Sun OpenDS Standard Edition 2.2 Deployment Planning Guide

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

Part No: 821–0561
December 2009

091210@23031

Contents

Preface

This book is intended to provide an overview of some of the deployments possible with the Sun OpenDS Standard Edition 2.2.

Who Should Use This Book

This book is intended for deployment architects and anyone who might be interested in configuring a Sun OpenDS Standard Edition server in a particular architecture. The aim is to present some typical deployment architectures possible with the Sun OpenDS Standard Edition.

For installation information, see the *Sun OpenDS Standard Edition 2.2 Installation Guide*.

Before You Read This Book

Before you read this book, you should be familiar with the *Sun OpenDS Standard Edition 2.2 Release Notes*.

Related Books

- *Sun OpenDS Standard Edition 2.2 Installation Guide*
- *Sun OpenDS Standard Edition 2.2 Administration Guide*
- *Sun OpenDS Standard Edition 2.2 Command-Line Usage Guide*

Related Third-Party Web Site References

Third-party URLs are referenced in this document and provide additional, related information.

Note – Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Documentation, Support, and Training

The Sun web site provides information about the following additional resources:

- Documentation (`http://www.sun.com/documentation/`)
- Support (`http://www.sun.com/support/`)
- Training (`http://www.sun.com/training/`)

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. To share your comments, go to `http://docs.sun.com` and click Feedback.

Typographic Conventions

The following table describes the typographic conventions that are used in this book.

TABLE P–1 Typographic Conventions

Typeface	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`
`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`** `Password:`
aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.

TABLE P–1 Typographic Conventions *(Continued)*

Typeface	Meaning	Example
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows the default UNIX® system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE P–2 Shell Prompts

Shell	Prompt
C shell	`machine_name%`
C shell for superuser	`machine_name#`
Bourne shell and Korn shell	`$`
Bourne shell and Korn shell for superuser	`#`

◆ ◆ ◆ CHAPTER 1

Overview of Sun OpenDS Standard Edition

Sun OpenDS Standard Edition is a comprehensive next generation directory service that is designed to address large deployments, to provide high performance, to be highly extensive and to be easy to deploy, manage, and monitor. This chapter provides a high-level overview of the Sun OpenDS Standard Edition directory service.

Sun OpenDS Standard Edition Components

Sun OpenDS Standard Edition can function in one of two modes:

- as an LDAP directory server
- as a proxy server

The mode in which the server runs depends on how you install the software. For more information, see "Installing the Directory Server" in *Sun OpenDS Standard Edition 2.2 Installation Guide* and "Installing the Proxy Server" in *Sun OpenDS Standard Edition 2.2 Installation Guide*.

◆ ◆ ◆ CHAPTER 2

Overview of the Directory Server

This chapter provides a brief overview of the directory server component of Sun OpenDS Standard Edition directory server. The chapter covers the following topics:

For additional information about deploying Sun OpenDS Standard Edition as a directory server, see Chapter 5, "Example Deployments Using the Directory Server."

Directory Server Features

The Sun OpenDS Standard Edition directory server is an LDAPv3 compliant directory server written entirely in Java. The directory server includes the following high-level functionality:

- *Full LDAPv3 compliance (RFC 4510–4519) with support for numerous standard and experimental extensions*
- *High performance and space effective data storage*
- *Ease of configuration and administration*
  - A highly extensible administrative framework that enables you to customize most of the features listed below.
  - An administration connector that manages all administration traffic to the server. The administration connector enables the separation of user traffic and administration traffic to simplify logging and monitoring, and to ensure that administrative commands take precedence over commands that manipulate user data.
  - A graphical control panel that displays server status information and enables you to perform basic server and data administration.
  - Several command-line utilities to assist with configuration, administration tasks, basic monitoring, and data management. The main configuration utility (`dsconfig`) provides an interactive mode that walks you through most configuration tasks.
- *An advanced replication mechanism*

- Enhanced multi-master replication across directory server instances
- An assured replication feature that ensures high availability of data and immediacy of data availability for specific deployment requirements
- Fractional replication capabilities
- Support for an external change log that publicizes all changes that have occurred in a directory server database

- *An extensible security model*
  - Support for various levels of authentication and confidentiality
  - Access to resources based on privileges
  - An advanced access control mechanism
- *Multi-faceted monitoring capabilities*
- *Rich user management functionality*
  - Password policies
  - Identity mapping
  - Account status notification
- *A DSML to LDAP gateway*

◆ ◆ ◆ CHAPTER 3

Overview of the Proxy Server

This chapter provides a brief overview of the proxy component of Sun OpenDS Standard Edition. The chapter covers the following topics:

For additional information about deploying Sun OpenDS Standard Edition as a proxy, see the following sections:

- Chapter 4, "Building Blocks of the Proxy Server"
- Chapter 6, "Example Deployments Using the Proxy Server"

What Is the Proxy Server?

The Sun OpenDS Standard Edition proxy is an LDAPv3 compliant server that does not store data but routes LDAP requests from clients to the directory servers that are spread across an enterprise.

The proxy is the entry point to a directory service deployment spread over multiple directory servers and/or multiple data centers. All client requests are routed by the proxy to the appropriate remote LDAP server. The Sun OpenDS Standard Edition proxy component can be used with any LDAP v3–compliant directory server, such as the Sun OpenDS Standard Edition directory server or Sun Directory Server Enterprise Edition 7.0.

In order to route data requests to the remote LDAP servers, the proxy component can be configured to use either *load balancing* or *data distribution*, or both.

You can deploy the Sun OpenDS Standard Edition proxy in very simple configurations, or in more complex, replicated scenarios, using `vdp-setup`. Some simple deployments are detailed in Chapter 6, "Example Deployments Using the Proxy Server."

Note – The proxy component cannot be used directly as a datastore.

As the interface between the client and the remote LDAP server, the proxy provides a number of security features, to ensure secure connection if and when required. For more information about security, see "Configuring Security Between the Proxy and the Data Source" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

For an in-depth presentation of the elements that constitute the Sun OpenDS Standard Edition proxy , see Chapter 4, "Building Blocks of the Proxy Server."

Why Use the Proxy Server?

The proxy manages all the connections between a client and a data source (be it a single server, replicated server, or data center). As such, it centralizes all the rules for client connections, including handling load balancing, data distribution and security with the data source.

To deploy a *highly available* directory service, you must have at least two replicated directory servers. To ensure that requests that fail to the first server are treated by the backup server, you must ensure that all the clients know the addresses for both data sources, and are coded to treat a failure on the primary server by re-sending the request to the backup server. The proxy handles the failover and *load balancing* of requests, thereby simplifying high availability and scalability.

Typically, if your deployment used only one server to store all the data, you would have performance issues if your data store was too large. You could resolve this issue by replacing the single server with several servers, and splitting the data across these servers. In this case, each client application would need to know which server to search for its data. With the proxy, there is no need to replicate the distribution information for each application, because the proxy manages the distribution of requests to the appropriate data source. Instead, the client application sends a request to the proxy. The proxy knows which partition holds the requested data and handles the request using *distribution*.

By including the proxy in your deployment, you ease the configuration and management of client applications. The proxy centralizes and handles all requests, ensuring load balancing and/or distribution of requests.

The proxy also provides a single access point for managing security in a directory service. You can use the proxy to authorize or restrict access to remote directory servers. In addition, if you want to perform maintenance or back up an LDAP server, you can simply modify your proxy deployment to avoid service interruption.

For a description of sample deployments, see Chapter 6, "Example Deployments Using the Proxy Server."

◆ ◆ ◆ CHAPTER 4

Building Blocks of the Proxy Server

This chapter describes in more detail the role of each of the proxy building blocks. The chapter covers the following topics:

Overview of the Proxy Server

As illustrated in Figure 4–1, a client request is managed by the proxy before being forwarded to the data source.

A client request follows this path through the proxy:

1. The request is attached to a network group based on the criteria and a QOS policy is assigned.
2. The network group forwards the request to a workflow, which defines the naming context.
3. The workflow forwards the request to a workflow element, which defines the how the data request will be treated. That is, if it will go through distribution or load balancing.
4. Once it has gone through the distribution or load balancing flow, the request is sent to the data source.

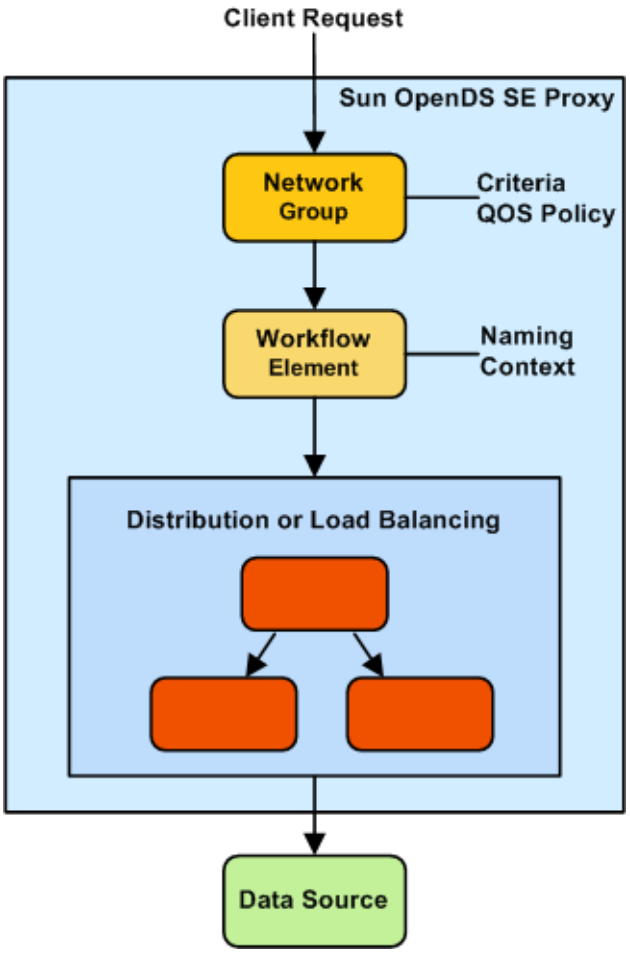


FIGURE 4–1 High Level Presentation of Proxy Elements

Network Groups

Network groups are the entry point of all client requests handled by the Sun OpenDS Standard Edition proxy.

The network groups handle all client interactions and dispatch them to workflows, based on:

- Criteria

  Criteria can include security authentication level, port number, client IP mask, client bind DN, domain name, and other criteria.
- Quality Of Service (QOS) policies

QOS policies can include LDAP referral policy, request filtering, client connection affinity, and resource limits.

Within the Sun OpenDS Standard Edition proxy, you can have more than one network group defined, each with different properties and different priorities. However, the incoming client connection can only be attached to one network group at a time. An incoming client connection is attached to the first network group for which a client connection complies with the criteria defined for that network group.

The client connection is assessed by each network group, in order of priority, until it complies with all the criteria of that network group. As illustrated in Figure 4–2, the request is first sent to the network group with the highest priority: `Network Group 1`. `Network Group 1` assesses if the request matches all the required criteria. If it does not match all of the criteria, it forwards the request to the next network group in the list: `Network Group 2`.

If the request matches all the properties of a network group, the network group assesses if the client connection matches the QOS policies of that network group. If it matches the QOS policies, it is routed to the associated *workflow*.

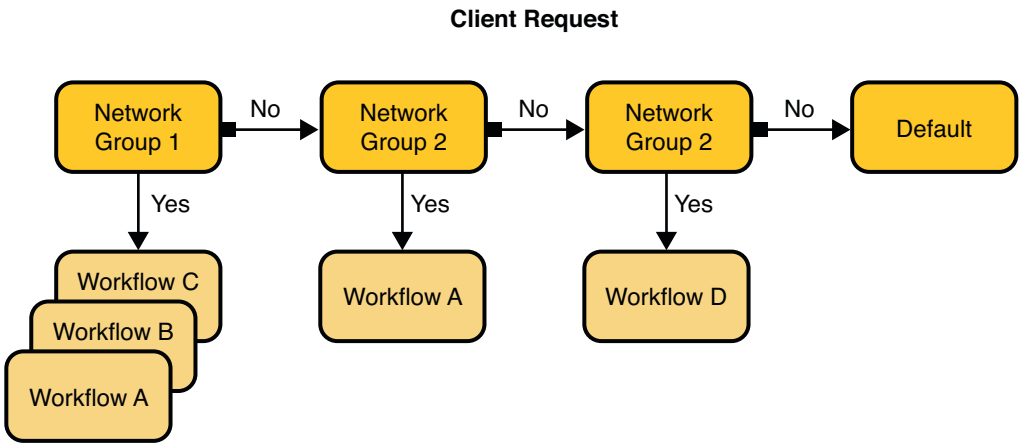


FIGURE 4–2 Network Group Selection

A network group can be associated with one or more workflows, each workflow corresponding to a different naming context. For more information of workflows, see "Workflows" on page 19. However, if the client connection matches the criteria of a network group, but not the QOS policies of that network group, the connection is not forwarded to the workflow, nor is it sent to the next network group. You will get an error message indicating the QOS policy that causes an error.

