</code>	Prints out the help message for <b>redhat-idm-console</b> .	
<code>-s</code>	Specifies the directory instance to access, either by specifying the DN of the server instance entry (SIE) or the instance name, such as <b>slapd-example</b> .	<code>-s slapd-example</code>
<code>-u</code>	Gives the user DN to use to log into the Console.	<code>-u "cn=Directory Manager"</code>

Argument	Description	Example
<code>-w</code>	Gives the password to use to log into the Console.	<code>-w secret</code>
<code>-w -</code>	Reads the password from the standard output.	
<code>-x options</code>	<p>Specifies extra options. There are three values for <i>extraOptions</i>:</p> <div> <p>nowinpos, which puts the Console window in the upper left corner of the screen</p> <p>nologo, which keeps the splash screen from being displayed and only opens the login dialog</p> <p>javalaaf, which uses the <i>Java look and feel</i> for the Console interface rather than the platform-specific styles</p> </div> <p>To use multiple options, separate them with a comma.</p>	<code>-x nologo,nowinpos</code>
<code>-y file</code>	Reads the password from the specified input file.	<code>-y password.txt</code>

### G.2.3. Opening a Directory or Admin Server Window

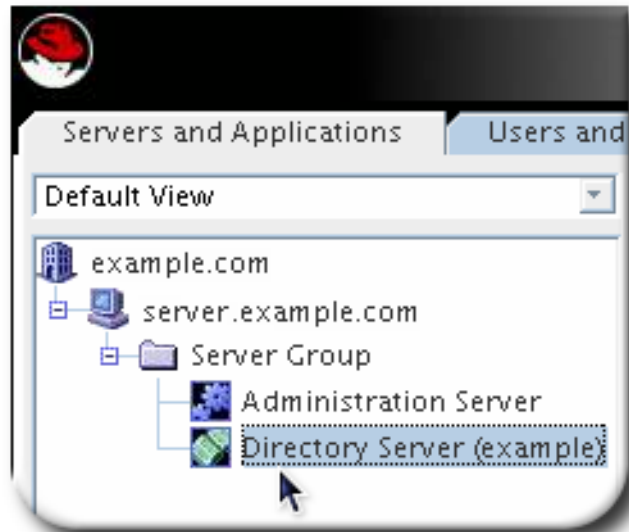
The Red Hat Management Console is the avenue to access instance-specific management windows for the Directory Server and Admin Server. To open a console window for a specific server instance:

1. Open the Red Hat Console.

```
redhat-idm-console
```

2. Click the **Servers and Applications** tab, which lists all of the Directory Server and Admin Server instances within the configured Directory Server domain.
3. In the navigation tree, click a server to select it.





4. In the right-hand panel, click **Open**.



Alternatively, double-click the server icon in the navigation tree.

## G.2.4. Changing the Console Appearance

The fonts used for different elements in the Console can be edited. The font settings and the location where the font profiles are stored can be customized. The default font settings can be restored easily.

This section also describes how to control other aspects of the appearance of the Console. For example, table columns can be easily rearranged. It is also possible to control which server instances are displayed (called a *navigation view*) which makes it easy to sort and find server instances.

Access control instructions can be applied to user interface elements, which is discussed in [Section G.5, “Setting Access Controls”](#).

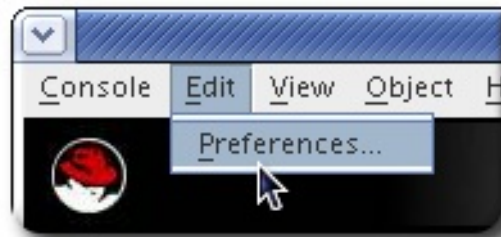
- [Section G.2.4.1, “Changing Profile Locations”](#)
- [Section G.2.4.2, “Restoring Default Font Settings”](#)
- [Section G.2.4.3, “Changing Console Fonts”](#)
- [Section G.2.4.4, “Reordering Table Columns”](#)
- [Section G.2.4.5, “Customizing the Main Window”](#)

### G.2.4.1. Changing Profile Locations

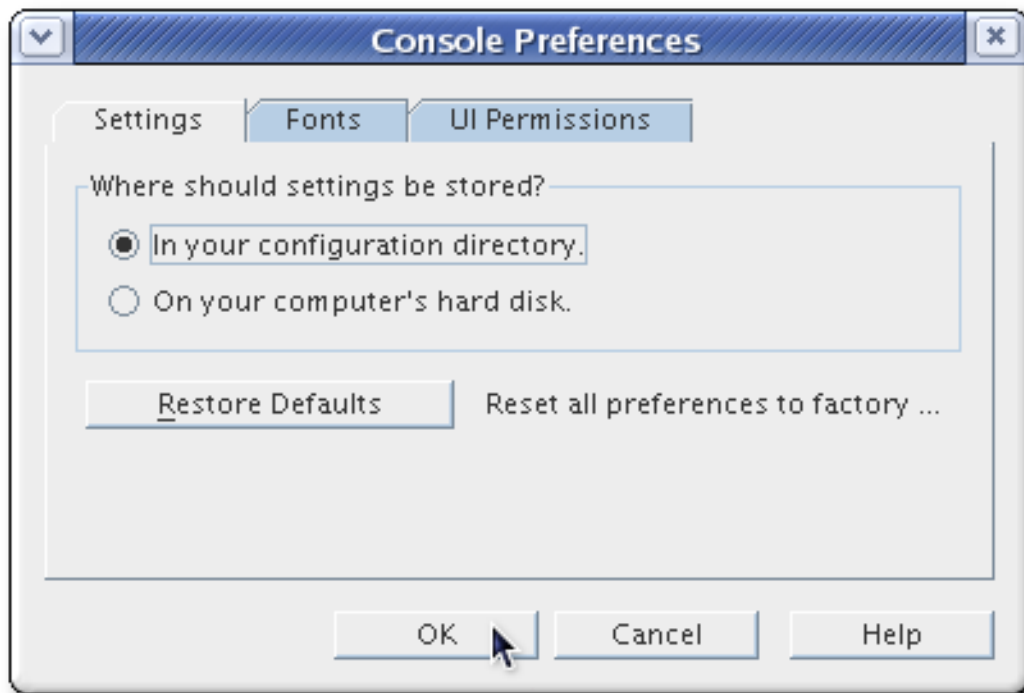
The Console formatting is stored in *profiles*. An entry's profiles can be stored locally, which means that they are only available at a specific workstation, or can be stored in the configuration directory, so they are accessible anywhere.

To set the profile location:

1. Click **Edit** in the top menu, and choose **Preferences**.



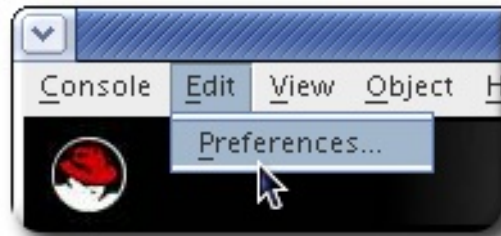
2. Click the **Settings** tab.
3. Select the radio button for the location to save the settings.



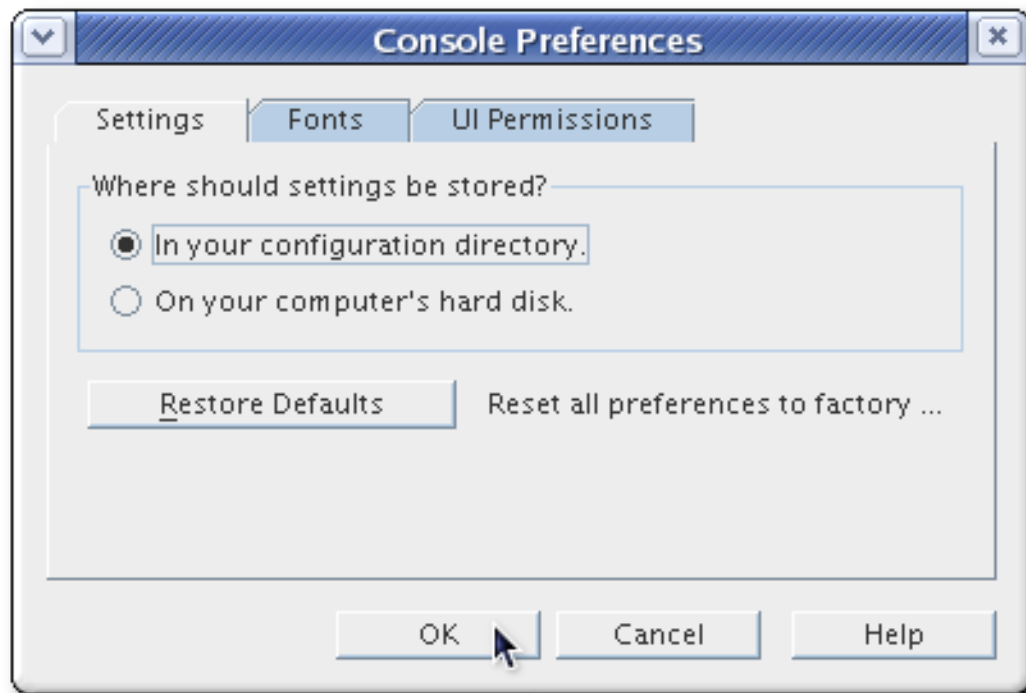
- *In your configuration directory* means that the settings are stored in the Directory Server configuration, making them available no matter where you log into the Console.
  - *On your computer's hard disk* stores the setting profiles locally. This is mainly useful if you want specific, different settings used by default on different Consoles, such as a workstation and a laptop.
4. Click **OK**.

#### G.2.4.2. Restoring Default Font Settings

1. Click **Edit** in the top menu, and choose **Preferences**.



2. Click the **Settings** tab.
3. Click the **Restore Defaults** button to revert to the default display settings.



4. Click **OK**.

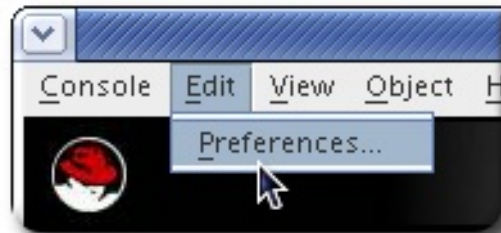
#### G.2.4.3. Changing Console Fonts

Different parts of the Console, such as table headings and regular text, have different font settings. The font settings are stored in *profiles*. The profiles define the font family, size, and formatting for every text element. There can be multiple font profiles available, and the font profiles can be private, such as settings for a specific user or group, or public, so that any user can access them.

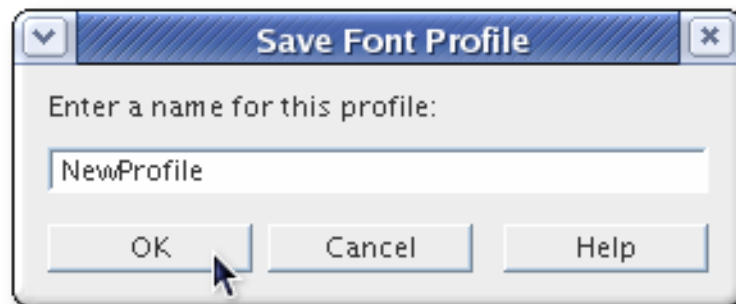
The default profile can be edited without having to create new profiles.

To edit or create a font profile:

1. In the main Red Hat Management Console window, from the **Edit** menu, choose **Preferences**.

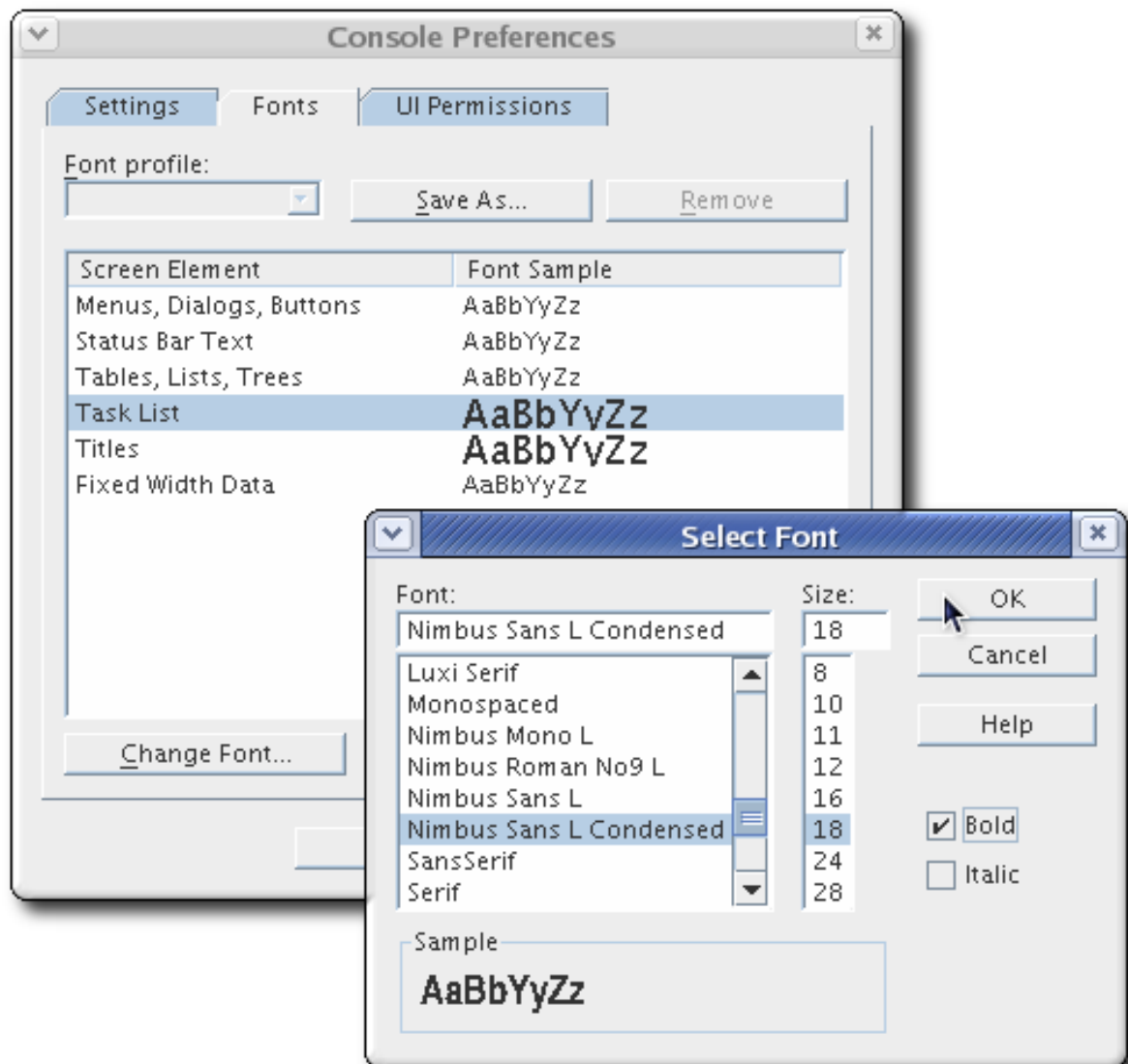


2. Click the **Fonts** tab.
3. To save the new settings as a new profile, click the **Save As** button, and fill in the profile name.



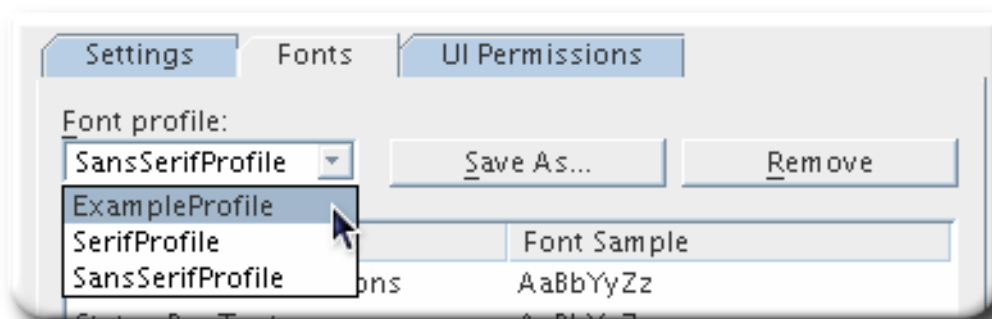
To edit the default (or current) profile, simply begin editing the fonts.

4. In the **Screen Element** column, click a screen element to edit, then click the **Change Font** button.
5. Edit the font for that specific element. There are three settings which can be changed: the font family, the size, and the formatting (bold or italic).



6. Click **OK** to save the profile.
7. Restart the Console to apply the changes.

To load and use a saved font profile, open the **Font** tab in the **Preference** dialog, and simply select the font profile to use and click **OK**.

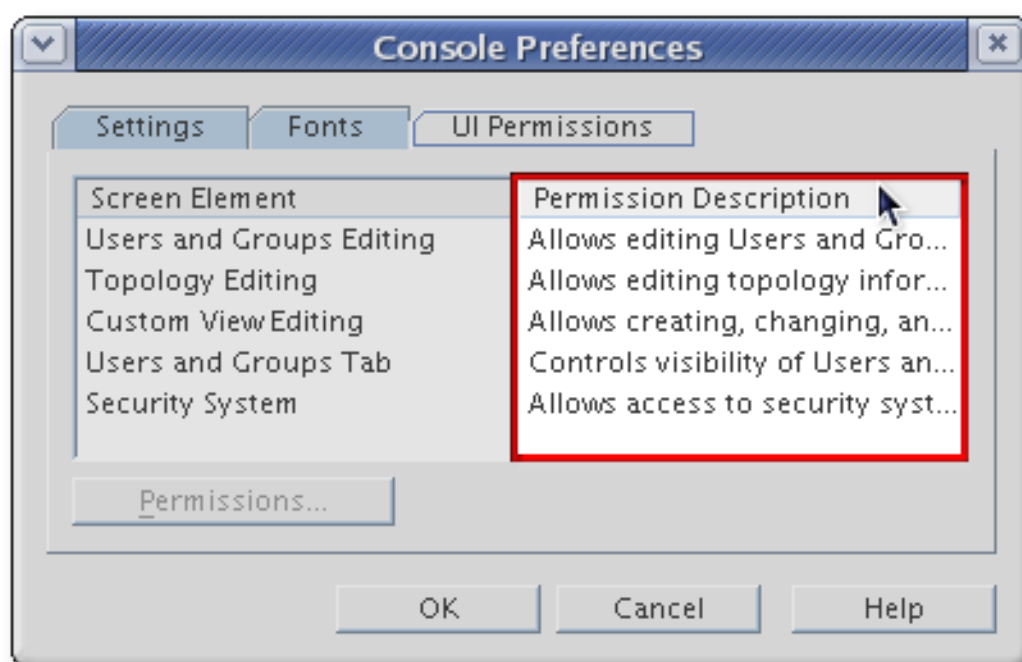


To delete a font profile, simply make sure that it is selected from the drop-down menu in the **Font** tab, and click the **Remove** button.

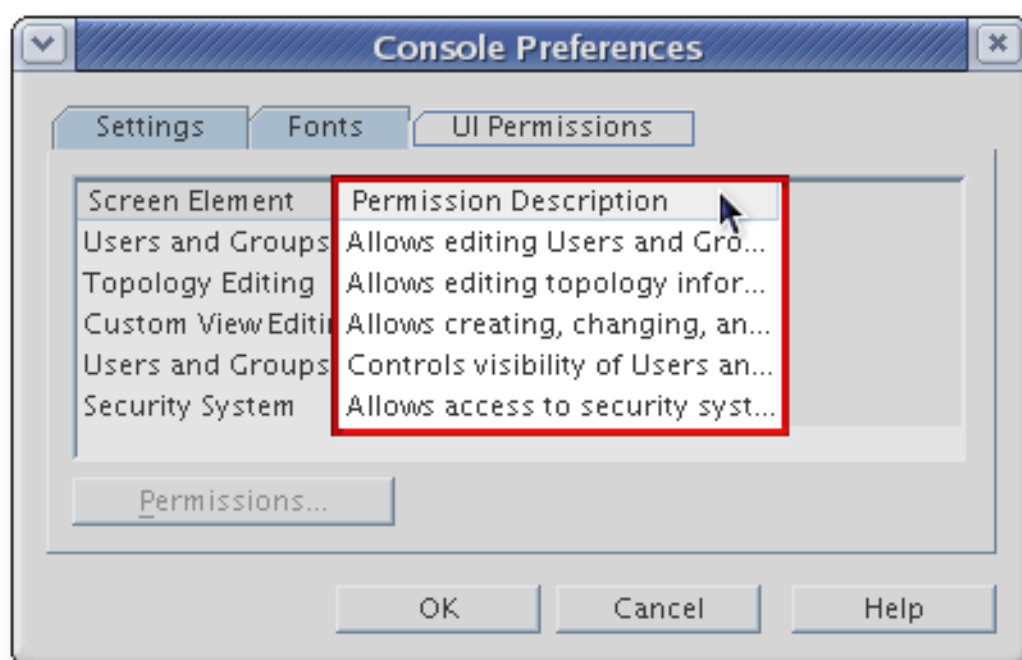
#### G.2.4.4. Reordering Table Columns

The columns in a table can be rearranged by dragging them into a new position.

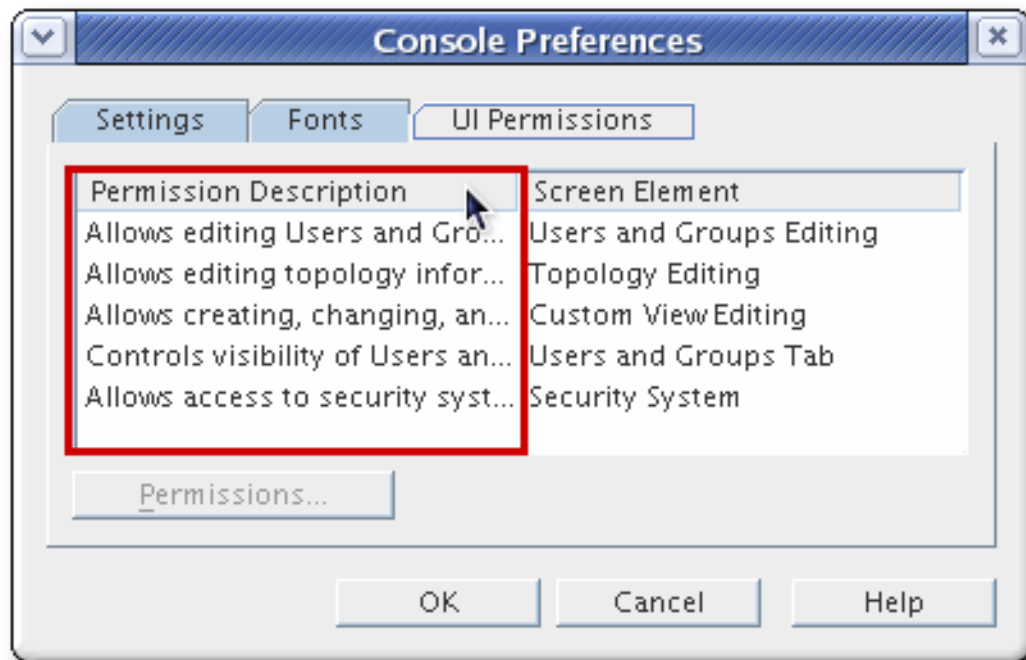
1. Click in the table heading.



2. Still holding down the left mouse button, drag the column to its new location. The other table columns will automatically shift down to their new positions.

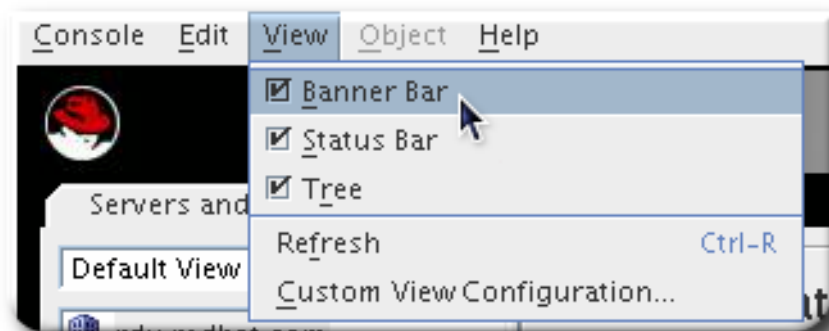


3. When you release the mouse button, the column snaps into its new position.

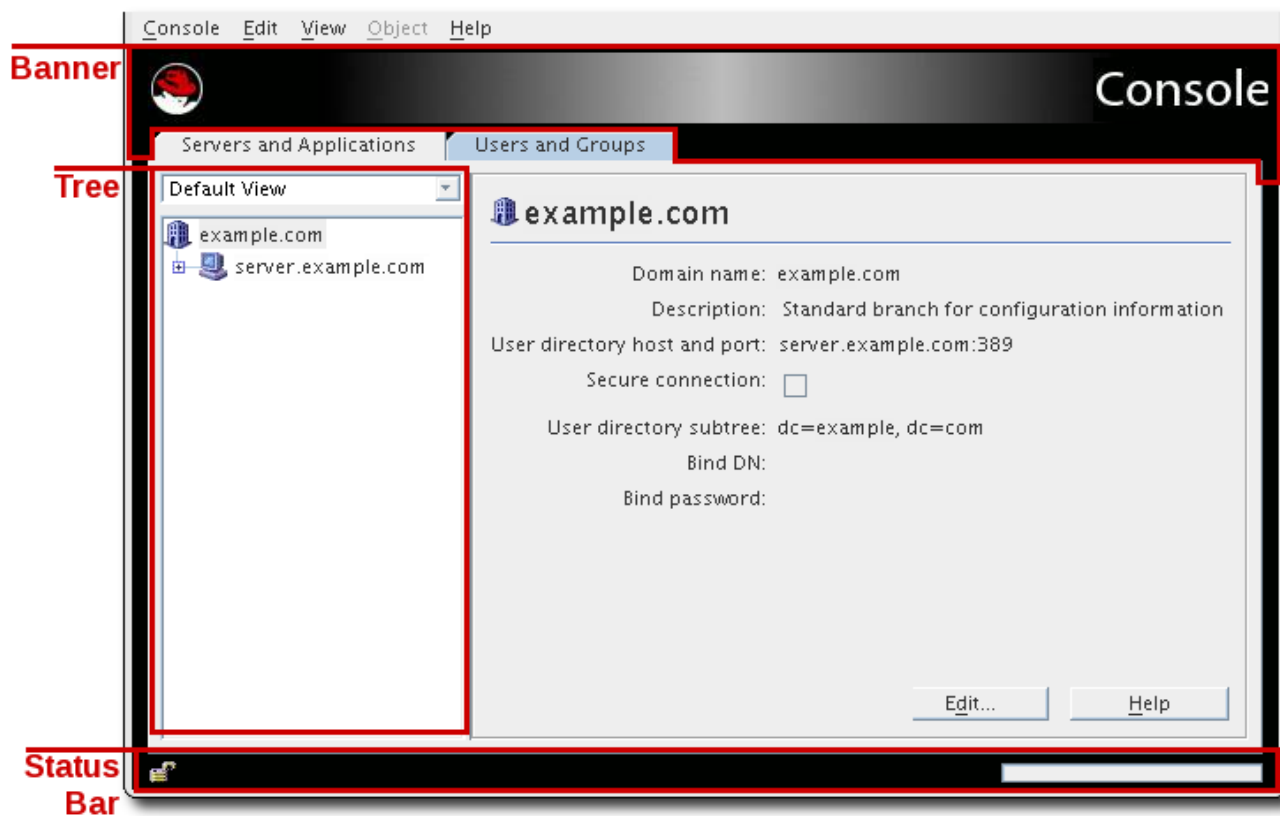


#### G.2.4.5. Customizing the Main Window

Different elements of the main Red Hat Management Console window can be displayed or hidden; this is set by check boxes in the **View** menu.



There are three parts of the Console which can be hidden: the navigation tree (the smaller panel on the left of the Console window); the decorative background and banner at the top of the Console window; and the status bar at the bottom of the Console.



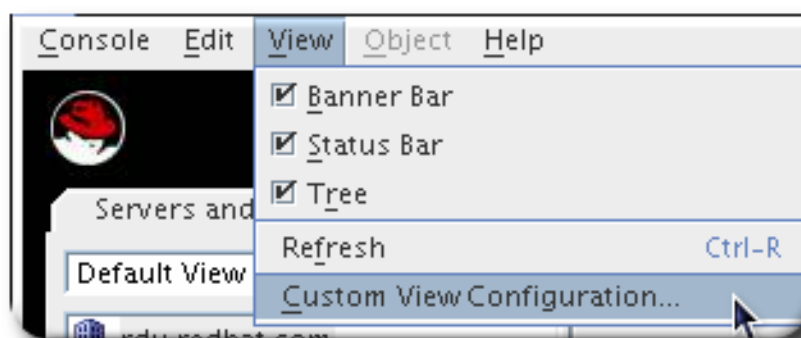
#### G.2.4.6. Working with Custom Views

The Console allows different *views* to be created to show different server and domain entries in the Red Hat Management Console window. Views show only a defined set of server entries; this makes it easier to maintain large numbers of instances or to have a quick way to perform specific tasks.

##### G.2.4.6.1. Creating Custom Views

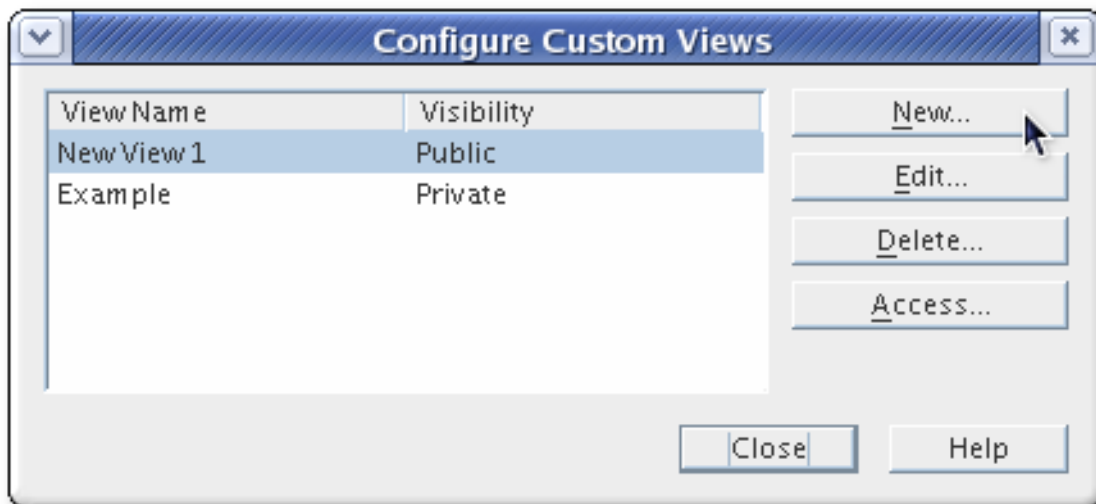
Custom views show different, defined server instances. Views are either public or private. A public view is visible to any user, while a private view is visible only to the person who created it.

