Framework 8.1

Load Distribution Server

User's Guide

About Genesys

Alcatel-Lucent's Genesys solutions feature leading software that manages customer interactions over phone, Web, and mobile devices. The Genesys software suite handles customer conversations across multiple channels and resources—self-service, assisted-service, and proactive outreach—fulfilling customer requests and optimizing customer care goals while efficiently using resources. Genesys software directs more than 100 million customer interactions every day for 4000 companies and government agencies in 80 countries. These companies and agencies leverage their entire organization, from the contact center to the back office, while dynamically engaging their customers. Go to www.genesyslab.com for more information.

Each product has its own documentation for online viewing at the Genesys Technical Support website or on the Documentation Library DVD, which is available from Genesys upon request. For more information, contact your sales representative.

Notice

Although reasonable effort is made to ensure that the information in this document is complete and accurate at the time of release, Genesys Telecommunications Laboratories, Inc., cannot assume responsibility for any existing errors. Changes and/or corrections to the information contained in this document may be incorporated in future versions.

Your Responsibility for Your System's Security

You are responsible for the security of your system. Product administration to prevent unauthorized use is your responsibility. Your system administrator should read all documents provided with this product to fully understand the features available that reduce your risk of incurring charges for unlicensed use of Genesys products.

Trademarks

Genesys, the Genesys logo, and T-Server are registered trademarks of Genesys Telecommunications Laboratories, Inc. All other trademarks and trade names referred to in this document are the property of other companies. The Crystal monospace font is used by permission of Software Renovation Corporation, www.SoftwareRenovation.com.

Technical Support from VARs

If you have purchased support from a value-added reseller (VAR), please contact the VAR for technical support.

Technical Support from Genesys

If you have purchased support directly from Genesys, please contact Genesys Technical Support at the regional numbers provided on page 8. For complete contact information and procedures, refer to the ***Genesys Technical Support Guide***.

Ordering and Licensing Information

Complete information on ordering and licensing Genesys products can be found in the ***Genesys Licensing Guide.***

Released by

Genesys Telecommunications Laboratories, Inc. www.genesyslab.com

Document Version: 81fr_us_lds_10-2011_v8.1.001.00

Table of Contents

Preface **7**

About Load Distribution Server 7
Intended Audience 8
Making Comments on This Document 8
Contacting Genesys Technical Support 8
Document Change History 9

Chapter 1 **Overview** **11**

Concepts 11
Overview 11
Application Types for LDS 12
Single T-Server Configuration 12
Resulting Distribution Modes 13
Load Distribution Mode 13
TProxy Mode 14
Tiered TProxy Mode 16
Broadcast Mode 17
Single T-Server LDS Mode 18
New for LDS in Release 8.1 20

Chapter 2 **Weighted Round Robin (WRR) Mode** **23**

Concept 23
Configuration Options 24

Chapter 3 **Installation** **25**

Installing LDS 25

Chapter 4 **Starting and Stopping LDS** **27**

Starting LDS 27
Stopping LDS 28

Chapter 5 **High-Availability (HA) Configuration 29**

LDS Backup Modes 29
Warm Standby 29
Hot Standby 30
Dynamic HA Model 30
Message-Synchronization Queue 30
Changes to Redundancy Types 31
Changes to HA Synchronization Level 31
Receiver Backup Modes 32

Chapter 6 **LDS Support (Load Distribution Mode) of Routing 33**

LDS and Routing 33
LDS Support of High-Availability Routing 34
LDS and Routing Components 34
Using LDS in Routing Solutions 35
System Configuration and LDS 35
URS as a Client to LDS 35
LDS and Network Routing 38
Scalability for LDS and URS Pairs 38
URS and Backup LDS 39
Backup LDS in Warm Standby 39
Backup LDS in Hot Standby 40
Additional Information for LDS with URS 40
LDS and Receiver Type 40
Resource Registration 40
Agent Reservation Options 41

Chapter 7 **LDS Support (Load Distribution Mode) of Call Concentrator 43**

Recommended Configuration 43

Chapter 8 **LDS Configuration Options and Log Messages 45**

LDS Section 45
LDS Options Configured in Receivers 55
Changes from Release 7.2 to 8.1 56
Log Messages 58

Chapter 9 **Common Configuration Options 61**

Setting Configuration Options 61
Mandatory Options 62
log Section 62

Log Output Options 68
Examples 72
Debug Log Options 73
log-extended Section 76
log-filter Section 78
log-filter-data Section 78
security Section 79
sml Section 79
common Section 81

Supplements **Related Documentation Resources 83**

Document Conventions 85

Index **..... 87**

Preface

Welcome to the *Framework 8.1 Load Distribution Server User's Guide.* This document introduces you to the concepts, terminology, and procedures relevant to Load Distribution Server (LDS).

This document is valid only for the 8.1 release of this product.

Note: For versions of this document created for other releases of this product, visit the Genesys Technical Support website, or request the Documentation Library DVD, which you can order by e-mail from Genesys Order Management at orderman@genesyslab.com.

This preface contains the following sections:

- About Load Distribution Server, page 7
- Intended Audience, page 8
- Making Comments on This Document, page 8
- Contacting Genesys Technical Support, page 8
- Document Change History, page 9

For information about related resources and about the conventions that are used in this document, see the supplementary material starting on page 83.

About Load Distribution Server

In brief, you will find the following information in this guide:

- A high-level description of Load Distribution Server (LDS) and its uses
- A description of the distribution modes you can configure for LDS
- Procedures for installing and configuring LDS
- A description of all LDS configuration options
- Additional information about using LDS with Genesys Routing solutions

Configuration guidelines for using LDS with Call Concentrator (CCon), which is part of the Genesys Reporting solution

Intended Audience

This document is primarily intended for system administrators in contact centers. It has been written with the assumption that you have a basic understanding of:

- Computer-telephony integration (CTI) concepts, processes, terminology, and applications
- Network design and operation
- Your own network configurations

You should also be familiar with Genesys Framework architecture and functions.

Making Comments on This Document

If you especially like or dislike anything about this document, feel free to e-mail your comments to Techpubs.webadmin@genesyslab.com.

You can comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this document. Please limit your comments to the scope of this document only and to the way in which the information is presented. Contact your Genesys Account Representative or Genesys Technical Support if you have suggestions about the product itself.

When you send us comments, you grant Genesys a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

Contacting Genesys Technical Support

If you have purchased support directly from Genesys, contact Genesys Technical Support at the following regional numbers:

Region	Telephone	E-Mail
North America and Latin America	+888-369-5555 (toll-free) +506-674-6767	support@genesyslab.com
Europe, Middle East, and Africa	+44-(0)-1276-45-7002	support@genesyslab.co.uk
Before contacting technical support, refer to the *Genesys Technical Support Guide* for complete contact information and procedures.		

Region	Telephone	E-Mail
Asia Pacific	+61-7-3368-6868	support@genesyslab.com.au
Malaysia	1-800-814-472 (toll-free in Malaysia) +61-7-3368-6868 (toll)	support@genesyslab.com.au
India	000-800-100-7136 (toll-free) +61-7-3368-6868 (International)	support@genesyslab.com.au
Japan	+81-3-6361-8950	support@genesyslab.co.jp
Before contacting technical support, refer to the *Genesys Technical Support Guide* for complete contact information and procedures.		

Document Change History

This is the first release of the *Framework 8.1 Load Distribution Server User's Guide*. In the future, this section will list topics that are new or that have changed significantly since the first release of this document.

Chapter

1 Overview

This chapter describes four of the five different LDS distribution modes. A fifth mode (`Weighted Round Robin` mode) is described in Chapter 2, "Weighted Round Robin (WRR) Mode," on page 23. This chapter contains the following topics:

- Concepts, page 11
- Load Distribution Mode, page 13
- TProxy Mode, page 14
- Broadcast Mode, page 17
- Single T-Server LDS Mode, page 18
- New for LDS in Release 8.1, page 20

Concepts

Overview

Purpose

LDS is designed to increase system performance in contact center environments with high call volumes. LDS enables load sharing in situations where the total traffic of a large installation exceeds the capacity of individual Receivers. Using LDS with multiple Receivers also increases redundancy in a configuration.

Terminology

LDS mediates between *Senders* and *Receivers*.

A Sender is a T-Server. Any premise or network T-Server can be a Sender.

A Receiver is a T-Server client.

Smart Distribution of Events

LDS divides the traffic into manageable portions and distributes it among Receivers by using *smart distribution* of T-Server events (*T-events*). Smart distribution means that LDS correctly identifies all T-events related to a given interaction and passes them on to the appropriate Receiver(s).

Start/Stop Overview

At startup, LDS connects to all T-Servers for which it is configured, without waiting for Receiver connections. If it cannot establish a connection at that time with a T-Server at startup, LDS repeats the connection attempt after a Receiver connects to it. LDS stays connected to the T-Servers as long as it is running, but it unregisters from all DNs after the last Receiver disconnects from it.

Receiver Types

From the T-Server's perspective, the type of Receiver is unimportant.

The Receiver type that an LDS instance supports is dynamically defined during runtime by the first Receiver to succeed in connecting to LDS. If you need to configure LDS for different Receiver types within the same system, you must use a separate instances of LDS for each Receiver type.

Note: The type of client application that can be a Receiver differs according to the LDS mode of operation.

Modes of Operation

You can use LDS in any of the following four distribution modes:

- `Load Distribution` mode
- `TProxy` mode
- `Broadcast` mode
- `Single T-Server LDS` mode

These modes are described in the remainder of this chapter. More detailed information on specific configurations and usages can be found in later chapters of this document.

Application Types for LDS

You can configure an LDS with application type:

- `LoadDistributionServer`
- `T-Server`

Single T-Server Configuration

Prior to release 7.0, LDS could operate only in a multi-T-Server configuration. In this configuration and in normal `Load Distribution` mode, LDS connects to multiple T-Servers to broker messages to multiple Receivers (see Chapter 6, "LDS Support (Load Distribution Mode) of Routing," on page 33 for an illustration of how this mode operates with the Universal Routing Server (URS) routing client).

From release 7.0 onward, LDS can also operate in a single–T-Server configuration using the functionalities described in this chapter.

You can use the proxy functions described in this chapter only in a single–T-Server environment where an LDS is configured with application type `T-Server`.

Resulting Distribution Modes

Configuration option `distribute-mode` specifies the default distribution mode of LDS.

As a result of the two application types, and the ability to use LDS in a single–T-Server environment, three distribution modes are available, as shown in Table 1 on page 13.

.

Table 1: LDS Distribution Modes

LDS with Application Type...	Distribution Mode		
	`distribute-mode = load`	`distribute-mode = auto`	`distribute-mode = proxy`
Load Distribution Server	Load Distribution	Load Distribution	Broadcast
T-Server (multiple T-Servers configured in `Connections` tab)	Load Distribution	TProxy	TProxy
T-Server (single T-Server configured in `Connections` tab)	Single T-Server LDS	TProxy	TProxy

Warning! Please keep in mind that `TProxy` mode is not compatible with the normal `Load Distribution` mode. You cannot mix these modes without breaking the load distribution model.

Load Distribution Mode

Concept

In `LoadDistribution` mode, LDS distributes requests (balances the load) among the Receivers by using smart distribution of T-events. Smart distribution means that LDS correctly identifies all T-events related to a given interaction and passes them on to the appropriate Receiver(s). Multiple–T-Server configurations can include any combination of premise and network T-Servers.

Supported Receiver Types

In `Load Distribution` mode, LDS supports only the following Receivers:

- URS release 6.5 and later
- CCon, release 6.1.001.12 and later

Illustration

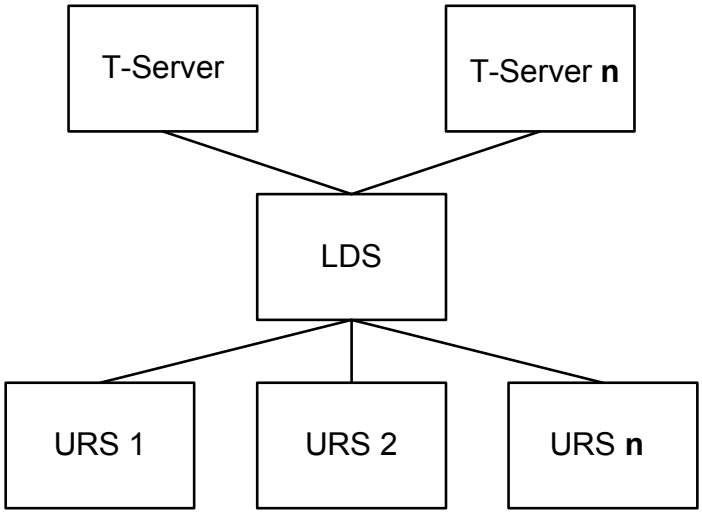

Figure 1: LDS in Load Distribution Mode

See Chapter 6, for samples of LDS configurations with Genesys Routing solutions.

Configuration

To configure LDS to operate in `Load Distribution` mode:

1. In Configuration Manager, create an `Application` object of type `LoadDistributionServer` using the `LDS_Server_810` application template.
2. Install LDS, choosing `Load Distribution` mode during the installation procedure.
3. On the `Connections` tab of the new application, add a connection to each T-Server in the configuration.
4. On the `Options` tab, set the value of LDS configuration option `distribute-mode` to `load`.
5. Configure Receiver connections to point to the new LDS application.

Note: Configuration of the `lds-query-dn` extension on the switch is mandatory. Configure this setting in the `query-dn` parameter option on the `LDS` application to avoid SIP Server (and any other T-Server) `EventError` messages.

TProxy Mode

Concept

Used in `TProxy` mode with a single T-Server, LDS reduces the amount of data transmitted over a WAN between remote T-Servers and T-Server clients in a central site. Instead of sending the same events multiple times—once for every client—T-Server sends the events a single time to a central LDS, which then

distributes this event to all clients on the central site that are registered for a particular DN.

`TProxy` mode has the potential to reduce the volume of data carried over the WAN between T-Server and LDS to 1/*N* of the volume of data carried over the WAN when clients connect directly to T-Server. This can lead to a substantial cost reduction for the customer in environments where costs are based on number of bytes transmitted.

Supported Receiver Types

When LDS is in `TProxy` mode, Receivers can be of any type and combination of types, provided that they are T-Library compliant. There are limitations to this, however. *(For example, ICON is currently not supported in the 8.1.0 release of LDS.)*

Illustration

Figure 2 illustrates how LDS in `TProxy` mode distributes TEvent 1. TEvent 1 is carried only once over the WAN before being distributed to all clients registered for the relevant DN. Redundant T-Server and LDS configurations are shown in red.