The Sun OpenDS Standard Edition proxy comes with a default network group. If a client connection does not match any of the network groups in your deployment, it is attached to this default network group. However, this network group does not have any workflows attached to it, so your request will not be treated. You will get an error message indicating: `No such entry`.

For information on managing network groups, see "Configuring Network Groups" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

EXAMPLE 4–1 Using Network Group Criteria to Route to Different Workflows

For example, if a Sun OpenDS Standard Edition proxy has the following network groups:

- `Network Group 1`: criteria set with bind DN `**,dc=example,dc=com`

  This network group is associated to `Workflow 1`, with naming context `dc=example,dc=com`
- `Network Group 2`: criteria set with bind DN `**,dc=test,dc=com`

  This network group is associated to `Workflow 2`, with naming context `dc=test,dc=com`

Depending on your bind DN, your search would be routed through `Network Group 1` or `Network Group 2`. For example, if your bind DN is `uid=user.1,dc=test,dc=com`, your request is not accepted by `Network Group 1`, but forwarded to and accepted by `Network Group 2`, and forwarded to `Workflow 2`.

EXAMPLE 4–2 Using Network Group QOS Policy to Filter Requests

For example, if a Sun OpenDS Standard Edition proxy has the following network groups:

- `Network Group 1`: criteria set with bind DN `**,ou=admin,dc=example,dc=com`

  QOS policy set with resource limits size limit=0, time limit=0. Therefore, for admin group, there are no limits.

  This network group is associated to `Workflow 1`, with naming context `dc=example,dc=com`.
- `Network Group 2`: criteria set with bind DN `**,dc=example,dc=com`

  QOS policy set with resource limits `size limit=100`, `time limit=30 s`. Therefore, for all connections other than admin group, there are limits set on the resources used.

  This network group is also associated to `Workflow 1`, with naming context `dc=example,dc=com`.

Therefore, as long as the bind DN is `dc=example,dc=com`, the requests will be forwarded to `Workflow 1`. The QOS policy set for `Network Group 2` gives restricted access to `Workflow 1`, for anyone that is not `admin`. Anyone who binds as `admin` will access `Workflow 1` through `Network Group 1`, and will have no limitations on resource limits.

Workflows

A workflow is defined by a *naming context* (base DN) and a workflow element that define how the Sun OpenDS Standard Edition proxy should handle an incoming request. A workflow must be registered with at least one network group, but can be attached to several network groups.

A network group can point to *several* workflows when the naming context of the workflows are different. However, several network groups can point to the *same* workflow when the network group QOS policies are different, but the naming context of the workflow is the same.

EXAMPLE 4–3 A Network Group Routing to Several Workflows

For example, if a Sun OpenDS Standard Edition proxy has the following network groups (as illustrated in Figure 4–2), where:

- `Network Group 1` with a bind DN of `**,l=fr,dc=example,dc=com`

  This network group is associated to `Workflow 1`, with naming context `l=fr,dc=sun,dc=com`
- `Network Group 2` with a bind DN of `**,l=uk,dc=example,dc=com`

  This network group is associated to `Workflow 2`, with naming context `l=uk,dc=example,dc=com`
- `Network Group 3` with a bind DN of `**,dc=example,dc=com`

  This network group is associated to `Workflow 1` and `Workflow 2`, with naming context `dc=example,dc=com`

A search with bind DN `**,l=uk,dc=sun,dc=com` would be handled by `Network Group 2` and sent to `Workflow 2`.

A search with bind DN `**,dc=sun,dc=com` would be handled by `Network Group 3` and sent to `Workflow 1` and `Workflow 2`.

Workflow Element

Each workflow contains at least one *workflow element*. Workflow elements are part of a routing structure.

Within the Sun OpenDS Standard Edition proxy there are different types of workflow elements:

- Load Balancing workflow elements.
- Distribution workflow elements.
- Proxy workflow elements.

The workflow elements can be classified in two types. The load balancing and distribution workflow elements act as a pointer, routing the request along one path. The proxy workflow element, however, has only one function and is a direct access to the data source.

Moreover, the Sun OpenDS Standard Edition proxy has a number of built-in workflow elements. These should not be modified or deleted.

Load Balancing Using the Proxy

You can use the Sun OpenDS Standard Edition proxy to route requests across multiple data sources or LDAP servers. These LDAP servers must be replicated in order to contain identical data.

When using the Sun OpenDS Standard Edition proxy, the client requests are routed to one of the data sources based on the *load balancing algorithm* set.

You can choose one of the following load balancing algorithms:

- Proportional. All the LDAP servers handle requests, based on the proportions (weight) set.
- Failover. There is one main LDAP server that handles all requests, until there is a failure.
- Saturation. There is one main LDAP server that handles all requests, until a saturation limit is reached.

Proportional Load Balancing

With the proportional load balancing algorithm, the Sun OpenDS Standard Edition proxy forwards requests across multiple routes to back-end LDAP servers or data sources, based on the proportions set. The proportion of requests handled by a route is identified by the weight that you set for each route in your configuration. The weight is represented as an integer value.

When you define your load balancing configuration in the Sun OpenDS Standard Edition proxy, you must indicate the proportion of requests handled by each LDAP server. In the example in Figure 4–3, Server 1 handles twice as many connections as Server 2, since the weight is set with a proportion of 2:1. Server 2 and Server 3 handle the same amount of requests (1:1).

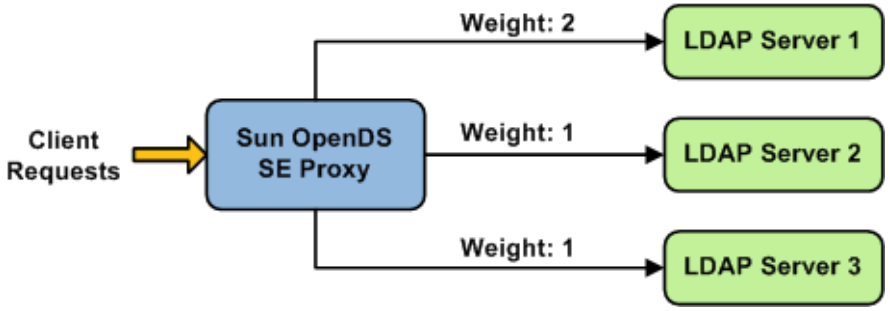


FIGURE 4–3 Proportional Load Balancing Example

With the Sun OpenDS Standard Edition proxy, you can configure a specific weight for each type of client operation, as illustrated in Figure 4–4. For example, in you want Server 1 to handle all the Bind operations, this is possible. To do so, set the weight of bind to `1` (or higher) for Server 1, and to `0` for Server 2 and Server 3.

In the example illustrated in Figure 4–4, Server 1 will handle three times as many Add requests as Server 2 and Server 3. However, Server 1 will handle only one half the Search requests handled by Server 2, and Server 3. Server 2 and Server 3 will handle the same amount of Add and Search requests, but will not handle Bind requests.

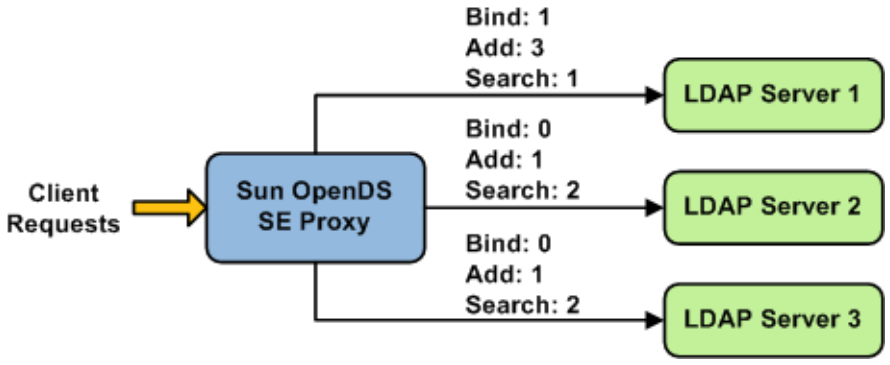


FIGURE 4–4 Proportional Load Balancing with Request Specific Management

If you do not modify the weights of operations other than Bind, Add, and Search, as illustrated in Figure 4–4, the servers will share the same load for all other operations (for example for Delete operations).

For more information on configuring the load balancing weights of routes when using proportional load balancing, see "Modifying the Load Balancing Route Properties" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

Failover Load Balancing

In load balancing with failover algorithm, the Sun OpenDS Standard Edition proxy routes requests to the remote LDAP server or data center with the highest priority. The Sun OpenDS Standard Edition proxy continues to send requests to the priority route until the back-end goes down. This may be caused by a network cut, a hardware failure, a software failure or some other problem. At failover time, the Sun OpenDS Standard Edition proxy routes incoming requests to the route with the second highest priority.

Figure 4–5 illustrates a typical failover load balancing configuration. In this example, there are three routes, each with a unique priority. All requests are treated by Server 1, since it has the highest priority, that is `priority=1`. If Server 1 goes down, the requests are sent to the server with the second highest priority, that is, priority 2.

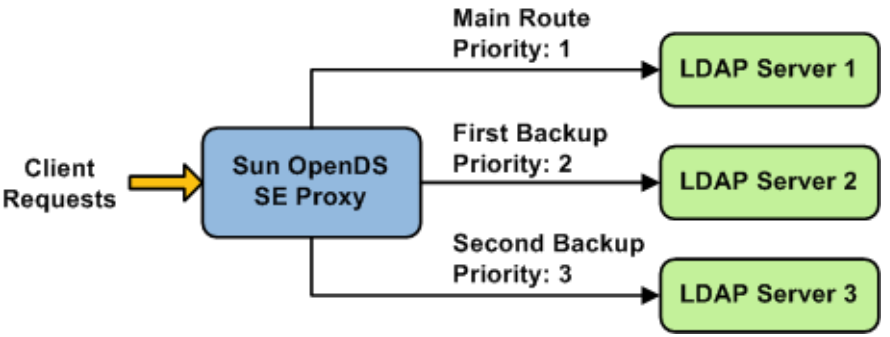


FIGURE 4–5 Failover Load Balancing Example

By default, the Sun OpenDS Standard Edition proxy does not immediately re-route requests to a server that has gone down, once it is running again. For example, if Server 1 goes down, the requests are sent to Server 2. Even when Server 1 is up again, Server 2 continues to handle incoming requests. However, if Server 2 goes down, and Server 1 is up again, Server 1 will now receive incoming requests. Only if both Server 1 and Server 2 are down at the same time will request be sent to Server 3. This default behavior can be changed with the switch-back flag. For more information on configuring the switch-back flag, see "Modifying the Load Balancing Algorithm Type" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

For failover to work effectively, the monitoring check interval must be set to be low enough so that the failover happens inside a time interval that suits your business needs. For details about setting the monitoring check interval, see "LDAP Data Source Monitoring Connection Properties" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

Saturation Load Balancing

With the saturation load balancing algorithm, the Sun OpenDS Standard Edition proxy sends requests to a chosen priority route. The Sun OpenDS Standard Edition proxy continues to send requests to the priority route until the back-end LDAP server on that route passes the saturation threshold set. The saturation threshold is represented as a percentage.

For example, if you want a back-end LDAP server to manage all incoming requests, set it as priority 1. If you want that same back-end LDAP server to stop handling requests when its saturation index reaches 70%, set the saturation threshold to 70%, as illustrated in Figure 4–6. In this way, the server handles all incoming requests until it becomes 70% saturated. The Sun OpenDS Standard Edition proxy then sends all new requests to the back-end LDAP server to Server 2, since it has the next highest priority. Server 2 will continue to handle requests until it reaches its own saturation threshold, or until Server 1 is no longer saturated.

In other words, if Server 1 reaches 70% saturation, Sun OpenDS Standard Edition proxy directs the requests to Server 2. If Server 1 is still at 70%, and Server 2 reaches 60%, then Sun OpenDS Standard Edition proxy directs the new requests to Server 3.

However, if while Server 2 is handling requests, the saturation level of Server 1 drops to 55%, thenSun OpenDS Standard Edition proxy will direct all new requests to Server 1, even if Server 2 has not reached its saturation threshold.