1. In the **View** menu, choose **Custom View Configuration**.



2. Click **New**.

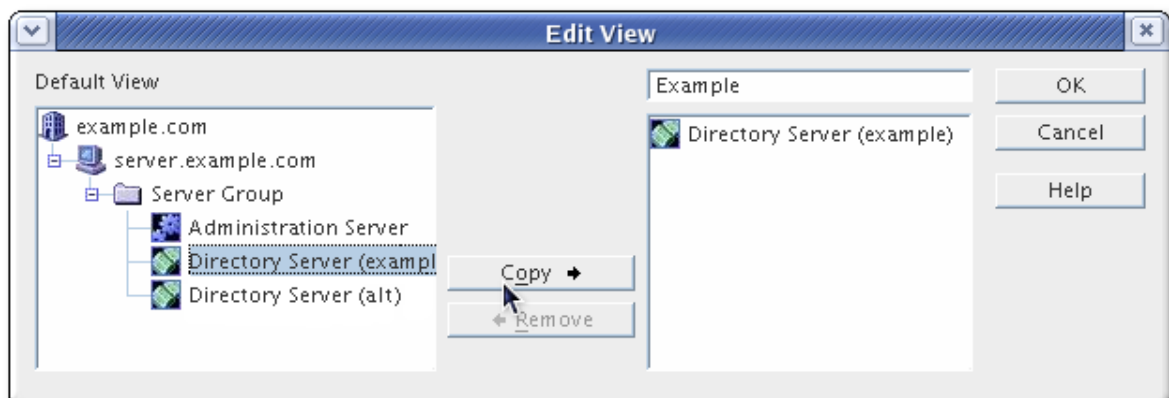




3. Choose whether the new view will be public or private, then click **OK**.



- A public view is visible to all Console users by default, but access control instructions (ACIs) can be set to restrict access. For more information, see [Section G.2.4.6.3, “Setting Access Permissions for a Public View”](#).
  - A private view is only visible to the user who sets it, and ACIs cannot be set to change the access to it.
4. In the **Edit View** window, enter a descriptive name for this view.
  5. Select a resource from the **Default View** navigation tree on the left. Click **Copy** to list it in the panel on the right and include it in the view.



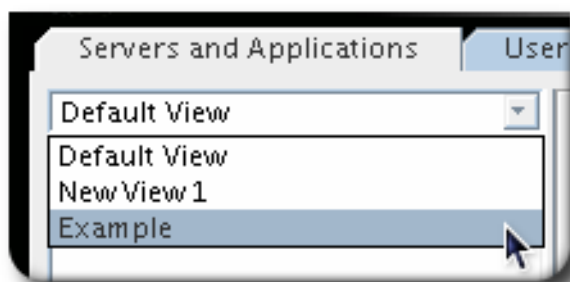
To select a range of resources, click the **SHIFT** key and select the first and last entries; select multiple, separate resources by holding down the **Ctrl** key and selecting the entries.

To edit a custom view, select it from the list, click the **Edit** button, and make the changes to the name or resources.

To delete a custom view, select it from the list, and click the **Remove** button.

#### G.2.4.6.2. Switching to a Custom View

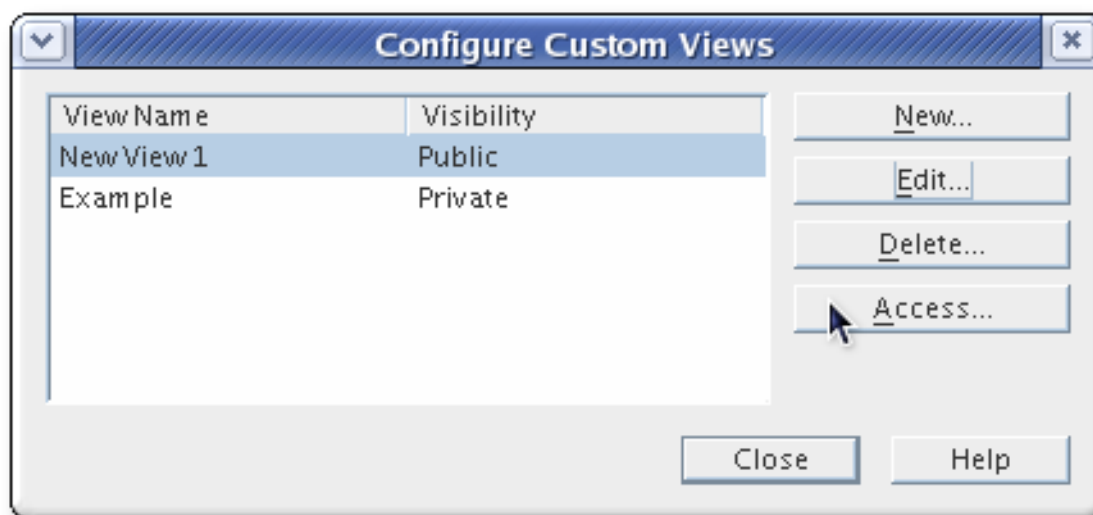
Choose the desired custom view from the drop-down list on the **Servers and Applications** tab.



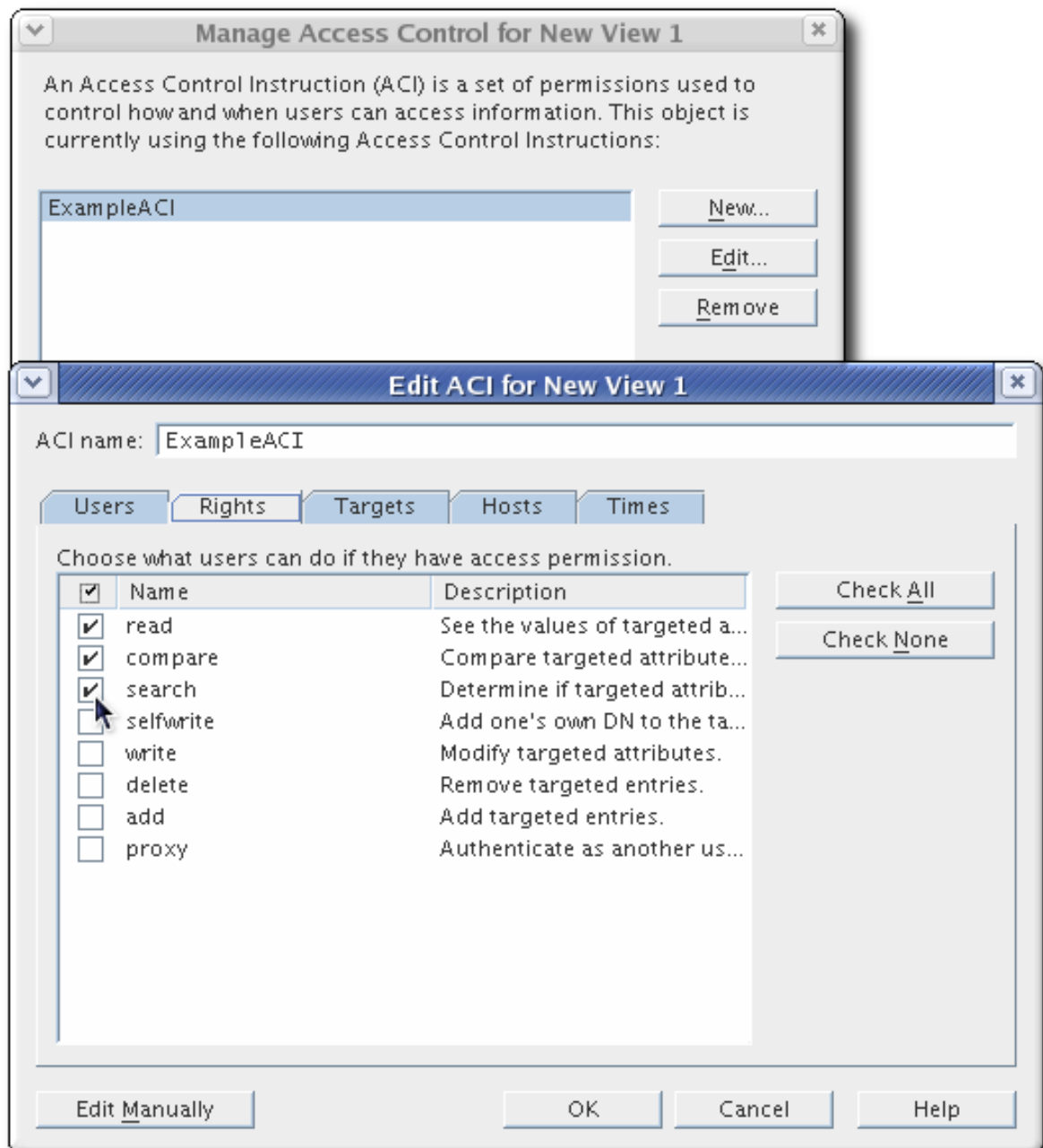
To return to the default view, choose **Default View** from the drop-down list.

#### G.2.4.6.3. Setting Access Permissions for a Public View

1. From the **View** menu, choose **Custom View Configuration**.
2. Choose a public **Custom View** from the list and click **Access**.



3. Set the access control instructions.



4. Click **OK** to save the ACI.

For more information on setting access permissions and creating access control instructions, see [Section G.5, “Setting Access Controls”](#).

## G.3. MANAGING SERVER INSTANCES

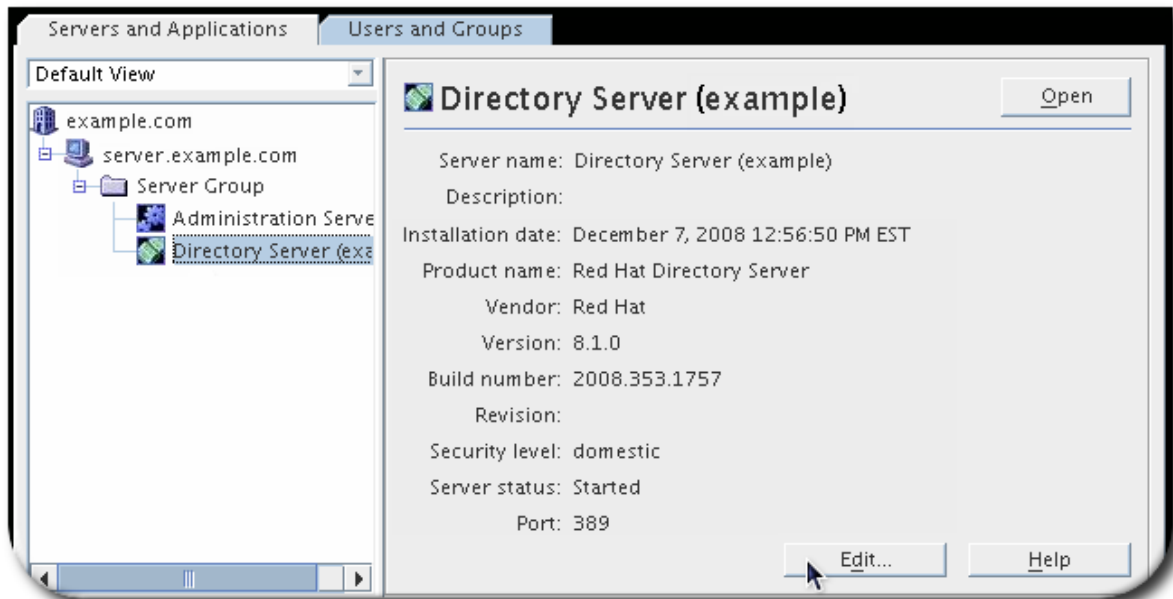
The server instances managed by the Red Hat Management Console are arranged in a hierarchy. At the top is the admin domain. Within the domain are hosts, representing different server machines. Each host has server groups, which identifies an inter-related group of Directory Servers using the same Admin Server instance. The individual Directory Server instances and a single Admin Server instance belong within a server group. There can only be one Admin Server instance per server group.

These high level entries can be created and managed in the Red Hat Management Console.

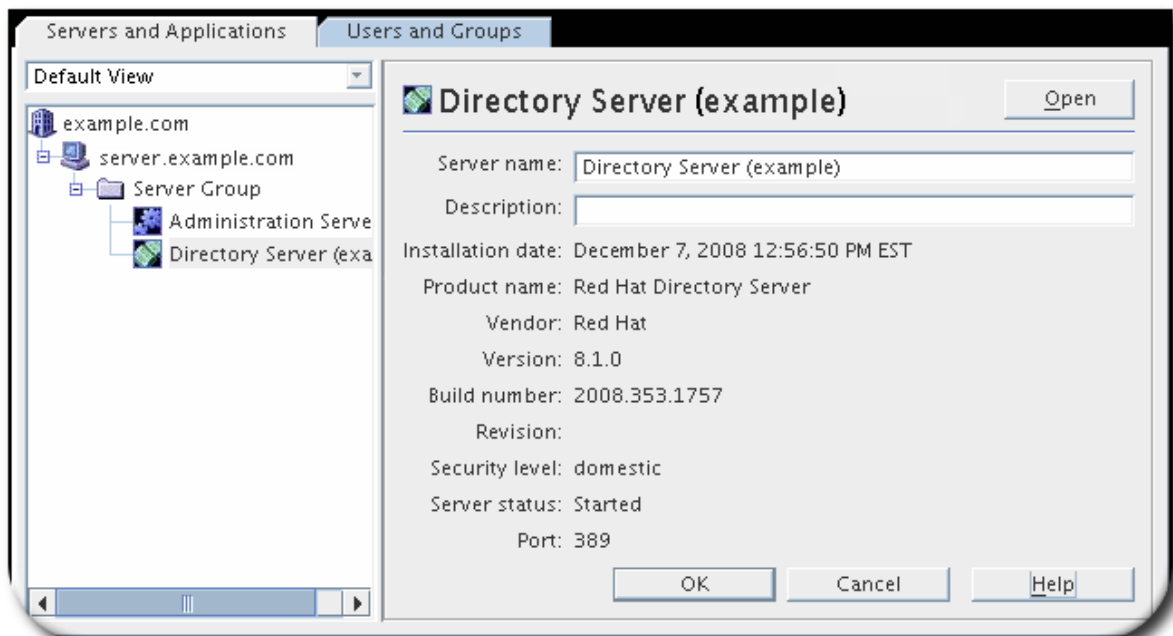
### G.3.1. Editing Domain, Host, Server Group, and Instance Information

The Red Hat Console displays some information about every admin domain, host, group, and server instances. Most of this information — such as the installation date and build number — are not editable, but some information is.

1. In the **Servers and Applications** tab, select the entry to modify.



2. Click **Edit**.
3. Edit the instance's information. Every entry has the option to change its name and description. The host, which is the physical machine on which the instances are installed, also has the option of changing the location.



4. Click **OK**.

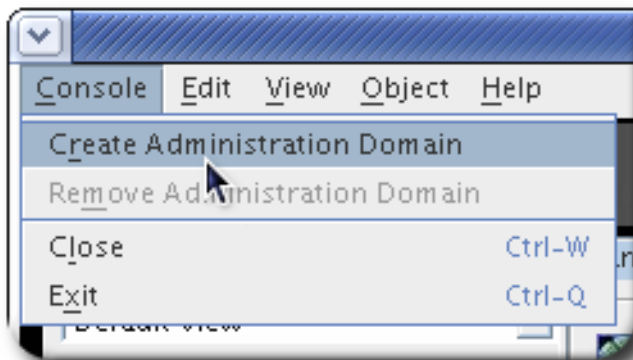
### G.3.2. Creating and Removing Admin Domains

An admin domain is a container entry for server groups (and each server group contains Directory Server instances which are configured to work with the same Configuration Directory Server and the same Admin Server, which is also in the server group).

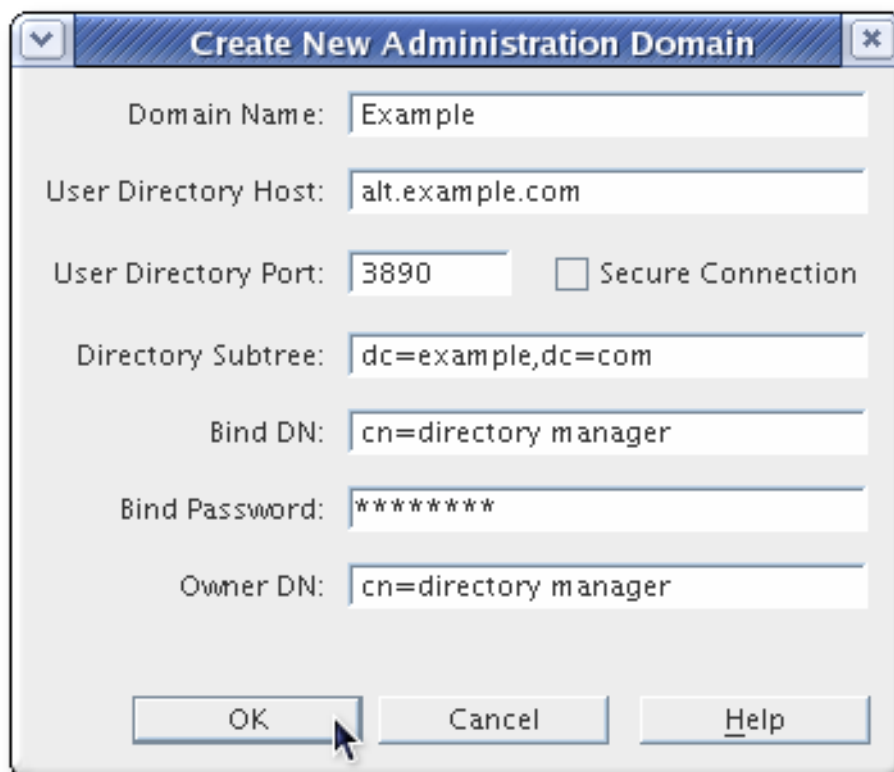
### G.3.2.1. Creating and Editing an Admin Domain

To create a new admin domain:

1. In the top menu, click the **Console** menu item.
2. Select **Create New Administration Domain**.



3. Fill in the admin domain's information, including information for a new Directory Server instance.



4. Click **OK**.

To edit an admin domain, select the entry in the server window and click the **Edit** button.

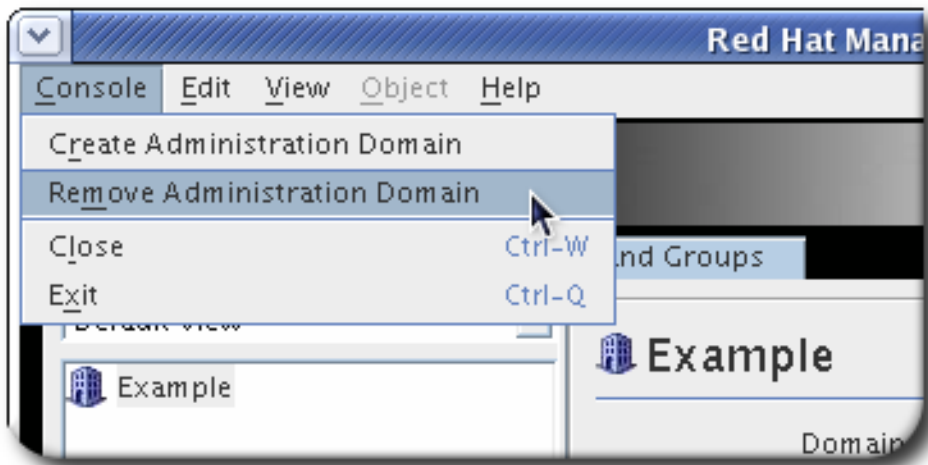
**WARNING**

The admin domain settings affect all servers within the domain. Making any changes to the admin domain settings means that all servers in the domain must be restarted.

**G.3.2.2. Removing an Admin Domain**

To remove an admin domain:

1. Highlight the admin domain to remove in the navigation tree.
2. In the top menu, click the **Console** menu item.
3. Select **Remove Administration Domain**.



4. Click **Yes**.

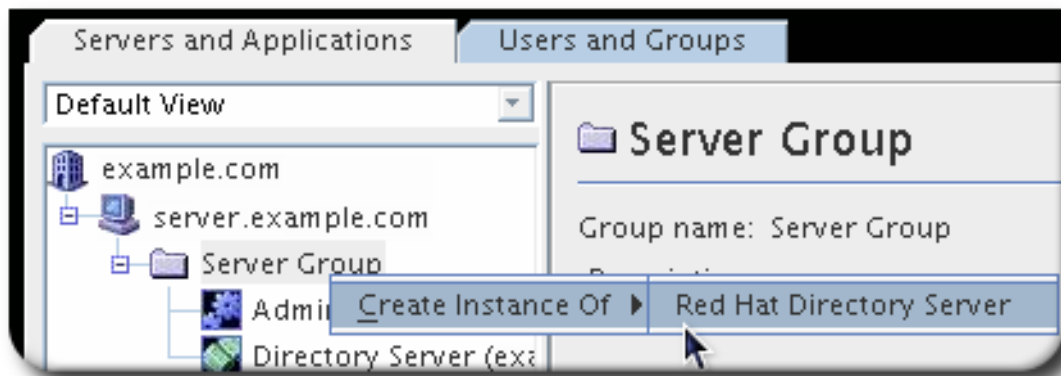
**NOTE**

Any server group and servers within the domain must be removed before the domain can be deleted.

**G.3.3. Creating a New Directory Server Instance**

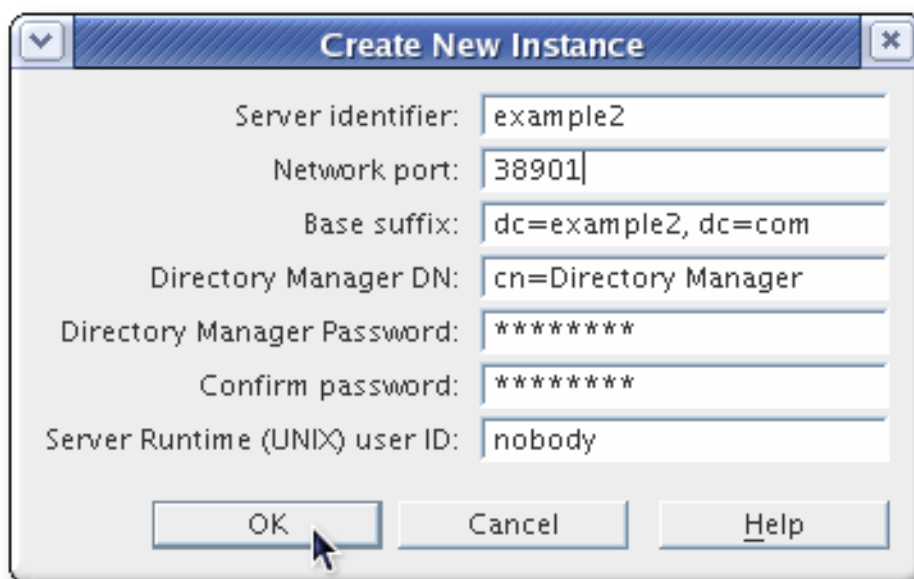
After the default Red Hat Directory Server and Admin Server instances are installed and configured, additional Directory Server instances can be created using the same schema and configuration and in the same installation directory, `/etc/dirsrv`. Having multiple instances on the same host makes it easier to maintain divisions between directories while simplifying administering multiple directories.

1. In Red Hat Management Console, select the server group that will contain the new server instance.
2. Right-click on the server group entry, and select **Create Instance Of**, and then **Red Hat Directory Server**.



Alternatively, click **Object** in the top menu bar, and select **Create Instance Of**.

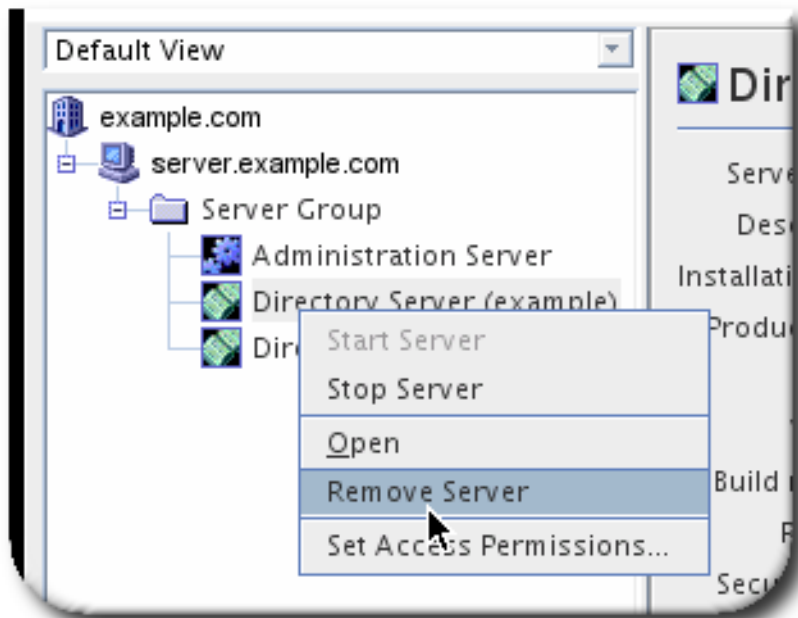
3. Fill in the information for the new instance of Directory Server, including the base DN, Directory Manager, and port.



4. Click **OK**.

### G.3.4. Deleting a Directory Server Instance

1. In the Red Hat Management Console, select the instance to delete.
2. Right-click the server instance, and select **Remove Server**.



3. Click **Yes** to confirm the deletion.

## G.4. MANAGING DIRECTORY SERVER USERS AND GROUPS

Users for both multiple Red Hat Directory Server instances and Admin Server can be created, edited, and searched for in the Red Hat Management Console. The main Console window can also be used to create organizational units and groups and to add entries to the new **ous** and groups.

[Section G.5, “Setting Access Controls”](#) describes how to work with user and group information when setting access privileges and other security information.

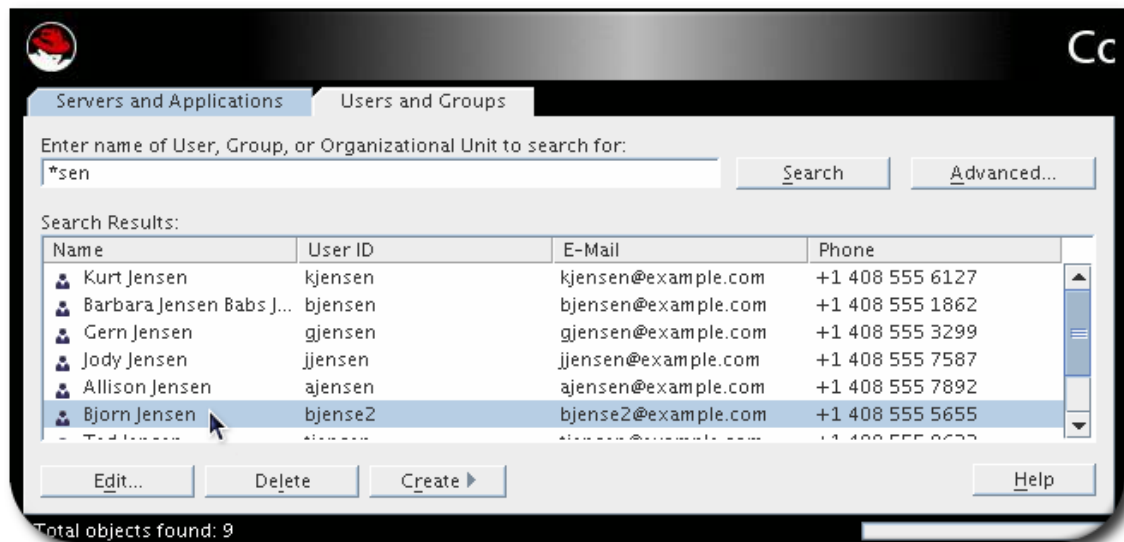
### G.4.1. Searching for Users and Groups

The **Users and Groups** searches for directory entries; by default, it looks in the default user directory configured for the Admin Server, but the directory can be changed to any Red Hat Directory Server instance.

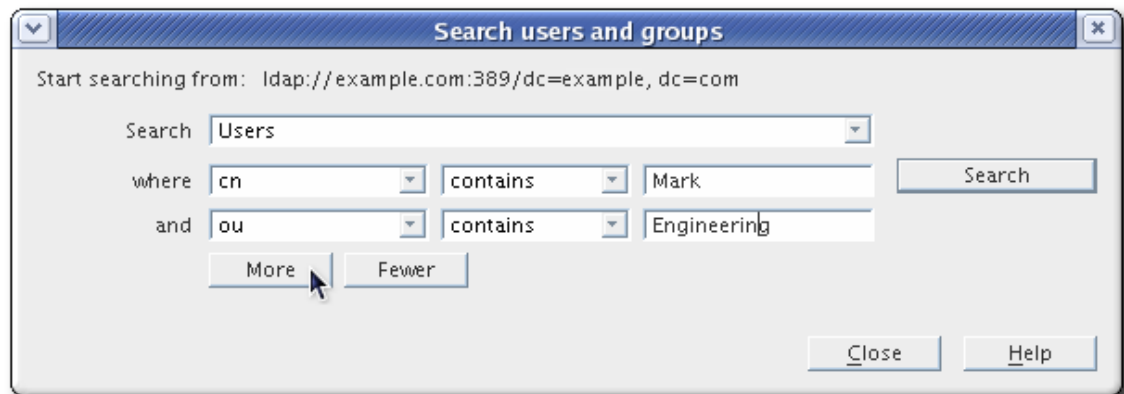
To search the directory:

1. Click the **Users and Groups** tab.
2. Enter the search criteria, and click **Search**.
  - For a simple search, enter all or part of an entry name in the text box. To return all entries, leave the search field blank or enter an asterisk (\*).





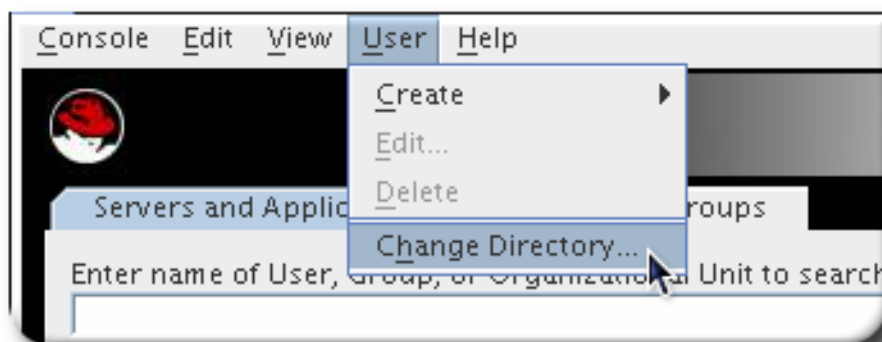
- For a more complex or focused search, click the **Advanced** button, and enter the attributes to search (such as **cn**, **givenname**, or **ou**), the kind of search, and the search term. To add or remove search criteria, click the **More** and **Fewer** buttons.



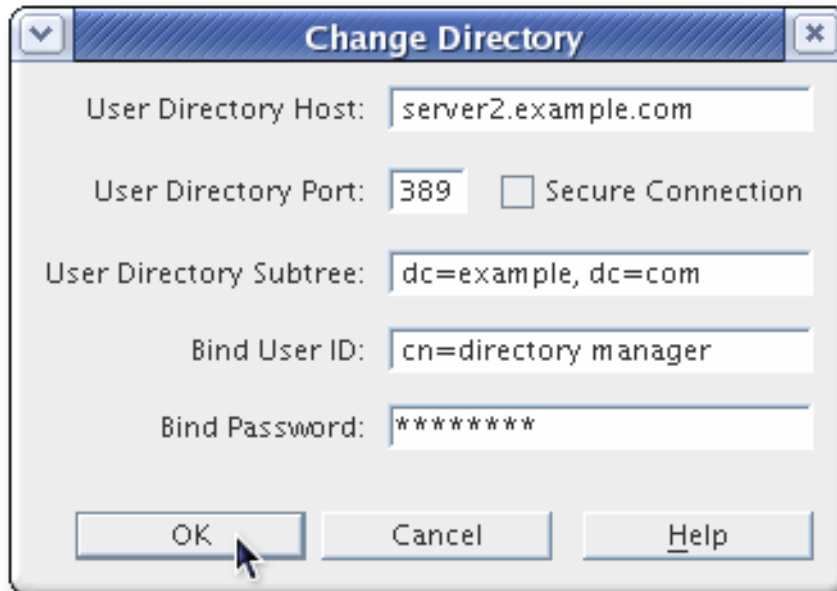
- Click **Search**. Results are displayed in the list box.