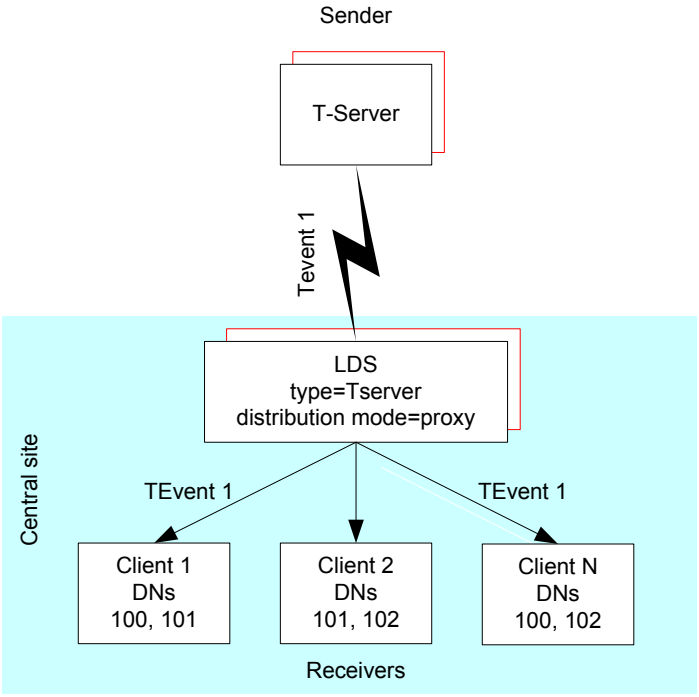

Figure 2: LDS in TProxy Mode

Configuration

To configure LDS to operate in `TProxy` mode:

1. In Configuration Manager, create an `Application` object of type `T-Server` using the `TProxy_Server_810` application template.
2. Install LDS, choosing `TProxy` mode during the installation procedure.
3. On the `Connections` tab of the new application, add a connection to a single T-Server.

4. On the `Switch` tab, add the same switch as is configured for the T-Server you added in Step 3.
5. On the `Options` tab, set the value of LDS configuration option `distribute-mode` to either `auto` or `proxy`.
6. Configure client connections to point to the new LDS application.

Note: From release 7.1, LDS configured in `TProxy` mode can control both active and passive Receivers simultaneously.

Note: Configuration of the `lds-query-dn` extension on the switch is mandatory. Configure this setting in the `query-dn` parameter option on the `LDS` application to avoid SIP Server (and any other T-Server) `EventError` messages.

Tiered TProxy Mode

In this section the term *TProxy* is used to describe an LDS configured in `TProxy` mode. So a client of such an LDS is a *TProxy client*, and so on.

Prior to release 7.1, the design of `TProxy` mode assumed either a TProxy directly connected to T-Server(s), or a TProxy-aware client that was able to access T-Server's switch configuration by using information from the LDS connection.

From release 7.1, you can create tiered configurations of TProxies, in which one TProxy can be a client of another TProxy. In this structure, a TProxy client can reach switch configuration details at the end of a chain of TProxies (or at any point within the chain where application type `T-Server` is found).

Such tiered configurations can be used to reduce the amount of network traffic flowing over the WAN between remote and central sites.

Example Configurations

Figure 3 on page 17 shows how tiered TProxies can be configured in different environments.

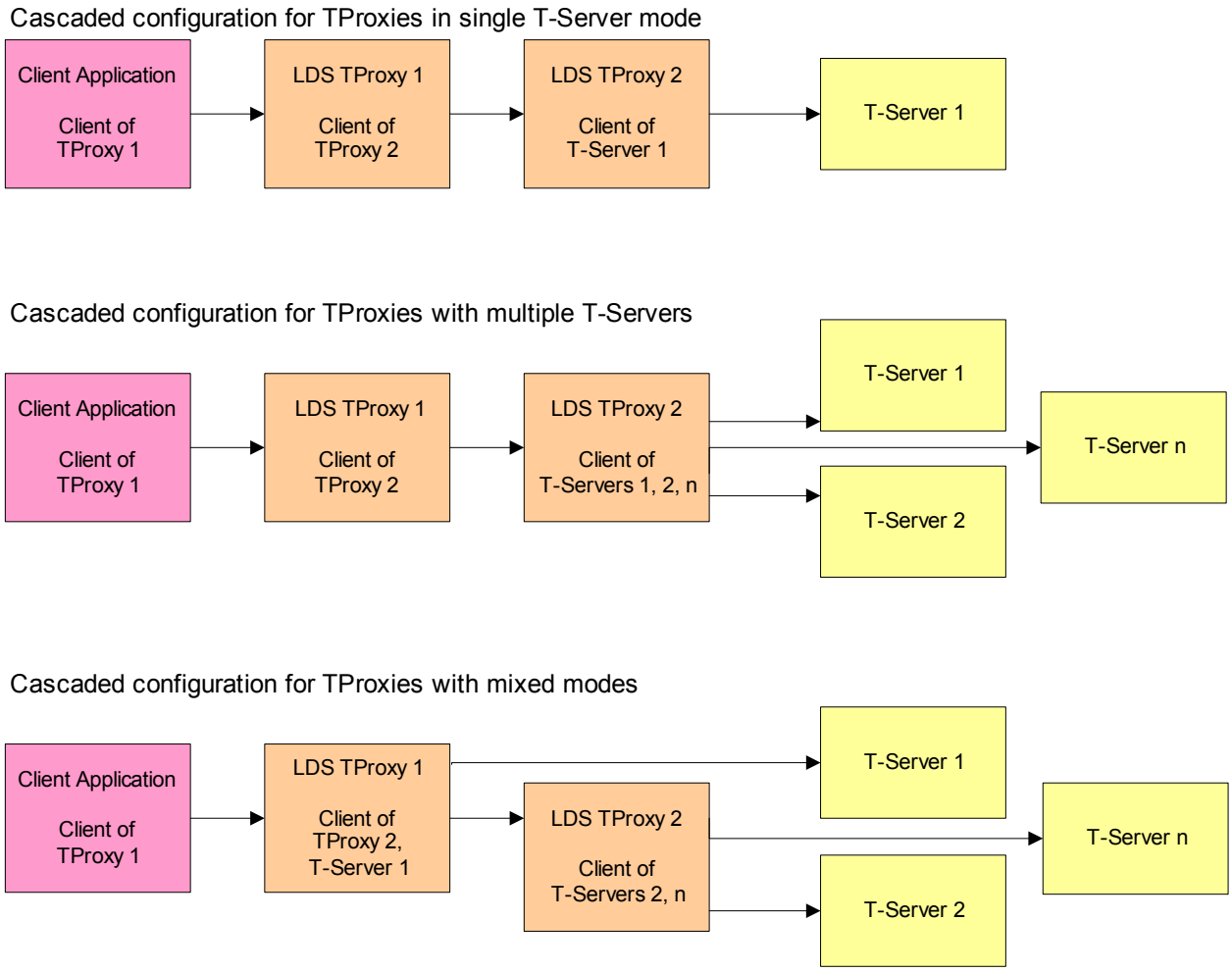

Figure 3: Tiered (Cascaded) Proxy Configuration Examples

Broadcast Mode

Concept

In `Broadcast` mode, LDS broadcasts events to any client that requests them. Only special T-Library clients which provide additional information in the connection request can work with LDS in `Broadcast` mode. At the time of this writing, only CCon, URS, and custom applications can connect in this mode.

LDS distributes events to clients accordingly to their subscription (registration on DN for DN based events, registration on call monitored events for call monitoring events, and so on).

Supported Receiver Types

In `Broadcast` mode, LDS currently supports the following Receivers:

- URS release 6.5 and later
- CCon, release 6.1.001.12 and later

Illustration Figure 4 on page 18 illustrates how `Broadcast` mode operates. Redundant T-Server and LDS configurations are shown in red.

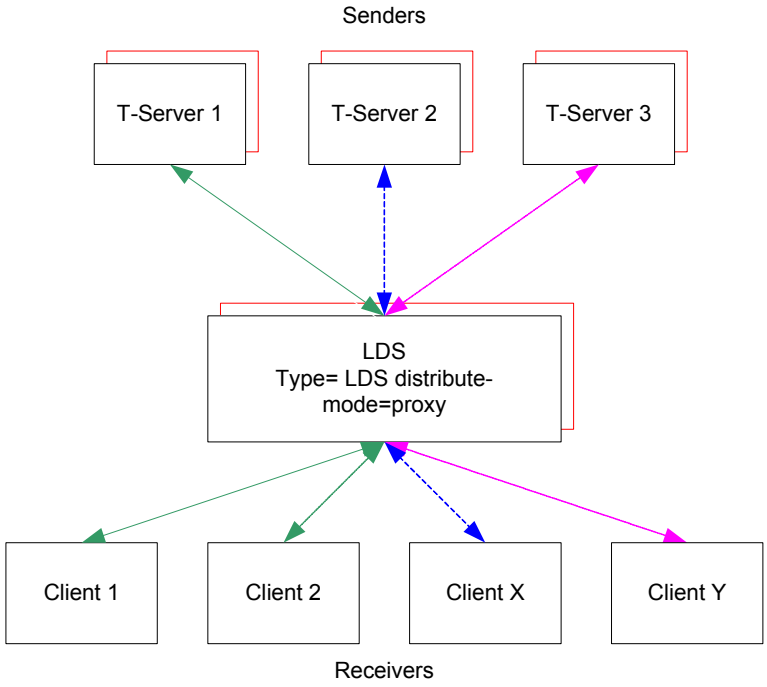

Figure 4: LDS in Broadcast Mode

Configuration To configure LDS to operate in `Broadcast` mode:

1. In Configuration Manager, create an `Application` object of type `LoadDistributionServer` using the `LDS_Server_810` application template.
2. Install LDS, choosing `Load Distribution` mode during the installation procedure.
3. On the `Connections` tab of the new application, configure the relevant connections to T-Servers.
4. On the `Options` tab of each instance of LDS, set the value of LDS configuration option `distribute-mode` to `proxy`.
5. Configure Receiver connections to point to the new LDS application.

Single T-Server LDS Mode

Concept When LDS is used in this mode, any Receiver can connect to LDS without modification, provided that LDS connects to only a single T-Server.

In earlier versions of LDS that do not support this mode of operation, you must modify Receivers to enable them to connect to multiple T-Servers—when connecting to LDS, the Receiver has to specify which T-Server it wants LDS to

connect to. When LDS connects to only one T-Server, you do not have to modify the Receiver.

Supported Receiver Types

With LDS in `Single T-Server LDS` mode, Receivers can be of any type, but must all be of the same type. If Receivers of different types are configured, load distribution becomes meaningless.

Illustration

Figure 5 illustrates how this mode operates.

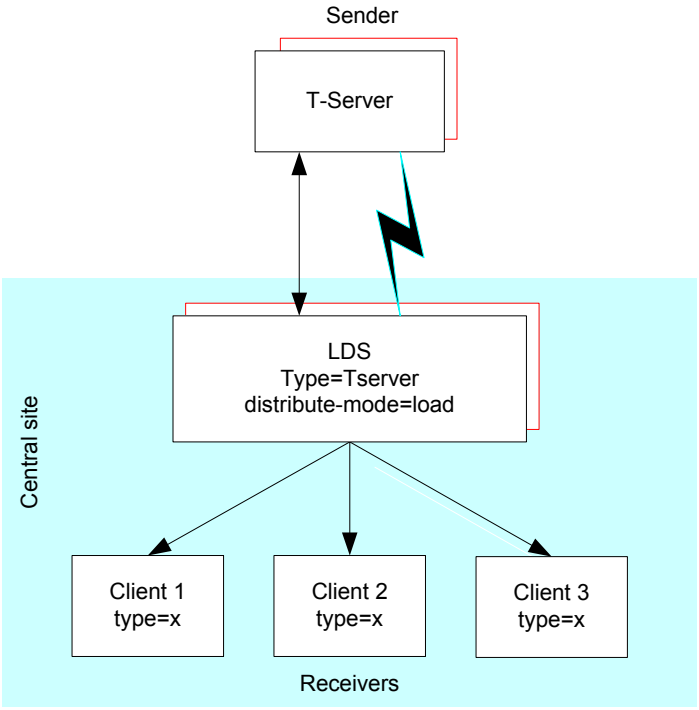

Figure 5: LDS in Single T-Server LDS Mode

Redundant T-Server and LDS configurations are shown in red.

Configuration

To configure LDS to operate in `Single T-Server LDS` mode, follow the steps below.

1. In Configuration Manager, create an `Application` object of type `T-Server` using the `TProxy_Server_810` application template.
2. Install LDS, choosing `TProxy` mode during the installation procedure.
3. On the `Connections` tab of the new application, add a connection to a single T-Server.
4. On the `Switch` tab, add the same switch as is configured for the T-Server you added in Step 3.
5. On the `Options` tab, set the value of LDS configuration option `distribute-mode` to `load`.
6. Configure Receiver connections to point to the new LDS application.

New for LDS in Release 8.1

This section gives summary details of new features in release 8.1.

- **Support for new operating systems and platforms:**
  - Microsoft Windows Server versions Windows 2008 64-bit
  - Red Hat Linux version 5.0 64-bit
  - IBM AIX version 7.1 64-bit
  - HP-UX IPF version 11i v3 Integrity Native
  - VMware vSphere 4 Hypervisor
- **Support for common component features:**
  - Acresso FLEXNet Publisher version 11.9
  - TCP/IP-v6. For more information, refer to the *Framework 8.1 Deployment Guide*.
  - LDS no longer connects to applications that have disabled status in the configuration environment.
  - The default value of the `background-processing` configuration option has been changed to `true`. See "background-processing" on page 46 for details.
- **Support for the Unresponsive Process Detection feature.** The following configuration options enable this feature:
  - "heartbeat-period" on page 79.
  - "hangup-restart" on page 80.

  For more information, refer to the *Framework 8.0 Management Layer User's Guide*.
- **Support for enhanced logging capability.** The maximum number of log files to be stored using the `expire` configuration option is now `1000`.See "expire" on page 63.
- **Support for an alarm for expired requests in STD log mode:** If any requests have expired based on the option `rq-expire-timeout`, STD log messages are written to the log file allowing the customer to generate an alarm.
- **Support for:**
  - **Genesys Administrator (GA).** LDS can be installed and configured using GA.
  - **Sender Link Bandwidth Monitoring.** LDS reports the number of incoming and outgoing messages on all sender links within the time interval specified by the configuration option `load-report-interval`, and reports this using a dedicated LMS message (see "Log Messages" on page 58).
  - **Sender Link Bandwidth Alarm Notification.** LDS can notify the Genesys Management Layer when set limits are exceeded by setting an alarm threshold value as a percentage of maximum throughput. The

value in the configuration option `use-link-bandwidth` represents 100% of bandwidth. High and low watermarks are specified in the options `link-alarm-high` and `link-alarm-low` as a percentage of the highest bandwidth.

- **Enhanced support of T-Server HA failover.** LDS now sends `EventPrimaryChanged` upon a T-Server role change even when the LDS role did not change.
- **Bulk request pacing.** LDS paces requests that were sent to the Sender on behalf of initializing clients, and applies the limit that is set in the configuration option `max-outstanding` for the following requests: `RequestRegisterAddress`, `RequestQueryAddress`, `RequestQueryCall`. The default value of the `background-processing` configuration option has been changed to `true` to accommodate this.

Chapter

2 Weighted Round Robin (WRR) Mode

This chapter provides information about the fifth of the five distribution modes, `Weighted Round Robin (WRR)` mode. See Chapter 1, "Overview," on page 11 for details of the other four.