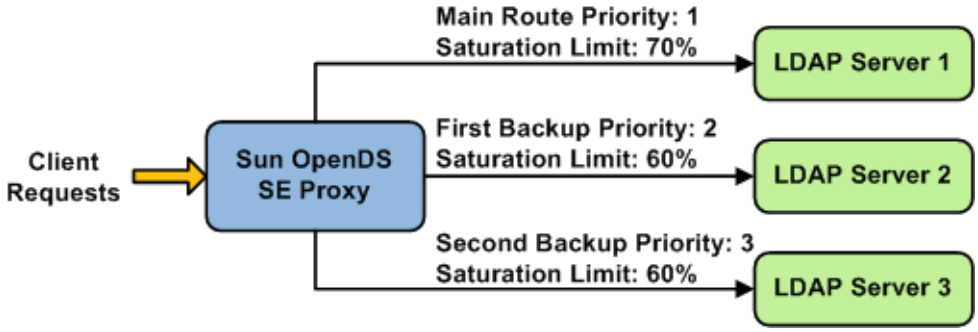


FIGURE 4–6 Saturation Load Balancing Example

If all routes have reached their saturation threshold, the Sun OpenDS Standard Edition proxy chooses the route with the lowest saturation.

With the Sun OpenDS Standard Edition proxy, you can set a saturation threshold alert. For example, if you set a saturation threshold alert to 60%, you will get a notification when the server reaches this limit, and thus, you can act before the server becomes too degraded.

Data Distribution Using the Proxy

The distribution feature in Sun OpenDS Standard Edition proxy addresses the challenge of large deployments, such as horizontal scalability, where all the entries cannot be held on a single data source, or LDAP server. Using distribution can also help you scale the number of updates per second.

When using the Sun OpenDS Standard Edition proxy with distribution, you must first split the data in your database into smaller chunks. To split the data, you can use the `split-ldif` command. These chunks of data are called *partitions*. Typically, each partition is stored on a separate server.

The split of the data is based on one of the following distribution algorithms:

- Numeric. Entries are split into partitions and distributed based on the numeric value of the naming attribute (for example uid).
- Lexico. Entries are split into partitions and distributed based on the alphabetical value of the naming attribute (for example cn).
- DN pattern. Entries are split into partitions and distributed based on the pattern (value) of the entry DN.

The type of data distribution you choose will depend on how the data in your directory service is organized. Numeric and lexico distribution have a very specific format for distribution. DN pattern can be adapted to match an existing data distribution model.

If a client request (except Add) cannot be linked to one of the distribution partitions, the Sun OpenDS Standard Edition proxy broadcasts the incoming request to all the partitions, unless a *global index catalog* has been configured.

However, if the request is clearly identified as outside the scope of the distribution, the request is returned with an error indicating that the entry does not exist. For example, if the distribution partitions includes data with uids from 1–100 (`partition1`) and 100–200 (`partition2`) but you run a search where the base DN is `uid=222,ou=people,dc=example,dc=com`, the Sun OpenDS Standard Edition proxy will indicate that the entry does not exist.

Moreover, for the numeric and lexico algorithms, it is the *first RDN* after the distribution base DN that is used to treat a request. For example, the following search will return an error, as the uid is not the first RDN after the distribution base DN, for example `ou=people,dc=example,dc=com`.

```
$ ldapsearch -b "uid=1010,o=admin,ou=people,dc=example,dc=com" "objectclass=*"
```

Consider the number of partitions carefully. When you define the number of partitions you want in your deployment, you should note that you cannot split and redistribute the data into new partitions without downtime. You can, however, add a new partition with data that has entries outside the initial ones.

For example, if the initial partitions cover data with uids from 1–100 (`partition1`) and 100–200 (`partition2`), you can later add a `partition3` which includes uids from 200–300. However, you cannot easily split `partition1` and `partition2` so that `partition1` includes uids 1–150 and `partition2` includes uids 150–300, for example. Splitting partitions is essentially like re-configuring a new distribution deployment.

Numeric Distribution

With a distribution using numeric algorithm, the Sun OpenDS Standard Edition proxy forwards requests to one of the partitions, based on the numeric value of the first RDN after the distribution base DN in the request. When you set up distribution with numeric algorithm in Sun OpenDS Standard Edition proxy, you split the data of your database into different partitions based on a numerical value of the attribute of your choice, as long as the attribute represents a numerical string. The Sun OpenDS Standard Edition proxy then forwards all client requests to the appropriate partition, using the same numeric algorithm.

For example, you could split your data into two partitions based on the uid of the entries, as illustrated in Figure 4–7.

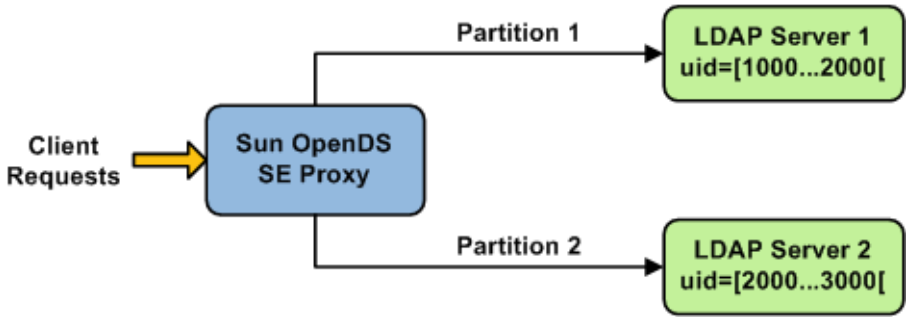


FIGURE 4–7 Numeric Distribution Example

In this example, a search for an entry with a uid of `1111` is sent to Partition 1, while a search for an entry with a uid of `2345` is sent to Partition 2. Any request for an entry with a uid outside the scope of the partitions defined will indicate that no such entry exists.

Note – The upper boundary limit of a distribution algorithm is exclusive. This means that a search for uid `3000` in the example above returns an error indicating that the entry does not exist.

EXAMPLE 4–4 Examples of Searches Using Numeric Distribution Algorithm

The following search will be successful:

```
$ ldapsearch -b "uid=1010,ou=people,cn=example,cn=com" "cn=Ben"
```

However, the following searches will indicate that the entry does not exist (with result code 32):

```
$ ldapsearch -b "uid=1010,o=admin,ou=people,cn=example,cn=com" "objectclass=*"

$ ldapsearch -b "uid=99,ou=people,cn=example,cn=com" "objectclass=*"
```

The following search will be broadcast, as the Sun OpenDS Standard Edition proxy cannot determine the partition to which the entry belongs, using the distribution algorithm defined above:

```
$ ldapsearch -b "ou=people,cn=example,cn=com" "uid=*"
```

Lexico Distribution

With a distribution using lexico algorithm, the Sun OpenDS Standard Edition proxy forwards requests to one of the partitions, based on the alphabetical value of the first RDN after the distribution base DN in the request. When you set up distribution with lexico algorithm in Sun OpenDS Standard Edition proxy, you split the data of your database into different partitions, based on an alphabetical value of the attribute of your choice. The Sun OpenDS Standard Edition proxy then forwards all client requests to the appropriate partition, using the same algorithm.

For example, you could split your data into two partitions based on the cn of the entries, as illustrated in Figure 4–8.

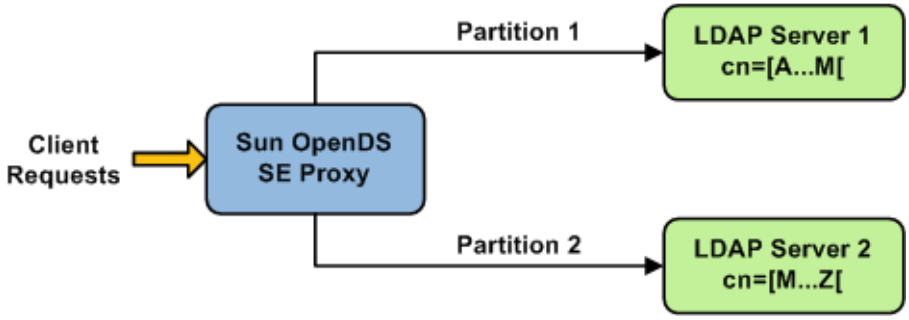


FIGURE 4–8 Lexico Distribution Example

In this example, any requests for an entry with a cn starting with `B` such as `Ben` are sent to Partition 1, while requests for an entry with a cn from `M`–`Y` are sent to Partition 2.

Note – The upper boundary limit of a distribution algorithm is exclusive. This means that a search for cn= `Zachary` in the example above will indicate that no such entry is found. In order to include entries starting with Z in the search boundaries, then you should use the `unlimited` keyword. For example, `cn=[M..unlimited[` will include all entries beyond M.

EXAMPLE 4–5 Examples of Searches Using Lexico Distribution Algorithm

The following search will be successful:

```
$ ldapsearch -b "cn=Ben,ou=people,cn=example,cn=com" "objectclass=*"
```

The following search will also be successful, as `cn=Ben` is the *first* RDN.

```
$ ldapsearch -b "uid=1010,cn=Ben,ou=people,cn=example,cn=com" "objectclass=*"
```

However, the following searches will indicate that the entry does not exist (with result code `32`):

```
$ ldapsearch -b "cn=Ben,o=admin,ou=people,cn=example,cn=com" "objectclass=*"
```

```
$ ldapsearch -b "cn=Zach,ou=people,cn=example,cn=com" "objectclass=*"
```

The following search cannot be determined by the distribution to which partition it belongs and will be broadcast:

```
$ ldapsearch -b "ou=people,cn=example,cn=com" "cn=*"
```

DN Pattern Distribution

With a distribution using DN pattern algorithm, the Sun OpenDS Standard Edition proxy forwards requests to one of the partitions, based on the match between a request base DN and a string pattern. The match is only perform on the relative part of the request base DN, that is, the part after the distribution base DN. For example, you could split your data into two partitions based on a the DN pattern in the uid of the entries, as illustrated in Figure 4–9.

Distribution using DN pattern is more onerous than distribution with numeric or lexico algorithm. If possible, use either numeric or lexico distribution algorithms.

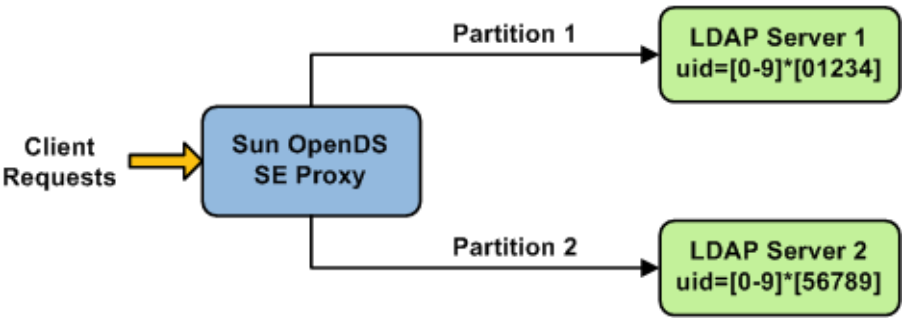


FIGURE 4–9 DN Pattern Distribution Example

In this example, all the data entries with a uid that ends with 0, 1, 2, 3, or 4 will be sent to Partition 1. Data entries with a uid that ends with 5, 6, 7, 8, or 9 will be sent to Partition 2.

This type of distribution, although using numerical values is quite different from numeric distribution. In numerical distribution, the data is partitioned based on a numerical *range*, while DN pattern distribution is based on a *pattern* in the data string.

Distribution using a DN pattern algorithm is typically used in cases where the distribution partitions do not correspond exactly to the distribution base DN. For example, if the data is distributed as illustrated in Figure 4–10, the data for Partition 1 and Partition 2 is in both base DN `ou=people,ou=region1` and `ou=people,ou=region2`. The only way to distribute the data easily is to use the DN pattern.

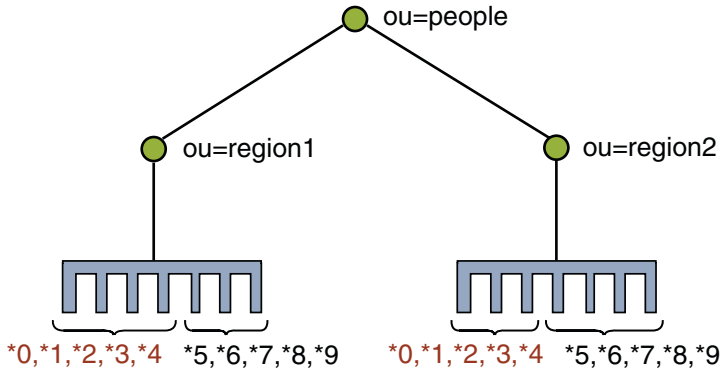


FIGURE 4–10 Example of Directory Information Tree

EXAMPLE 4–6 Example of DN Pattern Algorithm Split by Region

If the deployment of the information is based in two geographical locations, it may be easier to use the DN pattern distribution to distribute the data. For example, if employee numbers were 4 digit codes, where the first digit indicated the region, then you could have the following:

Region 1	Region 2
1000	2000
1001	2001
1002	2002
1003	2003
1004	2004
1005	2005
1006	2006
1007	2007
1008	2008
1009	2009
1010	2010
...	...