To change the search directory:

- Click the **Users and Groups** tab.
- In the top menu, select the **User** menu item, and choose **Change Directory**.



- Fill in the user directory information.



- **User Directory Host.** The fully qualified host name for the Directory Server instance.
- **User Directory Port** and **Secure Connection.** The port number for the connection and whether this is an SSL (LDAPS).
- **User Directory Subtree.** The DN of the subtree to search in the directory; for example, **dc=example, dc=com** for the base DN or **ou=Marketing, dc=example, dc=com** for a subtree.
- **Bind DN** and **Bind Password.** The credentials to use to authenticate to the directory.

4. Click **OK**.

## G.4.2. Creating Directory Entries

The Red Hat Management Console can be used to add, edit, and delete users, groups, and organization units in the **Users and Groups** tab. The different kinds of entries and options for creating entries is explained in more detail in the *Directory Server Administrator's Guide*.

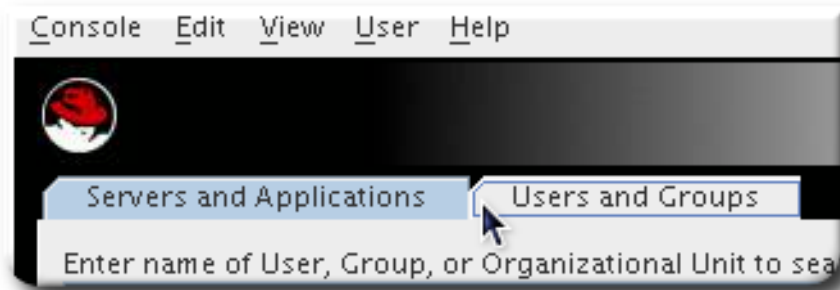
### G.4.2.1. Directory and Administrative Users

#### NOTE

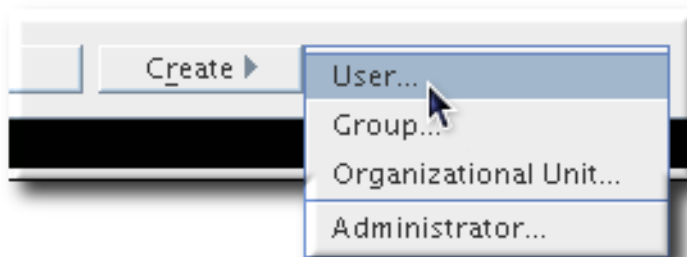
A user can be added to the Directory Server user database through the Console or a user can be added as an Admin Server administrator. The process is almost identical, with two exceptions:

- A Directory Server user is added by clicking the **Create** button, then the **Users** option, while an administrator is created by selecting the **Administrator** option.
- An administrator does not require selecting an organization unit, while the Directory Server user does, because the administrator is automatically added to **ou=Groups, ou=Topology, o=NetScapeRoot**.

1. Click the **Users and Groups** tab.

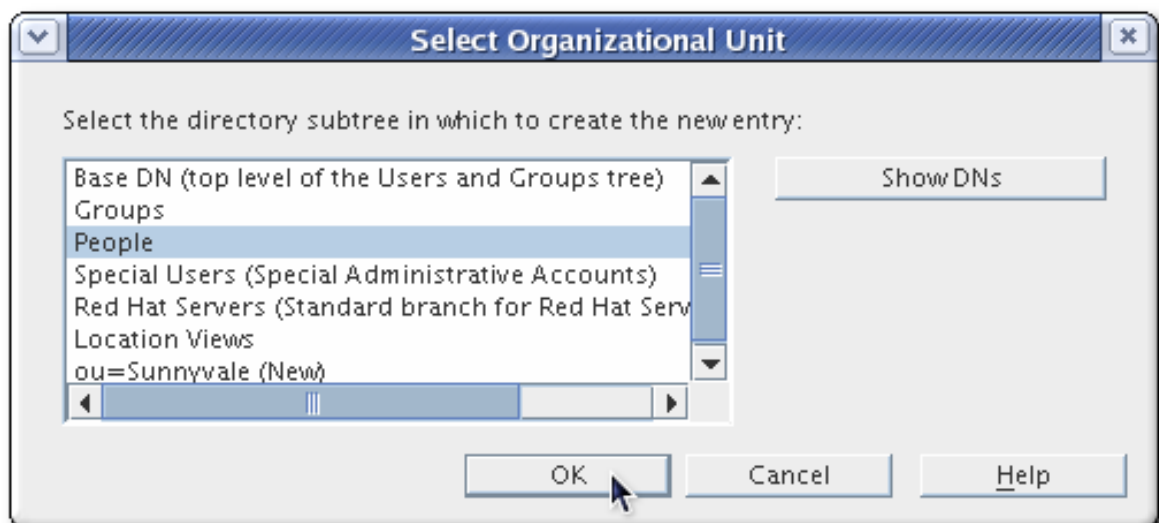


2. Click the **Create** button, and choose **User**.



Alternatively, open the **User** option in the top menu, and choose **Create > User**.

3. Select the are in the directory tree under which the entry is created.



#### NOTE

When creating an administrator, there is no option to select the **ou** to which to add the user as there is with a regular Directory Server user. This is because the administrator is added to **ou=Groups, ou=Topology, o=NetscapeRoot**, with the admin users.

The entry can be added to an **ou** or a view, if views have been added to the directory.

4. In the **Create User** window, enter user information. The **Common Name** and **User ID** fields are automatically filled in with the combined values the **First Name** and **Last Name** fields. These first, last, and common name fields are required; a password is also required for the user

to be able to log into the Directory Server and the Console, but is not a required attribute.

**Create User**

User  
Languages  
NT User  
Posix User

\* First Name: Timothy

\* Last Name: Jameson

\* Common Name(s): Timothy Jameson

User ID: TJameson

Password: \*\*\*\*\*

Confirm Password: \*\*\*\*\*

E-Mail: tjameson@example.com (e.g., user@company.com)

Phone: 919-555-0034

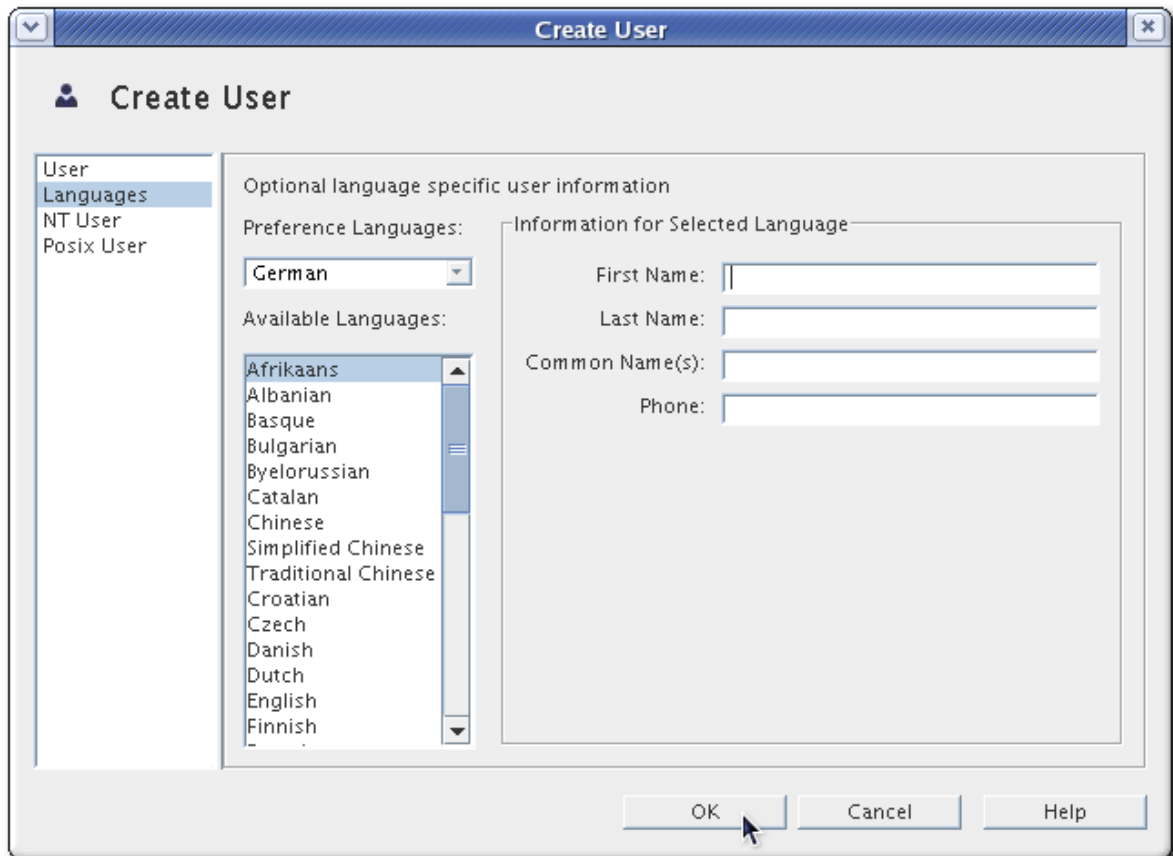
Fax: 919-555-6785

\* Indicates a required field

OK Cancel Help

5. Optionally, click the **Languages** link on the left, select an alternate language and fill in internationalized values for common attributes.

This option allows international users to select a language other than English and to represent their names in their preferred language. The pronunciation attribute allows for phonetic searching against the international name attributes.



6. Click **OK**.

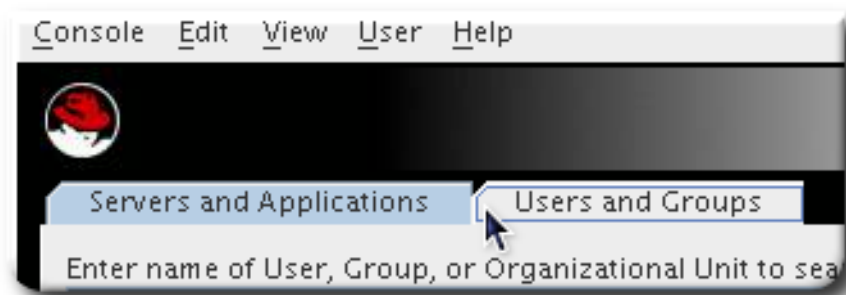
#### G.4.2.2. Groups

A group consists of users who share a common attribute or are part of a list. Red Hat Directory Server supports three types of groups: static, dynamic, and certificate. Each group differs by the way in which users, or *members*, are added to it:

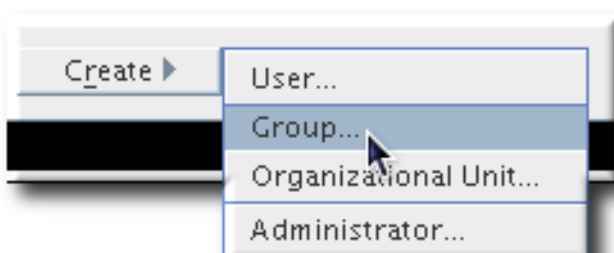
- A *static group* has members who are manually added to it, so it is *static* because the members do not change unless an administrator manually adds or removes users.
- A *dynamic group* automatically includes users based on one or more attributes in their entries; the attributes and values are determined using LDAP URLs. For example, a dynamic group can use an LDAP filter which searches for entries which contain the attributes and values **st=California** and **department=sales**. As entries are added to the directory with those two attributes, the users are automatically added as members to the dynamic group. If those attributes are removed from the entry, the entry is removed from the group.
- A *certificate group* includes all users who have a specific attribute-value pair in the subject name of the certificate. For example, the certificate group could be based on having the string **st=California,ou=Sales,ou=West** in the subject name. If a user logs onto a server using a certificate with those attributes in his certificate, the user is automatically added to the group and is granted all of the access privileges of that group.

To create a group:

1. Click the **Users and Groups** tab.

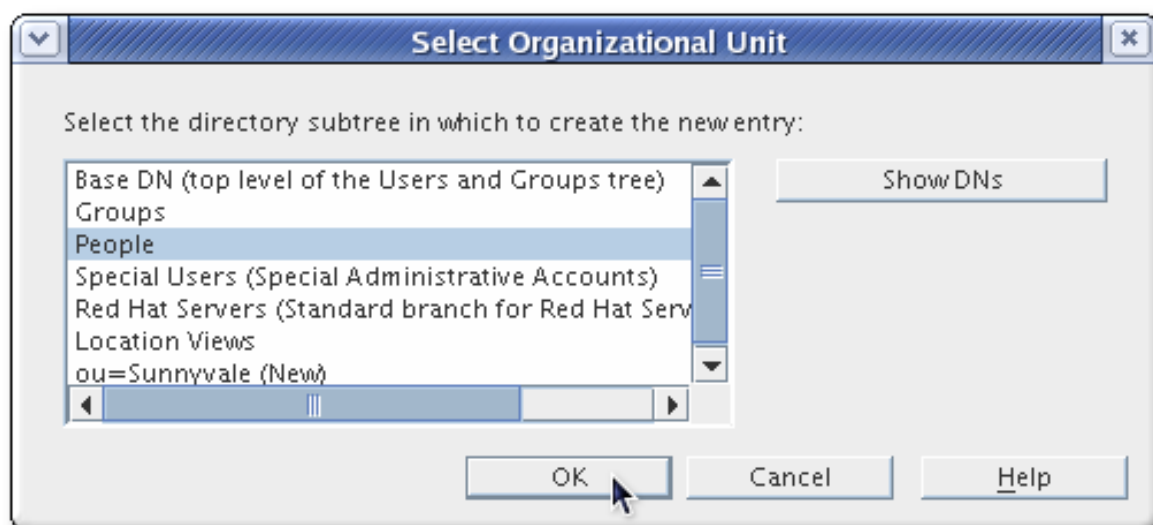


2. Click the **Create** button, and choose **Group**.



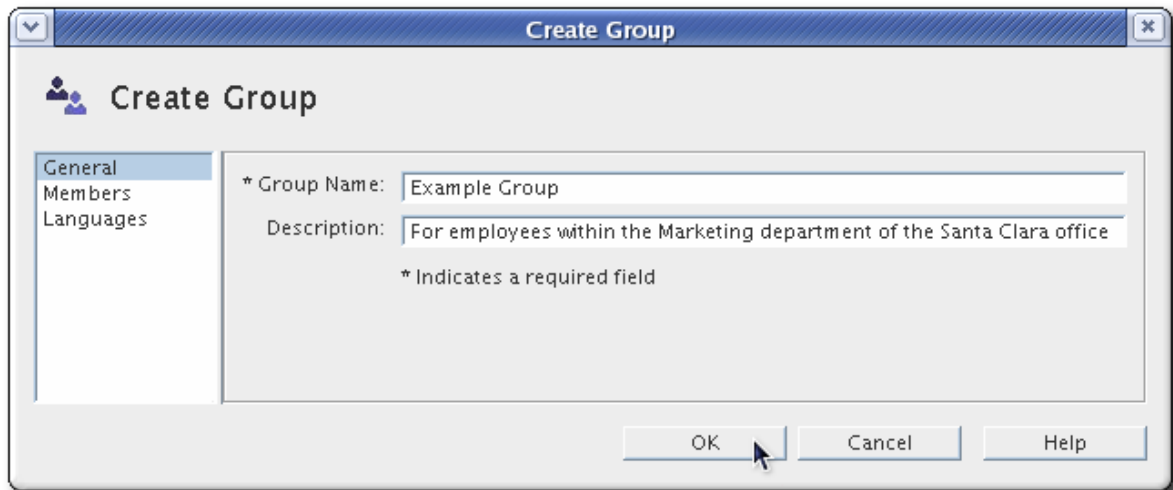
Alternatively, open the **User** option in the top menu, and choose **Create > Group**.

3. Select the area in the directory tree under which the entry is created.



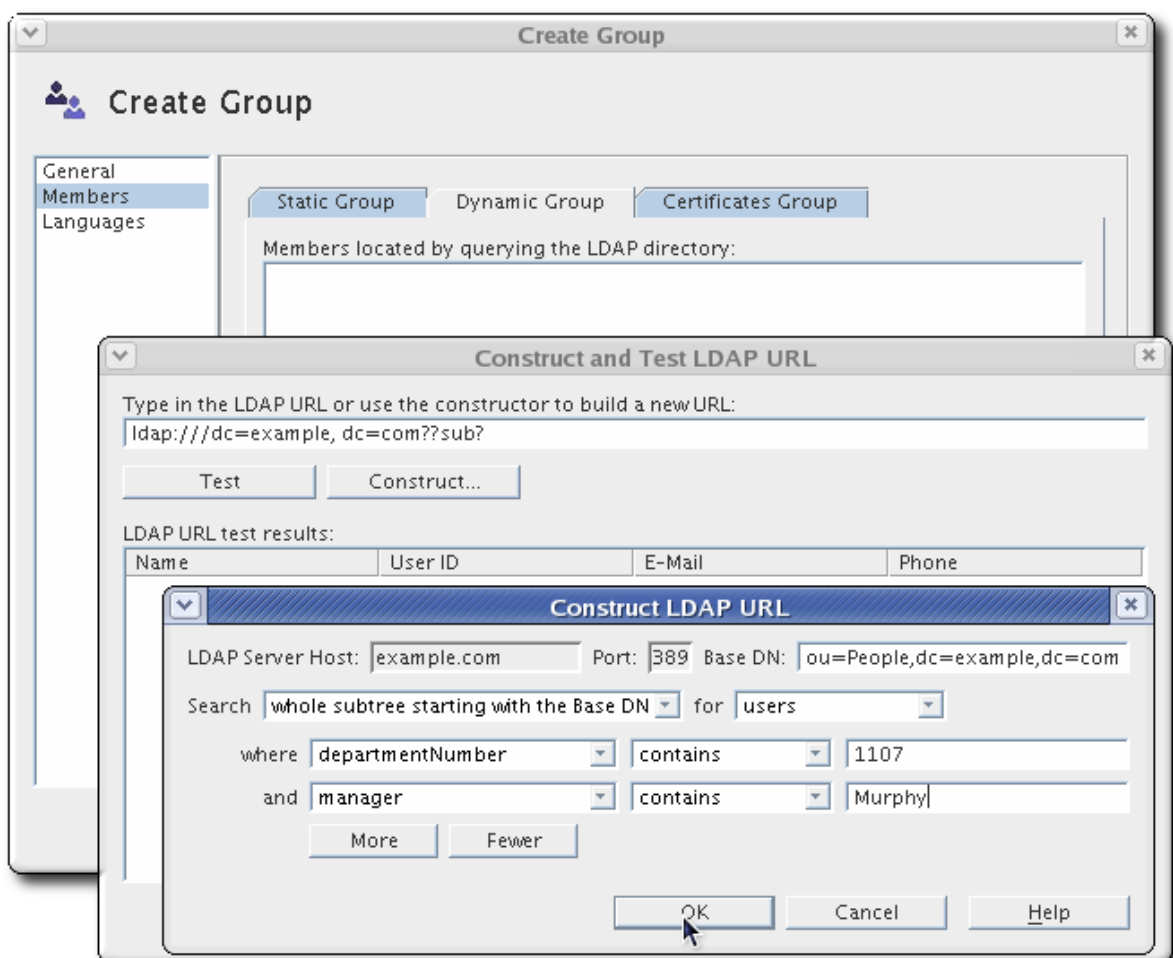
The subtree entry can be an **ou** or a view, if views have been added to the directory.

4. Enter the group's name and description.



It is possible to save the new group entry at this point, without adding members. Click **OK**.

5. Click the **Members** link to add members to the group, and click the tab of the type of group membership, **Static**, **Dynamic**, or **Certificate**.
6. Configure the members. For static groups, manually search for and add users; for dynamic groups, construct the LDAP URL to use to find entries; and for certificate groups, enter the values to search for in user certificate subject names.





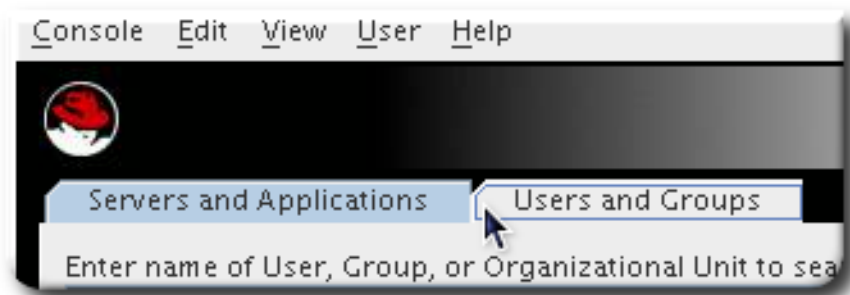
## NOTE

The different kinds of groups and how to configure their members are explained in more detail in the *Directory Server Administrator's Guide*.

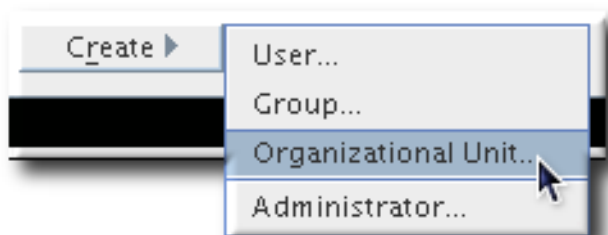
### G.4.2.3. Organizational Units

An organizational unit can include a number of groups and users. An org unit usually represents a distinct, logical division in an organization, such as different departments or geographical locations. Each **organizationalUnitName (ou)** is a new subtree branch in the directory tree. This is reflected in the relative distinguished name of the **ou**, such as **ou=People, dc=example, dc=com**, which becomes part of the distinguished names of its sub-entries.

1. Click the **Users and Groups** tab.



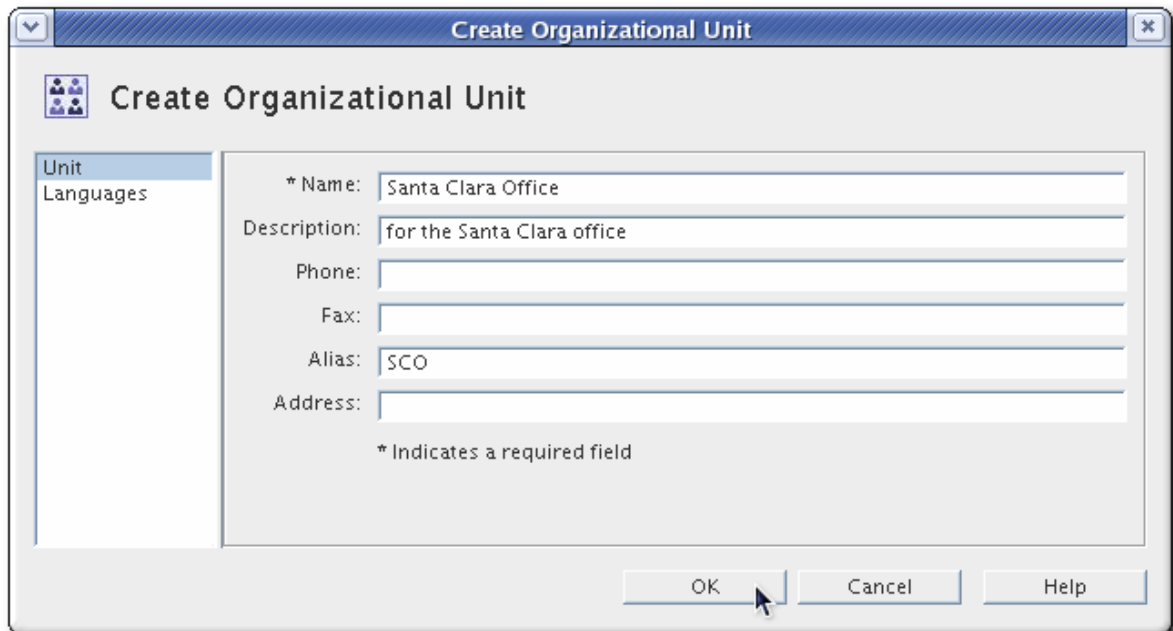
2. Click the **Create** button, and choose **Organizational Unit**.



Alternatively, open the **User** option in the top menu, and choose **Create > Organizational Unit**.

3. Select the directory subtree under which to locate the new organizational unit.
4. Fill in the organizational unit information. The **Alias** offers an alternative name for the organizational unit that can be used instead of the full name.





**Create Organizational Unit**

Unit  
Languages

\* Name: Santa Clara Office

Description: for the Santa Clara office

Phone:

Fax:

Alias: SCO

Address:

\* Indicates a required field

OK Cancel Help

- Click **OK**.

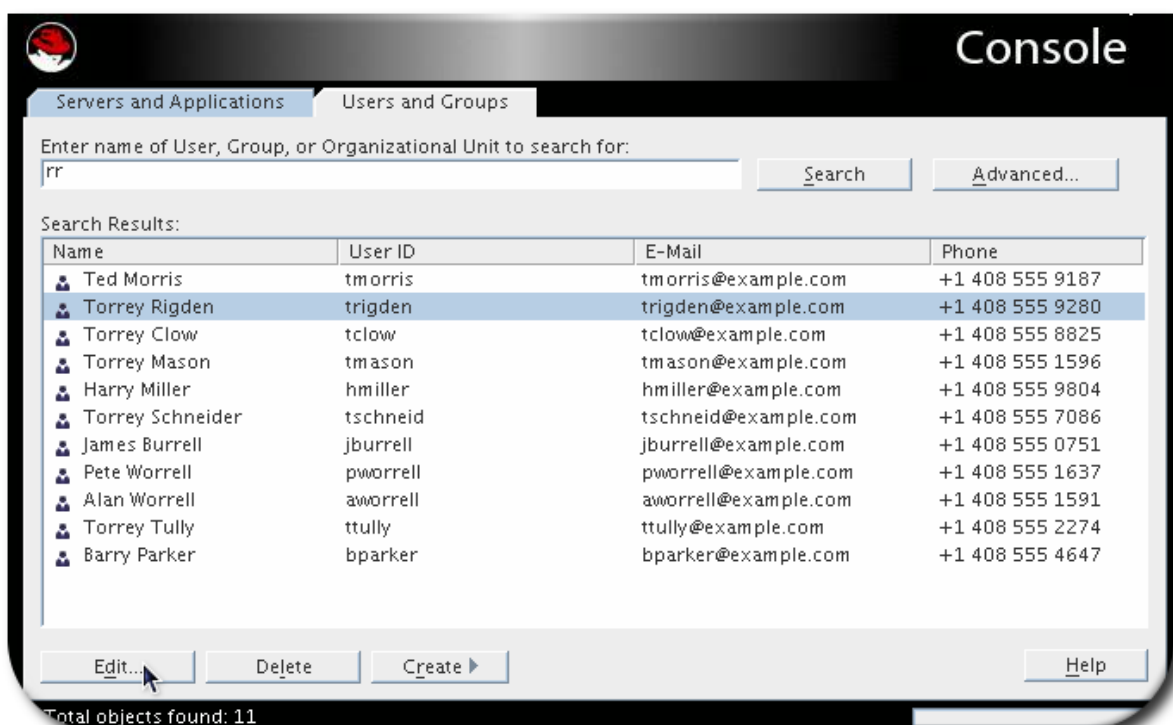
### G.4.3. Modifying Directory Entries

#### G.4.3.1. Editing Entries

- Search for the entry to edit.

See [Section G.4.1, “Searching for Users and Groups”](#) for more information on searching for entries.

- Select the entry, and click **Edit**.



**Console**

Servers and Applications Users and Groups

Enter name of User, Group, or Organizational Unit to search for:

rr Search Advanced...

Search Results:

Name	User ID	E-Mail	Phone
Ted Morris	tmorris	tmorris@example.com	+1 408 555 9187
Torrey Rigden	trigden	trigden@example.com	+1 408 555 9280
Torrey Clow	tclow	tclow@example.com	+1 408 555 8825
Torrey Mason	tmason	tmason@example.com	+1 408 555 1596
Harry Miller	hmiller	hmiller@example.com	+1 408 555 9804
Torrey Schneider	tschneid	tschneid@example.com	+1 408 555 7086
James Burrell	jburrell	jburrell@example.com	+1 408 555 0751
Pete Worrell	pworrell	pworrell@example.com	+1 408 555 1637
Alan Worrell	aworrell	aworrell@example.com	+1 408 555 1591
Torrey Tully	ttully	ttully@example.com	+1 408 555 2274
Barry Parker	bparker	bparker@example.com	+1 408 555 4647

Edit... Delete Create ► Help

Total objects found: 11

- Edit the entry information, and click **OK** to save the changes.

### G.4.3.2. Allowing Sync Attributes for Entries

Red Hat Directory Server and Active Directory synchronization unify some Unix and Windows-specific directory attributes; to carry over a Directory Server entry to Active Directory, the entry must have **ntUser** attributes. (Likewise, Windows entries must have **posixAccount** attributes.)

Windows (NT) attributes must be enabled on entries. By default, these attributes are added manually to individual entries. The user edit windows have links on the left for **NT User** to allow Directory Server entries to contain Windows-specific attributes for synchronization.

It is also possible to configure the server so that all new entries will automatically possess the **ntUser** object class; this is described in the Directory Server—Active Directory synchronization chapter of the *Directory Server Administrator's Guide*.



#### NOTE

Any Red Hat Directory Server entry must have the **ntUser** object class and required attributes added in order to be synchronized to Active Directory.

To enable synchronization:

1. Select or create a user, and click the **NT User** link.
2. Enable the NT account, and check how the entry will be synchronized (meaning, whether a new entry will be created and whether that entry should be deleted on Active Directory if it is deleted on Directory Server).

**Edit Entry**

**Torrey Rigden**  
Product Development  
Phone: +1 408 555 9280  
Fax: +1 408 555 8473

User  
Languages  
**NT User**  
Posix User

☒ Enable NT User Attributes

\* NT User ID: trigden

☒ Create New NT Account

☒ Delete NT Account If Person Deleted

Comment:

User Profile Path:

Logon Script:

Home Drive: C

Home Directory:

Logon Server: Logon Hours

User Workstations List:

Account Expiration Date: Change

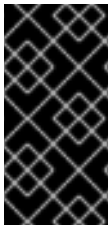
OK Cancel Help

3. Click **OK**.

### G.4.3.3. Changing Administrator Entries

When the Admin Server is installed, two entries are created with administrator access in the Console. The main entry is the *Configuration Administrator*, who is authorized to access and modify the entire configuration directory (**o=NetScapeRoot**). The Configuration Administrator entry is stored in the **uid=username, ou=Administrators, ou=TopologyManagement, o=NetScapeRoot** entry.

The Configuration Administrator's user name and password are automatically used to create the *Admin Server Administrator*, who can perform a limited number of tasks, such as starting, stopping, and restarting servers. The Admin Server Administrator is created so that a user can log into the Red Hat Management Console when the Directory Server is not running. The Admin Server Administrator does not have an LDAP entry; it exists in the Admin Server's configuration file, **/usr/share/dirsrv/properties/admpw**.



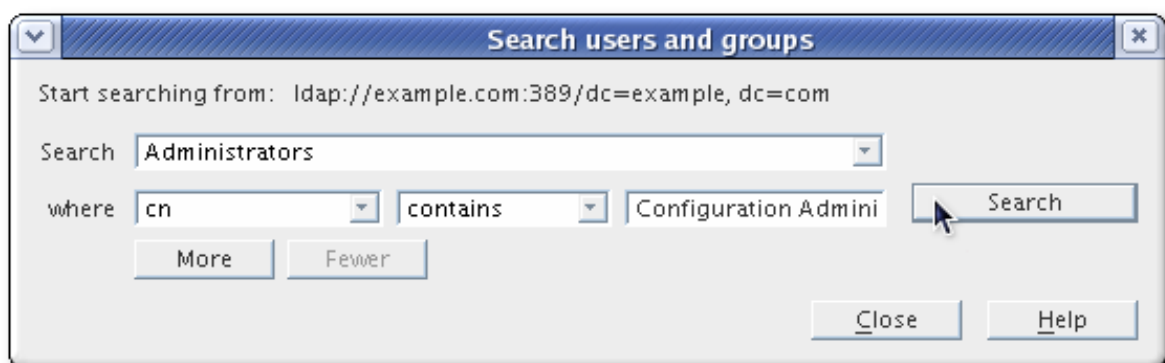
#### IMPORTANT

Even though they are created at the same time during installation, and are identical at that time, the Configuration Administrator and Admin Server Administrator are two separate entities. If the user name or password is changed for one, Red Hat Management Console does not automatically make the same changes for the other.