This chapter includes the following sections:

- Concept, page 23
- Configuration Options, page 24

Concept

`Weighted Round Robin (WRR)` mode is a variant of the standard `Load Distribution` mode that is achieved by configuring the Receiver application as described in this chapter. With `WRR` mode, you can alter the loading profile of an individual Receiver (or group of Receivers). So, for a given set of three Receivers, you can configure them to receive, for example, 50 per cent, 30 per cent and 20 per cent of the transactions, or any other proportion depending on your environment.

Receivers can also be grouped, and the loading profile can be set for a group.

Configuration Options

Set the two configuration options in this section in the `LDS` section of the relevant Receiver application. The section name within the Receiver application must be `LDS`.

Set the Receiver weighting using the configuration option `loading-coefficient` (see "loading-coefficient" on page 56).

Use the `group-id` option to group a set of Receivers together (see "group-id" on page 55). LDS treats such a group as a single Receiver.

Chapter

3 Installation

This chapter describes how to install LDS in UNIX/Linux and Microsoft Windows environments. It contains one section:

- Installing LDS, page 25

Installing LDS

This section describes the installation of LDS on UNIX/Linux and Windows.

Installing on UNIX/Linux

1. On the product CD, locate the appropriate shell script.
2. Run the script from the command prompt by typing `sh` followed by the file name.
3. When prompted, specify the `Host Name` of the computer on which to install LDS.
4. When prompted, specify the mode of LDS operation (`TProxy` or `LoadDistribution`).
5. When prompted, specify the following:
   - `Host Name` of the computer on which Configuration Server is running.
   - `Port` that client applications use to connect to Configuration Server.
   - `User Name` used to log in to the Configuration Layer.
   - `Password` used to log in to the Configuration Layer.
6. Depending on the mode selected in Step 4, installation displays a list of LDS applications of the relevant type configured for this host. Enter the number of the LDS application that you want to install.
7. Specify the full path of the destination directory into which you want to install LDS.
8. If asked, choose to install either the 32-bit or the 64-bit version, depending on your environment.

9. When prompted, specify the path to a valid license file.

As soon as the installation process is finished, a message appears announcing that the installation was successful. The process places LDS in the directory specified in Step 7.

Installing on Windows

1. From the product CD, locate and double-click the appropriate `Setup.exe` to start the installation.
2. When prompted, specify the mode of LDS operation (`TProxy` or `LoadDistribution`).
3. When prompted, specify the `Host` and `Port` of Configuration Server. Accept `ITCUtility` as the name of the Installation Configuration Utility application.
4. When prompted, specify the `User Name` and the `Password` used to log in to the Configuration Layer.
5. Confirm the `Host Name` of the computer on which to install LDS.
6. Depending on the mode selected in Step 2, installation displays a list of applications of the relevant type configured for this host. From the list, select the LDS application to install.
7. Specify the full path of the directory into which to install LDS.
8. Specify the `Program Folder` to which you want LDS added.
9. When prompted, specify the path to a valid license file.
10. Decide whether you want to install LDS as a Windows service. For more information, see the *Framework 8.1 Deployment Guide*.
11. When icons for LDS appear, click `Finish` to complete the installation

Note: LDS supports Configuration Server backup and Configuration Server proxy configurations.

Chapter

4 Starting and Stopping LDS

This chapter describes how to start and stop LDS. It contains the following topics:

- Starting LDS, page 27
- Stopping LDS, page 28

Starting LDS

Ensure that DB Server and Configuration Server are running. If you are using the Management Layer, ensure that all of its components are running, including Solution Control Interface. For instructions on starting and stopping LDS via the Management Layer, see *Solution Control Interface Help* in the Genesys Framework 8.1 documentation.

Command-Line Parameters

You must specify command-line parameters (also called command-line arguments) to operate LDS, whether you are operating it manually or with the Management Layer. In manual operation, you either enter parameters directly on a command line or invoke them from a batch file, which is invoked in turn either directly on a command line or via a shortcut on the Windows `Start` menu. With the Management Layer, you specify the parameters on the `Start Info` tab (in the `Command Line Arguments` field) of the LDS `Properties` window.

This section lists the required command-line parameters. See also the *Framework 8.1 Deployment Guide.*

Note: The first command-line parameter is always the name of the executable application file. On Windows, it is best to add the extension `.exe` to the executable file's name. For example, use `LDServer` on UNIX/Linux and use `LDServer.exe` on Windows.

These are the required command-line parameters:

`-host <confighost>`Represents the host running Configuration Server

`-port <configport>` Represents the port used by Configuration Server

`-app <appname>` Represents the name of the application as configured in Configuration Manager

`-l <license_file>` Represents either of:

The full path to, and the exact name of, the license file. For example, `-l /opt/mlink/license/license.dat`.

The `host name` and `port` of the license server, as specified in the `SERVER` line of the license file, in the format `port@host`. For example, `-l 7260@ABCserver`.

Stopping LDS

You can stop LDS from the Management Layer, or you can use any of the following manual procedures:

- Use the `Ctrl + C` command in the component's console window (on both Windows and UNIX/Linux).
- Use the `End Task` button in the Windows Task Manager.
- Use the `kill <processnumber>` command on UNIX/Linux.

Chapter

5 High-Availability (HA) Configuration

This chapter describes high-availability (HA) modes for LDS and Receivers. It contains the following these sections:

- LDS Backup Modes, page 29
- Dynamic HA Model, page 30
- Receiver Backup Modes, page 32

LDS Backup Modes

Note: Genesys recommends that LDS pairs used in HA configurations have identical configurations.

LDS supports both warm-standby and hot-standby configurations.

Warm Standby

In LDS `warm standby` mode:

- Both the primary and backup LDSs are connected to all T-Servers using identical configurations.
- The primary LDS accepts Receiver connections and registers to T-Servers for events.
- The backup LDS does not accept Receiver connections and does not register for T-Server events.
- The `ha-sync-level` configuration option is ignored.

Hot Standby

In LDS `hot standby` mode:

- Both the primary and backup LDSs (ideally with identical configurations) are connected to all T-Servers.
- From LDS release 6.5.3 onward, the primary and backup LDS can start successfully with different values set in configuration option `ha-sync-level`. When both LDSs are started, the backup LDS adopts the value set for the primary LDS.
- Both the primary and backup LDSs accept Receiver connections and registration requests for T-Server events.
- The primary LDS registers to T-Servers to receive events according to Receiver requests. The backup LDS registers for events in the same way as the primary, but it also asks T-Server to mask events in a way that corresponds to the value set for configuration option `ha-sync-level`. See "Dynamic HA Model" on page 30.
- Only the primary LDS, passes events to Receivers and returns responses to T-Servers.
- Both the primary and backup LDSs synchronize transaction context in real time.
- To prevent network overloading, primary and backup LDSs are not synchronized at startup.
- The HA LDS Application Programming Interface (API) enables Receivers to connect to the primary and standby LDSs simultaneously. Registration requests go to both LDSs, and transparently to Receivers. Failover to the backup LDS also occurs transparently.

The standby mode for LDS can be different from the standby mode of T-Servers and of Receivers, and different T-Servers can have different standby modes.

Note: From release 6.5.3 onwards, where there is more than one LDS, each instance of LDS starts by default in `Backup` mode, and Management Layer must switch one instance to `Primary` mode.

Dynamic HA Model

Message-Synchronization Queue

From LDS release 6.5.3 onward, a Message-Synchronization Queue feature has been implemented to provide uninterrupted event flow from LDS to Receivers after LDS switchover. The queue enables the backup LDS to track

messages that the primary LDS has already sent. After a switchover, the new primary LDS resumes message distribution from the same point in the message queue at which the old primary LDS had stopped.

Changes to Redundancy Types

Table 2 shows how LDS reacts to dynamic changes to its redundancy type.

Table 2: Effect of Dynamic Changes to LDS Redundancy Types

	Redundancy Type Changed To...			
Redundancy Type Changed From...		Not Specified	Warm	Hot
	Not specified		Switch to backup `warm standby` mode. Wait for instructions from Management Layer.	Switch to backup `hot standby` mode. Wait for instructions from Management Layer.
	Warm	Backup LDS creates listen port. No action for former primary.		Start LDS synchronization. Backup LDS creates listen port.
	Hot	Stop LDS synchronization. Backup LDS resumes event distribution to clients.	Stop LDS synchronization. Backup LDS closes listen port.	

Changes to HA Synchronization Level

Table 3 shows the effects of dynamic changes to the value of configuration option `ha-sync-level`.

Note: These changes apply only to LDS in `hot standby` mode.

.

Table 3: Effects of Dynamic Changes to HA Synchronization Level

	Value Changed To...			
Value Changed From...		**0**	**1**	**2**
	0		*Primary:* No action.	*Primary:* Initialize message-synchronization queue.
			Backup: Set `Input Mask` to `High Level`.	*Backup:* Set `Input Mask` to `High Level`. Initialize message-synchronization queue.
	1	*Primary:* No action.		*Primary:* Initialize message-synchronization queue.
		Backup: Set `Input Mask` to `Low Level`.		*Backup:* Set `Input Mask` to `High Level`. Initialize message-synchronization queue.
	2	Primary: Clear message-synchronization queue.	*Primary:* Clear message-synchronization queue.	
		Backup: Set `Input Mask` to `Low Level`. Clear message-synchronization queue.	*Backup:* Clear message-synchronization queue.	

Receiver Backup Modes

LDS supports backup configurations for Receivers in `hot standby` mode and for CCon in `parallel` mode.

Chapter

6 LDS Support (Load Distribution Mode) of Routing

Note: The information in this chapter applies only to LDS in `Load Distribution` mode. See Chapter 1, "Overview," on page 11 for a description of this mode.

This chapter provides information on LDS support of Genesys Routing solutions (including Enterprise Routing and Network Routing). It contains the following sections:

- LDS and Routing, page 33
- System Configuration and LDS, page 35
- URS and Backup LDS, page 39
- Additional Information for LDS with URS, page 40

The sections in this chapter explain:

- Routing (including high-availability routing) with LDS.
- The architecture supported by routing components, such as Universal Routing Server (URS) and Interaction Routing Designer (IRD).
- Application redundancy achieved by using LDS.
- Resource registration.
- Agent reservation features with LDS.

LDS and Routing

Use of LDS with Genesys Routing solutions provides a simple, scalable way to improve throughput by combining the processing power of multiple instances

of URS. The URS instances can run either on single-hardware platforms with multiple processors or on multiple-hardware platforms.

LDS Support of High-Availability Routing

The main reasons for using load distribution with Genesys Routing solutions in contact centers are:

- To meet customer requirements for several hundred interactions per second.
- To increase event throughput of a single URS process, which cannot be increased sufficiently by a hardware upgrade.
- To compensate for the decrease of single-URS throughput caused by complex routing-strategy requirements.
- To provide load distribution that is superior to multithreading, which is complex and does not solve all load distribution needs.
- To increase performance at a cost lower than that of a hardware upgrade.

For information on high-availability options for Enterprise Routing and Network Routing, see the *Universal Routing 8.1 Deployment Guide.*

LDS and Routing Components

LDS and URS

URS performs the following tasks:

- Executes the rules specified within a strategy created in IRD.
- Creates a list of destinations or targets based on the strategy.
- Uses Stat Server statistics to determine the most appropriate target.
- Instructs T-Server where to route an interaction

URS connects to LDS for every T-Server in the LDS `Connections` list.

Each URS operates independently from other URSs. URS can operate when its `Connections` list contains connections to both LDS and T-Servers and it can have more than one LDS in its `Connections` list.

When disconnecting from LDS, URS attempts to connect to a backup LDS (if one has been configured), not to a backup T-Server.

LDS and IRD

IRD is a user interface for creating, editing, loading, and monitoring routing strategies. IRD communicates with URS through Configuration Server. Through a Message Server, IRD also receives from URS real-time routing information about interactions, server status, and Routing Points.

In a non-LDS environment, a strategy is loaded on a Routing Point in only one URS, even if there are multiple URSs monitoring the same Routing Point. However, to ensure consistent routing results in an LDS environment, every URS that is connected to the same LDS and is monitoring the same Routing Point must have the same strategy loaded for that Routing Point.

For information about using IRD, see *Interaction Routing Designer 8.1 Help.*

Using LDS in Routing Solutions

You can use LDS in Routing solutions to distribute requests among primary URSs and to combine the processing power of URSs to increase total throughput.

In configurations with multiple URSs, LDS distributes interactions among the URSs. This reduces the probability of a URS failure because of decreased processing on any single instance of URS. (For more information on load distribution, see "URS as a Client to LDS".)

When using LDS, you can achieve redundancy in a configuration with any number of single instances of URS with no dedicated backup URS. In this configuration, when one URS fails, LDS redirects new routing requests to the remaining URS(s). After Management Layer restarts the failed URS, LDS resumes load distribution to all servers.

Redundant configurations using LDS with multiple instances of URS offer a way to scale up Routing solutions when interaction volume exceeds the capabilities of the existing hardware platform.

For more information on redundancy using LDS, see "Scalability for LDS and URS Pairs" on page 38.

System Configuration and LDS

This section describes some possible ways of configuring LDS with Routing solutions.

URS as a Client to LDS

For load distribution in configurations with URS as a client to LDS, connections between T-Server, LDS, and URS occur in the following ways:

- One T-Server to one LDS to multiple URSs
- Multiple T-Servers to one LDS to ***n*** URSs, where ***n*** refers to the number of single instances of URS required to meet the specified load
- Multiple T-Servers to multiple LDSs to multiple URSs

One T-Server, One LDS, and Multiple URSs

Use a configuration with one T-Server to one LDS to multiple URSs when the transaction rate (interactions/second) on the switch/T-Server pair is higher than a single instance of URS can handle. LDS will distribute routing requests (balance the load) among URSs. You can put one or multiple instances of URS on the same computer with multiple processors, or you can distribute URSs on several computers when the contact center's call volume requires additional processing power.

Multiple T-Servers, One LDS, and Multiple URSs

You can also connect multiple T-Servers to one LDS and multiple URSs to meet the specified load.

Figure 6 illustrates a configuration with LDS at the center of message distribution from multiple T-Servers to multiple URSs.

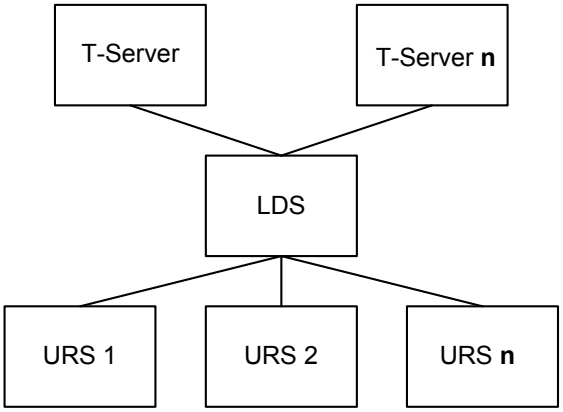

Figure 6: Multiple T-Servers to LDS to Multiple URSs

LDS distributes routing requests (balances the load) among the URSs. The multiple T-Servers can be either premise or network T-Servers.

You can also configure an environment using LDS and an ***n*** + 1 URS redundancy configuration without using a backup URS. (Here, ***n*** refers to the number of URSs required to meet the specified load.) When one URS shuts down, LDS redirects new routing requests to the remaining servers until Management Layer restarts the URS that shut down.