In order to spread the load of data, the entries in each location are split over two servers, where Server 1 contains all entries that end with 0, 1, 2, 3, and 4, while Server 2 contains all the entries that end with 5, 6, 7, 8, and 9, as illustrated in Figure 4–9.

Therefore, a search for DN pattern `1222` would be sent to partition 1, as would `2222`.

Global Index Catalog

A global index catalog can be used with a distribution deployment. The global index catalog maps the entries to the distribution partition in which the data is held. When Sun OpenDS Standard Edition proxy receives a request from the client, the distribution looks up the attribute entry in the global index catalog, and forwards the client request to the correct partition. This diminishes the need for broadcasts. Moreover, if a modify DN request is made, the global index catalog will ensure that the entry is always found.

A global index catalog maps the entries based on specific attributes, such as employee number or telephone number. The value of the attribute to be indexed must be unique across all the entries. You cannot use a global index to map entries based on country, for example, as that information is not unique.

A global index catalog can include several global indexes. Each global index maps a different attribute. For example, you can have one global index catalog called `GI-catalog`, which includes a global index mapping the entries based on the *telephone number* and one mapping the entries based on the *employee number*. This means that you can forward client requests to the right partition using either the telephone number or the employee number.

Global indexes catalogs and global index are created and configured using the `gicadm` command. For more information see "Configuring the Global Index" in *Sun OpenDS Standard Edition 2.2 Administration Guide* and "gicadm" in *Sun OpenDS Standard Edition 2.2 Command-Line Usage Guide*.

The global indexes can be populated with data from LDIF files. The data from one LDIF file can be split into partitions using the `split-ldif` command. For more information, see "split-ldif" in *Sun OpenDS Standard Edition 2.2 Command-Line Usage Guide*

A global index catalog is held in memory and should be replicated, in order to avoid a single point of failure. For information on replicating the global index catalog, see "Replication of Global Index Catalogs" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

EXAMPLE 4–7 Using a Global Index Catalog for Telephone Numbers

A typical example of a unique attribute which can be used to create a global index is a telephone number: the value of the attribute is unique, that is, only one person (employee, for example) can have that telephone number.

In the example below, the entries in the database have been split based on the telephone number. The global index includes the following information:

EXAMPLE 4–7 Using a Global Index Catalog for Telephone Numbers *(Continued)*

Value	Partition ID
4011233	1
4011234	1
7054477	2

The global index does not store the name of the employees, location, and other attribute values that may be associated to the telephone number. It only maps the attribute indexed to the partition. The data associated to the indexed value (here telephone number) is stored in the remote LDAP server.

If an employee has multiple phone numbers, these are regarded as multi-valued entries. In this case, if the global index is created based on the telephone number, there will be two global index entries that will result in finding one employee, say Ben Brown.

In the example above, employee Ben Brown could have both telephone number `4011233` and `7054477` assigned to him. In this case, a search on one of Ben Brown's telephone number would return the correct partition, and all the information associated to the telephone number, including the name Ben Brown, regardless that he has two phone numbers attributed to him.

◆ ◆ ◆ CHAPTER 5

Example Deployments Using the Directory Server

This chapter provides sample configurations for a replicated topology including multiple instances of the Sun OpenDS Standard Edition directory server.

For a complete understanding of how replication works in Sun OpenDS Standard Edition, see Chapter 6, "Directory Server Replication," in *Sun OpenDS Standard Edition 2.2 Architectural Reference*.

This chapter covers the following topics:

Small Replicated Topology

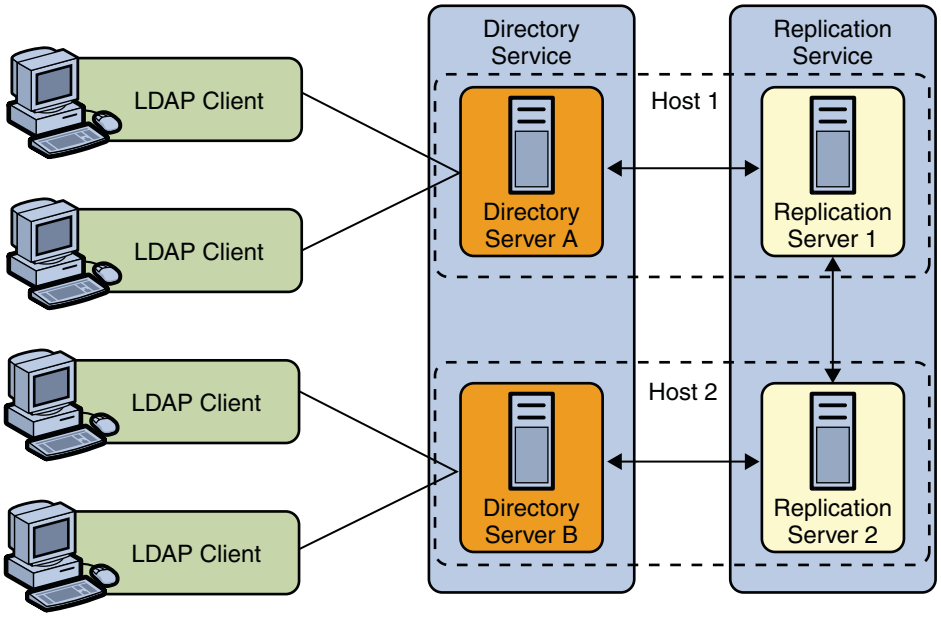
By replicating directory data across servers, you can reduce the access load on a single machine, improving server response time and providing horizontal read scalability. In addition, replication can be used to ensure availability of data in the event of machine failure.

Note that you cannot use replication to scale write operations because a write operation to one directory server results in a write operation to every other server in the topology. The only way to scale write operations horizontally is to split the directory data among multiple databases and place those databases on different servers.

The centralized replication model in Sun OpenDS Standard Edition separates user data from replication metadata. In this model, the server that stores the user data is called the directory server. The server that stores the replication metadata is called the replication server. This approach simplifies the management of replication topologies and can improve performance.

For small deployments, you can set up replication by putting the replication servers and directory servers on the same system. You can further simplify administration by running the replication server and the directory server on each system in a single process.

The following diagram shows how replication is used to ensure availability and to provide read scalability in a small topology.



The Role of Directory Servers in a Topology

Directory servers are responsible for the following tasks:

- Persistence of data and serving client requests
- Forwarding changes to specific replication servers

When a change is made on a directory server, that server forwards the change to a selected replication server. The replication server then replays the change to other replication servers in the topology, which in turn replay the change to all other directory servers in the topology.

Each directory server contains the following items:

- A list of the suffix DNs to be synchronized
- For each suffix DN, a list of replication servers to connect to

Applications should typically perform reads and writes on the same directory server instance. This prevents those applications from experiencing consistency problems due to loose consistency.

The Role of Replication Servers in a Topology

Replication servers are responsible for the following tasks:

- Managing connections from directory servers
- Connecting to other replication servers
- Listening for connections from other replication servers
- Receiving changes from directory servers
- Forwarding changes to directory servers and to other replication servers
- Saving changes to stable storage, which includes trimming older operations

Each replication server contains a list of all the other replication servers in the replication topology. Replication servers are also responsible for providing other servers with information about the replication topology. Even the smallest deployment must include two replication server instances, to ensure availability in case one of the replication server instances fails. There is usually no need for additional replication server instances unless the directory service must be able to survive more than one failure at a time, or unless the number of directory server instances must be very large.

Although replication servers do not store directory data, they are always LDAP servers or JMX servers. Like directory servers, replication servers can be configured, monitored, backed up and restored.

Multiple Data Center Topology

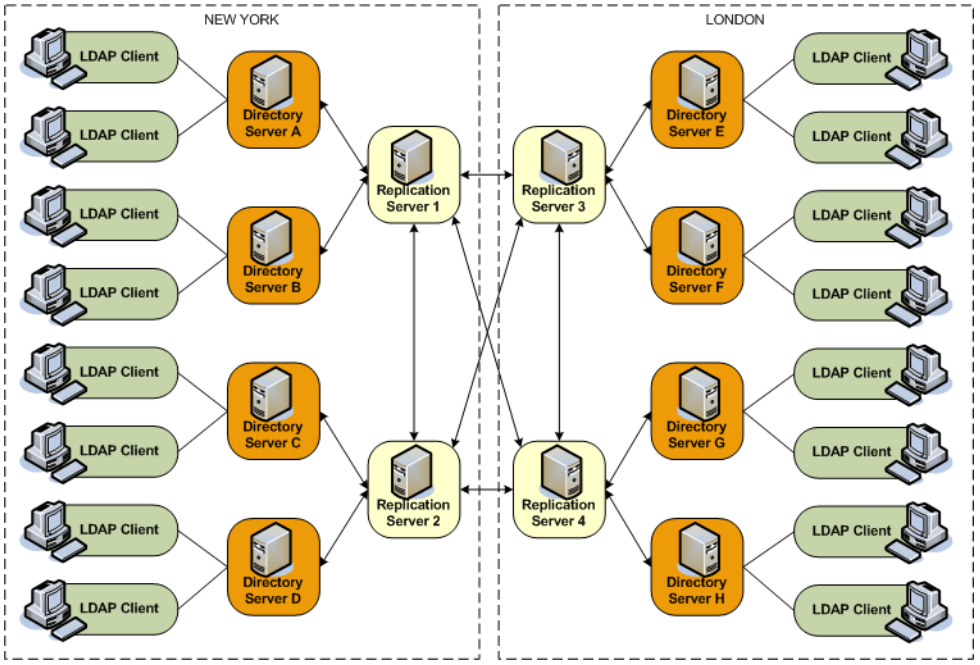
Replication enables geographic distribution of the directory service by providing identical copies of directory data on multiple servers across more than one data center. The basic principles of a replication deployment outlined in the small topology also apply to multiple data center deployments.

The Sun OpenDS Standard Edition directory server uses a custom replication protocol that is efficient over a wide area network (WAN). In the following scenario, an enterprise has two major data centers, one in London and the other in New York, separated by a WAN.

This deployment includes two replication server instances for availability in each data center, in case one of the replication server instances fails. The directory servers connect first to local replication servers. Directory servers only access replication servers in another data center if all local replication servers have failed. Client applications always connect to local directory server instances, and perform reads and writes on the same directory server instance.

The Sun OpenDS Standard Edition directory server supports an unlimited number of read/write directory servers in a replication topology. The number of directory servers can be scaled according to the read requirements of the organization. Note that increasing the number of directory servers does not scale the number of writes that can be processed because

ultimately all servers in the toplogy must process all the writes. Unless it is acceptable to have a topology that does not converge, the write throughput of the topology is limited to the write throughput of the slowest machine.



Multiple Data Centers and Replication Groups

Replication groups enable you to organize a replicated topology according to specific criteria, such as data center location. A replication group is identified by a unique ID that is applied to the replication servers and the directory servers in that group. Group IDs determine how a directory server domain connects to an available replication server. From the list of configured replication servers, a directory server first tries to connect to a replication server that has the same group ID as that of the directory server.

This sample deployment shows the use of replication groups across multiple data centers. The deployment assumes two data centers, connected by a wide area network (WAN), with the following configuration:

- Each replication server and directory server within a single data center has the same group ID.
- There is a unique group ID for the entire data center (one group ID per data center).

The following figure shows a disaster recovery deployment that includes two data centers with different group IDs.

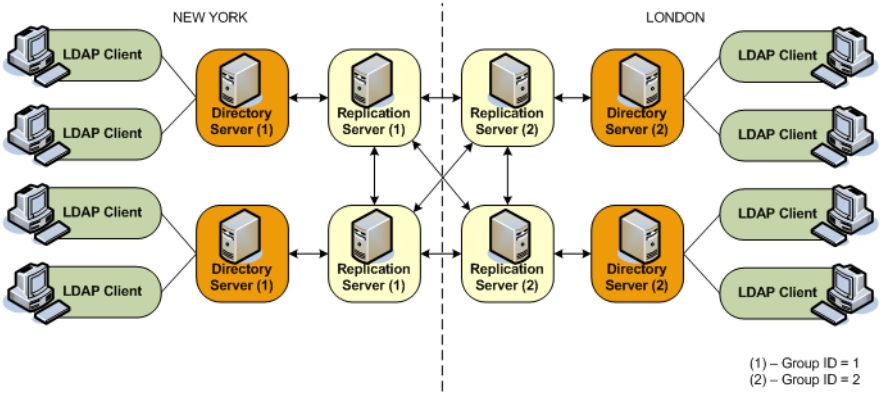


FIGURE 5–1 Replication Groups Over WAN

In this deployment, each directory server will attempt to connect to a replication server in its own data center, avoiding the latency associated with connection over a WAN. If all the replication servers in a data center fail, the directory server will connect to a remote replication server. This ensures that the replication service is maintained, albeit in a degraded manner (if the connection between data centers is slow). When one or more local replication servers is back online, the directory servers will automatically reconnect to a local replication server.