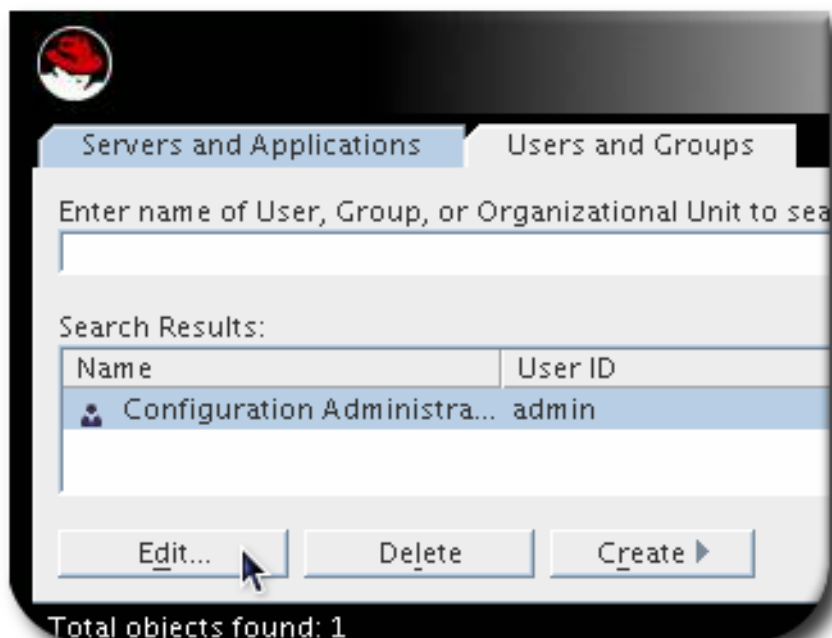
- [Section G.4.3.3.1, “Changing the Configuration Administrator and Password”](#)
- [Section G.4.3.3.2, “Changing the Admin Password”](#)
- [Section G.4.3.3.3, “Adding Users to the Configuration Administrators Group”](#)

#### G.4.3.3.1. Changing the Configuration Administrator and Password

1. In the **Users and Groups**, click **Advanced**.
2. Search for the Configuration Administrator. Select the **Administrators** object, and enter the administrator's user name, **Configuration Administrator** by default.



3. Select the Configuration Administrator from the list of search results, and then click **Edit**.



4. Change the administrator's **uid** and password. The **uid** is the naming attribute used to log into the Console and run commands.

5. Click **OK**.



## NOTE

If you are logged into the Console as the Configuration Administrator when you edited the Configuration Administrator entry, update the login information for the directory.

1. In the **Users and Groups** tab, click the **User** menu in the top menu and select **Change Directory**.
2. Update the **Bind DN** and **Bind Password** fields with the new information for the Configuration Administrator, and click **OK**.

#### G.4.3.3.2. Changing the Admin Password

1. Select the Admin Server in the **Servers and Applications** tab, and click **Open**.
2. Click the **Configuration** tab, and open the **Access** tab.
3. Set the new password.



#### WARNING

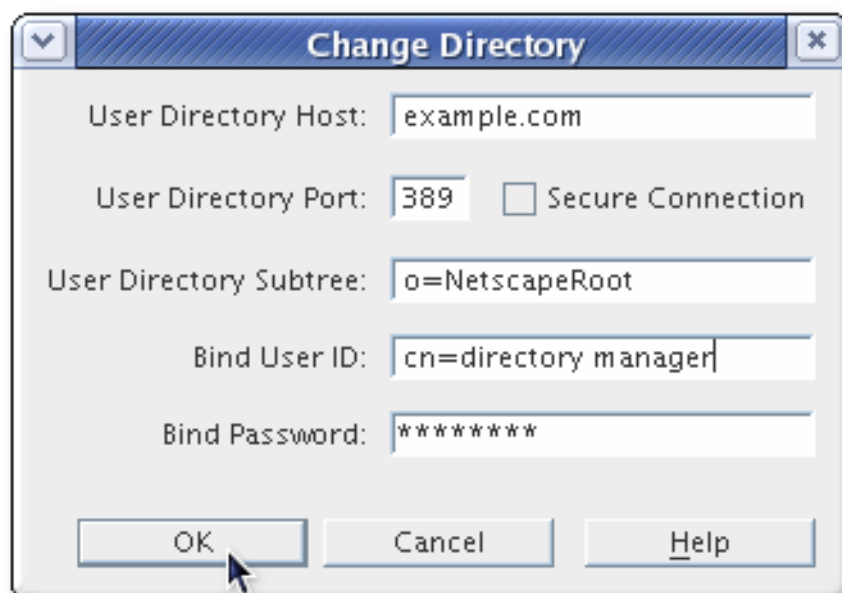
Do not change the admin user name.

4. Click **Save**.
5. Restart the Admin Server.

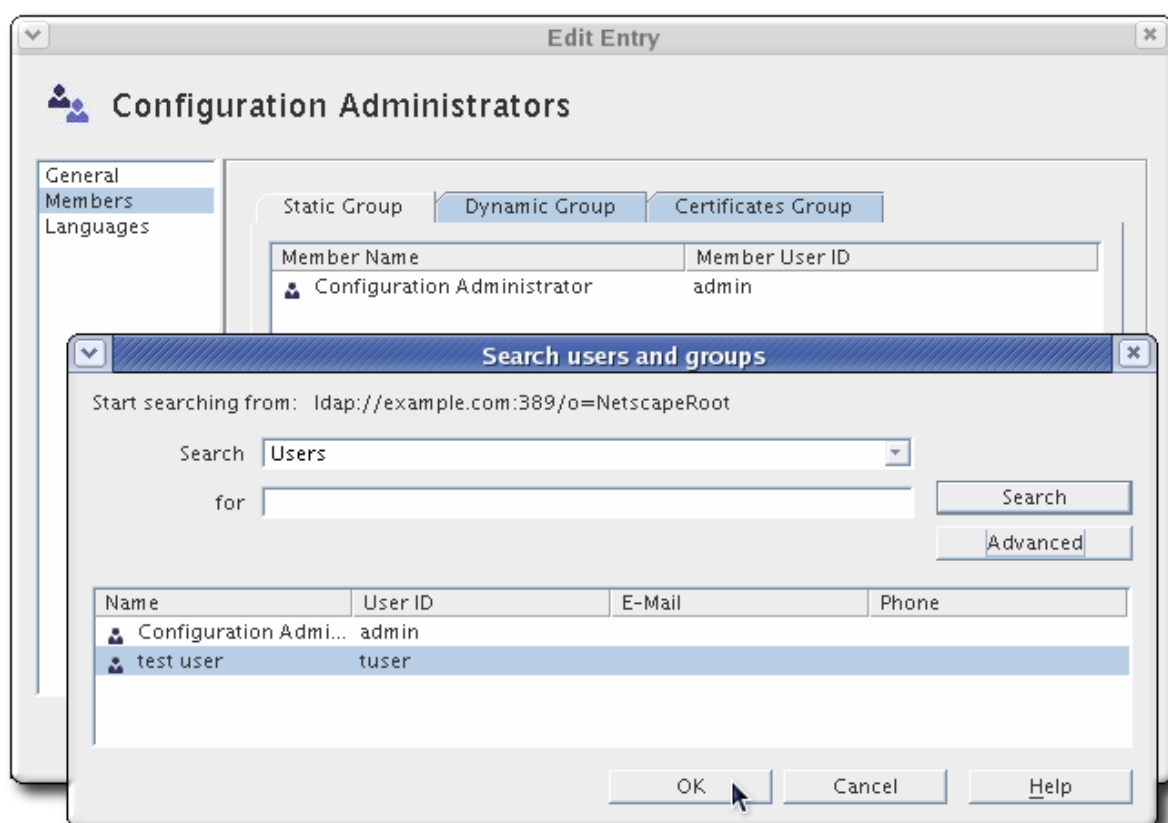
```
service dirsrv-admin restart
```

#### G.4.3.3.3. Adding Users to the Configuration Administrators Group

1. In the **Users and Groups** tab, click the **User** menu in the top menu and select **Change Directory**.
2. Change to the **o=NetScapeRoot** subtree, which contains the configuration information and the Configuration Administrators group.



3. Search for the **Configuration Administrators** group, and click **Edit**.
4. Click the **Members** link in the left of the edit window.
5. Click **Add**, and search for the user to add to the group.



## NOTE

Only users in the **o=NetscapeRoot** database can be added to the Configuration Administrators group. This means that the entry must be created as an administrator, not a regular user, when added through the Console. See [Section G.4.2.1, “Directory and Administrative Users”](#).

### G.4.3.4. Removing an Entry from the Directory

1. Search for the entry to deleted.

See [Section G.4.1, “Searching for Users and Groups”](#) for more information on searching for entries.



#### NOTE

All entries must be removed from under an organization unit before it can be deleted.

2. Select the entry in the results list, and click **Delete**. Click **OK** to confirm the deletion.

## G.5. SETTING ACCESS CONTROLS

Access control instructions (ACIs) can be set in the Red Hat Management Console to set limits on what users can see and what operations they can perform on Red Hat Directory Server and Admin Server instances managed in the Console.

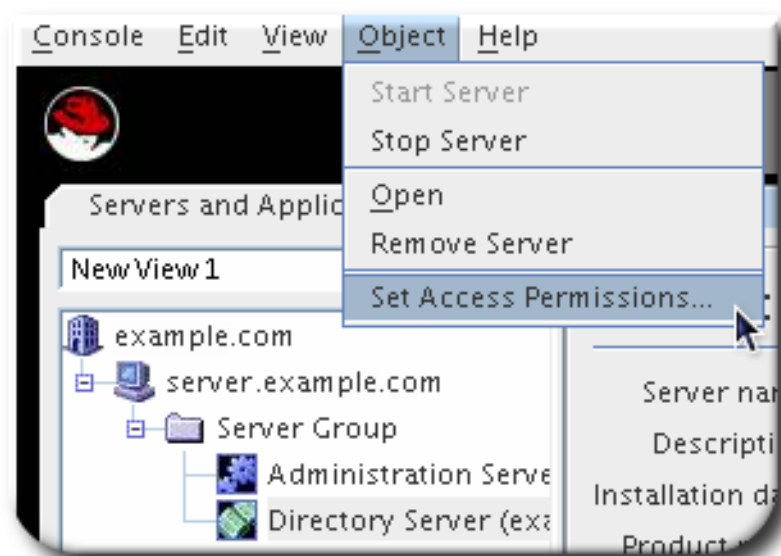
ACIs define what operations users can do with a specific instance of Red Hat Directory Server or Admin Server. ACIs set rules on areas of the subtree which can be accessed or modified, what operations are allowed, even what hosts can be used to access the server and what times of day access is allowed.

For Red Hat Management Console, access controls can be used to grant administrative privileges very easily to specific users and to set restrictions on different aspects of the main Console, such as searching the directory, adding and editing users and groups, and editing server or Console settings.

### G.5.1. Granting Admin Privileges to Users for Directory Server and Admin Server

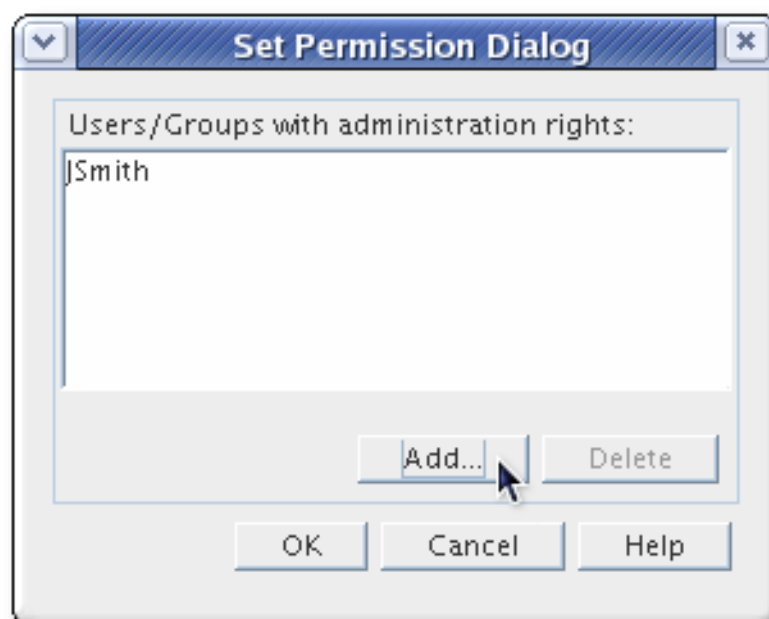
Users can be granted administrative privileges, the same as the **admin** user for the Admin Server and similar to the **cn=Directory Manager** user in Directory Server (though not exactly the same as the Directory Manager, which is a special user).

1. Highlight a server in the Console navigation tree.
2. Select the **Object** menu, and choose **Set Access Permissions**.

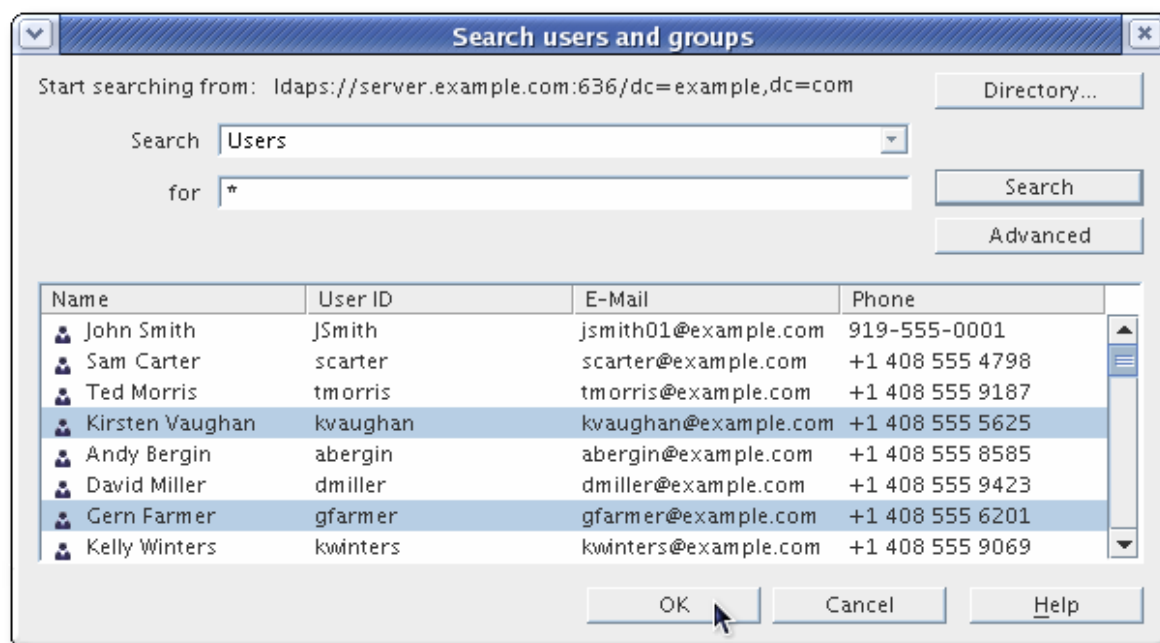


Alternatively, right-click the entry, and choose **Set Access Permissions**.

- Click **Add** to add a new user to the list of administrators for the server. The default users, **Directory Manager** for the Directory Server and **admin** for the Admin Server, are not listed in the **Set Permissions Dialog** box.



- Search for the users to add as administrators. In the results, highlight the selected users, and click **Add** to add them to the administrators list.



For more information on searching for users and groups, see [Section G.4.1, "Searching for Users and Groups"](#).

- Click **OK** to add the names to the **Set Permissions Dialog** list, then click **OK** again to save the changes and close the dialog.





## NOTE

Granting a user the right to administer a server does not automatically allow that user to give others the same right. To allow a user to grant administrative rights to other users, add that user to the Configuration Administrators group, as described in [Section G.4.3.3.3, “Adding Users to the Configuration Administrators Group”](#).

### G.5.2. Setting Access Permissions on Console Elements

There are five elements defined in the Console for access control rules:

- User and Groups Tab (viewing)
- User and Groups Tab (editing)
- Topology Tab (editing)
- Custom View Tab (editing)
- Server Security (editing)

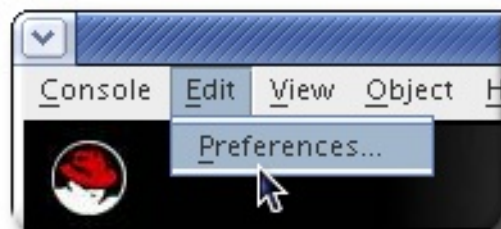
By default, each of these Console elements has five inherited ACIs:

- Enabling anonymous access
- Default anonymous access
- Configuration administrator's modifications
- Enabling group expansions
- SIE (host) group permissions

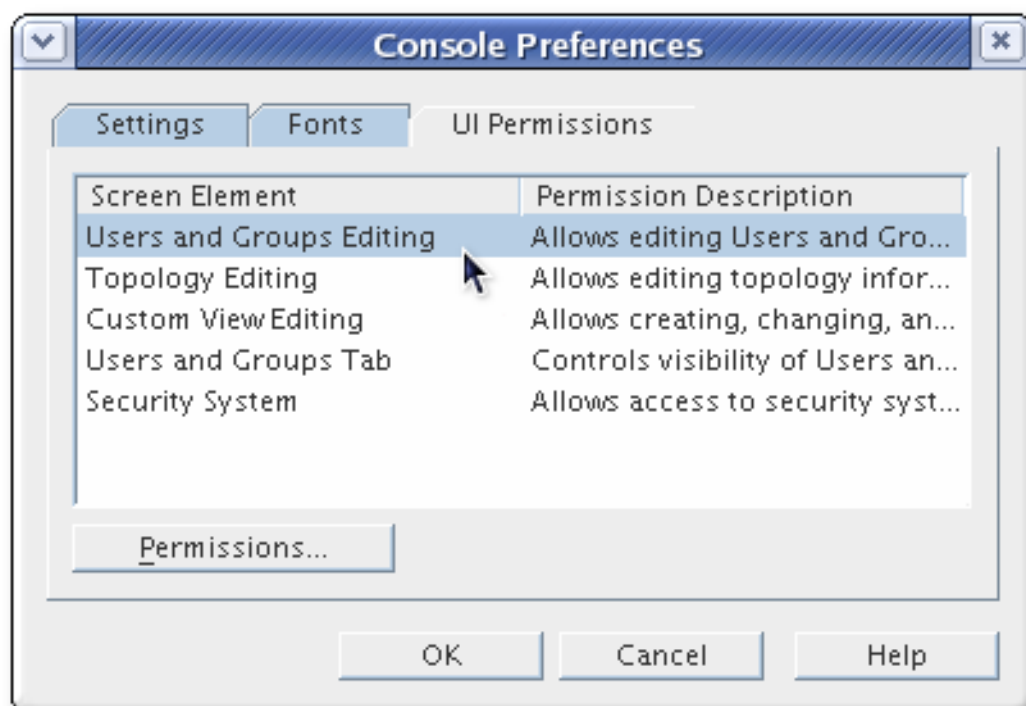
These inherited ACIs cannot be edited, but new ACIs can be added for each Console element in addition to these defaults. Additional ACIs can limit anonymous access, for example, and change other permissions within the Red Hat Management Console, which, in turn, affects access to the Directory Server and Admin Server instances.

To create new ACIs:

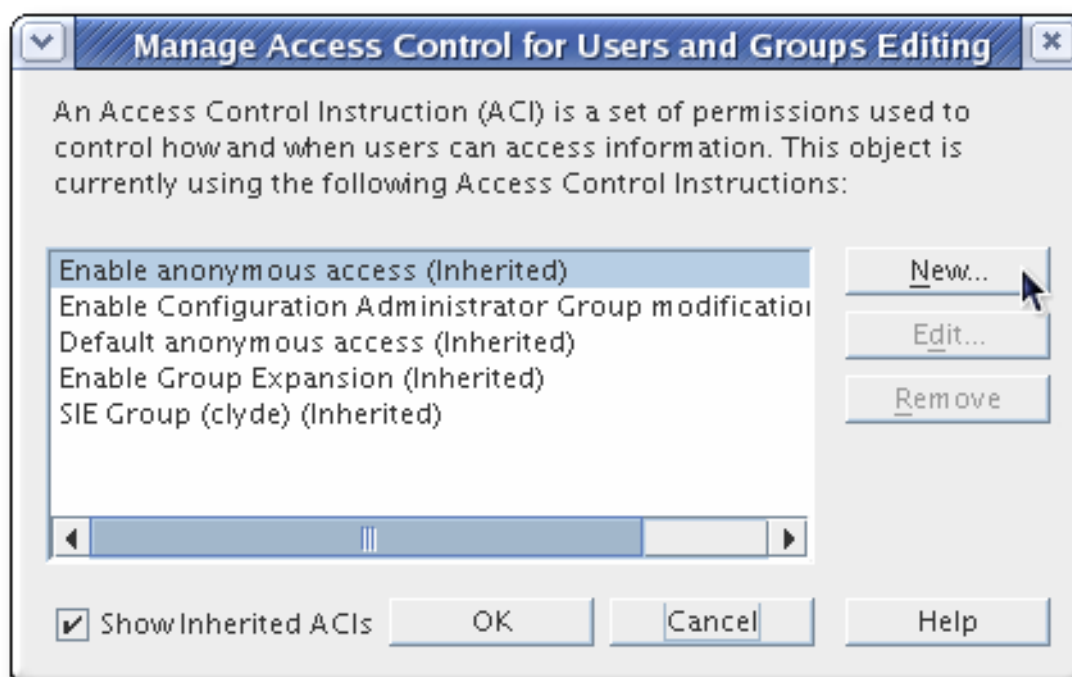
1. In the top menu, select **Edit** and then **Preferences**.



2. Select the Console element from the list, and click the **Permissions** button.

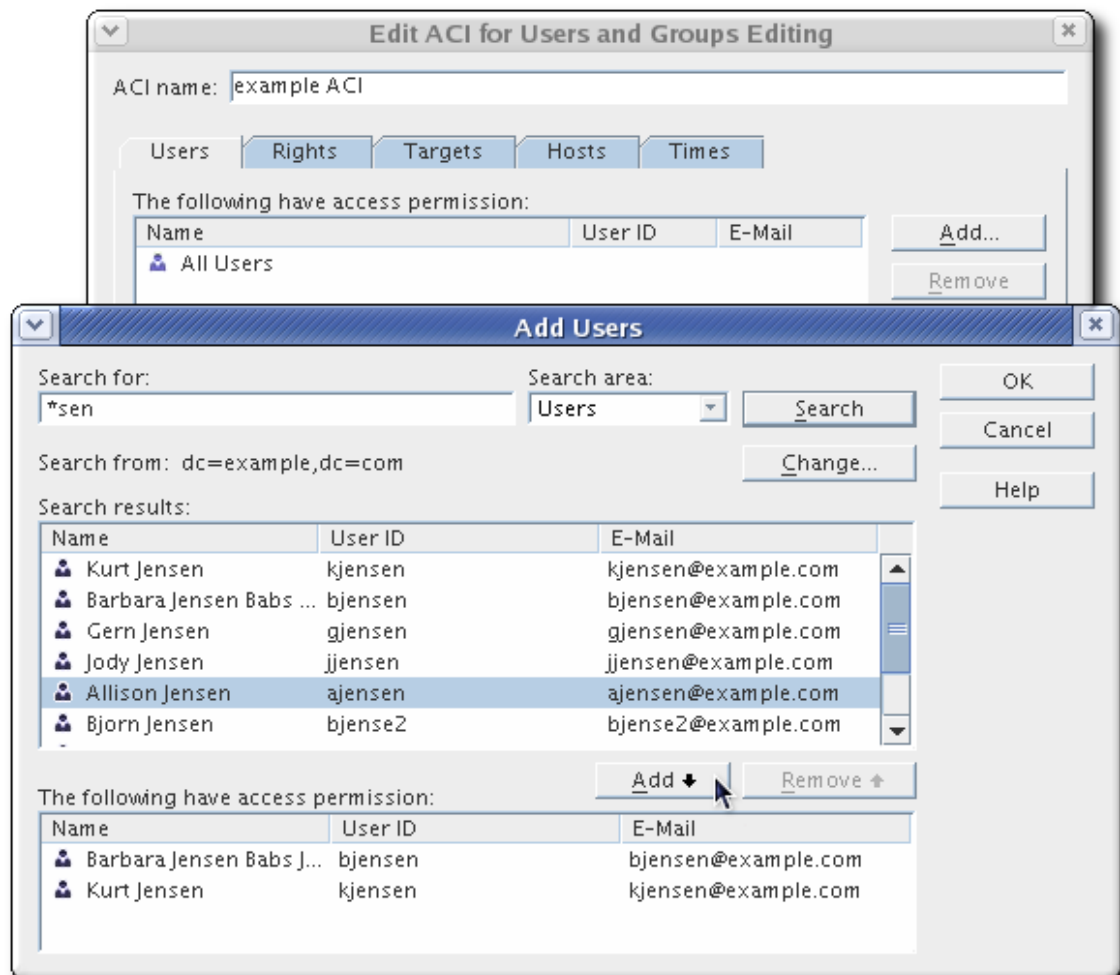


3. In the **ACI Manager** window, click the **New** button.



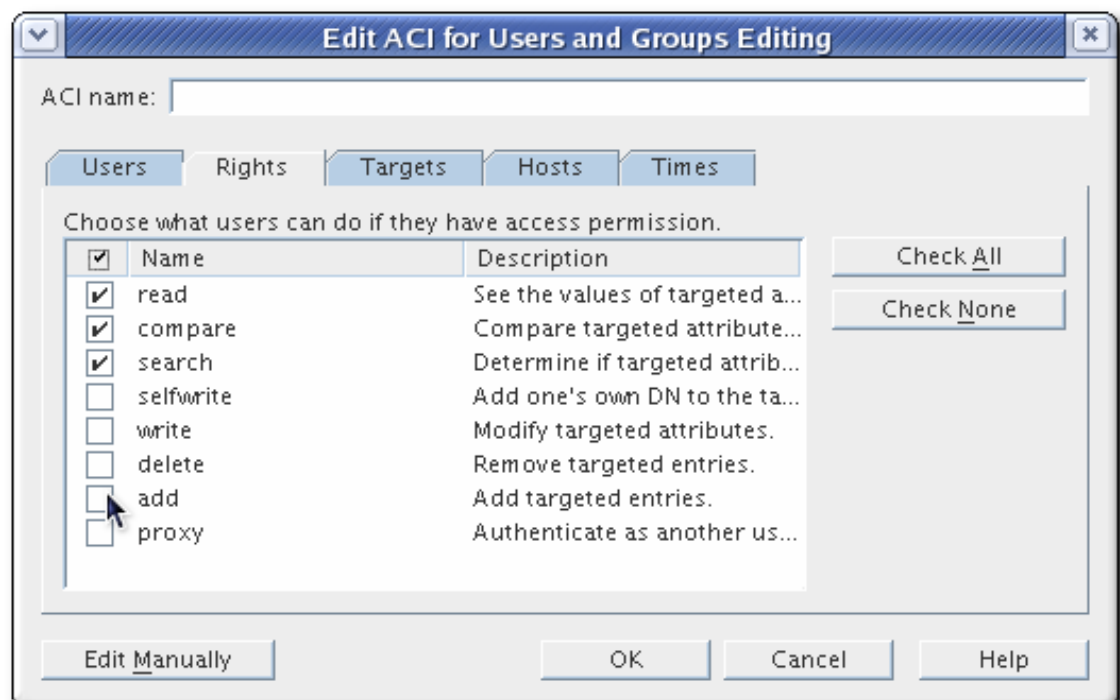
The five inherited ACIs are not displayed by default; to see them listed, click the **Show inherited ACIs** check box.

4. Configure the ACI by setting, at a minimum, the users to which it applies and the rights which are allowed. To configure the ACI in the wizard (visually):
  1. Enter a name for the ACI in the **ACI Name** field.
  2. In the **Users/Groups** tab, click the **Add** button to open the search window. Search for and add the users to which apply the ACI.



Select the users from the results list and click the **Add** button to include them. Click **OK** to save the list.

3. In the **Rights** tab, specify which operations are permitted as part of this ACI.

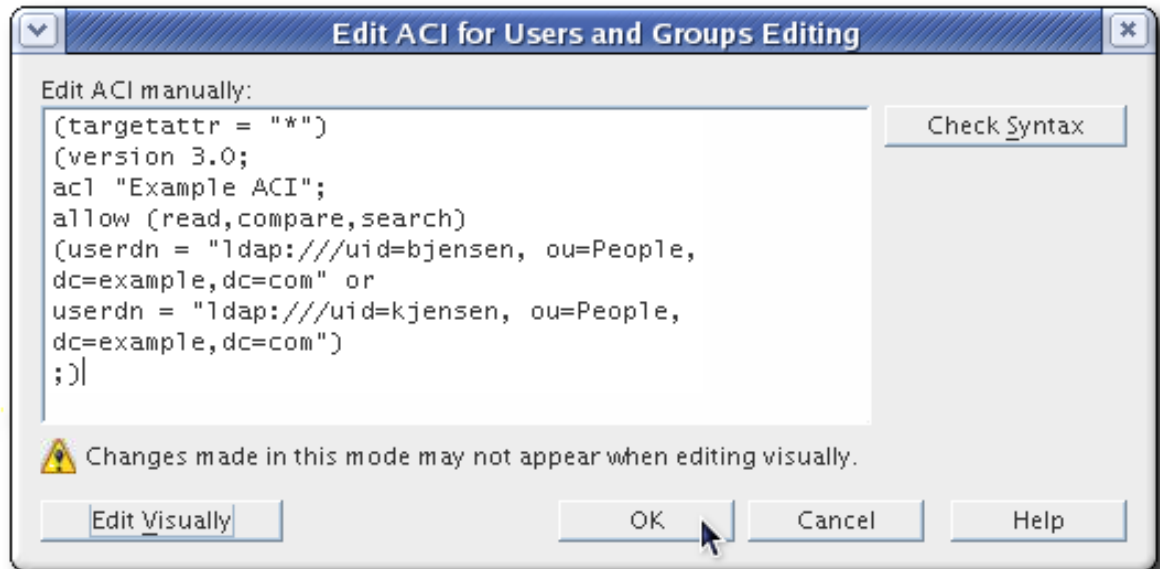


To hide a Console element entirely from the selected users, groups, and hosts, click **Check**

**None** to block any access.

4. Optionally, set the target entry in the subtree, host names, or times of day where the ACI is in effect.

More complex ACIs may not be able to be edited visually; in those cases, click the **Edit Manually** button, and configure the ACI entry directly.



Use the **Check syntax** button to validate the ACI.

5. Click **OK** to save the ACI.
6. Restart Red Hat Management Console to apply the new ACI.

## GLOSSARY

### A

#### access control instruction

See [ACI](#).

#### access control list

See [ACL](#).

#### access rights

In the context of access control, specify the level of access granted or denied. Access rights are related to the type of operation that can be performed on the directory. The following rights can be granted or denied: read, write, add, delete, search, compare, selfwrite, proxy and all.

#### account inactivation

Disables a user account, group of accounts, or an entire domain so that all authentication attempts are automatically rejected.

### ACI

An instruction that grants or denies permissions to entries in the directory.

See Also [access control instruction](#).

## ACL

The mechanism for controlling access to your directory.