This form of redundancy (an ***n*** + 1 URS redundancy configuration) is most likely used in configurations with two or more primary URSs performing load balancing without LDS. However, LDS offers a growth path for an enterprise to add routing services when interaction volume increases beyond the capabilities of the existing hardware and software.

Note: Because detection and notification of URS failure are communicated through network messaging, you must configure network bandwidths to ensure that no delay occurs between SCS (Solution Control Server), URS, and LDS.

URS can receive and process messages from multiple T-Servers (including T-Server for PBX and Network T-Server) of any media type through the same LDS.

You do not have to install instances of LDS and URS on the same computer. However, you can strategically position LDS and multiple URSs on the same computer to optimize network traffic.

Multiple T-Servers, LDSs, and URSs

You can connect a URS to two different LDSs of the same type. This configuration might be necessary when the transaction rate of a single LDS exceeds its capacity; for example, if one LDS handles the transaction load for multiple T-Servers.

Figure 7 illustrates a configuration with multiple T-Servers dividing the transaction load between two LDSs and distributing transactions to multiple URSs. In this figure, URS 3 is also connected to two different LDSs of the same type.

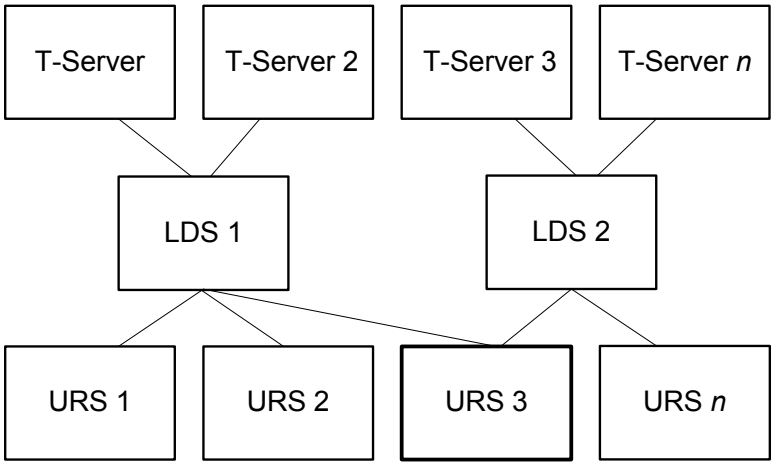

Figure 7: Multiple T-Servers to Multiple LDSs to Multiple URSs

LDS distributes routing requests according to the routing registration of URS. For information on this topic, see “Additional Information for LDS with URS” on page 40.

Note: Because LDS 1 and LDS 2 are not synchronized, there is no guarantee that the same router gets the call after a transfer. This may affect Inter Server Call Control (ISCC).

For information on a phased approach to introducing LDS into an existing architecture, see the *Universal Routing 8.1 Deployment Guide.*

LDS and Network Routing

You can add a new URS and LDS to an existing infrastructure to handle extra call volume and to provide scalability when the network's interaction volume increases beyond the capability of the existing infrastructure.

Figure 8 illustrates the addition of a new URS and LDS to a Network Routing solution.

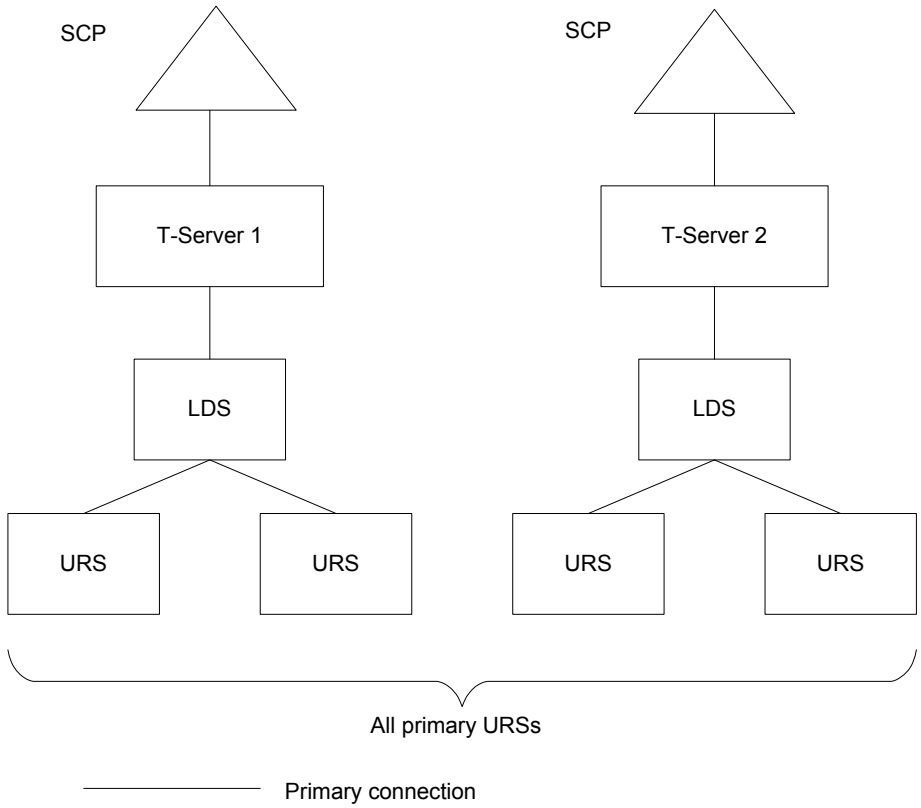

Figure 8: Adding a New URS and LDS to a Network Routing Solution

Scalability for LDS and URS Pairs

In Figure 9 on page 39, calls are distributed to 2*n* active servers, each handling approximately a 1/***n*** share of the load. Thus, the proportion of calls that cannot be routed when one server becomes unavailable is <= 1/(***n*–1**).

The connection between the `hot-standby` URS and its dependent servers is fully established. When a primary URS shuts down, new routing requests continue to be directed to the `hot-standby` URS in the pair. After the `hot-standby` URS becomes the primary URS, it replays the strategy for the pending interactions waiting in the Routing Point.

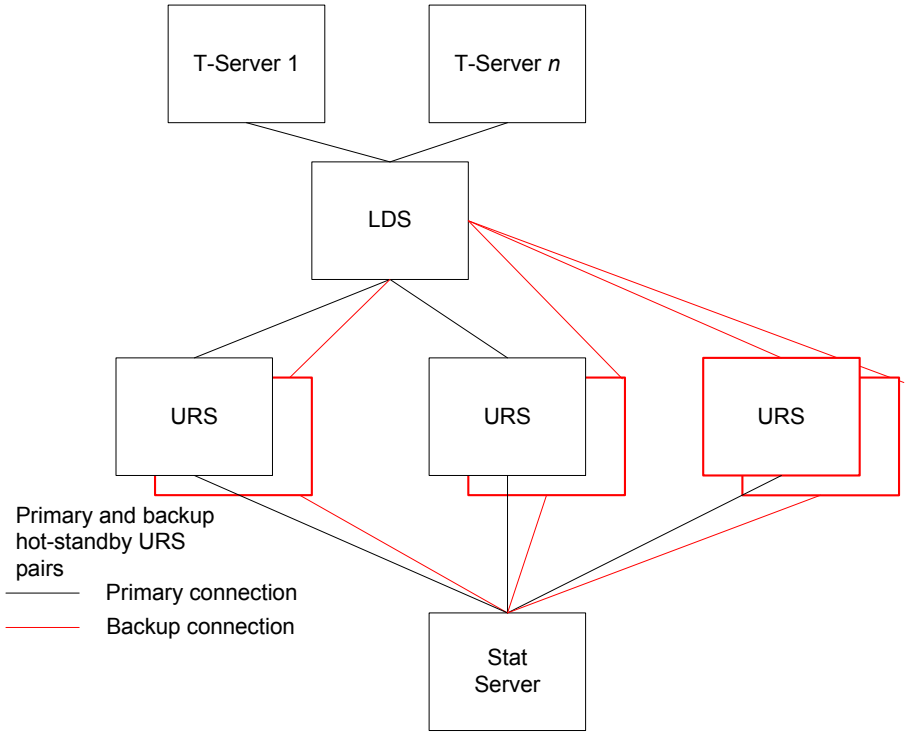

Figure 9: HA Routing Using an LDS and 2n URS Configuration

URS and Backup LDS

You can also configure LDS in primary and backup standby pairs. Only the primary LDS accepts connections from URS and registers for T-Server(s).

Backup LDS in Warm Standby

When a primary LDS shuts down, Management Layer notifies URS to reconnect to the backup LDS when the LDS failover process is complete.

Because a primary and backup LDS pair in `warm standby` mode are not synchronized, interactions in progress during failover are not recoverable. If there is a breakdown in communications between T-Server and URS during the failover period, the switch might default-route interactions.

Note: The fact that URS cannot route interactions *in this scenario* does not mean that they are irrecoverable. It means that URS cannot take control of these interactions or attempt to route them until after failover is complete.

For more information, see “High-Availability (HA) Configuration” on page 29.

Backup LDS in Hot Standby

You can also configure a primary and backup LDS pair in `hot standby` mode. Primary and `hot-standby` LDSs accept connections from URS and have connections to T-Server(s). However, only the primary LDS distributes events from T-Server(s).

Because primary and `hot-standby` LDSs are synchronized in a transaction context, T-Server events that the primary LDS submits are replayed by the `hot-standby` LDS after failover is complete. This means that no potential interactions are lost during the failover period.

See also "ha-sync-level" on page 48 and "Dynamic HA Model" on page 30.

Additional Information for LDS with URS

After you have configured and installed LDS, you can set up the routing environment. There are no routing-specific options that you need to set up in LDS or URS to enable routing to work with LDS: you have already set up options for LDS. (See page 45.) However, to get the most benefit from LDS, you need to take actions in respect of the following:

- LDS and Receiver type
- Resource registration
- Agent reservation options

LDS and Receiver Type

Currently, LDS is type specific. As a result, URS (the Receiver) can connect only to the LDS assigned to URS(s)—URS cannot connect to an LDS in Load Distribution mode that is already serving CCon clients.

LDS determines its type dynamically during runtime from the type of the first successful client connection to LDS; therefore, no explicit option settings are required.

Resource Registration

URS registers all Routing Points except those with option `event_arrive` configured and set to `none` in the Routing Point or virtual Routing Point properties. (This means that if you have not configured option `event_arrive`, URS registers for this Routing Point.) If URS is not registered to the specific Routing Point, URS receives no routing requests. Within a given LDS configuration, URS uses this mechanism to register to different Routing Points. Figure 10 illustrates this scenario.

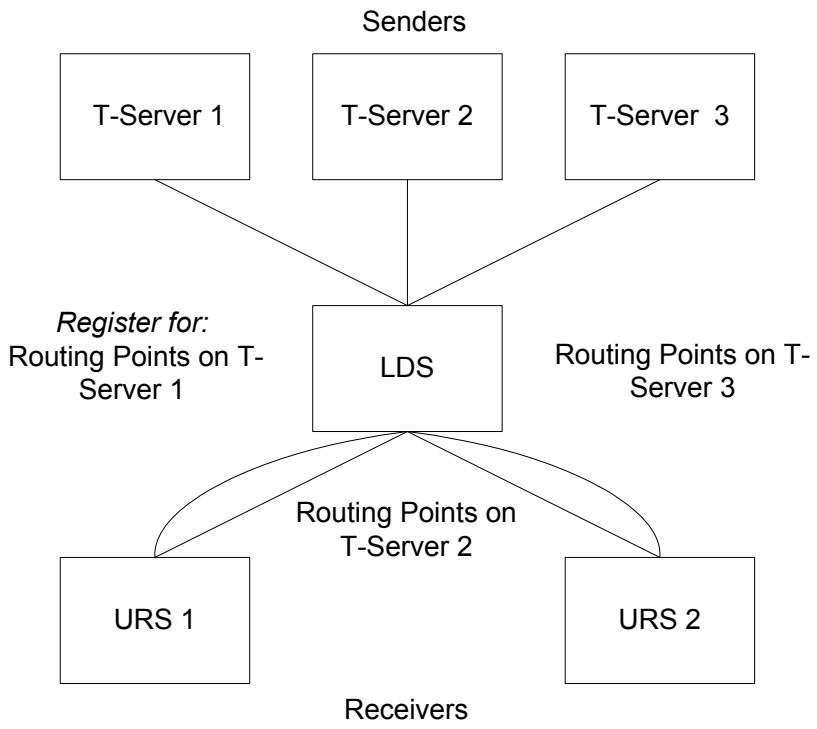

Figure 10: LDS and URS Registration of Routing Points

In Figure 10, URS 2 does not receive routing requests from T-Server 1. You establish this by configuring the `Annex` tab in the T-Server 1 `Application` object with a folder bearing the name of the URS 1 application at the individual Routing Point-level or virtual Routing Point-level.

LDS distributes the load (routing requests) according to the registration of Routing Points in URS. Interactions from T-Server 1 are sent only to URS 1. Interactions from T-Server 2 are distributed in `Load Distribution` mode between URS 1 and URS 2, and interactions from T-Server 3 are distributed only to URS 2. For load distribution of interactions from T-Server 2, the load on URS 1 and URS 2 (from processing interactions of T-Server 1 and T-Server 3) is taken into account.

Note: Using LDS does not affect Inter Server Call Control (external) routing functionality related to Routing.

Agent Reservation Options

Using LDS to achieve redundancy or load distribution always requires two or more URSs to route interactions from multiple T-Servers. In a configuration with LDS, the probability of having two URSs requesting the same routing target is higher. As a result, you must enable the Agent Reservation feature when using LDS.

Four options are related to the Agent Reservation feature at the URS application level (see the *Universal Routing Reference Manual* and the *Universal Routing Deployment Guide* for complete information about how these options function):

- `agent_reservation`
  This option is set at the URS Application level. It instructs URS to send a `reserve agent` request to T-Server and to wait for confirmation from T-Server before routing interactions to an agent.
- `transition_time`
  This option defines the minimum time (in seconds) that URS waits between the moment an interaction is routed to a target such as an agent, a place, or a DN, and any subsequent check for routing to the same target. To avoid repeated routing to the same target, you must set a nonzero value for this option.
- `reservation_pulling_time`
  This option temporarily eliminates the regular 2-second pause cycle for URS to select each routing target. With this option enabled, URS sends a reservation request for another ready agent immediately after a negative response to the preceding request. Enabling this option increases network traffic.
- `treatment_delay_time`
  This option delays treatment if `agent_reservation` is used.

Two options are related to the Agent Reservation feature at the T-Server application level. See the appropriate *Deployment Guide* for your specific T-Server for more details:

- `reservation_time`
  This option determines the time interval (in milliseconds) to reserve an agent. During that interval the agent cannot be reserved again. Use this option to handle race conditions caused by two or more routing requests for the same target. It does not work for multidirect (for example, direct transfers between agents) or multi-ACD interactions.
- `reject-subsequent-request`
  With value `true`, T-Server rejects subsequent requests for an agent reservation from the same client application for the same `Agent` object. With value `false`, a subsequent request prolongs the current reservation made by the same client application for the same agent.

Note: You must set the value of this option to `true` in all T-Servers that are LDS clients.