Multiple Data Centers and the Window Mechanism

The Sun OpenDS Standard Edition directory server provides a window mechanism which specifies that a certain number of update requests are sent without one server having to wait for an acknowledgement from the recipient server before continuing.

The window size represents the maximum number of update messages that can be sent without immediate acknowledgement from the recipient server. If the topology spans multiple data centers connected by a network with large latency, it might be worth increasing the window size beyond its default value of 100. To assess whether the window size is the limiting factor in replication thoughput, monitor the `current-send-window` and `current-rcv-window` attributes below `cn=monitor`.

If a server publishes a `current-send-window` to another server that is consistently zero or close to zero and the corresponding server publishes a `current-rcv-window` that is higher, it means that all the data are currently in the network. In this case, increasing the window size on the recipient server should increase replication speed and reduce replication delay. These improvements will result in the consumption of more resources on the recipient server.

◆ ◆ ◆ CHAPTER 6

Example Deployments Using the Proxy Server

There are many types of deployment in which the Sun OpenDS Standard Edition proxy can be used successfully. The following are suggested deployments, which will help familiarize you with how the proxy works.

Deciding Your Proxy Deployment Type

There are two main types of deployment with the Sun OpenDS Standard Edition proxy:

- Using load balancing
- Using distribution

In order to decide which type of deployment you want, consider this: where and how is your data stored? how much data do you handle?

- If all your data is stored on a replicated data store, then use a deployment with load balancing. See "Configuration 1: Simple Load Balancing" on page 40.
- If your data is partitioned or you have a large database and want to split your data so that it is partitioned on different data sources, then use a deployment with distribution. See "Configuration 2: Simple Distribution" on page 42.

More complex deployment scenarios can be defined, which layer load balancing and distribution. The main question will be, do you need load balancing, or distribution, or both?

Other than simple load balancing and simple distribution, the following example deployments will be presented:

- If you want to deploy data centers in different geographical locations, for example, you could deploy failover between two load-balanced data centers. See "Configuration 3: Failover Between Data Centers" on page 43.
- If you want to use distribution but want the data partitions to be replicated, then you can deploy Sun OpenDS Standard Edition proxy using distribution which routes to load balancer. See "Configuration 4: Distribution with Load Balancing" on page 45.
- If you want to use distribution with the data partitions replicated, but for availability and disaster recover you want the partitions to not only be replicated in one data center but also want to replicate the data centers in two different geographical locations, then you could deploy an architecture similar to "Configuration 5: Distribution with Failover Between Data Centers" on page 46.

You can add a *global index catalog* to deployments using distribution, to map entries to a specific partition. This will help minimize the use of broadcasts. For information on configuring a global index catalog, see "Configuring the Global Index" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

Configuration 1: Simple Load Balancing

When deploying the Sun OpenDS Standard Edition proxy using load balancing, all requests received through the Sun OpenDS Standard Edition proxy are routed to one of the remote LDAP servers. As illustrated in Figure 6–1, the remote LDAP servers are replicated and contain the same data. The number of supported remote LDAP servers is not limited.

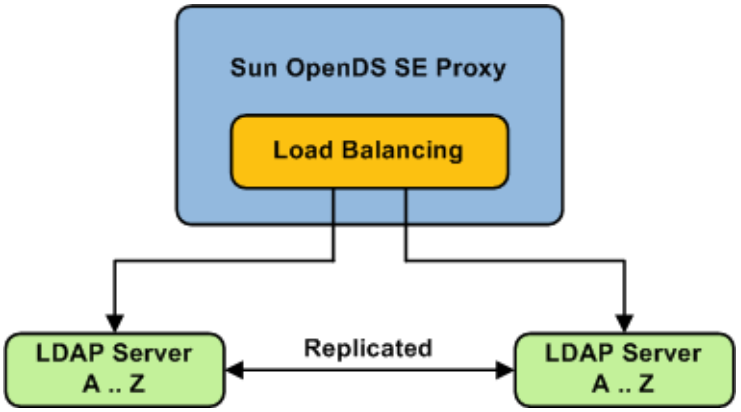


FIGURE 6–1 Simple Load Balancing

The requests are routed to one of the remote LDAP servers based on the load balancing algorithm set during deployment.

The load balancing algorithms are:

- Proportional
- Failover
- Saturation

For more information on the different load balancing algorithms, see "Load Balancing Using the Proxy" on page 20.

The algorithm can be bypassed by a client connection affinity. If you set client connection affinity, the Sun OpenDS Standard Edition proxy will use the load balancing algorithm for the first request, but for the following request will disregard the load balancing algorithm set and will try to reuse the same route for a new operation on the same client connection, for example, depending on the type of client affinity set. For more information, see "Setting Client Connection Affinity" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

The advantages of using load balancing deployment are the high availability of the data, as well as an adapted workload on the remote LDAP servers. For example, if one of the remote LDAP servers in your configuration becomes unavailable, the load balancing will route the request to another remote LDAP server. In this case, the failure is not visible to the client and there is no service disruption.

A simple load balancing deployment can be configured easily during the Sun OpenDS Standard Edition proxy installation, using the `vdp-setup` tool.

Configuration 2: Simple Distribution

When deploying the Sun OpenDS Standard Edition proxy using simple distribution, the data is split into partitions. Each partition of data is held on a separate remote LDAP server, as illustrated in Figure 6–2. All requests received through the Sun OpenDS Standard Edition proxy are routed to the remote LDAP server which contains the appropriate data.

The number of remote LDAP servers onto which the data is partitioned depends on the size of the database that you are splitting. The following example shows two partitions, but you can configure more.

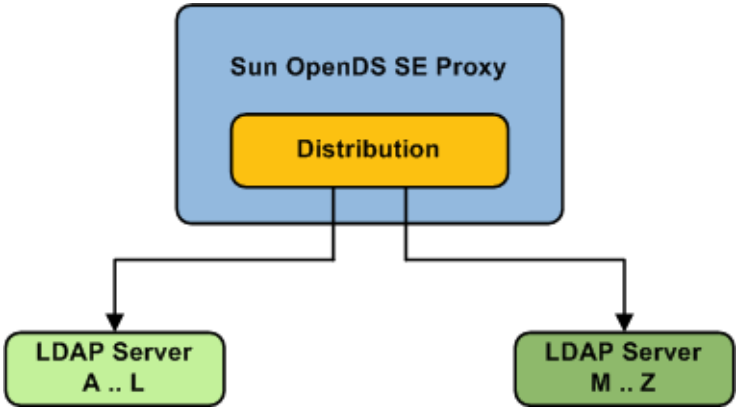


FIGURE 6–2 Simple Distribution

The requests are routed to one of the remote LDAP servers based on the distribution algorithm set during deployment.

The distribution algorithms are:

- Numeric
- Lexico
- DN pattern

For more information on the different distribution algorithms, see "Data Distribution Using the Proxy" on page 24.

The advantages of a deployment using distribution is that it allows you to scale the number of updates per second. In order to diminish the number of broadcasts when using distribution, you can add a global index catalog with the Sun OpenDS Standard Edition proxy. For more information on the global index catalog, see "Configuring the Global Index" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

A simple distribution deployment can be easily configured during the Sun OpenDS Standard Edition proxy installation, using the `vdp-setup` GUI.

Configuration 3: Failover Between Data Centers

When deploying failover between data centers, you are essentially deploying two levels of load balancers within the Sun OpenDS Standard Edition proxy. In this deployment, the data centers are replicated and the remote LDAP servers within the data centers are also replicated. The first load balancing element of the deployment can be either failover or saturation. The example assumes failover algorithm is selected for the initial load balancing element.

As illustrated in Figure 6–3, all of the requests are routed by the failover load balancer through the main route, to a second load balancing element, which sends the request to a server within Data Center 1. If Data Center 1 goes down or is degraded, then the traffic is routed by the failover load balancer to the backup route, to a server in Data Center 2.

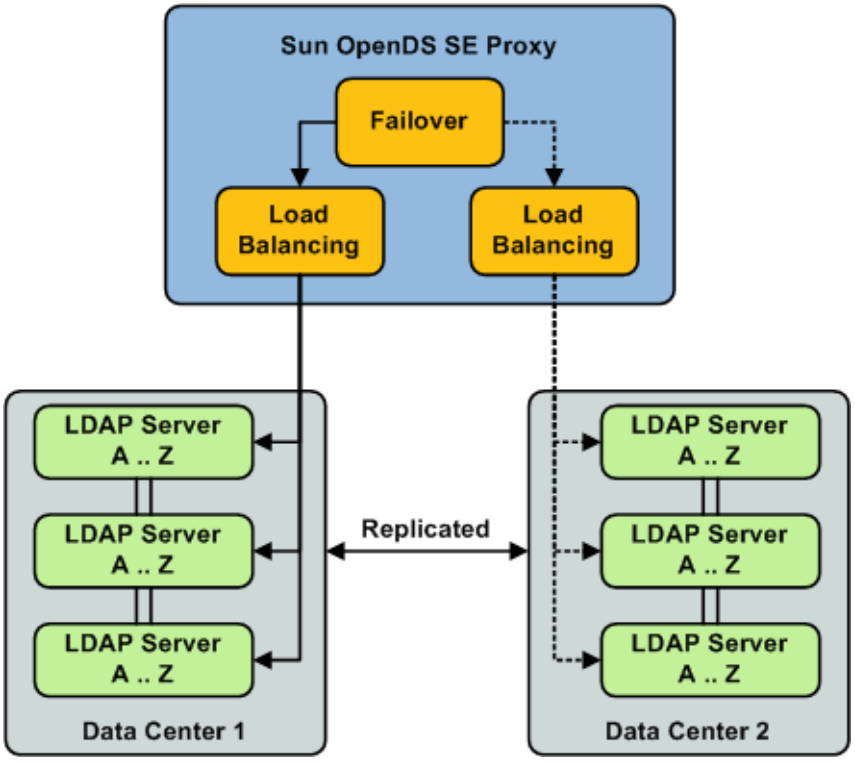


FIGURE 6–3 Failover Between Data Centers

The requests are routed to the remote LDAP servers within the data centers based on the load balancing algorithm set. The load balancing algorithm can be different for each data center. For example, you can set the load balancing in Data Center 1 as `proportional`, while the load balancing algorithm in Data Center 2 is set as `saturation`.

This type of deployment is typically used when deploying in two geographical areas. This adds high availability of data to a simple load balancing deployment, since not only are the remote LDAP servers replicated, but the data centers are also replicated.

Typically, you would have the two data centers in two different geographical locations. This way, if there was a problem in one location, the data center in the other location would act as backup. Another example would be setting the first load balancer to `saturation`. This way, if Data Center 1 in one geographical location (for example in one time-zone) becomes saturated, the other data center can pick up the excess traffic.

For more information on the different load balancing algorithms, see "Load Balancing Using the Proxy" on page 20.

For details on deploying this configuration, see "Configuring Failover Between Data Centers" on page 65.

Configuration 4: Distribution with Load Balancing

In a deployment of Sun OpenDS Standard Edition proxy using distribution deployment with load balancing, the data is split into partitions, and the data is replicated on the remote LDAP servers. Requests sent to the Sun OpenDS Standard Edition proxy are first distributed to the partition in which the data is stored, then the request is routed to one of the remote LDAP servers, depending on the load balancing algorithm set. The remote LDAP servers holding the partitioned data are replicated.

As illustrated in Figure 6–4, when the Sun OpenDS Standard Edition proxy receives a requests, it is filtered by the distribution to the correct partition. For example, a request for data `Garry` would be forwarded to partition 1, to the servers with data from `A..L`. The load balancer then forwards the request to one of the replicated remote LDAP servers.