See Also [access control list](#).

## All IDs Threshold

*Replaced with the ID list scan limit in Directory Server version 7.1.* A size limit which is globally applied to every index key managed by the server. When the size of an individual ID list reaches this limit, the server replaces that ID list with an All IDs token.

See Also [ID list scan limit](#).

## All IDs token

A mechanism which causes the server to assume that all directory entries match the index key. In effect, the All IDs token causes the server to behave as if no index was available for the search request.

## anonymous access

When granted, allows anyone to access directory information without providing credentials, and regardless of the conditions of the bind.

## approximate index

Allows for efficient approximate or "sounds-like" searches.

## attribute

Holds descriptive information about an entry. Attributes have a label and a value. Each attribute also follows a standard syntax for the type of information that can be stored as the attribute value.

## attribute list

A list of required and optional attributes for a given entry type or object class.

## authenticating directory server

In pass-through authentication (PTA), the authenticating Directory Server is the Directory Server that contains the authentication credentials of the requesting client. The PTA-enabled host sends PTA requests it receives from clients to the host.

## authentication

(1) Process of proving the identity of the client user to the Directory Server. Users must provide a bind DN and either the corresponding password or certificate in order to be granted access to the directory. Directory Server allows the user to perform functions or access files and directories based on the permissions granted to that user by the directory administrator.

(2) Allows a [client](#) to make sure they are connected to a secure server, preventing another computer from impersonating the server or attempting to appear secure when it is not.

## authentication certificate

Digital file that is not transferable and not forgeable and is issued by a third party. Authentication certificates are sent from server to client or client to server in order to verify and authenticate the other party.

## B

### **base distinguished name**

See [base DN](#).

### **base DN**

Base distinguished name. A search operation is performed on the base DN, the DN of the entry and all entries below it in the directory tree.

### **bind distinguished name**

See [bind DN](#).

### **bind DN**

Distinguished name used to authenticate to Directory Server when performing an operation.

### **bind rule**

In the context of access control, the bind rule specifies the credentials and conditions that a particular user or client must satisfy in order to get access to directory information.

### **branch entry**

An entry that represents the top of a subtree in the directory.

### **browser**

Software, such as Mozilla Firefox, used to request and view World Wide Web material stored as HTML files. The browser uses the HTTP protocol to communicate with the host server.

### **browsing index**

Speeds up the display of entries in the Directory Server Console. Browsing indexes can be created on any branch point in the directory tree to improve display performance.

See Also [virtual list view index](#) .

## C

### **CA**

See [Certificate Authority](#).

### **cascading replication**

In a cascading replication scenario, one server, often called the hub supplier, acts both as a consumer and a supplier for a particular replica. It holds a read-only replica and maintains a changelog. It receives updates from the supplier server that holds the master copy of the data and in turn supplies those updates to the consumer.

### **certificate**

A collection of data that associates the public keys of a network user with their DN in the directory. The certificate is stored in the directory as user object attributes.

**Certificate Authority**

Company or organization that sells and issues authentication certificates. You may purchase an authentication certificate from a Certification Authority that you trust. Also known as a [CA](#).

**CGI**

Common Gateway Interface. An interface for external programs to communicate with the HTTP server. Programs written to use CGI are called CGI programs or CGI scripts and can be written in many of the common programming languages. CGI programs handle forms or perform output parsing that is not done by the server itself.

**chaining**

A method for relaying requests to another server. Results for the request are collected, compiled, and then returned to the client.

**changelog**

A changelog is a record that describes the modifications that have occurred on a replica. The supplier server then replays these modifications on the replicas stored on replica servers or on other masters, in the case of multi-master replication.

**character type**

Distinguishes alphabetic characters from numeric or other characters and the mapping of upper-case to lower-case letters.

**ciphertext**

Encrypted information that cannot be read by anyone without the proper key to decrypt the information.

**class definition**

Specifies the information needed to create an instance of a particular object and determines how the object works in relation to other objects in the directory.

**class of service**

See [CoS](#).

**classic CoS**

A classic CoS identifies the template entry by both its DN and the value of one of the target entry's attributes.

**client**

See [LDAP client](#).

**code page**

An internal table used by a locale in the context of the internationalization plug-in that the operating system uses to relate keyboard keys to character font screen displays.

**collation order**

Provides language and cultural-specific information about how the characters of a given language are to be sorted. This information might include the sequence of letters in the alphabet or how to compare letters with accents to letters without accents.

**consumer**

Server containing replicated directory trees or subtrees from a supplier server.

**consumer server**

In the context of replication, a server that holds a replica that is copied from a different server is called a consumer for that replica.

**CoS**

A method for sharing attributes between entries in a way that is invisible to applications.

**CoS definition entry**

Identifies the type of CoS you are using. It is stored as an LDAP subentry below the branch it affects.

**CoS template entry**

Contains a list of the shared attribute values.

See Also [template entry](#).

**D****daemon**

A background process on a Unix machine that is responsible for a particular system task. Daemon processes do not need human intervention to continue functioning.

**DAP**

Directory Access Protocol. The ISO X.500 standard protocol that provides client access to the directory.

**data master**

The server that is the master source of a particular piece of data.

**database link**

An implementation of chaining. The database link behaves like a database but has no persistent storage. Instead, it points to data stored remotely.

**default index**

One of a set of default indexes created per database instance. Default indexes can be modified, although care should be taken before removing them, as certain plug-ins may depend on them.

**definition entry**

See [CoS definition entry](#).

**Directory Access Protocol**

See [DAP](#).

**Directory Manager**

The privileged database administrator, comparable to the root user in UNIX. Access control does not apply to the Directory Manager.



**directory service**

A database application designed to manage descriptive, attribute-based information about people and resources within an organization.

**directory tree**

The logical representation of the information stored in the directory. It mirrors the tree model used by most filesystems, with the tree's root point appearing at the top of the hierarchy. Also known as [DIT](#).

**distinguished name**

String representation of an entry's name and location in an LDAP directory.

**DIT**

See [directory tree](#).

**DM**

See [Directory Manager](#).

**DN**

See [distinguished name](#).

**DNS**

Domain Name System. The system used by machines on a network to associate standard IP addresses (such as 198.93.93.10) with host names (such as **www.example.com**). Machines normally get the IP address for a host name from a DNS server, or they look it up in tables maintained on their systems.

**DNS alias**

A DNS alias is a host name that the DNS server knows points to a different host specifically a DNS CNAME record. Machines always have one real name, but they can have one or more aliases. For example, an alias such as **www.yourdomain.domain** might point to a real machine called **realthing.yourdomain.domain** where the server currently exists.

**E****entry**

A group of lines in the LDIF file that contains information about an object.

**entry distribution**

Method of distributing directory entries across more than one server in order to scale to support large numbers of entries.

**entry ID list**

Each index that the directory uses is composed of a table of index keys and matching entry ID lists. The entry ID list is used by the directory to build a list of candidate entries that may match the client application's search request.

**equality index**

Allows you to search efficiently for entries containing a specific attribute value.

**F**

**file extension**

The section of a filename after the period or dot (.) that typically defines the type of file (for example, .GIF and .HTML). In the filename **index.html** the file extension is **html**.

**file type**

The format of a given file. For example, graphics files are often saved in GIF format, while a text file is usually saved as ASCII text format. File types are usually identified by the file extension (for example, .GIF or .HTML).

**filter**

A constraint applied to a directory query that restricts the information returned.

**filtered role**

Allows you to assign entries to the role depending upon the attribute contained by each entry. You do this by specifying an LDAP filter. Entries that match the filter are said to possess the role.

**G****general access**

When granted, indicates that all authenticated users can access directory information.

**GSS-API**

Generic Security Services. The generic access protocol that is the native way for UNIX-based systems to access and authenticate Kerberos services; also supports session encryption.

**H****host name**

A name for a machine in the form machine.domain.dom, which is translated into an IP address. For example, **www.example.com** is the machine **www** in the subdomain **example** and **com** domain.

**HTML**

Hypertext Markup Language. The formatting language used for documents on the World Wide Web. HTML files are plain text files with formatting codes that tell browsers such as the Mozilla Firefox how to display text, position graphics, and form items and to display links to other pages.

**HTTP**

Hypertext Transfer Protocol. The method for exchanging information between HTTP servers and clients.

**HTTPD**

An abbreviation for the HTTP daemon or service, a program that serves information using the HTTP protocol. The daemon or service is often called an httpd.

**HTTPS**

A secure version of HTTP, implemented using the Secure Sockets Layer, SSL.

**hub**

In the context of replication, a server that holds a replica that is copied from a different server, and, in turn, replicates it to a third server.

See Also [cascading replication](#).

## I

### **ID list scan limit**

A size limit which is globally applied to any indexed search operation. When the size of an individual ID list reaches this limit, the server replaces that ID list with an all IDs token.

### **index key**

Each index that the directory uses is composed of a table of index keys and matching entry ID lists.

### **indirect CoS**

An indirect CoS identifies the template entry using the value of one of the target entry's attributes.

### **international index**

Speeds up searches for information in international directories.

### **International Standards Organization**

See [ISO](#).

### **IP address**

*Also Internet Protocol address.* A set of numbers, separated by dots, that specifies the actual location of a machine on the Internet (for example, 198.93.93.10). Directory Server supports both IPv4 and IPv6 IP addresses.

## ISO

International Standards Organization.

## K

### **knowledge reference**

Pointers to directory information stored in different databases.

## L

### **LDAP**

Lightweight Directory Access Protocol. Directory service protocol designed to run over TCP/IP and across multiple platforms.

### **LDAP client**

Software used to request and view LDAP entries from an LDAP Directory Server.

See Also [browser](#).

### **LDAP Data Interchange Format**

See [LDAP Data Interchange Format](#).

**LDAP URL**

Provides the means of locating Directory Servers using DNS and then completing the query via LDAP. A sample LDAP URL is **ldap://ldap.example.com**.

**LDAPv3**

Version 3 of the LDAP protocol, upon which Directory Server bases its schema format.

**LDBM database**

A high-performance, disk-based database consisting of a set of large files that contain all of the data assigned to it. The primary data store in Directory Server.

**LDIF**

LDAP Data Interchange Format. Format used to represent Directory Server entries in text form.

**leaf entry**

An entry under which there are no other entries. A leaf entry cannot be a branch point in a directory tree.

**Lightweight Directory Access Protocol**

See [LDAP](#).

**locale**

Identifies the collation order, character type, monetary format and time / date format used to present data for users of a specific region, culture, and/or custom. This includes information on how data of a given language is interpreted, stored, or collated. The locale also indicates which code page should be used to represent a given language.

**M****managed object**

A standard value which the SNMP agent can access and send to the NMS. Each managed object is identified with an official name and a numeric identifier expressed in dot-notation.

**managed role**

Allows creation of an explicit enumerated list of members.

**management information base**

See [MIB](#).

**mapping tree**

A data structure that associates the names of suffixes (subtrees) with databases.

**master**

See [supplier](#).

**master agent**

See [SNMP master agent](#).

**matching rule**

Provides guidelines for how the server compares strings during a search operation. In an international search, the matching rule tells the server what collation order and operator to use.

**MD5**

A message digest algorithm by RSA Data Security, Inc., which can be used to produce a short digest of data that is unique with high probability and is mathematically extremely hard to produce; a piece of data that will produce the same message digest.

**MD5 signature**

A message digest produced by the MD5 algorithm.

**MIB**

Management Information Base. All data, or any portion thereof, associated with the SNMP network. We can think of the MIB as a database which contains the definitions of all SNMP managed objects. The MIB has a tree-like hierarchy, where the top level contains the most general information about the network and lower levels deal with specific, separate network areas.

**MIB namespace**

Management Information Base namespace. The means for directory data to be named and referenced. Also called the [directory tree](#).

**monetary format**

Specifies the monetary symbol used by specific region, whether the symbol goes before or after its value, and how monetary units are represented.

**multi-master replication**

An advanced replication scenario in which two servers each hold a copy of the same read-write replica. Each server maintains a changelog for the replica. Modifications made on one server are automatically replicated to the other server. In case of conflict, a time stamp is used to determine which server holds the most recent version.

**multiplexor**

The server containing the database link that communicates with the remote server.

**N****n + 1 directory problem**

The problem of managing multiple instances of the same information in different directories, resulting in increased hardware and personnel costs.

**name collisions**

Multiple entries with the same distinguished name.

**nested role**

Allows the creation of roles that contain other roles.

**network management application**

Network Management Station component that graphically displays information about SNMP managed devices, such as which device is up or down and which and how many error messages were received.

**network management station**

See [NMS](#).

**NIS**

Network Information Service. A system of programs and data files that Unix machines use to collect, collate, and share specific information about machines, users, filesystems, and network parameters throughout a network of computers.

**NMS**

Powerful workstation with one or more network management applications installed. Also [network management station](#).

**ns-slapd**

Red Hat's LDAP Directory Server daemon or service that is responsible for all actions of the Directory Server.

See Also [slapd](#).

**O****object class**

Defines an entry type in the directory by defining which attributes are contained in the entry.

**object identifier**

A string, usually of decimal numbers, that uniquely identifies a schema element, such as an object class or an attribute, in an object-oriented system. Object identifiers are assigned by ANSI, IETF or similar organizations.

See Also [OID](#).

**OID**

See [object identifier](#).

**operational attribute**

Contains information used internally by the directory to keep track of modifications and subtree properties. Operational attributes are not returned in response to a search unless explicitly requested.

**P****parent access**

When granted, indicates that users have access to entries below their own in the directory tree if the bind DN is the parent of the targeted entry.

**pass-through authentication**

See [PTA](#).

**pass-through subtree**

In pass-through authentication, the [PTA directory server](#) will pass through bind requests to the [authenticating directory server](#) from all clients whose DN is contained in this subtree.

**password file**

A file on Unix machines that stores Unix user login names, passwords, and user ID numbers. It is also known as `/etc/passwd` because of where it is kept.

**password policy**

A set of rules that governs how passwords are used in a given directory.

**PDU**

Encoded messages which form the basis of data exchanges between SNMP devices. Also [protocol data unit](#).

**permission**

In the context of access control, permission states whether access to the directory information is granted or denied and the level of access that is granted or denied.

See Also [access rights](#).

**pointer CoS**

A pointer CoS identifies the template entry using the template DN only.

**presence index**

Allows searches for entries that contain a specific indexed attribute.

**protocol**

A set of rules that describes how devices on a network exchange information.

**protocol data unit**

See [PDU](#).

**proxy authentication**

A special form of authentication where the user requesting access to the directory does not bind with its own DN but with a proxy DN.

**proxy DN**

Used with proxied authorization. The proxy DN is the DN of an entry that has access permissions to the target on which the client-application is attempting to perform an operation.

**PTA**

Mechanism by which one Directory Server consults another to check bind credentials. Also [pass-through authentication](#).

**PTA directory server**

In pass-through authentication ([PTA](#)), the PTA Directory Server is the server that sends (passes through) bind requests it receives to the [authenticating directory server](#).

**PTA LDAP URL**

In pass-through authentication, the URL that defines the [authenticating directory server](#), pass-through subtree(s), and optional parameters.

## R

### RAM

Random access memory. The physical semiconductor-based memory in a computer. Information stored in RAM is lost when the computer is shut down.

### rc.local

A file on Unix machines that describes programs that are run when the machine starts. It is also called **/etc/rc.local** because of its location.

### RDN

The name of the actual entry itself, before the entry's ancestors have been appended to the string to form the full distinguished name. Also [relative distinguished name](#).

### read-only replica

A replica that refers all update operations to read-write replicas. A server can hold any number of read-only replicas.

### read-write replica

A replica that contains a master copy of directory information and can be updated. A server can hold any number of read-write replicas.

### referential integrity

Mechanism that ensures that relationships between related entries are maintained within the directory.

### referral

- (1) When a server receives a search or update request from an LDAP client that it cannot process, it usually sends back to the client a pointer to the LDAP server that can process the request.
- (2) In the context of replication, when a read-only replica receives an update request, it forwards it to the server that holds the corresponding read-write replica. This forwarding process is called a referral.

### relative distinguished name

See [RDN](#).

### replica

A database that participates in replication.

### replica-initiated replication

Replication configuration where replica servers, either hub or consumer servers, pull directory data from supplier servers. This method is available only for legacy replication.

### replication

Act of copying directory trees or subtrees from supplier servers to replica servers.



**replication agreement**

Set of configuration parameters that are stored on the supplier server and identify the databases to replicate, the replica servers to which the data is pushed, the times during which replication can occur, the DN and credentials used by the supplier to bind to the consumer, and how the connection is secured.

**RFC**

Request for Comments. Procedures or standards documents submitted to the Internet community. People can send comments on the technologies before they become accepted standards.

**role**

An entry grouping mechanism. Each role has *members*, which are the entries that possess the role.

**role-based attributes**

Attributes that appear on an entry because it possesses a particular role within an associated CoS template.

**root**

The most privileged user available on Unix machines. The root user has complete access privileges to all files on the machine.

**root suffix**

The parent of one or more sub suffixes. A directory tree can contain more than one root suffix.

**S****SASL**

An authentication framework for clients as they attempt to bind to a directory. Also [Simple Authentication and Security Layer](#) .

**schema**

Definitions describing what types of information can be stored as entries in the directory. When information that does not match the schema is stored in the directory, clients attempting to access the directory may be unable to display the proper results.

**schema checking**

Ensures that entries added or modified in the directory conform to the defined schema. Schema checking is on by default, and users will receive an error if they try to save an entry that does not conform to the schema.

**Secure Sockets Layer**

See [SSL](#).

**self access**

When granted, indicates that users have access to their own entries if the bind DN matches the targeted entry.

**Server Console**

Java-based application that allows you to perform administrative management of your Directory Server from a GUI.

**server daemon**

The server daemon is a process that, once running, listens for and accepts requests from clients.

**Server Selector**

Interface that allows you select and configure servers using a browser.

**server service**

A process on Windows that, once running, listens for and accepts requests from clients. It is the SMB server on Windows NT.

**service**

A background process on a Windows machine that is responsible for a particular system task. Service processes do not need human intervention to continue functioning.

**SIE**

Server Instance Entry. The ID assigned to an instance of Directory Server during installation.

**Simple Authentication and Security Layer**

See [SASL](#).

**Simple Network Management Protocol**

See [SNMP](#).

**single-master replication**

The most basic replication scenario in which multiple servers, up to four, each hold a copy of the same read-write replicas to replica servers. In a single-master replication scenario, the supplier server maintains a changelog.

**SIR**

See [supplier-initiated replication](#).

**slapd**

LDAP Directory Server daemon or service that is responsible for most functions of a directory except replication.

See Also [ns-slapd](#).

**SNMP**

Used to monitor and manage application processes running on the servers by exchanging data about network activity. Also [Simple Network Management Protocol](#).

**SNMP master agent**

Software that exchanges information between the various subagents and the NMS.

**SNMP subagent**

Software that gathers information about the managed device and passes the information to the master agent. Also called a [subagent](#).

**SSL**

A software library establishing a secure connection between two parties (client and server) used to implement HTTPS, the secure version of HTTP. Also called [Secure Sockets Layer](#).

**standard index**

index maintained by default.

**sub suffix**

A branch underneath a root suffix.

**subagent**

See [SNMP subagent](#).

**substring index**

Allows for efficient searching against substrings within entries. Substring indexes are limited to a minimum of two characters for each entry.

**suffix**

The name of the entry at the top of the directory tree, below which data is stored. Multiple suffixes are possible within the same directory. Each database only has one suffix.

**superuser**

The most privileged user available on Unix machines. The superuser has complete access privileges to all files on the machine. Also called [root](#).

**supplier**

Server containing the master copy of directory trees or subtrees that are replicated to replica servers.

**supplier server**

In the context of replication, a server that holds a replica that is copied to a different server is called a supplier for that replica.

**supplier-initiated replication**

Replication configuration where [supplier](#) servers replicate directory data to any replica servers.

**symmetric encryption**

Encryption that uses the same key for both encrypting and decrypting. DES is an example of a symmetric encryption algorithm.

**system index**

Cannot be deleted or modified as it is essential to Directory Server operations.

**T****target**

In the context of access control, the target identifies the directory information to which a particular ACI applies.

**target entry**

The entries within the scope of a CoS.

**TCP/IP**

Transmission Control Protocol/Internet Protocol. The main network protocol for the Internet and for enterprise (company) networks.

**template entry**

See [CoS template entry](#).

**time/date format**

Indicates the customary formatting for times and dates in a specific region.

**TLS**

The new standard for secure socket layers; a public key based protocol. Also [Transport Layer Security](#).

**topology**

The way a directory tree is divided among physical servers and how these servers link with one another.

**Transport Layer Security**

See [TLS](#).

**U****uid**

A unique number associated with each user on a Unix system.

**URL**

Uniform Resource Locator. The addressing system used by the server and the client to request documents. It is often called a location. The format of a URL is *protocol://machine:port/document*. The port number is necessary only on selected servers, and it is often assigned by the server, freeing the user of having to place it in the URL.

**V****virtual list view index**

Speeds up the display of entries in the Directory Server Console. Virtual list view indexes can be created on any branch point in the directory tree to improve display performance.

See Also [browsing index](#).

**X****X.500 standard**

The set of ISO/ITU-T documents outlining the recommended information model, object classes and attributes used by directory server implementation.

---

# INDEX

## A

### access control

ACI attribute, [ACI Structure](#)

ACI syntax, [The ACI Syntax](#)

allowing or denying access, [Allowing or Denying Access](#)

and directory manager, [Setting Access Controls on Directory Manager](#)

and replication, [Access Control and Replication](#)

and schema checking, [Targeting Attributes](#)

anonymous access, [Anonymous Access \(anyone Keyword\)](#)

bind rules, [Bind Rules](#)

    access at specific time or day, [Defining Access at a Specific Time of Day or Day of Week](#)

    access based on value matching, [Defining Access Based on Value Matching](#)

    general access, [General Access \(all Keyword\)](#)

    user and group access, [Defining User Access - userdn Keyword](#)

Boolean bind rules, [Using Boolean Bind Rules](#)

compatibility with earlier versions, [Compatibility with Earlier Releases](#)

creating from console, [Creating ACIs from the Console](#)

dynamic targets, [LDAP URLs](#)

for a specific level of secure connection, [Requiring a Certain Level of Security in Connections](#)

from specific domain, [Defining Access from a Specific Domain](#)

from specific IP address, [Defining Access from a Specific IP Address](#)

logging information, [Logging Access Control Information](#)

overview, [Managing Access Control](#)

permissions, [Defining Permissions](#)

placement of ACIs, [ACI Placement](#)

rights, [Assigning Rights](#)

roles, [Using Roles Securely](#)

SASL authentication, [Defining Access Based on Authentication Method](#)

simple authentication, [Defining Access Based on Authentication Method](#)

SSL authentication, [Defining Access Based on Authentication Method](#)

structure of ACIs, [ACI Structure](#)

target DN

    containing comma, [Targeting a Directory Entry](#)

target DN containing comma, [Defining Permissions for DN's That Contain a Comma](#)

targeting, [Defining Targets](#)

targeting attribute values, [Targeting Attribute Values Using LDAP Filters](#)

targeting attributes, [Targeting Attributes](#)

targeting entries, [Targeting a Directory Entry](#)

targeting using filters, [Targeting Entries or Attributes Using LDAP Filters](#)

using the Access Control Editor, [Creating ACIs from the Console](#)

value matching, [Defining Access Based on Value Matching](#)

viewing

Access Control Editor, [Viewing ACIs](#)

get effective rights, [Checking Access Rights on Entries \(Get Effective Rights\)](#)

## Access Control

to navigation tree, [Granting Admin Privileges to Users for Directory Server and Admin Server](#)

## Access Control Editor

displaying, [Displaying the Access Control Editor](#)

access control instruction (ACI). See [ACI](#), [ACI Structure](#)

## access log

changing location and name

in the command line, [Changing the Log Location in the Command Line](#)

in the Console, [Changing the Log Name in the Console](#)

configuring

deletion policy, [Defining a Log File Deletion Policy](#)

rotation policy, [Defining a Log File Rotation Policy](#)

defined, [Viewing Logs](#)

manually rotating, [Manual Log File Rotation](#)

viewing, [Viewing Log Files](#)

viewing in command line, [Viewing Logs in the Command Line](#)

viewing in Console, [Viewing the Logs through the Console](#)

## access settings

for Admin Server, [Changing the Admin User's Name and Password](#)

## account inactivation, [Manually Inactivating Users and Roles](#)

from command line, [Inactivating and Activating Users and Roles Using the Command Line](#)

from console, [Activating and Inactivating Users and Roles Using the Console](#)

PAM pass-through authentication, [Setting PAM PTA Mappings](#)

## account lockout, [Configuring the Account Lockout Policy Using the Console](#)

configuration

attributes, [Configuring the Account Lockout Policy Using the Command Line](#)

configuring

using command line, [Configuring the Account Lockout Policy Using the Command Line](#)

using console, [Configuring the Account Lockout Policy Using the Console](#)

configuring password-based, [Configuring a Password-Based Account Lockout Policy](#)  
 configuring time-based, [Configuring Time-Based Account Lockout Policies](#)  
 disabling, [Configuring the Account Lockout Policy Using the Console](#)  
 enabling, [Configuring the Account Lockout Policy Using the Console](#)  
 lockout duration, [Configuring the Account Lockout Policy Using the Console](#)  
 password failure counter, [Configuring the Account Lockout Policy Using the Console](#)  
 replicating attributes, [Replicating Account Lockout Attributes](#)  
 replication, [Managing the Account Lockouts and Replication](#)

## account policy

configuring, [Configuring Time-Based Account Lockout Policies](#)

## ACI, [Access Control Principles](#)

and directory manager, [Setting Access Controls on Directory Manager](#)

assessment, [ACI Structure](#)

attribute, [ACI Placement](#)

authmethod keyword, [Defining Access Based on Authentication Method](#)

bind rules, [The ACI Syntax](#)

cascading chaining, [Configuring Cascading Chaining from the Command Line](#)

creating from console, [Creating a New ACI](#)

dayofweek keyword, [Defining Access at a Specific Time of Day or Day of Week](#)

deleting from console, [Deleting an ACI](#)

dns keyword, [Defining Access from a Specific Domain](#)

editing from console, [Editing an ACI](#)

evaluation, [ACI Evaluation](#)

examples of use, [Access Control Usage Examples](#)

groupdn keyword, [Defining Group Access - groupdn Keyword](#)

inheritance, [Using the userattr Keyword with Inheritance](#)

ip keyword, [Defining Access from a Specific IP Address](#)

### local evaluation

cascading chaining, [Configuring Cascading Chaining from the Command Line](#)

name, [The ACI Syntax](#)

permissions, [The ACI Syntax](#)

precedence rule, [ACI Evaluation](#)

proxy rights example, [Proxied Authorization ACI Example](#)

replication, [Access Control and Replication](#)

rights, [Assigning Rights](#)

roledn keyword, [Defining Role Access - roledn Keyword](#)

ssf keyword, [Requiring a Certain Level of Security in Connections](#)

structure, [ACI Structure](#)

syntax, [The ACI Syntax](#)

target, [The ACI Syntax](#)

**target DN**

containing comma, [Targeting a Directory Entry](#)

target DN containing comma, [Defining Permissions for DNs That Contain a Comma](#)

target keywords, [Defining Targets](#)

target overview, [Defining Targets](#)

targetattr keyword, [Targeting Attributes](#)

targetattrfilters keyword, [Targeting Attribute Values Using LDAP Filters](#)

targetfilter keyword, [Targeting Entries or Attributes Using LDAP Filters](#)

userattr and parent, [Using the userattr Keyword with Inheritance](#)

userattr keyword, [Using the userattr Keyword](#)

using macro ACIs, [Advanced Access Control: Using Macro ACIs](#)

value-based, [Targeting Attribute Values Using LDAP Filters](#)

viewing current, [Viewing ACIs](#)

wildcard in target, [Targeting a Directory Entry](#)

wildcards, [Wildcards](#)

**ACI attribute**

default index for, [Overview of System Indexes](#)

overview, [ACI Structure](#)

**ACI placement, [ACI Placement](#)**

ACI targets, [Targeting a Directory Entry](#)

ACL, [Access Control Principles](#)

**activating accounts**

from command line, [Inactivating and Activating Users and Roles Using the Command Line](#)

from console, [Activating and Inactivating Users and Roles Using the Console](#)

**Active Directory**

schema differences between Directory Server, [User Schema Differences between Red Hat Directory Server and Active Directory](#), [Group Schema Differences between Red Hat Directory Server and Active Directory](#)

add right, [Assigning Rights](#)

adding directory entries, [Adding Entries Using Idapmodify](#)

**admin domain**

creating, [Creating and Editing an Admin Domain](#)

**Admin Express**

configuring, [Configuring Admin Express](#)

directives, [Admin Express Directives](#)

file locations, [Admin Express File Locations](#)

files, [Admin Express Configuration Files](#)

for replication status, [Files for the Replication Status Appearance](#)



for server information page, [Files for the Server Information Page](#)

for the server logs page, [Files for the Server Logs Page](#)

for the welcome page, [Files for the Admin Server Welcome Page](#)

opening, [Opening Admin Express](#)

replication monitoring, [Monitoring Replication from Admin Express](#)

starting and stopping servers, [Starting and Stopping Servers](#)

viewing server information, [Viewing Server Information](#)

viewing server logs, [Viewing Server Logs](#)

## Admin Server

access settings for, [Changing the Admin User's Name and Password](#)

and replication, [Replicating o=NetscapeRoot for Admin Server Failover](#)

defined, [Introduction to Red Hat Admin Server](#)

directory settings for, [Changing Directory Server Settings](#)

enabling SSL, [Enabling SSL](#)

encryption settings for, [Working with SSL](#)

logging options for, [Viewing Logs](#)

login, [Opening the Admin Server Console](#)

password file, [Creating a Password File for the Admin Server](#)

port number, [Changing the Port Number](#)

in the command line, [Changing the Port Number in the Command Line](#)

in the Console, [Changing the Port Number in the Console](#)

requesting a certificate, [Requesting and Installing a Server Certificate](#)

restarting, [Starting and Stopping the Admin Server](#)

starting and stopping, [Starting and Stopping Admin Server](#)

command line, [Starting and Stopping Admin Server from the Command Line](#)

Console, [Starting and Stopping Admin Server from the Console](#)

starting and stopping servers, [Starting and Stopping Servers](#)

starting the Console, [Opening the Admin Server Console](#)

viewing logs, [Viewing Server Logs](#)

viewing server information, [Viewing Server Information](#)

## Admin Server Console

starting, [Opening the Admin Server Console](#)

## administration domain

defined, [The Servers and Applications Tab](#)

removing, [Removing an Admin Domain](#)

## Administration Server

defined, [Overview of the Directory Server Console](#)

**Administration Server Administrator**

changing user name or password for, [Changing the Admin Password](#)  
defined, [Changing the Admin User's Name and Password](#), [Changing Administrator Entries](#)

**administrators**

changing user name, [Changing the Admin User's Name and Password](#)  
resetting passwords, [Changing the Admin User's Name and Password](#)

administrators, overview of, [Changing Administrator Entries](#)

**algorithm**

metaphone phonetic algorithm, [Approximate Searches](#)  
search, [Overview of the Searching Algorithm](#)

All IDs Threshold, [Indexing Performance](#)

all keyword, [General Access \(all Keyword\)](#)

allowing access, [Allowing or Denying Access](#)

anonymous access, [Defining Access Based on Authentication Method](#)

example, [Examples](#)

overview, [Anonymous Access \(anyone Keyword\)](#)

**anonymous binds**

disabling, [Disabling Anonymous Binds](#)

resource limits, [Setting Resource Limits on Anonymous Binds](#)

anyone keyword, [Anonymous Access \(anyone Keyword\)](#)

approximate index, [About Index Types](#)

query string codes, [Approximate Searches](#)

approximate search, [Using Operators in Search Filters](#)