Chapter

7 LDS Support (Load Distribution Mode) of Call Concentrator

Note: The information in this chapter applies only to LDS in `Load Distribution` mode.

This chapter provides recommendations on using LDS and Call Concentrator (CCon). It contains:

- Recommended Configuration, page 43

Recommended Configuration

The recommended CCon configuration in an LDS (`Load Distribution` mode) environment comprises a redundant LDS process that receives T-events from all T-Servers, then distributes them to ***N*** + 1 CCon processes. (Here, ***N*** is the total expected traffic, divided by the traffic expected to be processed by a single CCon).

Additionally, you must set alarms in Management Layer against CCon's log events that indicate a database problem. It is also desirable to configure these alarms to automatically shut down the Call Concentrator process that reported the related event. Refer to Call Concentrator and Management Layer documentation for full details on how to do this.

This configuration ensures that if any Call Concentrator is closed down in this scenario, LDS automatically redistributes the load among the remaining processes that have enough processing power to handle it. Therefore, only the interactions in progress at the time of failure are affected.

Note: Setting the LDS configuration option `no-context-distribution` to value `all` allows Call Concentrator to populate `AREC` tables for non-call–related data, including custom states. Such data is most likely to be duplicated.

Chapter

8 LDS Configuration Options and Log Messages

This chapter describes the configuration options and log messages specific to LDS. It contains the following sections:

- LDS Section, page 45
- LDS Options Configured in Receivers, page 55
- Changes from Release 7.2 to 8.1, page 56
- Log Messages, page 58

Log options common to all servers are described in the "Common Configuration Options" chapter of this document. Common log messages are described in *Genesys 8.1 Combined Log Events Help*.

LDS Section

You can find the LDS configuration options in a section called `LDS` on the `Options` tab of the `Properties` window of the LDS `Application` object. The options in this section are listed in alphabetical order.

active-context-limit

Default Value: `1000000`
Valid Value: Any integer from `1-1000000`
Changes Take Effect: Immediately

Specifies the default maximum number of active transactions that each Receiver can process simultaneously. You can also configure this option per Receiver in the `LDS` section in the Receiver application. If you do so, that value overrides the default value set in the LDS application properties.

background-processing

Default Value: `true`
Valid Values: `true, false`
Changes Take Effect: Immediately

With Background Processing functionality enabled, LDS reads all messages immediately and waits until there are no messages before processing the message queue associated with LDS client requests. LDS reads all connection sockets immediately and places client requests in the input buffer, which prevents LDS clients from disconnecting because of configured timeouts.

When LDS processes client requests from the message queue, requests are processed in the order in which LDS received them.

When set to `false`, LDS processes all requests from one LDS client before proceeding to the requests from another LDS client, and so on.

background-timeout

Default Value: `60`
Valid Value: Any integer from `1-1000`
Changes Take Effect: Immediately

With the `background-processing` functionality enabled (option `background-processing` set to `true`), this option limits how long LDS will process requests from a non-empty background queue.

cleanup-timer

Default Value: `60`
Valid Value: Any integer from `10-60`
Changes Take Effect: Immediately

Defines (in seconds) how often LDS sends query requests to T-Server to check the status of non-active transactions.

context-cleanup

Default Value: `60`
Valid Value: Any integer from `15-11520`
Changes Take Effect: Immediately

Defines the time (in minutes) that LDS waits after the last event is received for an active transaction before querying T-Server to check whether the call still exists.

context-remove-delay

Default Value: `5`
Valid Value: Any integer from `5-60`
Changes Take Effect: Immediately

Specifies the interval (in seconds) for which the `Connection ID` of a call is kept after T-Server reports `End-Of-Call`. This interval is useful for call distribution in Inter Server Call Control (ISCC).

count-active-context

Default Value: `false`
Valid Values: `true, false`
Changes Take Effect: Immediately

With value `true`, LDS counts the total number of active transactions for each Receiver and issues an alarm when the number of transactions defined in the `active-context-limit` option is reached.

distribute-mode

Default Value: `auto`
Valid Values: `auto, load, proxy`

`auto`	With value `auto`, LDS selects its own mode depending on its actual configuration environment. With LDS configured in Configuration Manager with application type `T-Server` and one T-Server added on the LDS `Connection` tab, LDS operates in `proxy` mode. For all other possible configurations, LDS operates in `Load Distribution` mode.
`load`	With value `load`, LDS is forced into `LoadDistribution` mode.
`proxy`	With value `proxy`, LDS acts as proxy between T-Server and clients

Changes Take Effect: Immediately

Specifies the default distribution mode of LDS.

Note: Genesys recommends that you do not make changes without careful preparation, because the effect on client applications of changing between `Proxy` and `LoadDistribution` modes could cause unexpected effects.

enable-safe-handover

Default Value: `false`
Valid Values: `true`, `false`
Changes Take Effect: Immediately

This option instructs LDS to distribute the last `EventRouteRequest` for the transaction on the Routing Point to the new Receiver when the last Receiver that served this transaction is disconnected prior to completion of routing

With value `true`, an EventRouteRequest distributed to the disconnected Receiver will be re-sent to the new one. With value `false`, the new Receiver will receive the next event for the transaction.

ha-dly-switchover

Default Value: `false`
Valid Values: `true, false`
Changes Take Effect: Immediately

Determines whether the backup LDS delays switching into the `Running` state for Management Layer until it has been notified that target synchronization has completed. With value `false`, the backup LDS switches into `Running` state as soon as the backup link is established.

ha-sync-level

Default Value: `1`
Valid Values:

`0`	The backup LDS does not receive events from T-Server until it becomes primary. Synchronization between primary and backup LDS is performed with each transaction. Genesys advises against using this value in production environments with high call volumes because messages can be lost in failover scenarios.
`1`	Both primary and backup LDS receive all events from T-Server. Synchronization between primary and backup LDS is performed with each transaction. Using this value increases network traffic.
`2`	Both the primary and backup LDS receive all events from T-Server. Synchronization between primary and backup LDS is performed with each T-Server event. Using this value increases network traffic as well as the resource usage of LDS.

Changes Take Effect: Immediately

Defines the level of synchronization between a primary and backup LDS application. Increasing the value of this option reduces the risk of event loss in a switchover/failover scenario, but increases network traffic.

Note: The `ha-sync-level` option applies to primary/backup LDS Applications in `hot standby` mode only, and (from release 6.5.3 onward) is automatically adapted to the LDS redundancy level. If you have configured LDS in `warm standby` mode, this option is ignored. The backup and primary LDS can start with different values set for this option (though Genesys does not recommend this), but the backup LDS automatically adopts the value configured in the primary LDS.

---

intra-cluster-distribution

Default Value: `all`
Valid Values: `one, all`
Changes Take Effect: Immediately

Specifies whether to distribute transactions to one Receiver in a cluster (value `one`), or all Receivers (value `all`).

If a nominated Receiver in such a configuration fails, LDS distributes subsequent transactions to the next Receiver as per Warm Standby (that is, no synchronization).

keep-ext-key

Default Value: No default value
Valid Values: Comma-separated list of extensions
Changes Take Effect: Immediately

Specifies the list of extensions that should be preserved when LDS is removing ConnectionID-related extensions from events `EventRegistered` and `EventAddressInfo`.

When the connection ID found in extension "connid-N" is masked for particular Receivers (where N is a positive integer greater than or equal to 0), and all extensions with a prefix of N have not been distributed to them, LDS preserves the key in the format `ExtKey-N` when `ExtKey` is listed in this configuration option.

keep-taction-stat

Default Value: `5`
Valid Value: Any integer from `1-1440`
Changes Take Effect: Immediately

Defines the length of time (in minutes) that LDS maintains information about Receiver loading in order to perform even distribution.

license-file

Default Value: No default value
Valid Value: Valid path to a valid license file
Changes Take Effect: Immediately if command-line parameter `-l` is not specified

Specifies the location of the license file from which LDS obtains the license, if the license file location is not specified on startup (using the `-l` command-line parameter).

link-alarm-high

Default Value: `0`
Valid Value: Any integer from `0-100`
Changes Take Effect: Immediately

Specifies the percentage of the `use-link-bandwidth` option when LMS message `39570` (`MSG ALARM HIGH`) is generated. When set to `0`, the alarm feature is disabled.

link-alarm-low

Default Value: `0`
Valid Value: Any integer from `0-100`
Changes Take Effect: Immediately

Specifies the percentage of the `use-link-bandwidth` option when LMS message `39571` (`MSG ALARM LOW`) is generated. When set to `0`, the alarm feature is disabled.

link-by-originator

Default Value: `false`
Valid Values: `true`, `false`
Changes Take Effect: Immediately

With value `true`, LDS attempts to link a known in-progress interaction with a new one using `ThisDN`, `OtherDN`, or `ThirdPartyDN` information from the call events when the link cannot be established using `PreviousConnID`.

load-report-interval

Default Value: `15`
Valid Value: Any integer from `0-120`
Changes Take Effect: Immediately

Specifies (in minutes) how often LDS will output the LMS message `39572` (`MSG LOAD INFO`). This LMS message contains the number of outgoing and incoming messages across all Sender links during the configured interval.

max-outstanding

Default Value: `4`
Valid Value: Any integer from `0-2048`
Changes Take Effect: Immediately

Specifies the maximum number of messages of type `RequestRegisterAddress`, `RequestQueryAddress`, and `RequestQueryCall` that can be sent to a single Sender at any given time without being acknowledged.

max-update-rate

Default Value: `100 K`
Valid Values: See description
Changes Take Effect: Immediately

Defines the settings that manage the HA link load during initial transaction synchronization between the primary and backup LDSs.

`0`	Primary LDS informs backup about the transaction target only when the transaction is either created or became active (that is, a transaction-related event is received).
Non-`0` value	In addition to updates for new and active transactions, the primary LDS will send the specified number of updates for

inactive non-synchronized transactions to the backup LDS per second.

The option is defined using format *x*<K|M|G> and the following rules apply:

- A value *x* without a suffix instructs LDS to use that value in bps.
- A value *x* with suffix K instructs LDS to use that value in kbps.
- A value *x* with suffix M instructs LDS to use that value in mbps.
- A value *x* with suffix G instructs LDS to use that value in gbps.

The value defines overall traffic on the backup link. For transaction synchronization, LDS uses available "spare" bandwidth.

Example Primary/backup synchronization for 10 cps (calls per second) and 10 messages per call where ha-sync-level = 2 generates around 100 kbps of traffic (without overhead) on an HA link.

Note: The option is effective for a primary LDS running in HA Hot-Standby mode.

msg-duplication

Default Value: 0
Valid Value: Any integer from 0-10
Changes Take Effect: Immediately

Defines the number of additional Receivers that receive duplicate event messages. With value 0 (zero), only one Receiver receives the event flow for each transaction. With value1, two Receivers receive the event flow for each transaction, and so on.

This option relates to multiple Receivers in primary mode and does not conflict with HA Receiver mode, in which the same event is sent to two Receivers.

Warning! Use extreme caution when changing the value of this option from its default setting of 0 (zero). Setting nonzero values increases availability. Please consult the documentation for your Receiver application to determine whether it supports nonzero values for this option.

no-context-distribution

Default Value: first
Valid Values: none, first, all

none	Any events that do not have a call context are not distributed to Receivers.
first	Events with no call context are distributed to the first Receiver to connect to LDS at startup. If this Receiver is disconnected, LDS selects another Receiver to receive such events.

`all`	Events with no call context are broadcast to all connected Receivers that are registered for the device in question.

Changes Take Effect: Immediately

Defines which LDS Receiver or Receivers (none, only the first Receiver, or all Receivers) should receive messages without a call context. Currently this includes all T-Server events without a `Connection ID`.

Warning! If you are using URS version 7.5 or higher and you have more than one URS registering with LDS for same Routing Point, you need to set the value of this option to `all` to make sure all non-call-related events are sent from LDS to all URSs.Please be aware that this setting will impact performance. Genesys recommends not to use value `all` if you have Call Concentrator connected to the LDS.

Note: With value `first`, the first Receiver assignment may be changed when the set of Receivers is changed; for example, when a Receiver is dynamically added or deleted.

query-dn

Default Value: `lds-query-dn`
Valid Value: Any character string
Changes Take Effect: Immediately

Specifies the name of the DN that LDS uses to perform queries for calls.

query-timer

Default Value: `2`
Valid Value: Any integer from `1-60`
Changes Take Effect: Immediately

Defines the time (in minutes) between the receipt of the last event for an active transaction and an LDS query to T-Server to check whether the call still exists.

Note: The value of `query-timer` supersedes the value of option `context-cleanup` because LDS uses a different procedure for querying calls in T-Server releases 6.5.3 and later.

Genesys recommends setting the value of `query-timer` to half (or less) that of the value set for the `context-cleanup` option.

queue-expire-timeout

Default Value: `600`
Valid Values: Any integer from `1-3600`
Changes Take Effect: Immediately

Specifies the maximum time, in seconds, that LDS can keep a request queued before deleting it.

register-guard

Default Value: `5`
Valid Values: Any integer from `0-30`
Changes Take Effect: Immediately

Defines the timeout (in seconds) between LDS issuing a `LinkConnected` event (or a consecutive `RegisterAddress` event) to the client and the beginning of distribution of transactions to this client.

register-mode

Default Value: `tproxy`
Valid Values: `tproxy, tserver`
Changes Take Effect: Immediately

Defines how LDS in `TProxy` mode will process `RequestRegisterAddress` when different clients are required to register with different `RegisterMode` values. Please consult the *T-Library SDK Developers Guide* for more information on `TRegisterMode` support in T-Servers.

With value `tproxy`, LDS will override the `RegisterMode` received from the client and always use `ModeShare` when a request is sent to the Sender.

With value `tserver`, LDS internally emulates the Genesys T-Server handling for `RegisterMode`.

Note: Applicable for LDS in `TProxy` mode only.

rq-expire-timeout

Default Value: `120`
Valid Value: Any integer from `10-600`
Changes Take Effect: Immediately

Instructs LDS to delete a request from the client, as it has not received a response after the specified timeout. LDS sends EventError (Timeout) to the client requester. Depending on the event type, LDS either (1) does not distribute the event, or (2) strips the reference ID from the event if a response eventually arrives.

server-id

Default Value: `ApplicationDBID`
Valid Values: `0...16383`
Changes Take Effect: Immediately

Specifies the Server ID that LDS uses to generate Connection IDs and other unique identifiers. In a multi-site environment, you must assign each LDS a unique Server ID, in order to avoid confusion in reporting applications and

LDS behavior. Configuration of this option is necessary for Framework environments in which there are two or more instances of the Configuration Database.

Note: If you do not specify a value for this option, LDS populates it with the `ApplicationDBID` as reported by Configuration Server. Each data object in the Configuration Database is assigned a separate `DBID` that maintains a unique Server ID for each LDS configured in the database.

stat-calc-threshold

Default Value: `1`
Valid Value: Any integer from `0-100`
Changes Take Effect: Immediately

Specifies the frequency with which LDS recalculates internal statistics to sort available Receivers according to their loading. Value `1` means LDS recalculates with every transaction; value `10`, with every 10 transactions, and so on. The higher the value, the less frequent the recalculation.

strict-backup-name

Default Value: `false`
Valid Values: `true, false`
Changes Take Effect: Immediately

With value `true`, LDS does not accept clients requests for connection to non-running (or passive) T-Servers. With value `false`, LDS does accept such requests (pre-7.1 behavior).

tlib-verbose

Default Value: `0`
Valid Values: `0..2`
Changes Take Effect: Immediately

Enables T-Library debugging messages in the LDS log.

update-timestamp

Default Value: `false`
Valid Values: `true, false`
Changes Take Effect: Immediately

Defines if LDS uses the current LDS time to update the `timestamp` attribute in all events sent to Receivers. With value `false`, all LDS clients receive the T-Server timestamp in events.