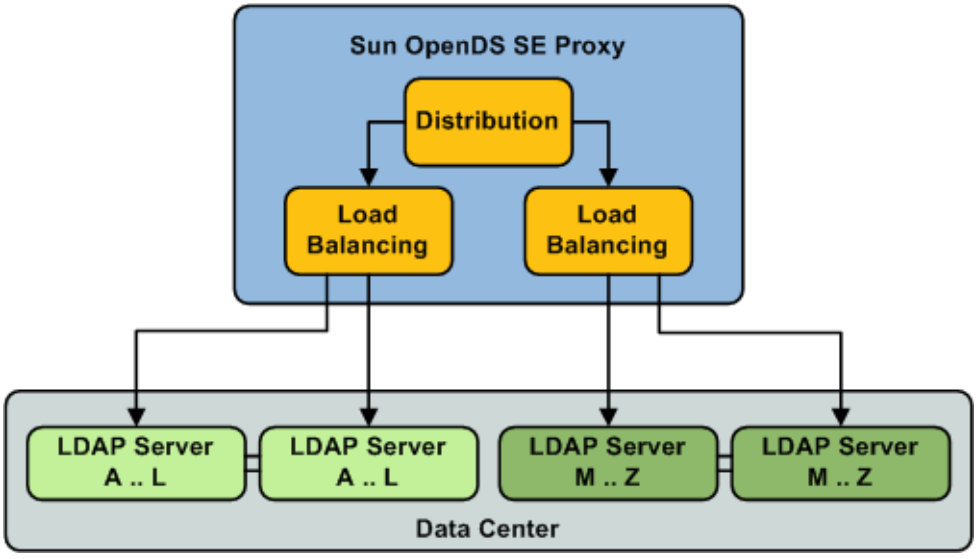


FIGURE 6–4 Distribution with Load Balancing

The requests are routed to the remote LDAP servers within the data centers based on the load balancing algorithm set. For more information on the different load balancing algorithms, see "Load Balancing Using the Proxy" on page 20.

The advantages of this deployment are the speed of the updates, because of the distribution of data, and high availability of the data.

For more information on the different distribution algorithms, see "Data Distribution Using the Proxy" on page 24.

For more information on the different load balancing algorithms, see "Load Balancing Using the Proxy" on page 20.

For details on deploying this configuration, see "Configuring Distribution and Load Balancing" on page 59.

Configuration 5: Distribution with Failover Between Data Centers

In a deployment of Sun OpenDS Standard Edition proxy using distribution with failover load balancing between two data centers, the data is split into partitions, where each partition is managed through a failover load balancing route. As illustrated in Figure 6–5, not only are the remote LDAP servers holding the partitioned data replicated within the data center, but in addition, the data centers are replicated, with one of the two acting as the backup.

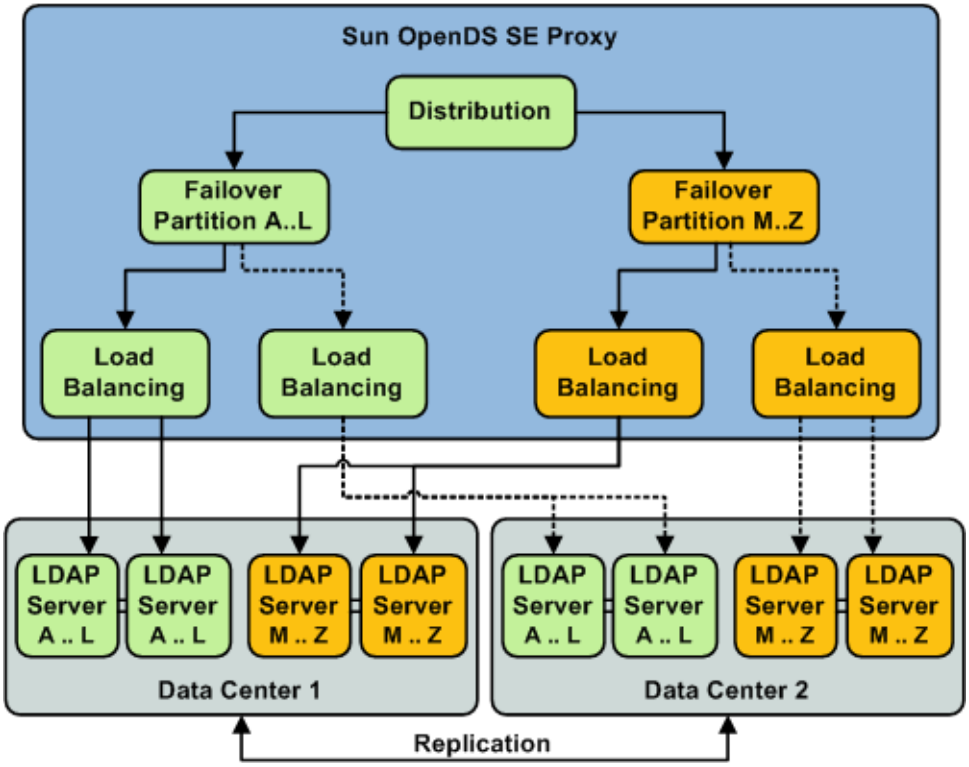


FIGURE 6–5 Distribution with Failover Between Data Centers

In other words, requests sent to the Sun OpenDS Standard Edition proxy deployed with this configuration are first distributed to the partition in which the data is stored. For example, a request for data `Garry` would be forwarded to partition 1. The failover load balancer then forwards the request through the main route, depending on the load balancing algorithm set, to one of the one of the remote LDAP servers holding the data for `A..L`.

In the deployment illustrated in Figure 6–5, Data Center 2 acts as a backup, and is only used on failure of the first data center. However, this same deployment could be configured to use `saturation`, rather than a failover load balancer. This way, if Data Center 1 in one geographical location (for example in one time-zone) becomes saturated, the other data center can pick up the excess traffic.

The advantages of this deployment are the speed of the reads through the distribution algorithm, and the high availability offered since the remote LDAP servers are replicated, and one data center acts as a backup.

For more information on the different distribution algorithms, see "Data Distribution Using the Proxy" on page 24.

For more information on the different load balancing algorithms, see "Load Balancing Using the Proxy" on page 20.

For details on deploying this configuration, see "Configuring Distribution with Failover Between Data Centers" on page 69.

Multiple Replicated Proxies

To reduce the exposure of your Sun OpenDS Standard Edition proxy configuration to Single Point of Failure, you should ensure that your deployment is redundant. Typically, this can be done by installing a third party hardware load balancer, as illustrated in Figure 6–6.

Using a hardware load balancer, you can manage multiple instances of the Sun OpenDS Standard Edition proxy on separate physical machines and/or in different geographical locations.

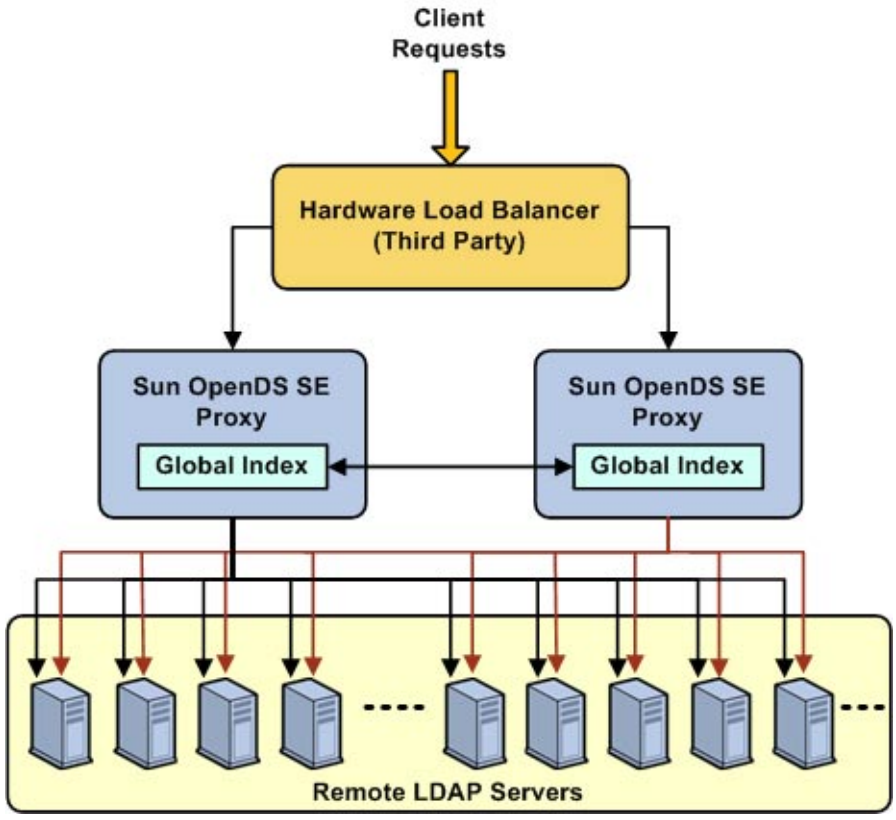


FIGURE 6–6 Multiple Instances of Sun OpenDS Standard Edition proxy

When running multiple instances of the Sun OpenDS Standard Edition proxy, if you are using a distribution deployment with a global index catalog, the global index catalog should be replicated. For more information on replicating the global index catalog, see "Replication of Global Index Catalogs" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

In order to deploy the same Sun OpenDS Standard Edition proxy deployment see the instructions in "Duplicating Your Proxy Installation" in *Sun OpenDS Standard Edition 2.2 Installation Guide*.

◆ ◆ ◆ CHAPTER 7

Simple Proxy Deployments Using the Command Line Interface

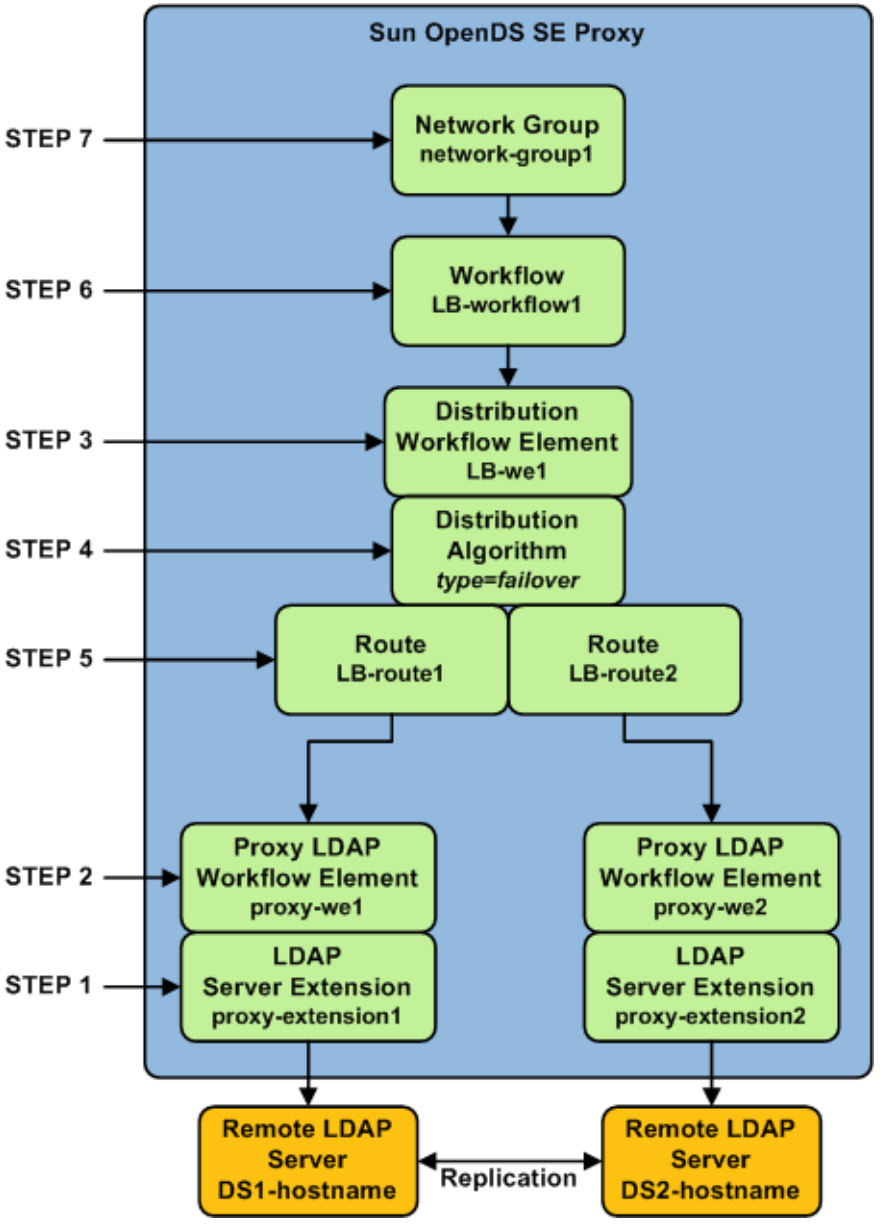
The following examples give the non-interactive commands, using the `dsconfig` command.

You can also complete the configuration in interactive mode. For information about using `dsconfig`in interactive mode, see "Using dsconfig in Interactive Mode" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

Configuring Load Balancing With the Command Line Interface

The following is a step by step procedure that defines all the different elements needed to set up a deployment using simple load balancing. The following example describes load balancing with failover on two LDAP servers. For more information on the different types of load balancing available, see "Load Balancing Using the Proxy" on page 20.

