Framework 7.6

SIP Server

Deployment Guide

About Genesys

Genesys Telecommunications Laboratories, Inc., a subsidiary of Alcatel-Lucent, is 100% focused on software for call centers. Genesys recognizes that better interactions drive better business and build company reputations. Customer service solutions from Genesys deliver on this promise for Global 2000 enterprises, government organizations, and telecommunications service providers across 80 countries, directing more than 100 million customer interactions every day. Sophisticated routing and reporting across voice, e-mail, and Web channels ensure that customers are quickly connected to the best available resource—the first time. Genesys offers solutions for customer service, help desks, order desks, collections, outbound telesales and service, and workforce management. Visit 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 following regional numbers:

Region	Telephone	E-Mail
North and Latin America	+888-369-5555 or +506-674-6767	support@genesyslab.com
Europe, Middle East, and Africa	+44-(0)-1276-45-7002	support@genesyslab.co.uk
Asia Pacific	+61-7-3368-6868	support@genesyslab.com.au
Japan	+81-3-6361-8950	support@genesyslab.co.jp

Prior to contacting technical support, please refer to the *Genesys Technical Support Guide* for complete contact information and procedures.

Ordering and Licensing Information

Complete information on ordering and licensing Genesys products can be found in the ***Genesys 7 Licensing Guide***.

Released by

Genesys Telecommunications Laboratories, Inc. www.genesyslab.com

Document Version: 76fr_dep-sip_03-2009_v7.6.003.00

Table of Contents

Index

List of Procedures

Preface

Welcome to the *Framework 7.6 SIP Server Deployment Guide*. This document introduces you to the concepts, terminology, and procedures relevant to T-Servers® in general and provides detailed reference information about SIP Server. The information includes, but is not limited to, configuration options, limitations, and switch-specific functionality. You must configure the configuration objects and options described in this document in the Framework Configuration Layer.

Use this document only after you have read through the *Framework 7.6 Deployment Guide*, and the Release Note for SIP Server.

This document is valid only for the 7.6 release of this product.

Note: For releases of this document created for other releases of this product, please 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 provides an overview of this document, identifies the primary audience, introduces document conventions, and lists related reference information:

- Intended Audience, page 14
- Chapter Summaries, page 14
- Document Conventions, page 16
- Related Resources, page 18
- Making Comments on This Document, page 19

SIP Server is the Genesys software component that provides an interface between your telephony hardware and the rest of the Genesys software components in your enterprise. It translates and keeps track of events and requests that come from, and are sent to the telephony device. SIP Server is a TCP/IP-based server that can also act as a messaging interface between SIP Server clients. It is the critical point in allowing your Genesys solution to facilitate and track the contacts that flow through your enterprise.

Intended Audience

This guide is intended primarily for system administrators, both those who are new to SIP Server and those who are familiar with it.

- If you are new to SIP Server, read the *Framework 7.6 Deployment Guide* and the Release Note mentioned earlier, and then read all of the sections of this document that apply to your software and its accompanying components. Refer back to the *Framework 7.6 Deployment Guide* as needed.
- If you are an experienced SIP Server user—someone with computer expertise, who is used to installing, configuring, testing, or maintaining Genesys software—you may find it more time efficient to go to the Index to see what is new or different in SIP Server release 7.6. If you take that approach, please also read Release Notes and refer to other related resources, such as the *Genesys 7 Events and Models Reference Manual*.

In general, this document assumes that you have a basic understanding of, and familiarity with:

- Network design and operation.
- Your own network configurations.
- Your telephony hardware and software.
- Genesys Framework architecture and functions.
- Configuration Manager interface and object-managing operations.

Based on your specific contact center environment and your responsibilities in it, you may need to be familiar with a much wider range of issues as you deploy T-Server.

Reading Prerequisites

You must read the *Framework 7.6 Deployment Guide* before using this *SIP Server Deployment Guide*. That book contains information about the Genesys software you must deploy before deploying SIP Server.

Chapter Summaries

This *SIP Server Deployment Guide* encompasses all information—including conceptual, procedural, and reference information—about Genesys T-Servers in general, and SIP Server in particular. Depending on the subject addressed in a particular section, the document style may move from narration, to instructions, to technical reference. To distinguish between SIP Server, general T-Server sections, and those chapters intended for DMX, this document is divided into three main parts.

Part One—SIP Server Deployment

Part One of this SIP Server document, “SIP Server Deployment,” consists of Chapters 1 through 8. These chapters contain deployment information specific to SIP Server.