Note: LDS does not perform any kind of conversion between time zones.

use-link-bandwidth

Default Value: `0`
Valid Value: Any integer from `0-999`
Changes Take Effect: Immediately

Specifies the number of messages per second throughput to indicate the maximum throughput. When set to `0`, this feature is disabled.

use-query-call

Default Value: `true`
Valid Values: `true, false`
Changes Take Effect: Immediately

Specifies whether, when working with Senders with no "call-end-support," LDS uses `QueryCall` to verify the existence of the call in the Sender (where supported).

With value `false`, no `QueryCall` request is made to the Sender.

LDS Options Configured in Receivers

Set the following three configuration options in the `LDS` section of the relevant Receiver application. The section name within the Receiver application must be `LDS`.

active-context-limit

Default Value: Value of the `active-context-limit` option in the LDS application
Valid Value: Any integer from `1-1000000`
Changes Take Effect: Immediately

Defines the maximum number of active transactions a specific Receiver is able to process simultaneously. This value overrides the value defined by the option `active-context-limit` in the default section in the LDS application for a specific Receiver.

group-id

Default Value: `0`
Valid Value: Any integer from `0-65535`
Changes Take Effect: At LDS restart

Specifies whether this Receiver is part of a group. When you configure two or more Receivers with the same value for this option, LDS treats them as a single group and they receive the same event set.

Configure this option with caution. The fact that grouped Receivers receive identical call information has potential impacts on Receiver functionality; for example, duplicated records in Call Concentrator, calls being routed by two different URSs, and so on).

Note: You must ensure that both Receivers in the Configuration Manager primary/backup pair have the same value for the `group-id` option (if the option is configured). From release 7.1, the Configuration Manager primary/backup setting takes precedence.

loading-coefficient

Default Value: `100`
Valid Value: Any integer from `0-100`
Changes Take Effect: Immediately

Defines the relative loading coefficient for each specific Receiver for `Weighted Round Robin (WRR)` mode. For example, a set of three identical Receivers configured with loading coefficients `100`, `70`, and `30` receive 50 percent, 35 percent and 15 percent, respectively, of the total number of transactions.

If you use the default value (`100`) for all Receivers (`100`), you disable `WRR` and transactions are distributed in `Load Distribution` mode. If you set any non-default value for any one Receiver (or Receiver group), you enable `WRR`.

Note: With value `0` (zero), a Receiver is excluded from transaction distribution. However, distribution to this Receiver is still possible, either when no other targets are available (for example, there are no more Receivers, or no more Receivers registered on a specific resource) or when this Receiver is available for non-context distribution.

Changes from Release 7.2 to 8.1

Table 4 on page 57 lists the configuration options that:

- Are new or changed in the 8.1 release of Load Distribution Server
- Have been added or changed since the most recent 7.2 release of this document

If a configuration option has been replaced with another that enables the same functionality, the new option name and its location in this chapter are noted.

Table 4: Option Changes from 7.2 to 8.1

Option Name	Option Values	Type of Change	Details
LDS Section			
background-processing	true, false	See Details	Default value changed to true. See the option description on page 46.
enable-safe-handover	true, false	New option in 8.1	See the option description on page 47.
link-alarm-high	0-100	New option in 8.1	See the option description on page 49.
link-alarm-low	0-100	New option in 8.1	See the option description on page 50.
load-report-interval	0-120	New option in 8.1	See the option description on page 50.
max-outstanding	0-2048	New option in 8.1	See the option description on page 50.
queue-expire-timeout	1-3600	New option in 8.1	See the option description on page 52.
server-id	0-16383	New option in 8.1	See the option description on page 53.
tlib-verbose	0-2	Option first added in 7.2.1	See the option description on page 54.
use-link-bandwidth	0-999	New option in 8.1	See the option description on page 55.

Log Messages

Table 5 lists the log messages specific to LDS.

Table 5: LDS Log Messages

Code	Type	Message
39500	STANDARD	The same Receiver [*text*] already connected to sender [*text*]
39501	STANDARD	Non T-Lib Receiver rejected
39502	STANDARD	Unknown Receiver [*text*] cannot be accepted
39503	STANDARD	Requested Sender [*text*] rejects connection
39504	STANDARD	Requested Sender [*text*] is not available
39505	STANDARD	Requested Sender [*text*] is not supported
39506	STANDARD	Requested Sender [*text*] unknown
39507	STANDARD	Unsupported Receiver type [*text*]-[*text*]
39508	STANDARD	Receiver [*text*]-[*text*] type conflict found, expected [*text*]
39510	TRACE	Sender [*text*] disabled
39511	STANDARD	Sender [*text*] removed
39512	STANDARD	Sender [*text*] connected
39513	STANDARD	Sender [*text*] disconnected
39514	TRACE	Sender [*text*] HA synchronization queue overflow
39515	TRACE	Sender [*text*] HA synchronization queue overflow end
39516	STANDARD	Sender [*text*] : no response on request [*text*] from client [*text*]
39517	STANDARD	Sender [*text*] : request [*text*] from client [*text*] expired
39520	TRACE	New transaction [*text*]
39521	TRACE	Transaction [*text*] sender changed
39522	TRACE	Transaction [*text*] disabled

Table 5: LDS Log Messages (Continued)

Code	Type	Message
39523	TRACE	Transaction [*text*] timeout expired
39524	STANDARD	Transaction [*text*] *2N+M* distribution detected
39525	STANDARD	Client [*text*] not registered on DN [*text*]. Transaction [*text*] event lost
39526	STANDARD	Transaction [*text*] event lost
39527	TRACE	Transaction [*text*] receiver changed
39528	TRACE	Transaction [*text*] receiver [*text*] disconnected
39530	TRACE	Unknown response from sender [*text*]
39531	STANDARD	Make Call request from Receiver [*text*] not supported
39532	STANDARD	Number of active transactions (number) on Receiver [*text*] is exceeded (number)
39533	STANDARD	Impossible to select the target
39534	STANDARD	No clients for Transaction [*text*]
39535	STANDARD	Event [*text*] with reference id [*text*] is discarded - no client-requestor
39536	STANDARD	Event [*text*] : removing reference ID [*text*] - no client-requestor
39540	STANDARD	Configuration option ha-sync-level is different for primary/backup LDS pair
39541	STANDARD	Backup Link configuration failure
39550	STANDARD	Equal weighting profile ON (that is, WRR is not active)
39551	STANDARD	Varied weighting profile ON (that is, WRR is active)
39552	STANDARD	Backup LDS requires transaction synchronization
39553	STANDARD	Backup LDS transaction synchronization complete
39554	STANDARD	LDS role changed to [*text*]

Table 5: LDS Log Messages (Continued)

Code	Type	Message
39555	STANDARD	Invalid (unknown) role for LDS was requested
39556	STANDARD	HA synchronization queue overflow
39557	STANDARD	HA synchronization queue overflow end
39570	STANDARD	Link bandwidth: *[text]* requests per second exceeded alarm threshold *[text]* messages per second
39571	STANDARD	Link bandwidth: *[text]* requests per second dropped below alarm threshold *[text]* messages per second
39572	STANDARD	Link bandwidth: *[text]* requests and *[text]* events per *[text]* minutes

Chapter

9 Common Configuration Options

Unless otherwise noted, the common configuration options that this chapter describes are common to all Genesys server applications and applicable to any Framework server component. This chapter includes the following sections:

- Setting Configuration Options, page 61
- Mandatory Options, page 62
- log Section, page 62
- log-extended Section, page 76
- log-filter Section, page 78
- log-filter-data Section, page 78
- security Section, page 79
- sml Section, page 79
- common Section, page 81

Note: Some server applications also support log options that are unique to them. For descriptions of a particular application's unique log options, refer to the chapter/document about that application.

Setting Configuration Options

Unless specified otherwise, set common configuration options in the `Options` of the `Application` object, using one of the following navigation paths:

- In Genesys Administrator—`Application` object > `Options` tab > `Advanced View (Options)`
- In Configuration Manager—`Application` object > `Properties` dialog box > `Options` tab

Warning! Configuration section names, configuration option names, and predefined option values are case-sensitive. Type them in Genesys Administrator or Configuration Manager exactly as they are documented in this chapter.

Mandatory Options

You do not have to configure any common options to start Server applications.

log Section

This section must be called `log`.

verbose

Default Value: `all`
Valid Values:

`all`	All log events (that is, log events of the Standard, Trace, Interaction, and Debug levels) are generated.
`debug`	The same as `all`.
`trace`	Log events of the Trace level and higher (that is, log events of the Standard, Interaction, and Trace levels) are generated, but log events of the Debug level are not generated.
`interaction`	Log events of the Interaction level and higher (that is, log events of the Standard and Interaction levels) are generated, but log events of the Trace and Debug levels are not generated.
`standard`	Log events of the Standard level are generated, but log events of the Interaction, Trace, and Debug levels are not generated.
`none`	No output is produced.

Changes Take Effect: Immediately

Determines whether a log output is created. If it is, specifies the minimum level of log events generated. The log events levels, starting with the highest priority level, are Standard, Interaction, Trace, and Debug. See also "Log Output Options" on page 68.

Note: For definitions of the Standard, Interaction, Trace, and Debug log levels, refer to the *Framework 8.0 Management Layer User's Guide*, *Framework 8.1 Genesys Administrator Help,* or to *Framework 8.0 Solution Control Interface Help.*

buffering

Default Value: `true`

Valid Values:

`true`	Enables buffering.
`false`	Disables buffering.

Changes Take Effect: Immediately

Turns on/off operating system file buffering. The option is applicable only to the `stderr` and `stdout` output (see page 68). Setting this option to `true` increases the output performance.

Note: When buffering is enabled, there might be a delay before log messages appear at the console.

segment

Default Value: `false`

Valid Values:

`false`	No segmentation is allowed.
`<number> KB` or `<number>`	Sets the maximum segment size, in kilobytes. The minimum segment size is `100 KB`.
`<number> MB`	Sets the maximum segment size, in megabytes.
`<number> hr`	Sets the number of hours for the segment to stay open. The minimum number is `1` hour.

Changes Take Effect: Immediately

Specifies whether there is a segmentation limit for a log file. If there is, sets the mode of measurement, along with the maximum size. If the current log segment exceeds the size set by this option, the file is closed and a new one is created. This option is ignored if log output is not configured to be sent to a log file.

expire

Default Value: `false`

Valid Values:

`false`	No expiration; all generated segments are stored.
`<number> file` or `<number>`	Sets the maximum number of log files to store. Specify a number from `1`–`1000`.
`<number> day`	Sets the maximum number of days before log files are deleted. Specify a number from `1`–`100`.

Changes Take Effect: Immediately

Determines whether log files expire. If they do, sets the measurement for determining when they expire, along with the maximum number of files (segments) or days before the files are removed. This option is ignored if log output is not configured to be sent to a log file.

Note: If an option's value is set incorrectly—out of the range of valid values— it will be automatically reset to `10`.

keep-startup-file

Default Value: `false`

Valid Values:

`false`	No startup segment of the log is kept.
`true`	A startup segment of the log is kept. The size of the segment equals the value of the `segment` option.
`<number> KB`	Sets the maximum size, in kilobytes, for a startup segment of the log.
`<number> MB`	Sets the maximum size, in megabytes, for a startup segment of the log.

Changes Take Effect: After restart

Specifies whether a startup segment of the log, containing the initial T-Server configuration, is to be kept. If it is, this option can be set to `true` or to a specific size. If set to `true`, the size of the initial segment will be equal to the size of the regular log segment defined by the `segment` option. The value of this option will be ignored if segmentation is turned off (that is, if the `segment` option set to `false`).

Note: This option applies only to T-Servers.

messagefile

Default Value: As specified by a particular application

Valid Values: `<string>.lms` (message file name)

Changes Take Effect: Immediately, if an application cannot find its `*.lms` file at startup

Specifies the file name for application-specific log events. The name must be valid for the operating system on which the application is running. The option value can also contain the absolute path to the application-specific `*.lms` file. Otherwise, an application looks for the file in its working directory.

Warning! An application that does not find its `*.lms` file at startup cannot generate application-specific log events and send them to Message Server.

message-format

Default Value: `short`
Valid Values:

`short`	An application uses compressed headers when writing log records in its log file.
`full`	An application uses complete headers when writing log records in its log file.

Changes Take Effect: Immediately

Specifies the format of log record headers that an application uses when writing logs in the log file. Using compressed log record headers improves application performance and reduces the log file's size.

With the value set to `short`:

- A header of the log file or the log file segment contains information about the application (such as the application name, application type, host type, and time zone), whereas single log records within the file or segment omit this information.
- A log message priority is abbreviated to `Std`, `Int`, `Trc`, or `Dbg`, for Standard, Interaction, Trace, or Debug messages, respectively.
- The message ID does not contain the prefix `GCTI` or the application type `ID`.

A log record in the full format looks like this:

```
2002-05-07T18:11:38.196 Standard localhost cfg_dbserver GCTI-00-05060
Application started
```

A log record in the short format looks like this:

```
2002-05-07T18:15:33.952 Std 05060 Application started
```

Note: Whether the full or short format is used, time is printed in the format specified by the `time_format` option.

time_convert

Default Value: `Local`
Valid Values:

`local`	The time of log record generation is expressed as a local time, based on the time zone and any seasonal adjustments. Time zone information of the application's host computer is used.
`utc`	The time of log record generation is expressed as Coordinated Universal Time (UTC).

Changes Take Effect: Immediately

Specifies the system in which an application calculates the log record time when generating a log file. The time is converted from the time in seconds since the Epoch (00:00:00 UTC, January 1, 1970).

time_format

Default Value: `time`
Valid Values:

`time`	The time string is formatted according to the `HH:MM:SS.sss` (hours, minutes, seconds, and milliseconds) format.
`locale`	The time string is formatted according to the system's locale.
`ISO8601`	The date in the time string is formatted according to the ISO 8601 format. Fractional seconds are given in milliseconds.

Changes Take Effect: Immediately

Specifies how to represent, in a log file, the time when an application generates log records.

A log record's time field in the ISO 8601 format looks like this:

```
2001-07-24T04:58:10.123
```

print-attributes

Default Value: `false`
Valid Values:

`true`	Attaches extended attributes, if any exist, to a log event sent to log output.
`false`	Does not attach extended attributes to a log event sent to log output.