The following figure illustrates all the objects that need to be created to configure a Sun OpenDS Standard Edition proxy using a simple load balancing distribution. The objects must be created in the order indicated.



All the commands in this procedure specify the proxy hostname (`-h`), the proxy admin port (`-p`), the bind DN for the initial root user (`-D`) and the proxy password you want to configure (`-w`). You must also indicate the authentication; if none is indicated and the client and the server are running in the same instance, the local authentication configuration is used.

▼ To Configure Simple Load Balancing

1 Create a proxy LDAP server extension:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-extension \
--extension-name proxy_extension1 \
--type ldap-server \
--set enabled:true \
--set remote-ldap-server-address:DS1_hostname \
--set remote-ldap-server-port:2389
```

The LDAP server extension is a link to the back-end LDAP server. For this use case, you will need at least two back-end LDAP server instances. Go through this step again, making sure to use a different LDAP hostname and port.

2 Create a proxy workflow element for each LDAP server extension:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-workflow-element \
--element-name proxy-we1 \
--type proxy-ldap\
--set enabled:true \
--set client-cred-mode:use-client-identity \
--set ldap-server-extension:proxy_extension1
```

The property `client-cred-mode` indicates the type of authentication used between the proxy and remote LDAP server. The client credential mode can be: `use-client-identity`, `use-specific-identity`, or `use-proxy-auth`.

3 Create a load balancing workflow element:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-workflow-element \
--element-name LB-we1 \
--type load-balancing \
--set enabled:true
```

You only need one load balancing workflow element to route requests to either of the two back-end LDAP servers.

4 Define the load balancing algorithm:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-load-balancing-algorithm \
--element-name LB-we1 \
--type failover
```

The type of load balancing algorithm can be `proportional`, `saturation` or `failover`. The properties of the load balancing algorithm (weight, threshold, or priority) are defined with the load balancing routes, in the next step.

5 **Define the load balancing routes for each proxy:**

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-load-balancing-route \
--element-name LB-we1 \
--route-name LB-route1 \
--type failover \
--set workflow-element:proxy-we1 \
--set priority:1
```

Make sure that you specify the same type when defining the routes as you did when defining the load balancing algorithm.

For this use case, you will need two load balancing routes. Go through this step again, specifying a different priority for each route.

The properties in the example above set the priority for failover load balancing. If you use proportional or saturation load balancing, the properties will differ. For more information on the setting different load balancing types, see "Modifying the Load Balancing Route Properties" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

6 **Create a workflow:**

This workflow associates the load balancing workflow element with the specified base dn.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-workflow \
--workflow-name LB-workflow1 \
--set enabled:true \
--set base-dn:dc=example,dc=com \
--set workflow-element:LB-we1
```

7 **Create the network group:**

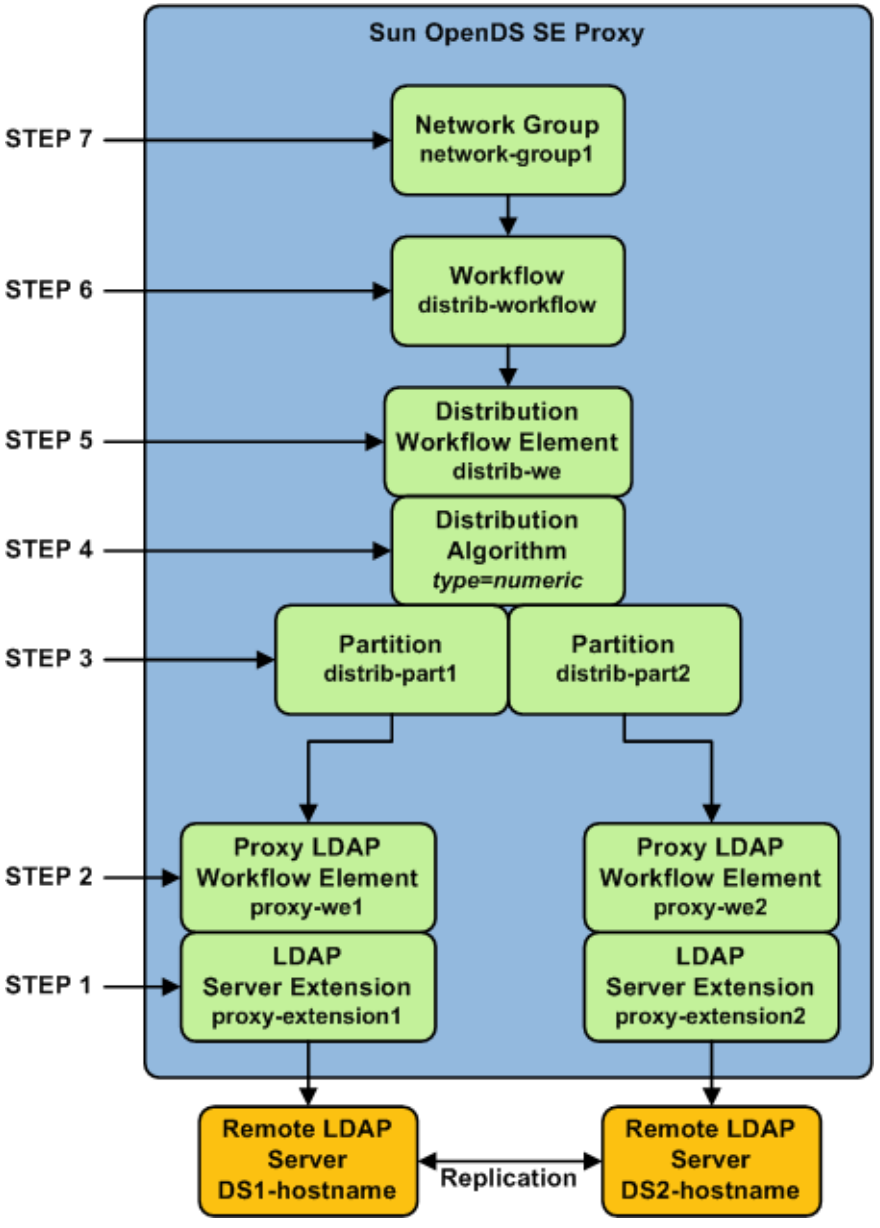
The network group handles all the requests between the client and the proxy.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-network-group \
--group-name network-group1 \
--set enabled:true \
--set workflow:LB-workflow1 \
--set priority:1
```

Configuring Distribution With the Command Line Interface

The following is a step by step procedure that defines all the different elements needed to set up a deployment using simple distribution. The following example shows a distribution split on two partitions. For more information on the different types of distribution available, see "Data Distribution Using the Proxy" on page 24.

The following figure illustrates all the objects that need to be created to configure a Sun OpenDS Standard Edition proxy using a simple distribution deployment. The objects must be created in the order indicated.



All the commands in this procedure specify the proxy hostname (`-h`), the proxy admin port (`-p`), the bind DN for the initial root user (`-D`) and the proxy password you want to configure (`-w`). You must also indicate the authentication; if none is indicated and the client and the server are running in the same instance, the local authentication configuration is used.

▼ To Configure Simple Distribution

1 Create a proxy LDAP server extension:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-extension \
--extension-name proxy_extension1 \
--type ldap-server \
--set enabled:true \
--set remote-ldap-server-address:DS1_hostname \
--set remote-ldap-server-port:2389
```

The LDAP server extension is a link to the back-end LDAP server. For this use case, you will need two back-end LDAP server instances. Go through this step again, making sure to use a different LDAP hostname and port.

2 Create a proxy workflow element for each LDAP server extension:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-workflow-element \
--element-name proxy-we1 \
--type proxy-ldap\
--set enabled:true \
--set client-cred-mode:use-client-identity \
--set ldap-server-extension:proxy_extension1
```

You will need at least two remote LDAP servers for a distribution architecture. Go through this step again. The LDAP server extension name should be the same as those created in step 1.

The property `client-cred-mode` indicates the type of authentication used between the proxy and back-end LDAP server. The client credential mode can be: `use-client-identity`, `use-specific-identity`, or `use-proxy-auth`.

3 Set up distribution by creating a distribution workflow element:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-workflow-element \
--element-name distrib-we \
--type distribution \
--set base-dn:dc=example,dc=com \
--set enabled:true
```

4 Set the distribution algorithm:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-distribution-algorithm \
--element-name distrib-we \
--type numeric \
--set distribution-attribute:uid
```

The type of distribution algorithm can be numeric, lexicographic, or by DN pattern match. The properties of the algorithm are defined when you create the distribution partitions, in the next step.

5 Define the distribution partitions:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-distribution-partition \
--element-name distrib-we \
--partition-name distrib-part1\
--type numeric \
--set lower-bound:0 \
--set upper-bound:1000 \
--set partition-id:1 \
--set workflow-element:proxy-we1
```

For this use case, you will need to create two partitions. Make sure that the partition ID and the partition name are unique for each workflow element. You must specify the same type when defining the partitions as you did when defining the load balancing algorithm.

Note – The upper boundary indicated is exclusive. This means that if you indicate 1000 as the upper boundary, the partition will only include values from 0 to 999, inclusive.

---

6 Create a workflow:

This workflow associates the distribution workflow element with the distribution partition.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-workflow \
--workflow-name distrib-workflow \
--set enabled:true \
--set base-dn:dc=example,dc=com \
--set workflow-element:distrib-we
```

7 Create the network group:

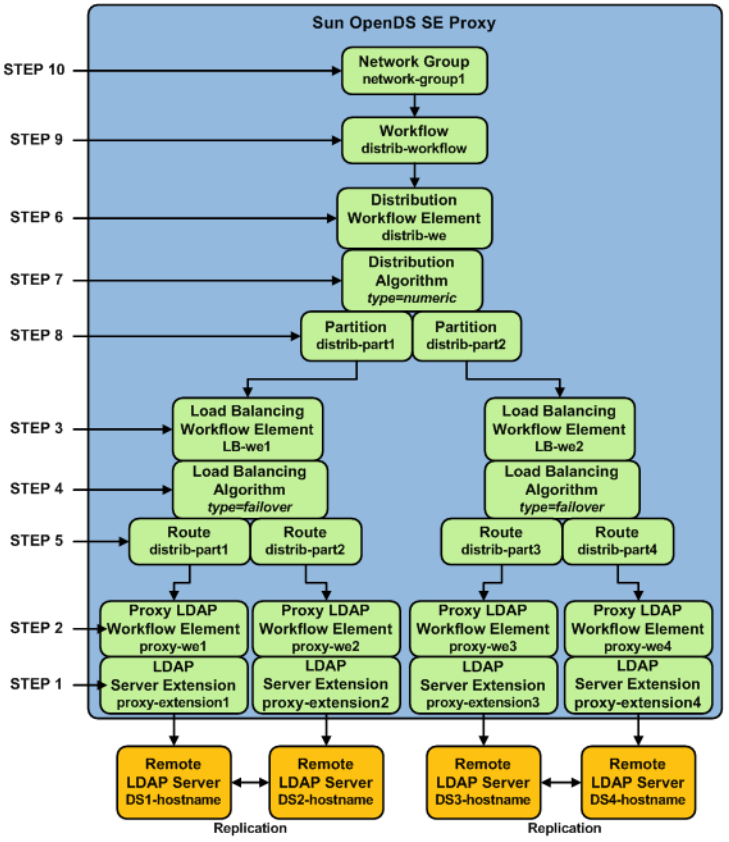
The network group handles all the requests between the client and the proxy.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-network-group \
--group-name network-group1 \
--set enabled:true \
--set workflow:distrib-workflow \
--set priority:1
```

Configuring Distribution and Load Balancing

This use case combines distribution with load balancing. As for all distribution deployments, you can add a global index, however, this is not included here. For information on creating a global index, see "Configuring the Global Index" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

The following figure illustrates all the objects that need to be created to deploy a Sun OpenDS Standard Edition proxy using distribution with load balancing. The objects must be created in the order indicated.



The following example presents a deployment with distribution over two partitions, with each partition load balanced onto two replicated LDAP servers. The distribution algorithm used to partition the data is numeric.

All the commands in this procedure specify the proxy hostname (`-h`), the proxy admin port (`-p`), the bind DN for the initial root user (`-D`) and the proxy password you want to configure (`-w`). You must also indicate the authentication; if none is indicated and the client and the server are running in the same instance, the local authentication configuration is used.