**attribute**

ACI, [ACI Structure](#)

adding, [Modifying an Entry Using LDIF](#)

adding multiple values, [Adding Attribute Values](#)

adding to entry, [Adding an Attribute to an Entry](#)

creating, [Creating Attributes](#)

defining in schema, [Creating Attributes](#), [Creating Custom Schema Files](#)

deleting, [Modifying an Entry Using LDIF](#), [Deleting Schema](#)

deleting using LDIF update statements, [Deleting All Values of an Attribute Using LDIF](#)

editing, [Editing Custom Schema Elements](#)

nsslapd-schemacheck, [Turning Schema Checking On and Off](#)

ref, [Creating Smart Referrals from the Command Line](#)

removing a value, [Adding Attribute Values](#)

searching for, [Using Attributes in Search Filters](#)

standard, [Overview of Schema](#)

targeting, [Targeting Attributes](#)  
very large, [Adding Very Large Attributes](#)  
viewing, [Viewing Attributes and Object Classes](#)

attribute encryption, [Configuring Attribute Encryption](#)  
importing and exporting encrypted databases, [Exporting and Importing an Encrypted Database](#)

attribute subtypes, [Adding an Attribute Subtype](#)  
adding, [Adding an Attribute Subtype](#)  
binary, [Adding an Attribute Subtype](#)  
language, [Adding an Attribute Subtype](#)  
pronunciation, [Adding an Attribute Subtype](#)

attribute type field (LDIF), [About the LDIF File Format](#)

attribute uniqueness plug-in, [Enforcing Attribute Uniqueness](#)  
configuring, [Configuring Attribute Uniqueness](#)  
creating an instance of, [Creating an Instance of the Attribute Uniqueness Plug-in](#)  
examples, [Attribute Uniqueness Plug-in Syntax Examples](#)  
markerObjectClass, [Using the markerObjectClass and requiredObjectClass Keywords](#)  
requiredObjectClass, [Using the markerObjectClass and requiredObjectClass Keywords](#)  
syntax, [Attribute Uniqueness Plug-in Syntax](#)

attribute value field (LDIF), [About the LDIF File Format](#)

attribute values  
adding, [Modifying an Entry Using LDIF](#)  
deleting, [Deleting a Specific Attribute Value Using LDIF](#)  
modifying, [Changing an Attribute Value Using LDIF](#)  
replacing, [Modifying an Entry Using LDIF](#)

## attributes

allowed, [Object Classes](#)  
defined, [Attributes](#)  
linked attributes, [Linking Attributes to Manage Attribute Values](#)  
about, [About Linking Attributes](#)  
creating instance, [Configuring Attribute Links](#)  
syntax, [Looking at the Linking Attributes Plug-in Syntax](#)

## linking

fixup-linkedattrs.pl, [Regenerating Linked Attributes Using fixup-linkedattrs.pl](#)

managing, [Managing Attributes and Values](#)  
required, [Object Classes](#)  
syntax, [Directory Server Attribute Syntaxes](#)  
unique number assignments, [Assigning and Managing Unique Numeric Attribute Values](#)

configuring, [Configuring Unique Number Assignments](#), [Editing the DNA Plug-in in the Console](#)

magic number, [Ranges and Assigning Numbers](#)

overview, [Assigning and Managing Unique Numeric Attribute Values](#)

syntax, [Looking at the DNA Plug-in Syntax](#)

usage, [Ranges and Assigning Numbers](#)

#### attributes values

targeting, [Targeting Attribute Values Using LDAP Filters](#)

#### audit log

##### configuring

deletion policy, [Defining a Log File Deletion Policy](#)

rotation policy, [Defining a Log File Rotation Policy](#)

disabling, [Enabling or Disabling Logs](#)

enabling, [Enabling or Disabling Logs](#)

viewing, [Viewing Log Files](#)

#### authentication, [Opening the Admin Server Console](#)

access control and, [Defining Access Based on Authentication Method](#)

##### autobind

configuring, [Configuring Autobind](#)

overview, [Overview of Autobind and LDAP](#)

bind DN, [Logging into Directory Server](#)

certificate-based, [Using Client \(Certificate-Based\) Authentication](#)

for database links, [Using Different Bind Mechanisms](#)

LDAP URLs, [Examples of LDAP URLs](#)

over TLS/SSL, [TLS/SSL in Directory Server](#)

SASL, [Setting up SASL Identity Mapping](#)

SASL mechanisms, [Authentication Mechanisms for SASL in Directory Server](#)

using PAM, [Using PAM for Pass-Through Authentication](#)

#### authmethod keyword, [Defining Access Based on Authentication Method](#)

##### autobind

configuring, [Configuring Autobind](#)

overview, [Overview of Autobind and LDAP](#)

## B

#### backing up data, [Backing up and Restoring Data](#)

all, [Backing up All Databases](#)

cn=tasks, [Backing up the Database through the cn=tasks Entry](#)

db2bak, [Backing up All Databases from the Command Line](#)  
db2bak.pl, [Backing up All Databases from the Command Line](#)  
dse.ldif, [Backing up the dse.ldif Configuration File](#)

bak2db script, [Using the bak2db Command-Line Script](#)

bak2db.pl perl script, [Using bak2db.pl Perl Script](#)

base 64 encoding, [Representing Binary Data](#)

base DN, ldapsearch and, [Using LDAP\\_BASEDN](#)

binary data, LDIF and, [Representing Binary Data](#)

binary subtype, [Adding an Attribute Subtype](#)

bind credentials

for database links, [Providing Bind Credentials](#)

bind DN

accessing the server, [Logging into Directory Server](#)

viewing current, [Viewing the Current Console Bind DN](#)

bind rules

access at specific time or day, [Defining Access at a Specific Time of Day or Day of Week](#)

access based on authentication method, [Defining Access Based on Authentication Method](#)

LDIF example, [Examples](#)

access based on value matching

overview, [Defining Access Based on Value Matching](#)

ACI syntax, [The ACI Syntax](#)

all keyword, [General Access \(all Keyword\)](#)

anonymous access, [Anonymous Access \(anyone Keyword\)](#)

example, [Examples](#)

LDIF example, [Examples](#)

anyone keyword, [Anonymous Access \(anyone Keyword\)](#)

authmethod keyword, [Defining Access Based on Authentication Method](#)

Boolean, [Using Boolean Bind Rules](#)

dayofweek keyword, [Defining Access at a Specific Time of Day or Day of Week](#)

dns keyword, [Defining Access from a Specific Domain](#)

general access, [General Access \(all Keyword\)](#)

example, [Examples](#)

group access, [Defining Group Access - groupdn Keyword](#)

group access example, [Granting a Group Full Access to a Suffix](#)

groupdn keyword, [Defining Group Access - groupdn Keyword](#)

ip keyword, [Defining Access from a Specific IP Address](#)

LDAP URLs, [LDAP URLs](#)

LDIF keywords, [Bind Rule Syntax](#)

overview, [Bind Rules](#)

parent keyword, [Parent Access \(parent Keyword\)](#)

role access, [Defining Role Access - roledn Keyword](#)

roledn keyword, [Defining Role Access - roledn Keyword](#)

self keyword, [Self Access \(self Keyword\)](#)

ssf keyword, [Requiring a Certain Level of Security in Connections](#)

timeofday keyword, [Defining Access at a Specific Time of Day or Day of Week](#)

user access

LDIF example, [Examples](#)

parent, [Parent Access \(parent Keyword\)](#)

self, [Self Access \(self Keyword\)](#)

user access example, [Granting Write Access to Personal Entries](#)

userattr keyword, [Using the userattr Keyword](#)

userdn keyword, [Defining User Access - userdn Keyword](#)

**binds**

anonymous, [Disabling Anonymous Binds](#)

requiring secure, [Requiring Secure Binds](#)

special types, [Enabling Different Types of Binds](#)

unauthenticated, [Allowing Unauthenticated Binds](#)

**Boolean bind rules**

example, [Using Boolean Bind Rules](#)

overview, [Using Boolean Bind Rules](#)

**Boolean operators, in search filters, [Using Compound Search Filters](#)**

**browsing index, [About Index Types](#)**

**browsing indexes**

creating

cn=tasks, [Using a cn=tasks Entry to Create a Browsing Index](#)

## C

**cache memory size**

and import operations, [Importing Entries with Large Attributes](#)

**caches**

DN, [Setting the DN Cache Size](#)

**cascading chaining**

client ACIs, [Configuring Cascading Chaining from the Command Line](#)

configuration attributes, [Summary of Cascading Chaining Configuration Attributes](#)

configuring from command line, [Configuring Cascading Chaining from the Command Line](#)  
configuring from console, [Configuring Cascading Chaining Using the Console](#)  
example, [Cascading Chaining Configuration Example](#)  
local ACI evaluation, [Configuring Cascading Chaining from the Command Line](#)  
loop detection, [Detecting Loops](#)  
overview, [Overview of Cascading Chaining](#)  
proxy admin user ACI, [Configuring Cascading Chaining from the Command Line](#)  
proxy authorization, [Configuring Cascading Chaining from the Command Line](#)

#### cascading replication

initializing the replicas, [Setting up the Replication Agreements](#)  
introduction, [Cascading Replication](#)  
setting up, [Configuring Cascading Replication](#)

#### certificate

mapping to a DN, [Using Client \(Certificate-Based\) Authentication](#)  
password, [Creating a Password File for the Directory Server](#)

#### certificate database

password, [TLS/SSL in Directory Server](#)

#### certificate group, [Groups](#)

certificate-based authentication, [Using Client \(Certificate-Based\) Authentication](#)  
setting up, [Using Client \(Certificate-Based\) Authentication](#)

#### certificates, [Requesting and Installing a Server Certificate](#)

for authenticating to the Directory Server, [Configuring Directory Server to Accept Certificate-Based Authentication from LDAP Clients](#)  
installing, [Installing a CA Certificate](#)

#### certmap.conf

defined, [Mapping DN's to Certificates](#)  
editing, [Editing the certmap.conf File](#)  
examples, [Example certmap.conf Mappings](#)

#### chaining

cascading, [Overview of Cascading Chaining](#)  
component operations, from command line, [Chaining Component Operations from the Command Line](#)  
component operations, from console, [Chaining Component Operations Using the Console](#)  
overview, [Creating and Maintaining Database Links](#)  
using SSL, [Creating a New Database Link Using the Console](#), [Providing an LDAP URL](#)

#### change operations, [Using LDIF Update Statements to Create or Modify Entries](#)

add, [Modifying an Entry Using LDIF](#)

delete, [Modifying an Entry Using LDIF](#)  
replace, [Modifying an Entry Using LDIF](#)

#### change type

add, [Adding an Entry Using LDIF](#)  
delete, [Deleting an Entry Using LDIF](#)  
LDIF, [Using LDIF Update Statements to Create or Modify Entries](#)  
modify, [Modifying an Entry Using LDIF](#)

#### changelog, [Changelog](#)

deleting, [Removing the Changelog](#)  
trimming, [Trimming the Replication Changelog](#)

#### character type, [About Locales](#)

#### ciphers, [Setting Encryption Ciphers](#)

none, MD5  
    MD5 message authentication, [Selecting the Encryption Cipher](#)  
  
overview, [Setting Encryption Ciphers](#)  
selecting, [Setting Encryption Ciphers](#)

#### cl-dump.pl script, [Troubleshooting Replication-Related Problems](#)

#### class of service (CoS), [Assigning Class of Service](#)

access control, [Access Control and CoS](#)

##### classic

example, [How a Classic CoS Works](#)  
overview, [How a Classic CoS Works](#)

cosPriority attribute, [Handling Multi-valued Attributes with CoS](#)

creating, [Creating a New CoS](#)

definition entry, [Creating the CoS Definition Entry from the Command Line](#)

editing, [Creating the CoS Template Entry](#)

##### indirect

example, [How an Indirect CoS Works](#)  
overview, [How an Indirect CoS Works](#)

##### pointer

example, [How a Pointer CoS Works](#)  
overview, [How a Pointer CoS Works](#)

#### qualifiers

merge-scheme, [Handling Multi-valued Attributes with CoS](#)  
override, [Handling Physical Attribute Values](#)

#### template entry



creating, [Creating the CoS Template Entry](#)  
 overview, [About the CoS Template Entry](#)

## classic CoS

example, [How a Classic CoS Works](#)  
 overview, [How a Classic CoS Works](#)

## client

using to find entries, [Finding Directory Entries](#)

client authentication, [Configuring Directory Server to Accept Certificate-Based Authentication from LDAP Clients](#)

cn=fixup linked attributes task, [Regenerating Linked Attributes Using Idapmodify](#)

cn=memberof task, [Initializing and Regenerating memberOf Attributes Using Idapmodify](#)

cn=schema reload task, [Reloading Schema Using Idapmodify](#)

## cn=task

cn=schema reload task, [Reloading Schema Using Idapmodify](#)

## cn=tasks

cn=backup, [Backing up the Database through the cn=tasks Entry](#)

cn=export, [Exporting through the cn=tasks Entry](#)

cn=fixup linked attributes, [Regenerating Linked Attributes Using Idapmodify](#)

cn=import, [Importing through the cn=tasks Entry](#)

cn=memberof task, [Initializing and Regenerating memberOf Attributes Using Idapmodify](#)

cn=restore, [Restoring the Database through the cn=tasks Entry](#)

creating browsing indexes, [Using a cn=tasks Entry to Create a Browsing Index](#)

creating indexes, [Using a cn=tasks Entry to Create an Index](#)

code page, [About Locales](#)

## collation order

international index, [Creating Indexes from the Server Console](#)

overview, [About Locales](#)

search filters and, [Searching an Internationalized Directory](#)

## command line

providing input from, [Providing Input from the Command Line](#)

## command-line scripts

db2bak, [Backing up All Databases from the Command Line](#)

db2bak.pl, [Backing up All Databases from the Command Line](#)

fixup-linkedattrs.pl, [Regenerating Linked Attributes Using fixup-linkedattrs.pl](#)

fixup-memberof.pl, [Initializing and Regenerating memberOf Attributes Using fixup-memberof.pl](#)

schema-reload.pl, [Reloading Schema Using schema-reload.pl](#)

**command-line utilities**

certificate-based authentication and, [Using Client \(Certificate-Based\) Authentication](#)

Idapdelete, [Deleting Entries Using Idapdelete](#)

Idapmodify, [Adding and Modifying Entries Using Idapmodify](#)

Idapsearch, [LDAP Search Filters](#)

Idif, [Base-64 Encoding](#)

Idif2db, [Running the db2index.pl Script](#)

commas, in DNSs, [Using Special Characters](#), [Targeting a Directory Entry](#)

using Idapsearch with, [Specifying DNSs That Contain Commas in Search Filters](#)

compare right, [Assigning Rights](#)

**compatibility**

ACIs, [Compatibility with Earlier Releases](#)

replication, [Replication with 4.x Versions of Directory Server](#)

compound search filters, [Using Compound Search Filters](#)

**Configuration Administrator**

changing user name or password for, [Changing Administrator Entries](#)

defined, [Changing the Admin User's Name and Password](#), [Changing Administrator Entries](#)

**Configuration Administrators group**

adding users to, [Adding Users to the Configuration Administrators Group](#)

**configuration attributes**

account lockout, [Configuring the Account Lockout Policy Using the Command Line](#)

cascading chaining, [Summary of Cascading Chaining Configuration Attributes](#)

password policy, [Configuring a Global Password Policy Using the Command Line](#)

suffix, [Creating Root and Sub Suffixes from the Command Line](#)

**configuration changes**

deleting core server configuration attributes, [Configuration Attributes Which Can Be Deleted](#)

requiring server restart, [Configuration Attributes Requiring Server Restart](#)

**configuration directory**

changing settings for, [Changing the Configuration Directory Host or Port](#)

defined, [Overview of the Directory Server Console](#)

overview, [Changing the Configuration Directory Host or Port](#)

connection restrictions, [Setting Host Restrictions](#)

setting in the command line, [Setting Host Restrictions in the Command Line](#)

setting in the Console, [Setting Host Restrictions in the Console](#)

**connections**

LDAPAPI (Unix sockets), [Overview of Autobind and LDAPAPI](#)

configuring, [Enabling LDAP](#)

monitoring, [Monitoring the Server from the Directory Server Console](#)

requiring secure, [Requiring Secure Connections](#)

viewing number of, [Monitoring the Server from the Directory Server Console](#)

consumer initialization

filesystem replica, [Filesystem Replica Initialization](#)

consumer server, [Suppliers and Consumers](#)

continued lines

in LDIF, [Continuing Lines in LDIF](#)

in LDIF update statements, [Using LDIF Update Statements to Create or Modify Entries](#)

core server configuration attributes

deleting, [Configuration Attributes Which Can Be Deleted](#)

CoS (class of service), [Assigning Class of Service](#)

CoS definition entry

attributes, [Creating the CoS Definition Entry from the Command Line](#)

object classes, [Creating the CoS Definition Entry from the Command Line](#)

CoS qualifiers

default, [Handling Physical Attribute Values](#)

merge-scheme, [Handling Multi-valued Attributes with CoS](#)

override, [Handling Physical Attribute Values](#)

CoS template entry, [About the CoS Template Entry](#)

creating, [Creating the CoS Template Entry](#)

cosPriority attribute, [Handling Multi-valued Attributes with CoS](#)

counter, password failures, [Configuring the Account Lockout Policy Using the Console](#)

country code, [Supported Locales](#)

creating a database

from the command line, [Creating a New Database for a Single Suffix from the Command Line](#)

from the console, [Creating a New Database for an Existing Suffix Using the Console](#)

creating a virtual DIT, [About Views](#)

creating the directory, [Defining Directories Using LDIF](#)

custom distribution function

adding to suffix, [Adding Multiple Databases for a Single Suffix](#)

custom distribution logic

adding databases, [Adding Multiple Databases for a Single Suffix](#)

adding to suffix, [Adding Multiple Databases for a Single Suffix](#)

custom schema files, [Creating Custom Schema Files](#)  
custom views, [Changing the Console Appearance](#)  
    changing to, [Switching to a Custom View](#)  
    creating, [Creating Custom Views](#)  
    editing, [Creating Custom Views](#)  
    removing, [Creating Custom Views](#)  
    setting ACLs on, [Setting Access Permissions for a Public View](#)  
    using, [Working with Custom Views](#)

## D

dash, in change operation, [Using LDIF Update Statements to Create or Modify Entries](#)  
data consistency  
    using referential integrity, [Maintaining Referential Integrity](#)

### database

and associated suffix, [Creating and Maintaining Suffixes](#)

#### backing up

    cn=tasks, [Backing up the Database through the cn=tasks Entry](#)  
    db2bak, [Backing up All Databases from the Command Line](#)  
    db2bak.pl, [Backing up All Databases from the Command Line](#)

backup, [Backing up and Restoring Data](#)

backup files, [Backing up All Databases from the Console](#)

backup from console, [Backing up All Databases](#)

creating from command line, [Creating a New Database for a Single Suffix from the Command Line](#)

creating from console, [Creating a New Database for an Existing Suffix Using the Console](#)

creating multiple, [Adding Multiple Databases for a Single Suffix](#)

creating using LDIF, [Defining Directories Using LDIF](#)

deleting, [Deleting a Database](#)

export, [Exporting Data](#)

    cn=tasks, [Exporting through the cn=tasks Entry](#)  
    db2ldif, [Exporting a Database Using db2ldif or db2ldif.pl](#)  
    db2ldif.pl, [Exporting a Database Using db2ldif or db2ldif.pl](#)  
    encrypted database, [Exporting and Importing an Encrypted Database](#)

export from console, [Exporting Directory Data to LDIF Using the Console](#)

import, [Importing Data](#)

    cn=tasks, [Importing through the cn=tasks Entry](#)  
    encrypted database, [Exporting and Importing an Encrypted Database](#)  
    ldif2db, [Importing Using the ldif2db Command-Line Script](#)  
    ldif2db.pl, [Importing Using the ldif2db.pl Perl Script](#)  
    ldif2ldap, [Importing Using the ldif2ldap Command-Line Script](#)

initialization, [Initializing a Database from the Console](#)  
making read-only, [Placing a Database in Read-Only Mode](#)  
monitoring from command line, [Monitoring Databases from the Command Line](#)  
monitoring from server console, [Monitoring Database Activity from the Directory Server Console](#)  
overview, [Creating and Maintaining Databases](#)  
read-only mode, [Placing a Database in Read-Only Mode](#)  
replication, [What Directory Units Are Replicated](#)  
restore, [Backing up and Restoring Data](#)  
restoring  
    bak2db, [Using the bak2db Command-Line Script](#)  
    bak2db.pl, [Using bak2db.pl Perl Script](#)  
    cn=tasks, [Restoring the Database through the cn=tasks Entry](#)  
  
restoring from console, [Restoring All Databases from the Console](#)  
selecting for monitoring, [Monitoring Database Activity](#)  
viewing backend information, [Monitoring Database Activity](#)

#### database link

##### cascading

    configuring from command line, [Configuring Cascading Chaining from the Command Line](#)  
    configuring from console, [Configuring Cascading Chaining Using the Console](#)  
    overview, [Overview of Cascading Chaining](#)

chaining with SSL, [Creating a New Database Link Using the Console](#), [Providing an LDAP URL](#)

configuration, [Creating a New Database Link](#)

configuration attributes, [Summary of Database Link Configuration Attributes](#)

configuration example, [Summary of Database Link Configuration Attributes](#)

configuring bind and authentication, [Using Different Bind Mechanisms](#)

configuring bind credentials, [Providing Bind Credentials](#)

configuring defaults, [Configuring Database Link Defaults](#)

configuring failover servers, [Providing a List of Failover Servers](#)

configuring LDAP URL, [Providing an LDAP URL](#)

configuring suffix, [Creating a Database Link from the Command Line](#)

creating from command line, [Creating a Database Link from the Command Line](#)

creating from console, [Creating a New Database Link Using the Console](#)

deleting, [Deleting Database Links](#)

maintaining remote server info, [Maintaining Database Links](#)

overview, [Creating and Maintaining Database Links](#)

#### database server parameters

    read-only, [Monitoring Database Activity from the Directory Server Console](#)

#### databases

caches

DN, [Setting the DN Cache Size](#)

in Directory Server, [Configuring Directory Databases](#)

date format, [About Locales](#)

dayofweek keyword, [Defining Access at a Specific Time of Day or Day of Week](#)

db2bak script, [Backing up All Databases from the Command Line](#)

db2bak utility, [Backing up All Databases from the Command Line](#)

db2bak.pl script, [Backing up All Databases from the Command Line](#)

db2ldif utility, [Exporting a Database Using db2ldif or db2ldif.pl](#)

db2ldif.pl, [Exporting a Database Using db2ldif or db2ldif.pl](#)

debug

and replication timeouts, [Setting Replication Timeout Periods](#)

default CoS qualifier, [Handling Physical Attribute Values](#)

default referrals

setting, [Setting Default Referrals](#)

setting from console, [Setting a Default Referral Using the Console](#)

settings from command line, [Setting a Default Referral from the Command Line](#)

defining

access control policy, [Creating ACIs from the Console](#)

attributes, [Creating Attributes](#)

object classes, [Creating Object Classes](#)

delete right, [Assigning Rights](#)

deleting

ACI, [Deleting an ACI](#)

attribute values, [Deleting a Specific Attribute Value Using LDIF](#)

attributes, [Modifying an Entry Using LDIF](#), [Deleting Schema](#)

core server configuration attributes, [Configuration Attributes Which Can Be Deleted](#)

database link, [Deleting Database Links](#)

Directory Server instance, [Deleting a Directory Server Instance](#)

dse.ldif file, [Configuration Attributes Which Can Be Deleted](#)

entries, [Deleting an Entry Using LDIF](#)

multiple attributes, [Modifying an Entry Using LDIF](#)

object classes, [Deleting Schema](#)

deleting directory entries, [Deleting Entries Using ldapdelete](#)

deleting schema elements, [Deleting Schema](#)

denying access, [Allowing or Denying Access](#)

precedence rule, [ACI Evaluation](#)

directives, [Admin Express Directives](#)

directory

changing the search directory, [Searching for Users and Groups](#)

directory creation, [Defining Directories Using LDIF](#)

directory entries

adding using LDIF, [Adding Entries Using LDIF](#)

creating, [Creating Directory Entries](#), [Creating Directory Entries](#)

deleting, [Deleting Directory Entries](#)

managing from command line, [Managing Entries from the Command Line](#)

managing from console, [Managing Entries from the Directory Console](#)

modifying, [Modifying Directory Entries](#)

removing, [Removing an Entry from the Directory](#)

searching for, [Searching for Users and Groups](#)

directory manager

and access control, [Setting Access Controls on Directory Manager](#)

Directory Manager

password, [Managing the Directory Manager Password](#)

Directory Server

basic administration, [Basic Red Hat Directory Server Settings](#)

binding to, [Logging into Directory Server](#)

changing bind DN, [Changing the Login Identity](#)

configuration, [Changing Directory Server Port Numbers](#)

configuration subtree, [Overview of the Directory Server Console](#)

configuring SASL authentication at startup, [Configuring SASL Authentication at Directory Server Startup](#)

connecting over LDAP (Unix sockets), [Overview of Autobind and LDAP](#)

controlling access, [Managing Access Control](#)

creating a root entry, [Creating a Root Entry](#)

creating content, [Populating Directory Databases](#)

creating entries, [Creating Directory Entries](#)

data, [Populating Directory Databases](#)

databases, [Configuring Directory Databases](#)

deleting entries, [Deleting Directory Entries](#)

deleting instance, [Deleting a Directory Server Instance](#)

file locations, [Directory Server File Locations](#), [Directory Server File Locations](#)

importing data, [Importing Data](#)

international character sets, [Internationalization](#)

login, [Logging into Directory Server](#)

managing attributes, [Managing Attributes and Values](#)

managing entries, [Creating Directory Entries](#)  
MIB, [Using the Management Information Base](#)  
modifying entries, [Modifying Directory Entries](#)  
monitoring, [Types of Directory Server Log Files](#)  
monitoring from command line, [Monitoring the Directory Server from the Command Line](#)  
monitoring with SNMP, [Monitoring Directory Server Using SNMP](#)  
overview, [Basic Red Hat Directory Server Settings](#)  
performance counters, [Monitoring Server Activity](#), [Enabling and Disabling Counters](#)  
    64-bit, [Monitoring Server Activity](#), [Monitoring Database Activity](#), [Using the Management Information Base](#)  
  
reloading schema, [Dynamically Reloading Schema](#)  
    cn=schema reload task, [Reloading Schema Using ldapmodify](#)  
    schema-reload.pl, [Reloading Schema Using schema-reload.pl](#)  
  
replication monitoring, [Monitoring Replication from Admin Express](#)  
role in managing resources and users, [Overview of the Directory Server Console](#)  
starting and stopping, [Starting and Stopping Directory Server from the Command Line](#)  
starting and stopping servers, [Starting and Stopping Servers](#)  
starting the Console, [Starting the Directory Server Console](#)  
suffixes, [Configuring Directory Databases](#)  
supported languages, [Supported Locales](#)  
user subtree, [Overview of the Directory Server Console](#)  
viewing information, [Viewing Server Information](#)  
viewing logs, [Viewing Server Logs](#)

## Directory Server Console

managing certificates, [Managing Certificates Used by the Directory Server Console](#)  
starting, [Starting the Directory Server Console](#)

## directory trees

finding entries in, [Using ldapsearch](#)

## disabling suffixes, [Disabling a Suffix](#)

## disk space

access log and, [Enabling or Disabling Logs](#)  
log files and, [Manual Log File Rotation](#)

## distributed number assignment, [Assigning and Managing Unique Numeric Attribute Values](#)

about ranges, [About Dynamic Number Assignments](#)  
basic example, [Looking at the DNA Plug-in Syntax](#)  
complete example, [Looking at the DNA Plug-in Syntax](#)  
configuring, [Configuring Unique Number Assignments](#), [Editing the DNA Plug-in in the Console](#)  
Directory Server behavior, [Assigning and Managing Unique Numeric Attribute Values](#)



for attributes, [Ranges and Assigning Numbers](#)  
overview, [Assigning and Managing Unique Numeric Attribute Values](#)  
scope, [Filters, Searches, and Target Entries](#)  
syntax, [Looking at the DNA Plug-in Syntax](#)

distribution function, [Adding Multiple Databases for a Single Suffix](#)

DN cache, [Setting the DN Cache Size](#)

dn field (LDIF), [About the LDIF File Format](#)

DNs

validating syntax, [Enabling Strict Syntax Validation for DNs](#)

dns keyword, [Defining Access from a Specific Domain](#)

ds-logpipe.py

using plug-ins, [Loading Plug-ins with the Named Pipe Log Script](#)

dse.ldif

deleting attributes, [Configuration Attributes Which Can Be Deleted](#)

editing, [Configuration Attributes Requiring Server Restart](#)

dse.ldif file

backing up, [Backing up the dse.ldif Configuration File](#)

restoring, [Restoring the dse.ldif Configuration File](#)

dynamic group, [Groups](#)

dynamic groups, [Creating Dynamic Groups in the Console](#)

creating, [Creating Dynamic Groups in the Console](#)

modifying, [Creating Dynamic Groups in the Console](#)

## E

editing

attributes, [Editing Custom Schema Elements](#)

dse.ldif file, [Configuration Attributes Requiring Server Restart](#)

object classes, [Editing Custom Schema Elements](#)

encryption

attribute, [Configuring Attribute Encryption](#)

database, [Configuring Attribute Encryption](#)

settings for Admin Server, [Working with SSL](#)

end of file marker, [Providing Input from the Command Line](#)

entity table, [Entity Table](#)

entries

adding an object class, [Adding or Removing an Object Class to an Entry](#)

adding attributes, [Adding an Attribute to an Entry](#)

adding using LDIF, [Adding Entries Using LDIF](#)

adding using LDIF update statements, [Adding an Entry Using LDIF](#)

adding very large attributes, [Adding Very Large Attributes](#)

creating, [Creating Directory Entries](#)

using LDIF, [Specifying Directory Entries Using LDIF](#)

deleting, [Deleting Directory Entries](#)

using ldapdelete, [Deleting Entries Using ldapdelete](#)

deleting and replication, [Managing Deleted Entries with Replication](#)

deleting using LDIF update statements, [Deleting an Entry Using LDIF](#)

distribution, [Creating Databases](#)

finding, [Using ldapsearch](#)

managing, [Creating Directory Entries](#)

managing from command line, [Managing Entries from the Command Line](#)

managing from console, [Managing Entries from the Directory Console](#)

modifying, [Modifying Directory Entries](#)

using ldapmodify, [Adding and Modifying Entries Using ldapmodify](#)

using LDIF update statements, [Modifying an Entry Using LDIF](#)

order of creation, [Providing Input from the Command Line](#)

order of deletion, [Deleting Entries Using ldapdelete](#)

removing an object class, [Adding or Removing an Object Class to an Entry](#)

root, [Defining Directories Using LDIF](#)

targeting, [Targeting a Directory Entry](#)

entry distribution, [Creating Databases](#)

entry ID list, [Indexing Performance](#)

entryUSN

import operations, [Setting EntryUSN Initial Values During Import](#)

initializing replicas and databases, [Setting EntryUSN Initial Values During Import](#)

entryUSN:

import operations, [Setting EntryUSN Initial Values During Import](#)

environment variables

LDAP\_BASEDN, [Using LDAP\\_BASEDN](#)

EOF marker, [Providing Input from the Command Line](#)

equality index, [About Index Types](#)

required for referential integrity, [How Referential Integrity Works](#)

equality search, [Using Operators in Search Filters](#)

example, [Using Attributes in Search Filters](#)

international example, [Equality Example](#)