Changes Take Effect: Immediately

Specifies whether the application attaches extended attributes, if any exist, to a log event that it sends to log output. Typically, log events of the Interaction log level and Audit-related log events contain extended attributes. Setting this option to `true` enables audit capabilities, but negatively affects performance. Genesys recommends enabling this option for Solution Control Server and Configuration Server when using audit tracking. For other applications, refer to *Genesys 8.0 Combined Log Events Help* to find out whether an application generates Interaction-level and Audit-related log events; if it does, enable the option only when testing new interaction scenarios.

check-point

Default Value: `1`
Valid Values: `0`–`24`
Changes Take Effect: Immediately

Specifies, in hours, how often the application generates a `check point` log event, to divide the log into sections of equal time. By default, the application generates this log event every hour. Setting the option to `0` prevents the generation of `check-point` events.

memory

Default Value: No default value
Valid Values: `<string>` (memory file name)
Changes Take Effect: Immediately

Specifies the name of the file to which the application regularly prints a snapshot of the memory output, if it is configured to do this (see “Log Output Options” on page 68). The new snapshot overwrites the previously written data. If the application terminates abnormally, this file will contain the latest log messages. Memory output is not recommended for processors with a CPU frequency lower than 600 MHz.

Note: If the file specified as the `memory` file is located on a network drive, an application does not create a snapshot file (with the extension `*.memory.log`).

memory-storage-size

Default Value: `2 MB`
Valid Values:

`<number> KB` or `<number>`	The size of the memory output, in kilobytes. The minimum value is `128 KB`.
`<number> MB`	The size of the memory output, in megabytes. The maximum value is `64 MB`.

Changes Take Effect: When memory output is created

Specifies the buffer size for log output to the memory, if configured. See also “Log Output Options” on page 68.

spool

Default Value: The application’s working directory
Valid Values: `<path>` (the folder, with the full path to it)
Changes Take Effect: Immediately

Specifies the folder, including full path to it, in which an application creates temporary files related to network log output. If you change the option value while the application is running, the change does not affect the currently open network output.

compatible-output-priority

Default Value: `false`
Valid Values:

`true`	The log of the level specified by “Log Output Options” is sent to the specified output.
`false`	The log of the level specified by “Log Output Options” and higher levels is sent to the specified output.

Changes Take Effect: Immediately

Specifies whether the application uses 6.x output logic. For example, you configure the following options in the `log` section for a 6.x application and for a 7.x application:

```
[log]
verbose = all
debug = file1
standard = file2
```

The log file content of a 6.x application is as follows:

- `file1` contains Debug messages only.
- `file2` contains Standard messages only.

The log file content of a 7.x application is as follows:

- `file1` contains Debug, Trace, Interaction, and Standard messages.
- `file2` contains Standard messages only.

If you set `compatible-output-priority` to `true` in the 7.x application, its log file content will be the same as for the 6.x application.

Warning! Genesys does not recommend changing the default value of this option unless you have specific reasons to use the 6.x log output logic—that is, to mimic the output priority as implemented in releases 6.x. Setting this option to `true` affects log consistency.

Log Output Options

To configure log outputs, set log level options (`all`, `alarm`, `standard`, `interaction`, `trace`, and/or `debug`) to the desired types of log output (`stdout`, `stderr`, `network`, `memory`, and/or `[filename]`, for log file output).

You can use:

- One log level option to specify different log outputs.
- One log output type for different log levels.
- Several log output types simultaneously, to log events of the same or different log levels.

You must separate the log output types by a comma when you are configuring more than one output for the same log level. See "Examples" on page 72.

Warnings!
- If you direct log output to a file on the network drive, an application does not create a snapshot log file (with the extension `*.snapshot.log`) in case it terminates abnormally.
- Directing log output to the console (by using the `stdout` or `stderr` settings) can affect application performance. Avoid using these log output settings in a production environment.

Note: The log output options are activated according to the setting of the `verbose` configuration option.

all

Default Value: No default value
Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`network`	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database. Setting the `all` log level option to the `network` output enables an application to send log events of the `Standard`, `Interaction`, and `Trace` levels to Message Server. `Debug`-level log events are neither sent to Message Server nor stored in the Log Database.
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends all log events. The log output types must be separated by a comma when more than one output is configured. For example:

```
all = stdout, logfile
```

Note: To ease the troubleshooting process, consider using unique names for log files that different applications generate.

alarm

Default Value: No default value
Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`network`	Log events are sent to Message Server, which resides anywhere on the network, and Message Server stores the log events in the Log Database.
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the Alarm level. The log output types must be separated by a comma when more than one output is configured. For example:

```
standard = stderr, network
```

standard

Default Value: No default value
Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`network`	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the `Standard` level. The log output types must be separated by a comma when more than one output is configured. For example:

```
standard = stderr, network
```

interaction

Default Value: No default value
Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`network`	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the `Interaction` level and higher (that is, log events of the `Standard` and `Interaction` levels). The log outputs must be separated by a comma when more than one output is configured. For example:

```
interaction = stderr, network
```

trace

Default Value: No default value
Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`network`	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the `Trace` level and higher (that is, log events of the `Standard`, `Interaction`, and `Trace` levels). The log outputs must be separated by a comma when more than one output is configured. For example:

```
trace = stderr, network
```

debug

Default Value: No default value
Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the `Debug` level and higher (that is, log events of the `Standard`, `Interaction`, `Trace`, and `Debug` levels). The log output types must be separated by a comma when more than one output is configured—for example:

```
debug = stderr, /usr/local/genesys/logfile
```

Note: `Debug`-level log events are never sent to Message Server or stored in the Log Database.

Log File Extensions

You can use the following file extensions to identify log files that an application creates for various types of output:

- `*.log`—Assigned to log files when you configure output to a log file. For example, if you set `standard = confservlog` for Configuration Server, it prints log messages into a text file called `confservlog.<time_stamp>.log`.
- `*.qsp`—Assigned to temporary (spool) files when you configure output to the network but the network is temporarily unavailable. For example, if you set `standard = network` for Configuration Server, it prints log messages into a file called `confserv.<time_stamp>.qsp` during the time the network is not available.
- `*.snapshot.log`—Assigned to files that contain the output snapshot when you configure output to a log file. The file contains the last log messages that an application generates before it terminates abnormally. For example, if you set `standard = confservlog` for Configuration Server, it prints the last log message into a file called `confserv.<time_stamp>.snapshot.log` in case of failure.

  Note: Provide `*.snapshot.log` files to Genesys Technical Support when reporting a problem.

- `*.memory.log`—Assigned to log files that contain the memory output snapshot when you configure output to memory and redirect the most recent memory output to a file. For example, if you set `standard = memory` and `memory = confserv` for Configuration Server, it prints the latest memory output to a file called `confserv.<time_stamp>.memory.log`.

Examples

This section presents examples of a `log` section that you might configure for an application when that application is operating in production mode and in two lab modes, debugging and troubleshooting.

Production Mode Log Section

```
[log]
verbose = standard
standard = network, logfile
```

With this configuration, an application only generates the log events of the `Standard` level and sends them to Message Server, and to a file named `logfile`, which the application creates in its working directory. Genesys recommends that you use this or a similar configuration in a production environment.

Warning! Directing log output to the console (by using the `stdout` or `stderr` settings) can affect application performance. Avoid using these log output settings in a production environment.

Lab Mode Log Section

```
[log]
verbose = all
all = stdout, /usr/local/genesys/logfile
trace = network
```

With this configuration, an application generates log events of the `Standard`, `Interaction`, `Trace`, and `Debug` levels, and sends them to the standard output and to a file named `logfile`, which the application creates in the `/usr/local/genesys/` directory. In addition, the application sends log events of the `Standard`, `Interaction`, and `Trace` levels to Message Server. Use this configuration to test new interaction scenarios in a lab environment.

Failure-Troubleshooting Log Section

```
[log]
verbose = all
standard = network
all = memory
memory = logfile
memory-storage-size = 32 MB
```

With this configuration, an application generates log events of the `Standard` level and sends them to Message Server. It also generates log events of the `Standard`, `Interaction`, `Trace`, and `Debug` levels, and sends them to the memory output. The most current log is stored to a file named `logfile`, which the application creates in its working directory. Increased memory storage allows an application to save more of the log information generated before a failure.

Note: If you are running an application on UNIX, and you do not specify any files in which to store the memory output snapshot, a core file that the application produces before terminating contains the most current application log. Provide the application's core file to Genesys Technical Support when reporting a problem.

Debug Log Options

The options in this section enable you to generate `Debug` logs containing information about specific operations of an application.

x-conn-debug-open

Default Value: `0`
Valid Values:

`0`	Log records are not generated.
`1`	Log records are generated.

Changes Take Effect: After restart

Generates `Debug` log records about "open connection" operations of the application.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-select

Default Value: `0`
Valid Values:

`0`	Log records are not generated.
`1`	Log records are generated.

Changes Take Effect: After restart

Generates `Debug` log records about "socket select" operations of the application.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-timers

Default Value: `0`
Valid Values:

`0`	Log records are not generated.
`1`	Log records are generated.

Changes Take Effect: After restart

Generates `Debug` log records about the timer creation and deletion operations of the application.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-write

Default Value: `0`
Valid Values:

`0`	Log records are not generated.
`1`	Log records are generated.

Changes Take Effect: After restart

Generates `Debug` log records about "write" operations of the application.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-security

Default Value: `0`

Valid Values:

`0`	Log records are not generated.
`1`	Log records are generated.

Changes Take Effect: After restart

Generates `Debug` log records about security-related operations, such as Transport Layer Security and security certificates.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-api

Default Value: `0`

Valid Values:

`0`	Log records are not generated.
`1`	Log records are generated.

Changes Take Effect: After restart

Generates `Debug` log records about connection library function calls.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-dns

Default Value: `0`

Valid Values:

`0`	Log records are not generated.
`1`	Log records are generated.

Changes Take Effect: After restart

Generates `Debug` log records about DNS operations.

Warning! Use this option only when requested by Genesys Technical Support.

x-conn-debug-all

Default Value: `0`
Valid Values:

`0`	Log records are not generated.
`1`	Log records are generated.

Changes Take Effect: After restart

Generates `Debug` log records about open connection, socket select, timer creation and deletion, write, security-related, and DNS operations, and connection library function calls. This option is the same as enabling or disabling all of the previous `x-conn-debug-<op type>` options.

Warning! Use this option only when requested by Genesys Technical Support.

log-extended Section

This section must be called `log-extended`.

level-reassign-<eventID>

Default Value: Default value of log event `<eventID>`
Valid Values:

`alarm`	The log level of log event `<eventID>` is set to `Alarm`.
`standard`	The log level of log event `<eventID>` is set to `Standard`.
`interaction`	The log level of log event `<eventID>` is set to `Interaction`.
`trace`	The log level of log event `<eventID>` is set to `Trace`.
`debug`	The log level of log event `<eventID>` is set to `Debug`.
`none`	Log event `<eventID>` is not recorded in a log.

Changes Take Effect: Immediately

Specifies a log level for log event `<eventID>` that is different than its default level, or disables log event `<eventID>` completely. If no value is specified, the log event retains its default level. This option is useful when you want to customize the log level for selected log events.

These options can be deactivated with the option `level-reassign-disable`.

Warning! Use caution when making these changes in a production environment.

Depending on the log configuration, changing the log level to a higher priority may cause the log event to be logged more often or to a greater number of outputs. This could affect system performance.

Likewise, changing the log level to a lower priority may cause the log event to be not logged at all, or to be not logged to specific outputs, thereby losing important information. The same applies to any alarms associated with that log event.

In addition to the preceding warning, take note of the following:

- Logs can be customized only by release 7.6 or later applications.
- When the log level of a log event is changed to any level except `none`, it is subject to the other settings in the `[log]` section at its new level. If set to `none`, it is not logged and is therefore not subject to any log configuration.
- Using this feature to change the log level of a log changes only its priority; it does not change how that log is treated by the system. For example, increasing the priority of a log to `Alarm` level does not mean that an alarm will be associated with it.
- Each application in a High Availability (HA) pair can define its own unique set of log customizations, but the two sets are not synchronized with each other. This can result in different log behavior depending on which application is currently in primary mode.
- This feature is not the same as a similar feature in Universal Routing Server (URS) release 7.2 or later. In this Framework feature, the priority of log events are customized. In the URS feature, the priority of debug messages only are customized. Refer to the *Universal Routing Reference Manual* for more information about the URS feature.
- You cannot customize any log event that is not in the unified log record format. Log events of the `Alarm`, `Standard`, `Interaction`, and `Trace` levels feature the same unified log record format.

Example

This is an example of using customized log level settings, subject to the following log configuration:

```
[log]
verbose=interaction
all=stderr
interaction=log_file
standard=network
```

Before the log levels of the log are changed:

- Log event 1020, with default level `standard`, is output to `stderr` and `log_file`, and sent to Message Server.
- Log event 2020, with default level `standard`, is output to `stderr` and `log_file`, and sent to Message Server.
- Log event 3020, with default level `trace`, is output to `stderr`.
- Log event 4020, with default level `debug`, is output to `stderr`.

Extended log configuration section:

```
[log-extended]
level-reassign-1020=none
level-reassign-2020=interaction
level-reassign-3020=interaction
level-reassign-4020=standard
```

After the log levels are changed:

- Log event 1020 is disabled and not logged.
- Log event 2020 is output to `stderr` and `log_file`.
- Log event 3020 is output to `stderr` and `log_file`.
- Log event 4020 is output to `stderr` and `log_file`, and sent to Message Server.

level-reassign-disable

Default Value: `false`
Valid Values: `true, false`
Changes Take Effect: Immediately

When this option is set to `true`, the original (default) log level of all log events in the `[log-extended]` section are restored. This option is useful when you want to use the default levels, but not delete the customization statements.

log-filter Section

The `log-filter` section contains configuration options used to define the default treatment of filtering data in log output. This section contains one configuration option, `default-filter-type`. Refer to the chapter "Hide Selected Data in Logs" in the *Genesys 8.x Security Deployment Guide* for complete information about this option.

log-filter-data Section

The `log-filter-data` section contains configuration options used to define the treatment of filtering data in log output on a key-by-key basis. This section contains one configuration option in the form of `<key name>`. Refer to the

chapter "Hide Selected Data in Logs" in the *Genesys 8.x Security Deployment Guide* for complete information about this option.

security Section

The `security` section contains configuration options used to specify security elements for your system. In addition to other options that may be required by your application, this section contains the configuration option `disable-rbac`, which is used to enable or disable Role-Based Access Control for an application. Refer to the chapter "Role-Based Access Control" in the *Genesys 8.x Security Deployment Guide* for complete information about this option.

sml Section

This section must be called `sml`.

Options in this section are defined in the `Annex` of the `Application` object, as follows:

- in Genesys Administrator—`Application` object > `Options` tab > `Advanced View (Annex)`
- in Configuration Manager— `Application` object > `Properties` dialog box > `Annex` tab

Warning! Use the first three options in this section (`heartbeat-period`, `heartbeat-period-thread-class-<n>`, and `hangup-restart`) with great care, and only with those applications of which support for this functionality has been announced. Failure to use these options properly could result in unexpected behavior, from ignoring the options to an unexpected restart of the application.

heartbeat-period

Default Value: None

Valid Values:

`0`	This method of detecting an unresponsive application is not used by this application.
`3-604800`	Length of timeout, in seconds; equivalent to 3 seconds–7 days.

Changes Take Effect: Immediately

Specifies the maximum amount of time, in seconds, in which heartbeat messages are expected from an application. If Local Control Agent (LCA) does not receive a heartbeat message from the application within this period, it assumes the application is not responding and carries out corrective action.