▼ To Configure Distribution with Load Balancing

1 Create the proxy LDAP server extensions:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-extension \
--extension-name proxy_extension1 \
--type ldap-server \
--set enabled:true \
--set remote-ldap-server-address:DS1_hostname \
--set remote-ldap-server-port:2389
```

The LDAP server extension is a link to the back-end LDAP server. For this use case, you will need four back-end LDAP server instances. Go through this step once for each back-end LDAP server, making sure to use a different LDAP hostname and port.

2 Create a proxy workflow element for each LDAP server extension:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-workflow-element \
--element-name proxy-we1 \
--type proxy-ldap\
--set enabled:true \
--set client-cred-mode:use-client-identity \
--set ldap-server-extension:proxy_extension1
```

For this use case, you will need four back-end LDAP server instances. Go through this step once for each back-end. The LDAP server extension name should be the same as those created in step 1.

The property `client-cred-mode` indicates the type of authentication used between the proxy and back-end LDAP server. The client credential mode can be: `use-client-identity`, `use-specific-identity`, or `use-proxy-auth`.

3 Create a load balancing workflow element:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-workflow-element \
--element-name LB-we1 \
--type load-balancing \
--set enabled:true
```

You only need one load balancing workflow element to route requests to either of the two back-end LDAP servers. In this use case, since you are using two load balancers, you will need to create two load balancing workflow elements.

4 Define the load balancing algorithm:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-load-balancing-algorithm \
--element-name LB-we1 \
--type failover
```

The type of load balancing algorithm can be proportional, saturation or failover. The properties of the load balancing algorithm (weight, threshold, or priority) are defined with the load balancing routes, in the next step. For this use case, you will need two load balancing algorithms.

5 Define the load balancing routes for each proxy:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-load-balancing-route \
--element-name LB-we1 \
--route-name LB-route1 \
--type failover \
--set workflow-element:proxy-we1 \
--set priority:1
```

For this use case, you will need four load balancing routes. Set two routes per load balancing workflow element (created in the previous step); for example, one route with priority 1 and the other route with priority 2.

Note – The properties in the example above set the priority for failover load balancing. If you use proportional or saturation load balancing, the properties will differ. For more information on the setting different load balancing types, see "Modifying the Load Balancing Route Properties" in *Sun OpenDS Standard Edition 2.2 Administration Guide*.

---

6 Set up distribution by creating a distribution workflow element:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-workflow-element \
--element-name distrib-we \
--type distribution \
--set base-dn:dc=example,dc=com \
--set enabled:true
```

For this use case, you will need only one distribution workflow element, which will point to the distribution algorithm.

7 Set the distribution algorithm:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-distribution-algorithm \
--element-name distrib-we \
--type numeric \
--set distribution-attribute:uid
```

The type of distribution algorithm can be numeric, lexicographic, or by DN pattern match. The boundaries are defined when you create the distribution partitions, in the next step.

8 Define the distribution partitions:

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-distribution-partition \
--element-name distrib-we \
--partition-name distrib-part1\
--type numeric \
--set lower-bound:0 \
--set upper-bound:1000 \
--set partition-id:1 \
--set workflow-element:LB-we1
```

For this use case, you will need to create two partitions. Make sure that the partition ID and the partition name are unique for each workflow element, and that each partition uses a different load balancing workflow element. You must specify the same type when defining the routes as you did when defining the load balancing algorithm.

Note – The upper boundary indicated is exclusive. This means that if you indicate 1000 as the upper boundary, the partition will only include values from 0 to 999, inclusive.

---

9 Create a workflow:

This workflow associates the distribution workflow element with the base DN.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-workflow \
--workflow-name workflow \
--set enabled:true \
--set base-dn:dc=example,dc=com \
--set workflow-element:distrib-we
```

10 Create the network group:

The network group handles all the requests between the client and the proxy.

```
$ dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password \
create-network-group \
--group-name network-group1 \
--set enabled:true \
--set workflow:workflow \
--set priority:1
```

◆ ◆ ◆ CHAPTER 8

Deploying Advanced Proxy Architectures

The following section assumes that you have a good knowledge of the concepts of Sun OpenDS Standard Edition proxy and a clear understanding of the architecture you want to deploy. The following examples can be modified as needed to add as many data sources as needed in your deployment.

Moreover, it is strongly recommended that your Sun OpenDS Standard Edition proxy be redundant. For more information, see the "Duplicating Your Proxy Installation" in *Sun OpenDS Standard Edition 2.2 Installation Guide*.

Configuring Failover Between Data Centers

Use the following script to set up a failover deployment between two data centers, as presented in "Configuration 3: Failover Between Data Centers" on page 43.

```
#Create a proxy LDAP extension for each back-end LDAP server
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension1 \
  --set enabled:true \
  --set remote-ldap-server-address:DS1_hostname \
  --set remote-ldap-server-port:3189

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension2 \
  --set enabled:true \
  --set remote-ldap-server-address:DS2_hostname \
  --set remote-ldap-server-port:3289
```

```
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension3 \
  --set enabled:true \
  --set remote-ldap-server-address:DS3_hostname \
  --set remote-ldap-server-port:3389

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension4 \
  --set enabled:true \
  --set remote-ldap-server-address:DS4_hostname \
  --set remote-ldap-server-port:3489

#Create a proxy workflow element for each LDAP server extension
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name proxy-we1 \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension1

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name proxy-we2 \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension2

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name proxy-we3 \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension3

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name proxy-we4 \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension4
```

```
# Create a load balancing workflow element for each data center
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name LB-we1 \
  --type load-balancing \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name LB-we2 \
  --type load-balancing \
  --set enabled:true

# Define the load balancing algorithm for each data center
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-algorithm \
  --element-name LB-we1 \
  --type proportional

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-algorithm \
  --element-name LB-we2 \
  --type proportional

# Define the load balancing routes for each proxy
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name LB-we1 \
  --route-name LB-route1 \
  --type proportional \
  --set workflow-element:proxy-we1

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name LB-we1 \
  --route-name LB-route2 \
  --type proportional \
  --set workflow-element:proxy-we2

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name LB-we2 \
  --route-name LB-route3 \
  --type proportional \
  --set workflow-element:proxy-we3

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
```

```
create-load-balancing-route \
  --element-name LB-we2 \
  --route-name LB-route4 \
  --type proportional \
  --set workflow-element:proxy-we4

# Set failover between the two data centers
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name FO-we \
  --type load-balancing \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-algorithm \
  --element-name FO-we \
  --type failover

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name FO-we \
  --route-name FO-route1 \
  --type failover \
  --set workflow-element:LB-we1 \
  --set priority:1

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name FO-we \
  --route-name FO-route2 \
  --type failover \
  --set workflow-element:LB-we2 \
  --set priority:2

# Create workflow
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow \
  --workflow-name FO-workflow \
  --set enabled:true \
  --set base-dn:dc=example,dc=com \
  --set workflow-element:FO-we

# Create network group
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-network-group \
  --group-name network-group1 \
  --set enabled:true \
  --set workflow:FO-workflow \
  --set priority:1
```

Configuring Distribution with Failover Between Data Centers

Use the following script to set up a failover deployment between two data centers, as presented in "Configuration 5: Distribution with Failover Between Data Centers" on page 46.

```
#Create the first failover route
#Create a proxy LDAP extension for each back-end LDAP server
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-1a \
  --set enabled:true \
  --set remote-ldap-server-address:DS1a_hostname \
  --set remote-ldap-server-port:3189

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-2a \
  --set enabled:true \
  --set remote-ldap-server-address:DS2a_hostname \
  --set remote-ldap-server-port:3289

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-1b \
  --set enabled:true \
  --set remote-ldap-server-address:DS1b_hostname \
  --set remote-ldap-server-port:3389

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-2b \
  --set enabled:true \
  --set remote-ldap-server-address:DS2b_hostname \
  --set remote-ldap-server-port:3489

#Create a proxy workflow element for each LDAP server extension
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name proxy-we-1a \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-1a
```

```
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name proxy-we-2a \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-2a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name proxy-we-1b \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-1b

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name proxy-we-2b \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-2b

# Create a load balancing workflow element for each data center
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name LB-we-1a \
  --type load-balancing \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name LB-we-1b \
  --type load-balancing \
  --set enabled:true

# Define the load balancing algorithm for each data center
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-algorithm \
  --element-name LB-we-1a \
  --type proportional

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-algorithm \
  --element-name LB-we-1b \
  --type proportional
```

```
# Define the load balancing routes for each proxy
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name LB-we-1a \
  --route-name LB-route-1a \
  --type proportional \
  --set workflow-element:proxy-we-1a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name LB-we-1a \
  --route-name LB-route-2a \
  --type proportional \
  --set workflow-element:proxy-we-2a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name LB-we-1b \
  --route-name LB-route-1b \
  --type proportional \
  --set workflow-element:proxy-we-1b

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name LB-we-1b \
  --route-name LB-route-2b \
  --type proportional \
  --set workflow-element:proxy-we-2b

# Set failover between the two data centers
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name FO-we1 \
  --type load-balancing \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-algorithm \
  --element-name FO-we1 \
  --type failover

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name FO-we1 \
  --route-name FO-route-1a \
  --type failover \
  --set workflow-element:LB-we-1a \
  --set priority:1
```

```
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name FO-we1 \
  --route-name FO-route-1b \
  --type failover \
  --set workflow-element:LB-we-1b \
  --set priority:2

#Create the second failover route
#Create a proxy LDAP extension for each back-end LDAP server
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-3a \
  --set enabled:true \
  --set remote-ldap-server-address:DS3a_hostname \
  --set remote-ldap-server-port:3189

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-4a \
  --set enabled:true \
  --set remote-ldap-server-address:DS4a_hostname \
  --set remote-ldap-server-port:3289

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-3b \
  --set enabled:true \
  --set remote-ldap-server-address:DS3b_hostname \
  --set remote-ldap-server-port:3389

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-extension \
  --type ldap-server \
  --extension-name proxy-extension-4b \
  --set enabled:true \
  --set remote-ldap-server-address:DS4b_hostname \
  --set remote-ldap-server-port:3489

#Create a proxy workflow element for each LDAP server extension
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name proxy-we-3a \
  --type proxy-ldap \
```

```
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-3a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name proxy-we-4a \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-4a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name proxy-we-3b \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-3b

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name proxy-we-4b \
  --type proxy-ldap \
  --set enabled:true \
  --set client-cred-mode:use-client-identity \
  --set ldap-server-extension:proxy-extension-4b

# Create a load balancing workflow element for each data center
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name LB-we-2a \
  --type load-balancing \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name LB-we-2b \
  --type load-balancing \
  --set enabled:true

# Define the load balancing algorithm for each data center
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-algorithm \
  --element-name LB-we-2a \
  --type proportional

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
```

```
create-load-balancing-algorithm \
  --element-name LB-we-2b \
  --type proportional

# Define the load balancing routes for each proxy
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name LB-we-2a \
  --route-name LB-route-3a \
  --type proportional \
  --set workflow-element:proxy-we-3a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name LB-we-2a \
  --route-name LB-route-4a \
  --type proportional \
  --set workflow-element:proxy-we-4a

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name LB-we-2b \
  --route-name LB-route-3b \
  --type proportional \
  --set workflow-element:proxy-we-3b

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name LB-we-2b \
  --route-name LB-route-4b \
  --type proportional \
  --set workflow-element:proxy-we-4b

# Set failover between the two data centers
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name FO-we2 \
  --type load-balancing \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-algorithm \
  --element-name FO-we2 \
  --type failover

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name FO-we2 \
```

```
  --route-name FO-route-2a \
  --type failover \
  --set workflow-element:LB-we-2a \
  --set priority:1

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-load-balancing-route \
  --element-name FO-we2 \
  --route-name FO-route-2b \
  --type failover \
  --set workflow-element:LB-we-2b \
  --set priority:2

# Create distribution to the two failover routes
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow-element \
  --element-name distrib-we \
  --type distribution \
  --set base-dn:dc=example,dc=com \
  --set enabled:true

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-distribution-algorithm \
  --element-name distrib-we \
  --type numeric \
  --set distribution-attribute:uid

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-distribution-partition \
  --element-name distrib-we \
  --partition-name distrib-part1\
  --type numeric \
  --set lower-bound:0 \
  --set upper-bound:1000 \
  --set partition-id:1 \
  --set workflow-element:FO-we1

dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-distribution-partition \
  --element-name distrib-we \
  --partition-name distrib-part2\
  --type numeric \
  --set lower-bound:1000 \
  --set upper-bound:2000 \
  --set partition-id:2 \
  --set workflow-element:FO-we2
```

```
# Create workflow
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-workflow \
  --workflow-name Distrib-workflow \
  --set enabled:true \
  --set base-dn:dc=example,dc=com \
  --set workflow-element:distrib-we

# Create network group
dsconfig -p 4444 -h localhost -D"cn=Directory Manager" -w password -X -n \
create-network-group \
  --group-name network-group1 \
  --set enabled:true \
  --set workflow:Distrib-workflow \
  --set priority:1
```