**error log**

access control information, [Logging Access Control Information](#)

changing location and name

in the command line, [Changing the Log Location in the Command Line](#)

in the Console, [Changing the Log Name in the Console](#)

configuring

deletion policy, [Defining a Log File Deletion Policy](#)

rotation policy, [Defining a Log File Rotation Policy](#)

defined, [Viewing Logs](#)

manually rotating, [Manual Log File Rotation](#)

viewing, [Viewing Log Files](#)

viewing in command line, [Viewing Logs in the Command Line](#)

viewing in Console, [Viewing the Logs through the Console](#)

**example**

cascading chaining, [Cascading Chaining Configuration Example](#)

**exporting data, [Exporting Data](#)**

cn=tasks, [Exporting through the cn=tasks Entry](#)

db2ldif, [Exporting a Database Using db2ldif or db2ldif.pl](#)

db2ldif.pl, [Exporting a Database Using db2ldif or db2ldif.pl](#)

encrypted database, [Exporting and Importing an Encrypted Database](#)

using console, [Exporting Directory Data to LDIF Using the Console](#)

extending the directory schema, [Managing the Directory Schema](#)

**F****failover servers**

for database links, [Providing a List of Failover Servers](#)

File locations, [Directory Server File Locations](#), [Directory Server File Locations](#)

**files**

access log, [Types of Directory Server Log Files](#)

database backup, [Backing up All Databases from the Console](#)

EOF marker, [Providing Input from the Command Line](#)

id2entry.db4, [Overview of Standard Indexes](#)

Filesystem Hierarchy Standard, [Directory Server File Locations](#), [Directory Server File Locations](#)

filesystem replica initialization, [Filesystem Replica Initialization](#)

**filtered role**

creating, [Creating a Filtered Role](#)

example, [Creating a Filtered Role through the Command Line](#)

**finding**

attributes, [Using Attributes in Search Filters](#)

entries, [Using Idapsearch](#)

fixup-linkedattrs.pl, [Regenerating Linked Attributes Using fixup-linkedattrs.pl](#)

fixup-memberof.pl, [Initializing and Regenerating memberOf Attributes Using fixup-memberof.pl](#)

**fonts**

changing, [Changing Console Fonts](#)

format, LDIF, [LDAP Data Interchange Format](#)

fractional replication, [Replicating a Subset of Attributes with Fractional Replication](#)

**G****general access**

example, [Examples](#)

overview, [General Access \(all Keyword\)](#)

get effective rights, [Checking Access Rights on Entries \(Get Effective Rights\)](#)

return codes, [Get Effective Rights Return Codes](#)

global password policy, [Configuring the Global Password Policy](#)

glue entries, [Solving Orphan Entry Conflicts](#)

**greater than or equal to search**

international example, [Greater-Than or Equal-to Example](#)

overview, [Using Operators in Search Filters](#)

groupdn keyword, [Defining Group Access - groupdn Keyword](#)

LDIF examples, [Defining Group Access - groupdn Keyword](#)

groupdnattr keyword, [Using the userattr Keyword](#)

**groups**

access control, [Defining User Access - userdn Keyword](#)

access control example, [Granting a Group Full Access to a Suffix](#)

access to directory, [Defining Group Access - groupdn Keyword](#)

configuring the memberOf plug-in, [Configuring an Instance of the MemberOf Plug-in](#), [Editing the MemberOf Plug-in from the Console](#), [Editing the MemberOf Plug-in from the Command Line](#)

creating, [Groups](#)

differences between Directory Server and Active Directory, [Group Schema Differences between Red Hat Directory Server and Active Directory](#)

dynamic, [Creating Dynamic Groups in the Console](#)

creating, [Creating Dynamic Groups in the Console](#)

modifying, [Creating Dynamic Groups in the Console](#)

editing, [Editing Entries](#)

fixup-memberof.pl, [Initializing and Regenerating memberOf Attributes Using fixup-memberof.pl](#)

locating, [Searching for Users and Groups](#)

memberOf

cn=memberOf task, [Initializing and Regenerating memberOf Attributes Using ldapmodify](#)

overview, [Using Groups](#)

removing, [Removing an Entry from the Directory](#)

static, [Creating Static Groups in the Console](#)

creating, [Creating Static Groups in the Console](#)

modifying, [Creating Static Groups in the Console](#)

types, [Groups](#)

GSS-API, [Authentication Mechanisms for SASL in Directory Server](#)

## H

host information, modifying, [Editing Domain, Host, Server Group, and Instance Information](#)

host restriction, [Setting Host Restrictions](#)

setting in the command line, [Setting Host Restrictions in the Command Line](#)

setting in the Console, [Setting Host Restrictions in the Console](#)

hub, [Suppliers and Consumers](#)

## I

id field (LDIF), [About the LDIF File Format](#)

id2entry.db4 file, [Overview of Standard Indexes](#)

identity mapping

default, [Default SASL Mappings for Directory Server](#)

importing

buffer size, [Importing Entries with Large Attributes](#)

failures, [Importing Large Numbers of Entries](#)

large attributes, [Importing Entries with Large Attributes](#)

large numbers of entries, [Importing Large Numbers of Entries](#)

importing data, [Importing Data](#)

cn=tasks, [Importing through the cn=tasks Entry](#)

encrypted database, [Exporting and Importing an Encrypted Database](#)

from console, [Importing a Database from the Console](#)

ldif2ldap, [Importing Using the ldif2ldap Command-Line Script](#)

using ldif2db, [Importing Using the ldif2db Command-Line Script](#)

using ldif2db.pl, [Importing Using the ldif2db.pl Perl Script](#)

inactivating accounts, [Manually Inactivating Users and Roles](#)

inactivating roles, [Making a Role Inactive or Active](#)

index types, [About Index Types](#)

approximate index, [About Index Types](#)

browsing index, [About Index Types](#)

equality index, [About Index Types](#)

international index, [About Index Types](#)

presence index, [About Index Types](#)

substring index, [About Index Types](#)

virtual list view index, [About Index Types](#)

indexes

creating

cn=tasks, [Using a cn=tasks Entry to Create an Index](#)

creating dynamically, [Creating Indexes from the Command Line](#)

dynamic changes to, [Creating Indexes from the Command Line](#)

matching rules, [Using Matching Rules](#)

presence, [Overview of System Indexes](#)

required for referential integrity, [How Referential Integrity Works](#)

indexing, [About Index Types](#)

creating indexes from console, [Creating Indexes from the Server Console](#)

system indexes, [Overview of System Indexes](#)

indirect CoS

example, [How an Indirect CoS Works](#)

overview, [How an Indirect CoS Works](#)

init scripts

configuring SASL authentication, [Configuring SASL Authentication at Directory Server Startup](#)

initialization

and entryUSN values, [Setting EntryUSN Initial Values During Import](#)

and suppliers in MMR, [Setting EntryUSN Initial Values During Import](#)

manual consumer creation, [Manual Consumer Initialization Using the Command Line](#)

online consumer creation, [Online Consumer Initialization Using the Console](#)

initializing databases, [Initializing a Database from the Console](#)

initializing replicas

cascading replication, [Setting up the Replication Agreements](#)

filesystem replica, [Filesystem Replica Initialization](#)

interaction table, [Interaction Table](#)

international charactersets, [Internationalization](#)

international index, [About Index Types](#)

collation order, [Creating Indexes from the Server Console](#)

international searches, [Searching an Internationalized Directory](#)

equality, [Equality Example](#)

examples, [International Search Examples](#)

greater than, [Greater-Than Example](#)

greater than or equal to, [Greater-Than or Equal-to Example](#)

less than, [Less-Than Example](#)

less than or equal to, [Less-Than or Equal-to Example](#)

substring, [Substring Example](#)

using OIDs, [Matching Rule Formats](#)

internationalization

character type, [About Locales](#)

collation order, [About Locales](#)

country code, [Supported Locales](#)

date format, [About Locales](#)

language tag, [Supported Locales](#)

locales and, [About Locales](#)

location of files, [About Locales](#)

modifying entries, [Modifying an Entry in an Internationalized Directory](#)

monetary format, [About Locales](#)

object identifiers and, [Supported Locales](#)

of LDIF files, [Storing Information in Multiple Languages](#)

search filters and, [Searching an Internationalized Directory](#)

supported locales, [Supported Locales](#)

time format, [About Locales](#)

ip keyword, [Defining Access from a Specific IP Address](#)

## J

jpeg images, [Representing Binary Data](#)

## K

Kerberos, [Using Kerberos GSS-API with SASL](#), [Authentication Mechanisms for SASL in Directory Server](#)

realms, [About Principals and Realms](#)

keytabs

with SELinux, [Managing SELinux Labels for Files Used by the Directory Server](#)

## L

**language code**

in LDIF entries, [Storing Information in Multiple Languages](#)

list of supported, [Supported Locales](#)

**language subtype, [Adding an Attribute Subtype](#)****language support**

language tag, [Supported Locales](#)

searching and, [Searching an Internationalized Directory](#)

specifying using locales, [Supported Locales](#)

**language tags**

described, [Supported Locales](#)

in international searches, [Using a Language Tag for the Matching Rule](#)

in LDIF update statements, [Modifying an Entry in an Internationalized Directory](#)

**LDAP clients**

authentication over SSL, [Configuring Directory Server to Accept Certificate-Based Authentication from LDAP Clients](#)

certificate-based authentication and, [Using Client \(Certificate-Based\) Authentication](#)

monitoring database with, [Monitoring Databases from the Command Line](#)

monitoring server with, [Monitoring the Directory Server from the Command Line](#)

using to find entries, [Finding Directory Entries](#)

**LDAP Data Interchange Format, see LDIF, [Using LDIF Update Statements to Create or Modify Entries](#)**

**LDAP search filters**

DNs with commas and, [Specifying DNs That Contain Commas in Search Filters](#)

in targets, [Targeting Entries or Attributes Using LDAP Filters](#)

example, [Setting a Target Using Filtering](#)

examples, [Targeting Entries or Attributes Using LDAP Filters](#)

**LDAP URLs**

components of, [Components of an LDAP URL](#)

examples, [Examples of LDAP URLs](#)

for database links, [Providing an LDAP URL](#)

in access control, [LDAP URLs](#)

security, [Examples of LDAP URLs](#)

syntax, [Components of an LDAP URL](#)

**ldapcompare command-line utility**

examples, [Comparing Entries](#)

**ldapdelete utility, [Adding and Modifying Entries Using ldapmodify](#)**

deleting entries, [Deleting Entries Using ldapdelete](#)



DNs with commas and, [Using Special Characters](#)  
example, [Deleting Entries Using Idapdelete](#)

## LDAPAPI

enabling, [Enabling LDAPAPI](#)  
overview, [Overview of Autobind and LDAPAPI](#)

Idapmodify utility, [Adding and Modifying Entries Using Idapmodify](#)

attributes with language tags, [Modifying an Entry in an Internationalized Directory](#)  
creating a root entry, [Creating a Root Entry from the Command Line](#)  
creating entries, [Adding Entries Using Idapmodify](#)  
DNs with commas and, [Using Special Characters](#)  
example, [Adding Entries Using Idapmodify](#)  
example of use, [Adding Entries Using Idapmodify](#)  
modifying entries, [Adding and Modifying Entries Using Idapmodify](#)  
schema checking and, [Adding and Modifying Entries Using Idapmodify](#)  
vs. Idapdelete, [Adding and Modifying Entries Using Idapmodify](#)

Idappasswd command-line utility

changing user password, [Changing Passwords](#)  
generating user password, [Changing Passwords](#)  
prompting for new password, [Changing Passwords](#)

Idapsearch command-line utility

extended operations, [Running Extended Operations](#)  
SASL options, [Using SASL with LDAP Client Tools](#)

Idapsearch utility

base DN and, [Using LDAP\\_BASEDN](#)  
commonly used options, [Commonly Used Idapsearch Options](#)  
DNs with commas and, [Using Special Characters](#)  
example of use, [Examples of Common Idapsearches](#)  
format, [Idapsearch Command-Line Format](#)  
international searches, [Searching an Internationalized Directory](#)  
limiting attributes returned, [Displaying Subsets of Attributes](#)  
search filters, [LDAP Search Filters](#)  
specifying files, [Displaying Subsets of Attributes](#)  
using, [Using Idapsearch](#)

LDAP\_BASEDN environment variable, [Using LDAP\\_BASEDN](#)

## LDIF

access control keywords  
groupdnattr, [Using the userattr Keyword](#)  
userattr, [Using the userattr Keyword](#)

- adding entries, [Adding Entries Using LDIF](#)
- binary data, [Representing Binary Data](#)
- change type, [Using LDIF Update Statements to Create or Modify Entries](#)
- entry format, [LDAP Data Interchange Format](#)
  - organization, [Specifying Domain Entries](#)
  - organizational person, [Specifying Organizational Person Entries](#)
  - organizational unit, [Specifying Organizational Unit Entries](#)

- example, [Defining Directories Using LDIF](#)
- internationalization and, [Storing Information in Multiple Languages](#)
- line continuation, [Continuing Lines in LDIF](#)
- Server Console and, [Adding Entries Using LDIF](#)
- specifying entries
  - organization, [Specifying Domain Entries](#)
  - organizational person, [Specifying Organizational Person Entries](#)
  - organizational unit, [Specifying Organizational Unit Entries](#)

- update statements, [Using LDIF Update Statements to Create or Modify Entries](#)
- using to create directory, [Defining Directories Using LDIF](#)

## LDIF entries

- binary data in, [Representing Binary Data](#)
- creating, [Specifying Directory Entries Using LDIF](#)
  - organizational person, [Specifying Organizational Person Entries](#)
  - organizational units, [Specifying Organizational Unit Entries](#)
  - organizations, [Specifying Domain Entries](#)

- internationalization and, [Storing Information in Multiple Languages](#)

## LDIF files

- continued lines, [Continuing Lines in LDIF](#)
- creating directory using, [Defining Directories Using LDIF](#)
- creating multiple entries, [Adding Entries Using LDIF](#)
- example, [Defining Directories Using LDIF](#)
- importing from Server Console, [Adding Entries Using LDIF](#)
- internationalization and, [Storing Information in Multiple Languages](#)

LDIF format, [LDAP Data Interchange Format](#)

- LDIF update statements, [Using LDIF Update Statements to Create or Modify Entries](#)
  - adding attributes, [Adding Attributes to Existing Entries Using LDIF](#)
  - adding entries, [Adding an Entry Using LDIF](#)
  - continued lines, [Using LDIF Update Statements to Create or Modify Entries](#)
  - deleting attribute values, [Deleting a Specific Attribute Value Using LDIF](#)
  - deleting attributes, [Deleting All Values of an Attribute Using LDIF](#)

deleting entries, [Deleting an Entry Using LDIF](#)  
modifying attribute values, [Changing an Attribute Value Using LDIF](#)  
modifying entries, [Modifying an Entry Using LDIF](#)  
syntax, [Using LDIF Update Statements to Create or Modify Entries](#)

#### ldif utility

converting binary data to LDIF, [Base-64 Encoding](#)

#### ldif2db utility, [Importing Using the ldif2db Command-Line Script](#)

options, [Running the db2index.pl Script](#)

#### ldif2db.pl perl script, [Importing Using the ldif2db.pl Perl Script](#)

#### ldif2ldap utility, [Importing Using the ldif2ldap Command-Line Script](#)

#### legacy consumer

configuration, [Configuring Legacy Replication](#)

#### legacy replication plug-in

overview, [Replication with 4.x Versions of Directory Server](#)

#### less than or equal to search

international example, [Less-Than or Equal-to Example](#)

syntax, [Using Operators in Search Filters](#)

#### less than search

international example, [Less-Than Example](#)

syntax, [Using Operators in Search Filters](#)

#### linked attributes, [Linking Attributes to Manage Attribute Values](#)

about, [About Linking Attributes](#)

and replication, [About Linking Attributes](#)

attribute requirements, [About Linking Attributes](#)

creating, [Configuring Attribute Links](#)

data consistency and ACIs, [About Linking Attributes](#)

scope, [About Linking Attributes](#)

syntax, [Looking at the Linking Attributes Plug-in Syntax](#)

#### local password policy, [Configuring a Local Password Policy](#)

#### locales

defined, [About Locales](#)

location of files, [About Locales](#)

supported, [Supported Locales](#)

#### locked accounts, [Configuring the Account Lockout Policy Using the Console](#)

#### lockout duration, [Configuring the Account Lockout Policy Using the Console](#)

#### log files, [Types of Directory Server Log Files](#)

- access log, [Types of Directory Server Log Files](#)
- audit log, [Types of Directory Server Log Files](#)
- deletion policy, [Defining a Log File Deletion Policy](#)
- error log, [Types of Directory Server Log Files](#)
- location of, [Manual Log File Rotation](#)
- manually rotating, [Manual Log File Rotation](#)
- rotation policy, [Defining a Log File Rotation Policy](#)
- viewing, [Viewing Log Files](#)
- viewing when server is down, [Viewing Log Files](#)

## logging

- for WinSync, [Troubleshooting](#)

## logging into Console

- logging in, [Launching the Console](#)

## login identity

- changing, [Changing the Login Identity](#)
- viewing, [Viewing the Current Console Bind DN](#)

## logs

- changing location and name
  - in the command line, [Changing the Log Location in the Command Line](#)
  - in the Console, [Changing the Log Name in the Console](#)

## named pipe script

- plug-ins, [Loading Plug-ins with the Named Pipe Log Script](#)

## transaction

- moving, [Configuring Transaction Logs for Frequent Database Updates](#)

- users shown for proxy authorization, [Proxied Authorization ACI Example](#)

- viewing access, [Viewing the Logs through the Console](#), [Viewing Logs in the Command Line](#)

- viewing error, [Viewing the Logs through the Console](#), [Viewing Logs in the Command Line](#)

## loop detection

- cascading chaining, [Detecting Loops](#)

## M

### macro ACIs

- example, [Macro ACI Example](#)
- overview, [Advanced Access Control: Using Macro ACIs](#)
- syntax, [Macro ACI Syntax](#)

### managed device

overview, [About SNMP](#)

managed object, [About SNMP](#)

managed role

creating, [Creating a Managed Role](#)

example, [Creating Managed Roles through the Command Line](#)

management window

opening for Directory or Admin Server, [Opening a Directory or Admin Server Window](#)

manually rotating log files, [Manual Log File Rotation](#)

markerObjectClass keyword, [Using the markerObjectClass and requiredObjectClass Keywords](#)

matching rules, [Using Matching Rules](#)

international formats, [Matching Rule Formats](#)

list of supported, [Using Matching Rules](#)

matchingRule format

using language tag, [Using a Language Tag for the Matching Rule](#)

using language tag and suffix, [Using a Language Tag and Suffix for the Matching Rule](#)

using OID, [Matching Rule Formats](#)

using OID and suffix, [Using an OID and Suffix for the Matching Rule](#)

memberOf plug-in

configuring, [Configuring an Instance of the MemberOf Plug-in](#)

from the command line, [Editing the MemberOf Plug-in from the Command Line](#)

from the console, [Editing the MemberOf Plug-in from the Console](#)

menus, in Red Hat Management Console, [Red Hat Management Console Menus](#)

metaphone phonetic algorithm, [Approximate Searches](#)

**MIB**

Directory Server, [Using the Management Information Base](#)

redhat-directory.mib, [Using the Management Information Base](#)

entity table, [Entity Table](#)

entries table, [Entries Table](#)

interaction table, [Interaction Table](#)

operations table, [Operations Table](#)

modifying

attribute values, [Changing an Attribute Value Using LDIF](#)

entries, [Modifying an Entry Using LDIF](#)

international entries, [Modifying an Entry in an Internationalized Directory](#)

modutil

loading PKCS#11 modules, [Installing PKCS#11 Modules Through the Command Line](#)

monetary format, [About Locales](#)

#### monitoring

database from command line, [Monitoring Databases from the Command Line](#)

database from server console, [Monitoring Database Activity from the Directory Server Console](#)

Directory Server, [Types of Directory Server Log Files](#)

from console, [Monitoring Server Activity](#)

log files, [Types of Directory Server Log Files](#)

replication status, [Monitoring Replication Status](#)

threads, [Monitoring the Server from the Directory Server Console](#)

with SNMP, [Monitoring Directory Server Using SNMP](#)

monitoring from console, [Monitoring Server Activity](#)

#### multi-master replication

introduction, [Multi-Master Replication](#)

preventing monopolization of the consumer, [Preventing Monopolization of the Consumer in Multi-Master Replication](#)

setting up, [Configuring Multi-Master Replication](#)

multiple search filters, [Using Compound Search Filters](#)

## N

#### named pipe script

using plug-ins, [Loading Plug-ins with the Named Pipe Log Script](#)

#### naming conflicts

in replication, [Solving Naming Conflicts](#)

#### navigation tree

overview, [The Servers and Applications Tab](#)

setting access permissions to, [Granting Admin Privileges to Users for Directory Server and Admin Server](#)

#### nested role

creating, [Creating a Nested Role](#)

example, [Creating Nested Role through the Command Line](#)

#### NetscapeRoot

and replication, [Replicating o=NetscapeRoot for Admin Server Failover](#)

nsds5ReplicaBusyWaitTime, [Preventing Monopolization of the Consumer in Multi-Master Replication](#)

nsds5ReplicaSessionPauseTime, [Preventing Monopolization of the Consumer in Multi-Master Replication](#)

nsslapd-maxbersize, [Adding Very Large Attributes](#)

nsslapd-schemacheck attribute, [Turning Schema Checking On and Off](#)

`nsview`, [About Views](#)

`nsviewfilter`, [About Views](#)

## O

### object class

adding to an entry, [Adding or Removing an Object Class to an Entry](#)

allowed attributes, [Object Classes](#)

creating, [Creating Object Classes](#)

defined, [Object Classes](#)

defining in schema, [Creating Object Classes](#), [Creating Custom Schema Files](#)

deleting, [Deleting Schema](#)

editing, [Editing Custom Schema Elements](#)

inheritance, [Object Classes](#)

parent object class, [Object Classes](#)

referral, [Creating Smart Referrals from the Command Line](#)

removing from an entry, [Adding or Removing an Object Class to an Entry](#)

required attributes, [Object Classes](#)

standard, [Overview of Schema](#)

user-defined, [Viewing Attributes and Object Classes](#)

viewing, [Viewing Attributes and Object Classes](#)

object identifier, [Managing Object Identifiers](#)

object identifier (OID), [Supported Locales](#)

in `matchingRule`, [Matching Rule Formats](#)

matching rule, [Using Matching Rules](#)

`objectClass` field (LDIF), [About the LDIF File Format](#)

### OID

getting and assigning, [Managing Object Identifiers](#)

OID, See object identifier, [Supported Locales](#)

operations, [Monitoring the Server from the Directory Server Console](#)

operations table, [Operations Table](#)

### operators

Boolean, [Using Compound Search Filters](#)

international searches and, [Supported Search Types](#)

search filters and, [Using Operators in Search Filters](#)

suffix, [Supported Search Types](#)

organization, specifying entries for, [Specifying Domain Entries](#)

organizational person, specifying entries for, [Specifying Organizational Person Entries](#)

organizational unit, specifying entries for, [Specifying Organizational Unit Entries](#)

organizational units

creating, [Organizational Units](#)

removing, [Removing an Entry from the Directory](#)

override CoS qualifier, [Handling Physical Attribute Values](#)

## P

PAM pass-through authentication, [Using PAM for Pass-Through Authentication](#)

and account inactivation, [Setting PAM PTA Mappings](#)

and password policies, [Using PAM for Pass-Through Authentication](#)

configuration options, [PAM Pass-Through Authentication Configuration Options](#)

configuring, [Configuring PAM Pass-Through Authentication](#)

entry mapping methods, [Setting PAM PTA Mappings](#)

example, [Configuring PAM Pass-Through Authentication](#)

general settings, [Configuring General PAM PTA Settings](#)

target suffixes, [Specifying the Suffixes to Target for PAM PTA](#)

parent access, [Parent Access \(parent Keyword\)](#)

parent keyword, [Parent Access \(parent Keyword\)](#)

parent object class, [Object Classes](#)

pass-through authentication

PAM, [Using PAM for Pass-Through Authentication](#)

pass-through authentication (PTA), [Using Pass-Through Authentication](#)

password

changing for a user or administrator, [Editing Entries](#)

password change extended operation, [Changing Passwords Stored Externally](#)

password file

Admin Server, [Creating a Password File for the Admin Server](#)

SSL certificate, [Creating a Password File for the Directory Server](#)

password policy

account lockout, [Configuring the Account Lockout Policy Using the Console](#)

attributes, [Configuring a Global Password Policy Using the Command Line](#)

configuring

using command line, [Configuring a Global Password Policy Using the Command Line](#)

using console, [Configuring a Global Password Policy Using the Console](#)

configuring global, [Configuring the Global Password Policy](#)

configuring local, [Configuring a Local Password Policy](#)

global, [Configuring the Global Password Policy](#)

lockout duration, [Configuring the Account Lockout Policy Using the Console](#)

managing, [Managing the Password Policy](#)

password failure counter, [Configuring the Account Lockout Policy Using the Console](#)



passwordChange, [Configuring a Global Password Policy Using the Command Line](#)  
passwordCheckSyntax, [Configuring a Global Password Policy Using the Command Line](#)  
passwordExp, [Configuring a Global Password Policy Using the Command Line](#)  
passwordGraceLimit, [Configuring a Global Password Policy Using the Command Line](#)  
passwordHistory, [Configuring a Global Password Policy Using the Command Line](#)  
passwordInHistory, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMaxAge, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMaxRepeats, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMin8bit, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMinAge, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMinAlphas, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMinCategories, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMinDigits, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMinLength, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMinLower, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMinSpecials, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMinTokenLength, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMinUppers, [Configuring a Global Password Policy Using the Command Line](#)  
passwordMustChange, [Configuring a Global Password Policy Using the Command Line](#)  
passwordStorageScheme, [Configuring a Global Password Policy Using the Command Line](#)  
passwordTrackUpdateTime, [Configuring a Global Password Policy Using the Command Line](#)  
passwordWarning, [Configuring a Global Password Policy Using the Command Line](#)  
replicating account lockout attributes, [Replicating Account Lockout Attributes](#)  
replication, [Managing the Account Lockouts and Replication](#)  
subtree-level, [Configuring a Local Password Policy](#)  
user-level, [Configuring a Local Password Policy](#)

## Password Sync, [Managing the Password Sync Service](#)

installation directory, [Step 4: Install the Password Sync Service](#)  
installed files, [Step 4: Install the Password Sync Service](#)  
installing, [Step 4: Install the Password Sync Service](#)  
modifying, [Modifying Password Sync](#)  
setting up SSL, [Step 5: Configure the Password Sync Service](#)  
starting and stopping, [Starting and Stopping the Password Sync Service](#)  
uninstalling, [Uninstalling Password Sync Service](#)

passwordChange attribute, [Configuring a Global Password Policy Using the Command Line](#)  
passwordCheckSyntax attribute, [Configuring a Global Password Policy Using the Command Line](#)  
passwordExp attribute, [Configuring a Global Password Policy Using the Command Line](#)  
passwordGraceLimit attribute, [Configuring a Global Password Policy Using the Command Line](#)  
passwordHistory attribute, [Configuring a Global Password Policy Using the Command Line](#)  
passwordInHistory attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMaxAge attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMaxRepeats attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMin8bit attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMinAge attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMinAlphas attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMinCategories attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMinDigits attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMinLength attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMinLowers attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMinSpecials attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMinTokenLength attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMinUppers attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordMustChange attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwords, [Changing the Admin User's Name and Password](#)

- account lockout, [Configuring the Account Lockout Policy Using the Console](#)
- certificate, [Creating a Password File for the Directory Server](#)
- changing, [Changing Passwords Stored Externally](#)
- failure counter, [Configuring the Account Lockout Policy Using the Console](#)
- lockout duration, [Configuring the Account Lockout Policy Using the Console](#)
- policy
  - differences between Directory Server and Active Directory, [Password Policies](#)
- setting, [Setting User Passwords](#)
- synchronizing, [Synchronizing Passwords](#)
- syncing with Active Directory, [Managing the Password Sync Service](#)

passwordStorageScheme attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordTrackUpdateTime attribute, [Configuring a Global Password Policy Using the Command Line](#)

passwordWarning attribute, [Configuring a Global Password Policy Using the Command Line](#)

PDUs, [About SNMP](#)

performance

- turning DN cache, [Setting the DN Cache Size](#)

performance counters, [Monitoring Database Activity from the Directory Server Console](#)

- configuring 64-bit, [Monitoring Server Activity](#), [Monitoring Database Activity](#), [Using the Management Information Base](#)
- configuring 64-bit integers, [Enabling and Disabling Counters](#)
- monitoring the server with, [Monitoring Server Activity](#)
- server attributes, [Enabling and Disabling Counters](#)

permissions

ACI syntax, [The ACI Syntax](#)

allowing or denying access, [Allowing or Denying Access](#)

assigning rights, [Assigning Rights](#)

overview, [Defining Permissions](#)

precedence rule, [ACI Evaluation](#)

PKCS#11 modules, [Using Hardware Security Modules](#)

installing through the command line, [Installing PKCS#11 Modules Through the Command Line](#)

plug-ins

and SELinux, [Managing SELinux Labels for Files Used by the Directory Server](#)

directory manager ACI, [Setting Access Controls on Directory Manager](#)

disabling, [Enabling Plug-ins in the Directory Server Console](#), [Enabling Plug-ins in the Command Line](#)

displaying details in the Console, [Enabling Plug-ins in the Directory Server Console](#)

distributed number assignment, [Assigning and Managing Unique Numeric Attribute Values](#)

configuring, [Configuring Unique Number Assignments](#), [Editing the DNA Plug-in in the Console](#)

overview, [Assigning and Managing Unique Numeric Attribute Values](#)

syntax, [Looking at the DNA Plug-in Syntax](#)

enabling, [Enabling Plug-ins in the Directory Server Console](#), [Enabling Plug-ins in the Command Line](#)

linked attributes, [Linking Attributes to Manage Attribute Values](#)

about, [About Linking Attributes](#)

creating instance, [Configuring Attribute Links](#)

scope, [About Linking Attributes](#)

syntax, [Looking at the Linking Attributes Plug-in Syntax](#)

setting precedence, [Setting the Plug-in Precedence](#)

pointer CoS

example, [How a Pointer CoS Works](#)

overview, [How a Pointer CoS Works](#)

port number, [Changing Standard Port Numbers](#), [Changing the Port Number](#)

changing in the command line, [Changing the Port Number in the Command Line](#)

changing in the Console, [Changing the Port Number in the Console](#)

Directory Server configuration, [Changing Directory Server Port Numbers](#)

for SSL communications, [Changing SSL Port Numbers](#)

precedence rule

ACI, [ACI Evaluation](#)

preferences, [Changing the Console Appearance](#)

font, [Changing Console Fonts](#)

UI permissions, [Changing the Console Appearance](#)

presence index, [About Index Types](#)

defaults, [Overview of System Indexes](#)

required for referential integrity, [How Referential Integrity Works](#)

presence search

example, [Using Attributes in Search Filters](#)

syntax, [Using Operators in Search Filters](#)

preventing monopolization of the consumer in multi-master replication, [Preventing Monopolization of the Consumer in Multi-Master Replication](#)

pronunciation subtype, [Adding an Attribute Subtype](#)

Property Editor

displaying, [Modifying Directory Entries](#)

protocol data units. See PDUs, [About SNMP](#)

proxy authorization

ACI example, [Proxied Authorization ACI Example](#)

users in logs, [Proxied Authorization ACI Example](#)

with cascading chaining, [Configuring Cascading Chaining from the Command Line](#)

proxy DN, [Proxied Authorization ACI Example](#)

proxy right, [Assigning Rights](#)

PTA plug-in

configuring, [Configuring the PTA Plug-in](#)

examples, [PTA Plug-in Syntax Examples](#)

syntax, [PTA Plug-in Syntax](#)

use in Directory Server, [Using Pass-Through Authentication](#)

## Q

quotation marks, in parameter values, [Using Special Characters](#)