This option can also be used to specify the maximum heartbeat interval for threads registered with class zero (0). This thread class is reserved for use by the Management Layer only.

If this option is not configured or is set to zero (`0`), heartbeat detection is not used by this application.

heartbeat-period-thread-class-<n>

Default Value: None

Valid Values:

`0`	Value specified by `heartbeat-period` in application is used.
`3-604800`	Length of timeout, in seconds; equivalent to 3 seconds–7 days.

Changes Take Effect: Immediately

Specifies the maximum amount of time, in seconds, in which heartbeat messages are expected from a thread of class <n> registered by an application. If a heartbeat message from the thread is not received within this period, the thread is assumed to be not responding, and therefore, the application is unable to provide service.

If this option is not configured or is set to zero (`0`), but the application has registered one or more threads of class <n>, the value specified by the value of `heartbeat-period` for the application will also be applied to these threads.

Refer to application-specific documentation to determine what thread classes, if any, are used.

hangup-restart

Default Value: `true`

Valid Values: `true`, `false`

Changes Take Effect: Immediately

If set to `true` (the default), specifies that LCA is to restart the unresponsive application immediately, without any further interaction from Solution Control Server.

If set to `false`, specifies that LCA is only to generate a notification that the application has stopped responding.

suspending-wait-timeout

Default Value: `10`

Valid Values: `5-600`

Changes Take Effect: Immediately

Specifies a timeout (in seconds) after the Stop Graceful command is issued to an application during which the status of the application should change to `Suspending` if the application supports graceful shutdown. If the status of the application does not change to `Suspending` before the timeout expires, it is assumed that the application does not support graceful shutdown, and it is stopped ungracefully.

Use this option if you are unsure whether the Application supports graceful shutdown.

Note: Genesys recommends that you do not set this option for any Management Layer component (Configuration Server, Message Server, Solution Control Server, or SNMP Master Agent) or any DB Server. These components by definition do not support graceful shutdown, so this option is not required.

common Section

This section must be called `common`.

enable-async-dns

Default Value: `off`
Valid Values:

`off`	Disables asynchronous processing of DNS requests.
`on`	Enables asynchronous processing of DNS requests.

Changes Take Effect: Immediately

Enables the asynchronous processing of DNS requests such as, for example, host-name resolution.

Warnings!
- Use this option only when requested by Genesys Technical Support.

rebind-delay

Default Value: `10`
Valid Values: `0`–`600`
Changes Take Effect: After restart

Specifies the delay, in seconds, between socket-bind operations that are being executed by the server. Use this option if the server has not been able to successfully occupy a configured port.

Warning! Use this option only when requested by Genesys Technical Support.

Related Documentation Resources

The following resources provide additional information that is relevant to this software. Consult these additional resources as necessary.

Solution Documentation

- Documentation for the solution with which you are using LDS (Routing or Reporting).

Genesys

- *Genesys Technical Publications Glossary,* which ships on the Genesys Documentation Library DVD and which provides a comprehensive list of the Genesys and computer-telephony integration (CTI) terminology and acronyms used in this document.
- *Genesys Migration Guide*, which ships on the Genesys Documentation Library DVD, and which provides documented migration strategies for Genesys product releases. Contact Genesys Technical Support for more information.
- Release Notes and Product Advisories for this product, which are available on the Genesys Technical Support website at http://genesyslab.com/support.

Information about supported hardware and third-party software is available on the Genesys Technical Support website in the following documents:

- *Genesys Supported Operating Environment Reference Manual*
- *Genesys Supported Media Interfaces Reference Manual*

Consult these additional resources as necessary:

- *Genesys Hardware Sizing Guide,* which provides information about Genesys hardware sizing guidelines for Genesys 8.x releases.

- *Genesys Interoperability Guide,* which provides information on the compatibility of Genesys products with various Configuration Layer Environments; Interoperability of Reporting Templates and Solutions; and Gplus Adapters Interoperability.
- *Genesys Licensing Guide,* which introduces you to the concepts, terminology, and procedures relevant to the Genesys licensing system.

For additional system-wide planning tools and information, see the release-specific listings of System Level Documents on the Genesys Technical Support website, accessible from the system level documents by release tab in the Knowledge Base Browse Documents Section.

Genesys product documentation is available on the:

- Genesys Technical Support website at http://genesyslab.com/support.
- Genesys Documentation Library DVD, which you can order by e-mail from Genesys Order Management at orderman@genesyslab.com.

Document Conventions

This document uses certain stylistic and typographical conventions—introduced here—that serve as shorthands for particular kinds of information.

Document Version Number

A version number appears at the bottom of the inside front cover of this document. Version numbers change as new information is added to this document. Here is a sample version number:

81fr_us_lds_10-2011_v8.1.001.00

You will need this number when you are talking with Genesys Technical Support about this product.

Screen Captures Used in This Document

Screen captures from the product graphical user interface (GUI), as used in this document, may sometimes contain minor spelling, capitalization, or grammatical errors. The text accompanying and explaining the screen captures corrects such errors *except* when such a correction would prevent you from installing, configuring, or successfully using the product. For example, if the name of an option contains a usage error, the name would be presented exactly as it appears in the product GUI; the error would not be corrected in any accompanying text.

Type Styles

Table 6 describes and illustrates the type conventions that are used in this document.

Table 6: Type Styles

Type Style	Used For	Examples
Italic	• Document titles • Emphasis • Definitions of (or first references to) unfamiliar terms • Mathematical variables Also used to indicate placeholder text within code samples or commands, in the special case where angle brackets are a required part of the syntax (see the note about angle brackets on page 86).	Please consult the *Genesys Migration Guide* for more information. Do *not* use this value for this option. A *customary and usual* practice is one that is widely accepted and used within a particular industry or profession. The formula, $x + 1 = 7$ where x stands for . . .
Monospace font (Looks like `teletype` or `typewriter text`)	All programming identifiers and GUI elements. This convention includes: • The *names* of directories, files, folders, configuration objects, paths, scripts, dialog boxes, options, fields, text and list boxes, operational modes, all buttons (including radio buttons), check boxes, commands, tabs, CTI events, and error messages. • The values of options. • Logical arguments and command syntax. • Code samples. Also used for any text that users must manually enter during a configuration or installation procedure, or on a command line.	Select the `Show variables on screen` check box. In the `Operand` text box, enter your formula. Click `OK` to exit the `Properties` dialog box. T-Server distributes the error messages in `EventError` events. If you select `true` for the `inbound-bsns-calls` option, all established inbound calls on a local agent are considered business calls. Enter `exit` on the command line.
Square brackets ([ ])	A particular parameter or value that is optional within a logical argument, a command, or some programming syntax. That is, the presence of the parameter or value is not required to resolve the argument, command, or block of code. The user decides whether to include this optional information.	`smcp_server -host [/flags]`
Angle brackets (< >)	A placeholder for a value that the user must specify. This might be a DN or a port number specific to your enterprise. **Note:** In some cases, angle brackets are required characters in code syntax (for example, in XML schemas). In these cases, italic text is used for placeholder values.	`smcp_server -host <confighost>`

Index

Symbols

[] (square brackets) . . . 86
< > (angle brackets) . . . 86
<key name>
  common log option . . . 78

A

active-context-limit . . . 45, 55
Agent Reservation options . . . 41
alarm
  common log option . . . 69
all
  common log option . . . 69
angle brackets . . . 86
application types
  TServer . . . 12
audience, for document . . . 8

B

background-processing . . . 46
background-timeout . . . 46
brackets
  angle . . . 86
  square . . . 86
buffering
  common log option . . . 62

C

Call Concentrator . . . 13, 17, 43
  recommendations for use with LDS . . . 43
cascaded proxy configuration . . . 16
CCon . . . 13, 17, 43
changing HA synchronization level . . . 30
check-point
  common log option . . . 66
cleanup-timer . . . 46
command-line parameters . . . 27
  -app . . . 28
  -host . . . 27
  -l . . . 28
  -port . . . 28
commenting on this document . . . 8
common configuration options . . . 62–81
  common section . . . 81
  disable-rbac . . . 79
  enable-async-dns . . . 81
  hangup-restart . . . 80
  heartbeat-period . . . 79
  heartbeat-period-thread-class-<n> . . . 80
  log section . . . 62–76
  log-extended section . . . 76–78
  log-filter section . . . 78
  log-filter-data section . . . 78–79
  mandatory . . . 62
  rebind-delay . . . 81
  security section . . . 79
  setting . . . 61
  sml section . . . 79–81
  suspending-wait-timeout . . . 80
common log options . . . 62–78
  <key name> . . . 78
  alarm . . . 69
  all . . . 69
  buffering . . . 62
  check-point . . . 66
  compatible-output-priority . . . 67
  debug . . . 71
  default-filter-type . . . 78
  expire . . . 63
  interaction . . . 70
  keep-startup-file . . . 64
  level-reassign-<eventID> . . . 76
  level-reassign-disable . . . 78
  log section . . . 62–76
  log-extended section . . . 76–78
  log-filter section . . . 78
  log-filter-data section . . . 78–79

mandatory options . . . 62
memory . . . 67
memory-storage-size . . . 67
messagefile . . . 64
message-format . . . 65
print-attributes . . . 66
segment . . . 63
setting . . . 61
spool . . . 67
standard . . . 70
time_convert . . . 65
time_format . . . 66
trace . . . 71
verbose . . . 62
x-conn-debug-all . . . 76
x-conn-debug-api . . . 75
x-conn-debug-dns . . . 75
x-conn-debug-open . . . 74
x-conn-debug-security . . . 75
x-conn-debug-select . . . 74
x-conn-debug-timers . . . 74
x-conn-debug-write . . . 74
common options
common log options . . . 62–78
common section . . . 81
mandatory options . . . 62
sml section . . . 79–81
common section
common options . . . 81
compatible-output-priority
common log option . . . 67
components
Interaction Routing Designer . . . 34
Configuration options . . . 45
configuration options
common log options . . . 62–78
common options . . . 62–81
configuration options . . . 50
ha-dly-switchover . . . 48
intra-cluster-distribution . . . 48
keep-ext-key . . . 49
mandatory options . . . 62
register-guard . . . 53
register-mode . . . 53
setting
common . . . 61
strict-backup-name . . . 54
use-query-call . . . 55
configuring distribution modes
Broadcast . . . 18
Load Distribution . . . 14, 35
Single T-Server LDS . . . 19
TProxy . . . 15
connections . . . 12
context-cleanup . . . 46
conventions
in document . . . 85
type styles . . . 86
count-active-context . . . 47

D

debug
common log option . . . 71
default-filter-type
common log option . . . 78
disable-rbac
common configuration option . . . 79
distribute-mode . . . 47
distribution modes
Broadcast . . . 17
Load Distribution . . . 13
Single T-Server LDS . . . 18
TProxy . . . 14
document
audience . . . 8
change history . . . 9
conventions . . . 85
errors, commenting on . . . 8
version number . . . 85
dynamic HA model . . . 30

E

enable-async-dns
common configuration option . . . 81
enable-safe-handover . . . 47
expire
common log option . . . 63

F

font styles
italic . . . 86
monospace . . . 86

G

group-id . . . 24, 55

H

HA synchronization . . . 30
ha-dly-switchover
configuration options . . . 48
hangup-restart
common configuration option . . . 80
ha-sync-level . . . 48
heartbeat-period

common configuration option 79
heartbeat-period-thread-class-<n>
common configuration option 80
high availability 34

I

installing
Message Server 25
intended audience 8
interaction
common log option 70
Interaction Routing Designer 34
intra-cluster-distribution
configuration options 48
italics 86

K

keep-ext-key
configuration options 49
keep-startup-file
common log option 64
keep-taction-stat 49

L

LDS and CCon 43
level-reassign-<eventID>
common log option 76
level-reassign-disable
common log option 78
license-file 49
link-alarm-high 49
link-alarm-low 50
link-by-originator 50
load distribution 34, 35, 41
loading-coefficient 56
loading-coefficient (WWR mode) 58
load-report-interval 50
log configuration options 62–68
log messages 45
log options 45
log section
common log options 62–76
log-extended section
common log options 76–78
log-filter section
common log options 78
log-filter-data section
common log options 78–79

M

Management Layer 27
max-outstanding 50
max-update-rate 50
memory
common log option 67
memory-storage-size
common log option 67
Message Server
installing 25
message synchronization queue 30
messagefile
common log option 64
message-format
common log option 65
monospace font 86
msg-duplication 51

N

new in release 8.1 20
no-context-distribution 51

O

options
active-context-limit 45, 55
background-processing 46
background-timeout 46
cleanup-timer 46
configured in Receiver applications 55
context-cleanup 46
context-remove-delay 46
count-active-context 47
distribute-mode 47
enable-safe-handover 47
group-id 24, 55
ha-sync-level 48
keep-taction-stat 49
LDS Receivers
group-id 55
loading-coefficient 56
license-file 49
link-alarm-high 49
link-alarm-low 50
link-by-originator 50
loading-coefficient (WWR mode) 58
load-report-interval 50
max-outstanding 50
msg-duplication 51
no-context-distribution 51
query-dn 52
query-timer 52
queue-expire-timeout 52

rq-expire-timeout . . . 53
server-id . . . 53
stat-calc-threshold . . . 54
tlib-verbose . . . 54
update-timestamp . . . 54
use-link-bandwidth . . . 55

P

print-attributes
common log option . . . 66

Q

query-dn . . . 52
query-timer . . . 52
queue-expire-timeout . . . 52

R

rebind-delay
common configuration option . . . 81
Receiver type . . . 17, 40
redundancy . . . 35, 41
redundant configurations . . . 35, 36
register-guard
configuration options . . . 53
register-mode
configuration options . . . 53
Resource Registration . . . 40
routing components . . . 34
Universal Routing Server . . . 34
rq-expire-timeout . . . 53

S

security section
common configuration options . . . 79
segment
common log option . . . 63
server-id . . . 53
setting configuration options
common . . . 61
single T-Server configurations . . . 12
sml section
common options . . . 79–81
spool
common log option . . . 67
square brackets . . . 86
standard
common log option . . . 70
stat-calc-threshold . . . 54
strict-backup-name
configuration options . . . 54
suspending-wait-timeout
common configuration option . . . 80

T

tiered proxy mode . . . 16
time_convert
common log option . . . 65
time_format
common log option . . . 66
tlib-verbose . . . 54
trace
common log option . . . 71
type styles
conventions . . . 86
italic . . . 86
monospace . . . 86
typographical styles . . . 85, 86

U

Universal Routing Server . . . 34
use-link-bandwidth . . . 55
use-query-call
configuration options . . . 55
user interface
Interaction Routing Designer . . . 34

V

verbose
common log option . . . 62
version numbering, document . . . 85

W

Weighted Round Robin mode . . . 56
WWR mode . . . 56, 59

X

x-conn-debug-all
common log option . . . 76
x-conn-debug-api
common log option . . . 75
x-conn-debug-dns
common log option . . . 75
x-conn-debug-open
common log option . . . 74
x-conn-debug-security
common log option . . . 75
x-conn-debug-select
common log option . . . 74

x-conn-debug-timers
common log option 74
x-conn-debug-write
common log option 74