## R

read right, [Assigning Rights](#)

read-only mode, [Monitoring Database Activity from the Directory Server Console](#)

database, [Placing a Database in Read-Only Mode](#)

read-only replica, [Read-Write and Read-Only Replicas](#)

read-write replica, [Read-Write and Read-Only Replicas](#)

Red Hat Console

overview of, [Overview of the Directory Server Console](#)

## Red Hat Management Console

defined, [Overview of the Directory Server Console](#)  
information panel, [The Servers and Applications Tab](#)  
logging into, [Launching the Console](#)  
menus, [Red Hat Management Console Menus](#)  
tabs, [Red Hat Management Console Tabs](#)

## redhat-directory.mib, [Using the Management Information Base](#)

entity table, [Entity Table](#)  
entries table, [Entries Table](#)  
interaction table, [Interaction Table](#)  
operations table, [Operations Table](#)

## ref attribute, [Creating Smart Referrals from the Command Line](#)

## refer command, [Starting the Server in Referral Mode](#)

## referential integrity

attributes, [How Referential Integrity Works](#)  
disabling, [Enabling and Disabling Referential Integrity in the Console](#)  
enabling, [Enabling and Disabling Referential Integrity in the Console](#)  
log file, [How Referential Integrity Works](#)  
modifying attributes, [Modifying the Attribute List from the Console](#)  
overview, [Maintaining Referential Integrity](#)  
required indexes, [How Referential Integrity Works](#)  
with replication, [Using Referential Integrity with Replication](#)

## referral mode, [Starting the Server in Referral Mode](#)

## referral object class, [Creating Smart Referrals from the Command Line](#)

## referrals

creating smart referrals, [Creating Smart Referrals](#)  
creating suffix, [Creating Suffix Referrals](#)  
on update, [Creating Suffix Referrals Using the Console](#)  
setting default, [Setting Default Referrals](#)  
suffix, [Creating Suffix Referrals Using the Console](#)

## reloading schema, [Dynamically Reloading Schema](#)

cn=schema reload task, [Reloading Schema Using ldapmodify](#)  
schema-reload.pl, [Reloading Schema Using schema-reload.pl](#)

## replacing attribute values, [Modifying an Entry Using LDIF](#)

## replica

exporting to LDIF, [Exporting a Replica to LDIF](#)  
read-only, [Read-Write and Read-Only Replicas](#)  
read-write, [Read-Write and Read-Only Replicas](#)

`replicate_now.sh` script, [Forcing Replication Updates from the Command Line](#)

replication

account lockout attributes, [Replicating Account Lockout Attributes](#)

and access control, [Access Control and Replication](#)

and `ou=NetscapeRoot`, [Replicating `o=NetscapeRoot` for Admin Server Failover](#)

and password policy, [Managing the Account Lockouts and Replication](#)

and referential integrity, [Using Referential Integrity with Replication](#)

and SSL, [Replication over SSL](#)

and the Admin Server, [Replicating `o=NetscapeRoot` for Admin Server Failover](#)

cascading, [Configuring Cascading Replication](#)

changelog, [Changelog](#)

compatibility with earlier versions, [Replication with 4.x Versions of Directory Server](#)

configuring from the command line, [Configuring Replication from the Command Line](#)

configuring legacy replication, [Configuring Legacy Replication](#)

configuring SSL, [Replication over SSL](#)

consumer server, [Suppliers and Consumers](#)

creating the supplier bind DN, [Creating the Supplier Bind DN Entry](#)

errors

RUV does not contain element, [Resolving Errors for Obsolete/Missing Suppliers](#)

forcing synchronization, [Forcing Replication Updates](#)

fractional, [Replicating a Subset of Attributes with Fractional Replication](#)

hub, [Suppliers and Consumers](#)

managing, [Managing Replication](#)

monitoring status, [Monitoring Replication Status](#)

multi-master, [Configuring Multi-Master Replication](#)

of ACIs, [Access Control and Replication](#)

overview, [Replication Overview](#)

purging RUV, [Resolving Errors for Obsolete/Missing Suppliers](#)

removing supplier and RUV, [Removing a Supplier from the Replication Topology](#)

`replicate_now.sh` script, [Forcing Replication Updates from the Command Line](#)

replication manager entry, [Replication Identity](#)

session hooks, [Setting Replication Session Hooks](#)

single-master, [Configuring Single-Master Replication](#)

solving conflicts, [Solving Common Replication Conflicts](#)

supplier bind DN, [Replication Identity](#)

supplier server, [Suppliers and Consumers](#)

supplier-initiated, [Suppliers and Consumers](#)

suspending, [Temporarily Suspending Replication](#)

timeout periods, [Setting Replication Timeout Periods](#)

tombstone entries

purging, [Managing Deleted Entries with Replication](#)

troubleshooting, [Troubleshooting Replication-Related Problems](#)  
unit of, [What Directory Units Are Replicated](#)  
using cl-dump.pl script, [Troubleshooting Replication-Related Problems](#)

replication agreement, [Replication Agreement](#)

replication agreements

suspending replication, [Temporarily Suspending Replication](#)

replication manager, [Replication Identity](#)

replication monitoring, [Monitoring Replication from Admin Express](#)

requiredObjectClass keyword, [Using the markerObjectClass and requiredObjectClass Keywords](#)

resource limits

setting

for anonymous binds, [Setting Resource Limits on Anonymous Binds](#)

using command line, [Setting User and Global Resource Limits Using the Command Line](#)

using console, [Setting Resource Limits on a Single User](#)

Resource Summary

viewing, [Monitoring the Server from the Directory Server Console](#)

resource use

connections, [Monitoring the Server from the Directory Server Console](#)

monitoring, [Monitoring the Server from the Directory Server Console](#)

restart

Admin Server, [Starting and Stopping the Admin Server](#)

restarting server

requirement for certain configuration changes, [Configuration Attributes Requiring Server Restart](#)

restoring data, [Backing up and Restoring Data](#)

bak2db, [Using the bak2db Command-Line Script](#)

bak2db.pl, [Using bak2db.pl Perl Script](#)

cn=tasks, [Restoring the Database through the cn=tasks Entry](#)

dse.ldif, [Restoring the dse.ldif Configuration File](#)

from console, [Restoring All Databases from the Console](#)

replicated entries, [Restoring Databases That Include Replicated Entries](#)

retro changelog

and access control, [Retro Changelog and the Access Control Policy](#)

attributes, [Using the Retro Changelog Plug-in](#)

object class, [Using the Retro Changelog Plug-in](#)

searching, [Retro Changelog and the Access Control Policy](#)

trimming, [Trimming the Retro Changelog](#)



**retro changelog plug-in**

enabling, [Enabling the Retro Changelog Plug-in](#)

overview, [Replication with 4.x Versions of Directory Server](#)

**rights**

list of, [Assigning Rights](#)

roledn keyword, [Defining Role Access - roledn Keyword](#)

**roles, [Using Roles](#)**

access control, [Using Roles Securely](#)

access to directory, [Defining Role Access - roledn Keyword](#)

activating, [Activating and Inactivating Users and Roles Using the Console](#)

assigning, [Editing and Assigning Roles to an Entry](#)

**filtered**

creating, [Creating a Filtered Role](#)

example, [Creating a Filtered Role through the Command Line](#)

inactivating, [Making a Role Inactive or Active](#)

**managed**

creating, [Creating a Managed Role](#)

example, [Creating Managed Roles through the Command Line](#)

**nested**

creating, [Creating a Nested Role](#)

example, [Creating Nested Role through the Command Line](#)

overview, [About Roles](#)

root DSE, [Searching the Root DSE Entry](#)

root entry creation, [Defining Directories Using LDIF](#)

**root suffix, [Creating Suffixes](#)**

creating from command line, [Creating Root and Sub Suffixes from the Command Line](#)

creating from console, [Creating a New Root Suffix Using the Console](#)

**RUV**

purging old supplier entries, [Resolving Errors for Obsolete/Missing Suppliers](#)

**S**

SASL, [Setting up SASL Identity Mapping](#)

authentication, [Defining Access Based on Authentication Method](#)

**configuring**

KDC server, [About the KDC Server and Keytabs](#)

configuring authentication at startup, [Configuring SASL Authentication at Directory Server Startup](#)



configuring server to server mappings, [About SASL Identity Mapping](#)  
identity mapping, [About SASL Identity Mapping](#)  
    configuring from the Console, [Configuring SASL Identity Mapping from the Console](#)  
    configuring from the command line, [Configuring SASL Identity Mapping from the Command Line](#)  
    default, [Default SASL Mappings for Directory Server](#)

#### KDC server

    configuration example, [About the KDC Server and Keytabs](#)

Kerberos, [Using Kerberos GSS-API with SASL](#)

Kerberos realms, [About Principals and Realms](#)

mechanisms, [Authentication Mechanisms for SASL in Directory Server](#)

    CRAM-MD5, [Authentication Mechanisms for SASL in Directory Server](#)

    DIGEST-MD5, [Authentication Mechanisms for SASL in Directory Server](#)

    EXTERNAL, [Authentication Mechanisms for SASL in Directory Server](#)

    GSS-API, [Authentication Mechanisms for SASL in Directory Server](#)

    PLAIN, [Authentication Mechanisms for SASL in Directory Server](#)

overview, [Setting up SASL Identity Mapping](#)

password change extended operation, [Changing Passwords Stored Externally](#)

requiring for connections, [Requiring Secure Connections](#)

requiring secure binds, [Requiring Secure Binds](#)

using with Idapsearch, [Using SASL with LDAP Clients](#)

with SELinux, [Managing SELinux Labels for Files Used by the Directory Server](#)

#### schema

    adding new attributes, [Creating Attributes](#), [Creating Custom Schema Files](#)

    assigning OIDs, [Managing Object Identifiers](#)

    checking, [Turning Schema Checking On and Off](#)

    creating new attributes, [Creating Attributes](#)

    creating new object classes, [Creating Object Classes](#)

    custom files, [Creating Custom Schema Files](#)

    deleting attributes, [Deleting Schema](#)

    deleting elements, [Deleting Schema](#)

    deleting object classes, [Deleting Schema](#)

    differences between Directory Server and Active Directory, [User Schema Differences between Red Hat Directory Server and Active Directory](#), [Group Schema Differences between Red Hat Directory Server and Active Directory](#)

        cn, [Values for cn Attributes](#)

        initials, [Constraints on the initials Attribute](#)

        street and streetAddress, [Values for street and streetAddress](#)

    editing attributes, [Editing Custom Schema Elements](#)

editing object classes, [Editing Custom Schema Elements](#)  
extending, [Managing the Directory Schema](#)  
nsslapd-schemacheck attribute, [Turning Schema Checking On and Off](#)  
reloading, [Dynamically Reloading Schema](#)  
    cn=schema reload task, [Reloading Schema Using Idapmodify](#)  
    schema-reload.pl, [Reloading Schema Using schema-reload.pl](#)

standard, [Managing the Directory Schema](#)  
viewing attributes, [Viewing Attributes and Object Classes](#)  
viewing object classes, [Viewing Attributes and Object Classes](#)

#### schema checking

and access control, [Targeting Attributes](#)  
Idapmodify and, [Adding and Modifying Entries Using Idapmodify](#)  
overview, [Turning Schema Checking On and Off](#)  
turning on or off, [Turning Schema Checking On and Off](#)  
turning on or off in the command line, [Turning Schema Checking On and Off](#)

schema-reload.pl, [Reloading Schema Using schema-reload.pl](#)

#### scripts

cl-dump.pl, [Troubleshooting Replication-Related Problems](#)

#### search filters, [LDAP Search Filters](#)

Boolean operators, [Using Compound Search Filters](#)  
contained in file, [Displaying Subsets of Attributes](#)  
examples, [LDAP Search Filters](#)  
matching rule, [Using Matching Rules](#)  
operators in, [Using Operators in Search Filters](#)  
specifying attributes, [Using Attributes in Search Filters](#)  
syntax, [LDAP Search Filters](#)  
using compound, [Using Compound Search Filters](#)  
using multiple, [Using Compound Search Filters](#)

#### Search Performance, [Search Performance and Resource Limits](#)

search right, [Assigning Rights](#)

#### search types

list of, [Using Operators in Search Filters](#)

#### searches

approximate, [Using Operators in Search Filters](#)  
equality, [Using Operators in Search Filters](#)  
example, [Examples of Common Idapsearches](#)  
greater than or equal to, [Using Operators in Search Filters](#)  
international, [Searching an Internationalized Directory](#)

international examples, [International Search Examples](#)

less than, [Less-Than Example](#)

less than or equal to, [Using Operators in Search Filters](#)

of directory tree, [Using Idapsearch](#)

presence, [Using Operators in Search Filters](#)

specifying scope, [Commonly Used Idapsearch Options](#)

substring, [Using Operators in Search Filters](#)

## searching

changing the search directory, [Searching for Users and Groups](#)

for directory entries, [Searching for Users and Groups](#)

## searching algorithm

overview, [Overview of the Searching Algorithm](#)

## Secure Sockets Layer (SSL), [TLS/SSL in Directory Server](#)

### security

LDAP URLs, [Examples of LDAP URLs](#)

setting encryption ciphers, [Setting Encryption Ciphers](#)

security strength factor, [Requiring Secure Connections](#)

self access, [Self Access \(self Keyword\)](#)

LDIF example, [Examples](#)

self keyword, [Self Access \(self Keyword\)](#)

selfwrite right, [Assigning Rights](#)

example, [Allowing Users to Add or Remove Themselves from a Group](#)

## SELinux, [Managing SELinux with the Directory Server](#), [Managing SELinux for the Admin Server](#)

and CGIs, [SELinux Definitions for the Admin Server](#)

and SSL, [Labeling SSL/TLS Ports](#)

and SSL ports, [Labeling SSL/TLS Ports](#)

Directory Server domains, [SELinux Definitions for the Directory Server](#)

Directory Server file labels, [Managing SELinux Labels for Files Used by the Directory Server](#)

Directory Server security contexts, [SELinux Definitions for the Directory Server](#)

domains, [SELinux Definitions for the Admin Server](#)

editing (GUI), [Viewing and Editing SELinux Policies for the Directory Server](#)

editing files (command line), [Managing SELinux Labels for Files Used by the Directory Server](#)

editing ports (command line), [Labeling SSL/TLS Ports](#)

files which need relabeled, [Managing SELinux Labels for Files Used by the Directory Server](#)

for SNMP, [SELinux Definitions for the SNMP Agent](#)

packages, [SELinux Definitions for the Directory Server](#)

security contexts, [SELinux Definitions for the Admin Server](#)

SNMP security contexts, [SELinux Definitions for the SNMP Agent](#)

starting servers, [Starting the Directory Server Confined by SELinux](#), [Starting the Admin Server Confined by SELinux](#)

starting the Admin Server, [SELinux Definitions for the Admin Server](#)

viewing, [Viewing and Editing SELinux Policies for the Directory Server](#)

viewing and editing (GUI), [Viewing SELinux Policies for the Admin Server](#)

with custom plug-ins, [Managing SELinux Labels for Files Used by the Directory Server](#)

with GSS-API, [Managing SELinux Labels for Files Used by the Directory Server](#)

with SASL, [Managing SELinux Labels for Files Used by the Directory Server](#)

## server

defined, [The Servers and Applications Tab](#)

opening a management window for, [Opening a Directory or Admin Server Window](#)

## server group

defined, [The Servers and Applications Tab](#)

modifying information for, [Editing Domain, Host, Server Group, and Instance Information](#)

## server instance

creating, [Creating a New Directory Server Instance](#)

modifying information for, [Editing Domain, Host, Server Group, and Instance Information](#)

## server parameters

### database

read-only, [Monitoring Database Activity from the Directory Server Console](#)

## server restart

after configuration changes, [Configuration Attributes Requiring Server Restart](#)

setting access controls, [Creating ACIs from the Console](#)

setting passwords, [Setting User Passwords](#)

simple authentication, [Defining Access Based on Authentication Method](#)

Simple Authentication and Security Layer, [Setting up SASL Identity Mapping](#)

Simple Authentication and Security Layer (SASL), [Defining Access Based on Authentication Method](#)

## simple binds

requiring secure connections, [Requiring Secure Binds](#)

Simple Network Management Protocol. See SNMP, [About SNMP](#)

Simple Sockets Layer (SSL), [Defining Access Based on Authentication Method](#)

## single-master replication

introduction, [Single-Master Replication](#)

setting up, [Configuring Single-Master Replication](#)

## smart referrals

creating, [Creating Smart Referrals](#)

creating from command line, [Creating Smart Referrals from the Command Line](#)  
creating from console, [Creating Smart Referrals Using the Directory Server Console](#)

## SNMP

configuring

Directory Server, [Configuring the Directory Server for SNMP](#)

managed device, [About SNMP](#)

managed objects, [About SNMP](#)

master agent, [About SNMP](#)

configuring, [Configuring the Master Agent](#)

MIB, [Testing the Subagent](#)

entity table, [Entity Table](#)

entries table, [Entries Table](#)

interaction table, [Interaction Table](#)

operations table, [Operations Table](#)

monitoring the Directory Server, [Monitoring Directory Server Using SNMP](#)

overview, [About SNMP](#)

subagent, [About SNMP](#)

configuration file, [Creating the Subagent Configuration File](#)

location, [Configuring the Subagent](#)

starting, [Starting the Subagent](#)

testing the subagent, [Testing the Subagent](#)

## SSF, [Requiring Secure Connections](#)

ACI example, [Setting an ACI to Require a Certain Security Strength Factor for Some Operations](#)

and SASL, [Requiring Secure Connections](#)

and Start TLS, [Requiring Secure Connections](#)

bind rule keyword, [Requiring a Certain Level of Security in Connections](#)

operators, [Requiring a Certain Level of Security in Connections](#)

setting minimum, [Requiring Secure Connections](#)

ssf keyword, [Requiring a Certain Level of Security in Connections](#)

## SSL, [Working with SSL](#)

Admin Server password file, [Creating a Password File for the Admin Server](#)

and replication, [Replication over SSL](#)

and SELinux, [Labeling SSL/TLS Ports](#), [Labeling SSL/TLS Ports](#)

authentication, [TLS/SSL in Directory Server](#)

CA certificate error messages, [Managing Certificates Used by the Directory Server Console](#)

certificate password, [Creating a Password File for the Directory Server](#)

certificate-based authentication, [Using Client \(Certificate-Based\) Authentication](#)  
certificates, [Requesting and Installing a Server Certificate](#)  
chaining with, [Creating a New Database Link Using the Console](#), [Providing an LDAP URL](#)  
client authentication, [Configuring Directory Server to Accept Certificate-Based Authentication from LDAP Clients](#)  
configuring clients to use, [Configuring Directory Server to Accept Certificate-Based Authentication from LDAP Clients](#)  
enabling, [TLS/SSL in Directory Server](#)  
installing certificates, [Installing a CA Certificate](#)  
loading PKCS#11 modules, [Using Hardware Security Modules](#)  
    command line, [Installing PKCS#11 Modules Through the Command Line](#)  
  
managing certificates for the Directory Server Console, [Managing Certificates Used by the Directory Server Console](#)  
port number, [Changing SSL Port Numbers](#)  
requiring for connections, [Requiring Secure Connections](#)  
requiring secure binds, [Requiring Secure Binds](#)  
setting encryption ciphers, [Setting Encryption Ciphers](#)  
starting the server with, [TLS/SSL in Directory Server](#)  
using hardware security modules, [Using Hardware Security Modules](#)  
using with Admin Server, [Enabling SSL](#)

SSL authentication, [Defining Access Based on Authentication Method](#)  
standard

    attributes, [Overview of Schema](#)  
    index files, [Overview of Standard Indexes](#)  
    object classes, [Overview of Schema](#)  
    schema, [Managing the Directory Schema](#)

Start TLS, [Command-Line Functions for Start TLS](#)

starting and stopping

    Directory Server and Admin Server, [Starting and Stopping Servers](#)  
    SELinux, [Starting the Directory Server Confined by SELinux](#), [Starting the Admin Server Confined by SELinux](#)

Starting and stopping

    Admin Server Console, [Opening the Admin Server Console](#)  
    Directory Server and Admin Server, [Starting and Stopping the Admin Server](#)  
    Directory Server Console, [Starting the Directory Server Console](#)

starting and stopping servers, [Starting and Stopping Servers](#)

starting the Directory Server

    with TLS/SSL, [TLS/SSL in Directory Server](#)

static group, [Groups](#)

- static groups, [Creating Static Groups in the Console](#)
  - creating, [Creating Static Groups in the Console](#)
  - modifying, [Creating Static Groups in the Console](#)
- sub suffix, [Creating Suffixes](#)
  - creating from command line, [Creating Root and Sub Suffixes from the Command Line](#)
  - creating from console, [Creating a New Sub Suffix Using the Console](#)
- substring index, [About Index Types](#)
  - required for referential integrity, [How Referential Integrity Works](#)
- substring index limitation, [About Index Types](#)
- substring search, [Using Operators in Search Filters](#)
  - international example, [Substring Example](#)
- subtree-level password policy, [Configuring a Local Password Policy](#)
- subtypes
  - of attributes, [Adding an Attribute Subtype](#)
- suffix
  - and associated database, [Creating and Maintaining Suffixes](#)
  - configuration attributes, [Creating Root and Sub Suffixes from the Command Line](#)
  - creating, [Creating a Root Entry](#)
  - creating from command line, [Creating Root and Sub Suffixes from the Command Line](#)
  - creating root suffix, [Creating a New Root Suffix Using the Console](#)
  - creating sub suffix, [Creating a New Sub Suffix Using the Console](#)
  - custom distribution function, [Adding Multiple Databases for a Single Suffix](#)
  - custom distribution logic, [Adding Multiple Databases for a Single Suffix](#)
  - disabling, [Disabling a Suffix](#)
  - in Directory Server, [Configuring Directory Databases](#)
  - using referrals, [Creating Suffix Referrals Using the Console](#)
    - on update only, [Creating Suffix Referrals Using the Console](#)
  - with multiple databases, [Adding Multiple Databases for a Single Suffix](#)
- suffix referrals
  - creating, [Creating Suffix Referrals](#)
  - creating from command line, [Creating Suffix Referrals from the Command Line](#)
  - creating from console, [Creating Suffix Referrals Using the Console](#)
- supplier bind DN, [Replication Identity](#)
- supplier server, [Suppliers and Consumers](#)
- suppliers
  - purging old entries from the RUV, [Resolving Errors for Obsolete/Missing Suppliers](#)

**synchronization****POSIX attributes**

configuring sync for, [Synchronizing POSIX Attributes for Users and Groups](#)

not syncing object classes, [Synchronizing POSIX Attributes for Users and Groups](#)

**symbols**

", in ldapsearch, [Using Special Characters](#)

-, in change operation, [Using LDIF Update Statements to Create or Modify Entries](#)

::, in LDIF statements, [Base-64 Encoding](#)

<, in LDIF statements, [Standard LDIF Notation](#)

quotation marks, in ldapmodify commands, [Using Special Characters](#)

**synchronization**

passwordTrackUpdateTime, [Configuring a Global Password Policy Using the Command Line](#)

subtree scope and deleting entries, [Handling Entries That Move Out of the Synced Subtree](#)

**synchronization agreement**

changing, [Modifying the Sync Agreement](#), [Adding and Editing the Sync Agreement in the Command Line](#)

**synchronization options**

enabling, [Allowing Sync Attributes for Entries](#)

overview, [Allowing Sync Attributes for Entries](#)

**synchronizing**

passwords, [Synchronizing Passwords](#)

**syntax**

ACI statements, [The ACI Syntax](#)

LDAP URLs, [Components of an LDAP URL](#)

ldapsearch, [ldapsearch Command-Line Format](#)

LDIF update statements, [Using LDIF Update Statements to Create or Modify Entries](#)

matching rule filter, [Using Matching Rules](#)

search filter, [LDAP Search Filters](#)

**syntax validation, [Using Syntax Validation](#)**

and error logging, [Enabling Syntax Validation Warnings \(Logging\)](#)

and warnings, [Enabling Syntax Validation Warnings \(Logging\)](#)

command-line perl script, [Validating the Syntax of Existing Attribute Values](#)

enabling and disabling, [Enabling or Disabling Syntax Validation](#)

enforcing DNS, [Enabling Strict Syntax Validation for DNS](#)

related RFCs, [About Syntax Validation](#)

**syntax-validate.pl, [Validating the Syntax of Existing Attribute Values](#)****system**



ulimit for import operations, [Importing Large Numbers of Entries](#)

## system connections

monitoring, [Monitoring the Server from the Directory Server Console](#)

system indexes, [Overview of System Indexes](#)

## system resources

monitoring, [Monitoring the Server from the Directory Server Console](#)

## T

### tables

changing column position in, [Reordering Table Columns](#)

tabs, in Red Hat Management Console, [Red Hat Management Console Tabs](#)

### target

ACI syntax, [The ACI Syntax](#)

attribute values, [Targeting Attribute Values Using LDAP Filters](#)

attributes, [Targeting Attributes](#)

keywords in ACIs, [Defining Targets](#)

overview, [Defining Targets](#)

using LDAP search filters, [Targeting Entries or Attributes Using LDAP Filters](#)

using LDAP URLs, [LDAP URLs](#)

### target DN

containing commas, [Targeting a Directory Entry](#)

target keyword, [Targeting a Directory Entry](#)

targetattr keyword, [Targeting Attributes](#)

targetattrfilters keyword, [Targeting Attribute Values Using LDAP Filters](#)

targetfilter keyword, [Targeting Entries or Attributes Using LDAP Filters](#)

### targeting

directory entries, [Targeting a Directory Entry](#)

### tasks

purging old entries from the RUV, [Resolving Errors for Obsolete/Missing Suppliers](#)

template entry. See CoS template entry., [About the CoS Template Entry](#)

### thread

monitoring, [Monitoring the Server from the Directory Server Console](#)

time format, [About Locales](#)

timeofday keyword, [Defining Access at a Specific Time of Day or Day of Week](#)

### timeout period

for replication, [Setting Replication Timeout Periods](#)

## TLS

requiring for connections, [Requiring Secure Connections](#)

## tombstone entries

purging, [Managing Deleted Entries with Replication](#)

## topology

defined, [The Servers and Applications Tab](#)

## transaction logs

moving, [Configuring Transaction Logs for Frequent Database Updates](#)

## U

unauthenticated binds, [Allowing Unauthenticated Binds](#)

user access, [Defining User Access - userdn Keyword](#)

example, [Granting Write Access to Personal Entries](#)

LDIF example, [Examples](#)

to child entries, [Parent Access \(parent Keyword\)](#)

to own entry, [Self Access \(self Keyword\)](#)

LDIF example, [Examples](#)

## user and group management

referential integrity, [Maintaining Referential Integrity](#)

## user directory

settings, [Changing the User Directory Host or Port](#)

## user entries

changing passwords for, [Editing Entries](#)

creating, [Directory and Administrative Users](#)

editing, [Editing Entries](#)

locating, [Searching for Users and Groups](#)

removing, [Removing an Entry from the Directory](#)

user passwords, [Setting User Passwords](#)

user-defined object classes, [Viewing Attributes and Object Classes](#)

user-level password policy, [Configuring a Local Password Policy](#)

userattr keyword, [Using the userattr Keyword](#)

restriction on add, [Granting Add Permission Using the userattr Keyword](#)

userdn keyword, [Defining User Access - userdn Keyword](#)

## users

activating, [Activating and Inactivating Users and Roles Using the Console](#)

inactivating, [Manually Inactivating Users and Roles](#)

Users and Groups tab, changing the search directory for, [Searching for Users and Groups](#)  
UTF-8, [Internationalization](#)

## V

value-based ACI, [Targeting Attribute Values Using LDAP Filters](#)

### viewing

access control

get effective rights, [Checking Access Rights on Entries \(Get Effective Rights\)](#)

attributes, [Viewing Attributes and Object Classes](#)

object classes, [Viewing Attributes and Object Classes](#)

viewing server information, [Viewing Server Information](#)

viewing server logs, [Viewing Server Logs](#)

virtual list view index, [About Index Types](#)

vlvindex command-line tool, [About Index Types](#)

## W

### wildcard

in LDAP URL, [Wildcards](#)

in target, [Targeting a Directory Entry](#)

### wildcards

in matching rule filters, [LDAP Search Filters](#)

WinSync, [Synchronizing Red Hat Directory Server with Microsoft Active Directory](#)

about, [About Windows Sync](#)

changing the sync agreement, [Modifying the Sync Agreement](#), [Adding and Editing the Sync Agreement in the Command Line](#)

checking sync status, [Checking Synchronization Status](#)

configuring, [Steps for Configuring Windows Sync](#)

deleting entries, [Deleting and Resurrecting Entries](#)

groups, [Synchronizing Groups](#)

logging levels, [Troubleshooting](#)

manually updating, [Sending Synchronization Updates](#)

Password Sync service, [Step 4: Install the Password Sync Service](#), [Managing the Password Sync Service](#)

modifying, [Modifying Password Sync](#)

setting up SSL, [Step 5: Configure the Password Sync Service](#)

starting and stopping, [Starting and Stopping the Password Sync Service](#)

uninstalling, [Uninstalling Password Sync Service](#)

resurrecting deleted entries, [Resurrecting Entries](#)

schema differences, [User Schema Differences between Red Hat Directory Server and Active Directory](#), [Group Schema Differences between Red Hat Directory Server and Active Directory](#)

troubleshooting, [Troubleshooting](#)

users, [Synchronizing Users](#)

write performance, [Indexing Performance](#)

write right, [Assigning Rights](#)

## APPENDIX H. REVISION HISTORY

Note that revision numbers relate to the edition of this manual, not to version numbers of Red Hat Directory Server.

<b>Revision 9.1-13</b>	<b>Jun 26, 2017</b>	<b>Marc Muehlfeld</b>
Added a statement that this documentation is deprecated and no longer maintained.		
<b>Revision 9.1-12</b>	<b>May 29, 2017</b>	<b>Marc Muehlfeld</b>
Updated section "Setting Access Control for VLV Information".		
<b>Revision 9.1-11</b>	<b>Mar 16, 2017</b>	<b>Marc Muehlfeld</b>
Added section "The Replication Keep-alive Entry".		
<b>Revision 9.1-10</b>	<b>Feb 24, 2017</b>	<b>Marc Muehlfeld</b>
Added "Fine Grained ID List Size" and "Trimming the Replication Changelog" sections. Other minor fixes.		
<b>Revision 9.1-9</b>	<b>Dec 15, 2016</b>	<b>Marc Muehlfeld</b>
Updated Replacing Log Files with a Named Pipe section. Other minor fixes.		
<b>Revision 9.1-8</b>	<b>Nov 11, 2016</b>	<b>Marc Muehlfeld</b>
Updated supported JRE version. Added ACI for o=NetscapeRoot suffix for multi-master replication.		
<b>Revision 9.1-7</b>	<b>Jun 21, 2016</b>	<b>Petr Bokoč</b>
Added information to avoid using owner nobody:nobody. Updated Schema Replication section. Other minor fixes.		
<b>Revision 9.1-6</b>	<b>June 30, 2013</b>	<b>Ella Deon Lackey</b>
Added DN cache information and disk monitoring.		
<b>Revision 9.1-2</b>	<b>March 8, 2013</b>	<b>Ella Deon Lackey</b>
Updated example for setting root DN ACI. Other minor updates.		
<b>Revision 9.1-1</b>	<b>February 21, 2013</b>	<b>Ella Deon Lackey</b>
Updates for Red Hat Enterprise Linux 6.4.		
<b>Revision 9.0-3</b>	<b>July 2, 2012</b>	<b>Ella Deon Lackey</b>
Added Automembership Plug-in information. Added logconv options and logconv usage examples.		
<b>Revision 9.0-1</b>	<b>January 30, 2012</b>	<b>Ella Deon Lackey</b>
Added information for the Account Policy Plug-in.		
<b>Revision 9.0-0</b>	<b>December 6, 2011</b>	<b>Ella Deon Lackey</b>
Initial version for Red Hat Directory Server version 9.0.		