- Chapter 1, “SIP Server Fundamentals,” on page 25, provides information about SIP Server architectures and deployment considerations.
- Chapter 2, “SIP Server General Deployment,” on page 33, presents configuration and installation procedures for SIP Server.
- Chapter 3, “High-Availability Deployment,” on page 51, describes how to implement a high-availability (HA) configuration for SIP Server.
- Chapter 4, “Starting and Stopping SIP Server,” on page 69, describes how, and in what order, to start up SIP Server among other Framework components. It also provides possible stopping commands.
- Chapter 5, “SIP Devices Support,” on page 77, describes compatibility and configuration information specific to SIP Server, including how to set the DN properties and recommendations for the device configuration.
- Chapter 6, “SIP Server Feature Support,” on page 97, describes which features SIP Supports and how to configure them.
- Chapter 7, “T-Library Functionality Support,” on page 177, describes T-Library functionality that SIP Server supports, known limitations, and error messages.
- Chapter 8, “SIP Server Configuration Options,” on page 197, describes configuration options specific to SIP Server.

Although you must refer to Part Two if you have never before configured or installed SIP Server. You might also use the chapters in Part Two, even if you are already familiar with SIP Server, to discover any changes to functionality, configuration, and installation since you last deployed this component.

Genesys recommends that you use wizards to deploy SIP Server. If you do, first read Chapter 2 to familiarize yourself with SIP Server general deployment, and then proceed with the deployment process using Framework wizards.

Part Two—T-Server Common Functions and Procedures

Part Two of this SIP Server document, “T-Server Common Functions and Procedures,” consists of Chapters 9 through 12. These chapters contain architectural, functional, and procedural information common to all T-Servers. They also contain information about configuration options that are common to all T-Servers.

- Chapter 9, “T-Server Fundamentals,” on page 255, describes T-Server, its place in the Framework 7 architecture, T-Server redundancy, and multi-site issues. It does not, however, provide configuration and installation information.
- Chapter 10, “Multi-Site Support,” on page 267, describes the variations available for T-Server implementations across geographical locations.
- Chapter 11, “Common Configuration Options,” on page 321, describes log configuration options common to all Genesys server applications.
- Chapter 12, “T-Server Common Configuration Options,” on page 343, describes configuration options common to all T-Server types including options for multi-site configuration.

Part Three—DMX Reference Information

Part Three of this SIP Server document, “DMX Reference Information,” consists of Chapters 13 and 14. These chapters contain reference information specific to DMX.

- Chapter 13, “DMX Deployment,” on page 369, describes how to configure, install, and start DMX.
- Chapter 14, “DMX Reference,” on page 393, provides detailed information about DMX.

Appendix

- The appendix “Session Initiation Protocol (SIP) Overview” on page 401, provides basic information about the SIP protocol.

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:

76fr_sip_06-2008_v7.6.001.00

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

Type Styles

Italic

In this document, italic is used for emphasis, for documents' titles, for definitions of (or first references to) unfamiliar terms, and for mathematical variables.

Examples:

- Please consult the *Genesys Migration Guide* for more information.
- *A customary and usual practice* is one that is widely accepted and used within a particular industry or profession.
- Do *not* use this value for this option.
- The formula, x +1 = 7 where x stands for . . .

Monospace Font

A monospace font, which looks like `teletype or typewriter text`, is used for 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; and code samples.

Examples:

- Select the `Show variables on screen` check box.
- Click the `Summation` button.
- In the `Properties` dialog box, enter the value for the host server in your environment.
- In the `Operand` text box, enter your formula.
- Click `OK` to exit the `Properties` dialog box.
- The following table presents the complete set of error messages T-Server distributes 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.

Monospace is also used for any text that users must manually enter during a configuration or installation procedure, or on a command line:

Example:

- Enter `exit` on the command line.

Screen Captures Used in This Document

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

Square Brackets

Square brackets indicate that a particular parameter or value is optional within a logical argument, a command, or some programming syntax. That is, the parameter's or value's presence is not required to resolve the argument, command, or block of code. The user decides whether to include this optional information. Here is a sample:

```
smcp_server -host [/flags]
```

Angle Brackets

Angle brackets indicate a placeholder for a value that the user must specify. This might be a DN or port number specific to your enterprise. Here is a sample:

```
smcp_server -host <confighost>
```

Related Resources

Consult these additional resources as necessary:

- The *Framework 7.6 Deployment Guide*, which will help you configure, install, start, and stop Framework components.
- The *Framework 7.6 Stream Manager Deployment Guide*, which will help you configure, install, start, and stop Stream Manager.
- The *Framework 7.6 Configuration Options Reference Manual*, which will provide you with descriptions of configuration options for other Framework components.
- The *Framework 7.6 Configuration Manager Help*, which will help you use Configuration Manager.
- The *Genesys Migration Guide*, also on the Genesys Documentation Library DVD, which contains a documented migration strategy from Genesys product releases 5.x and later to all Genesys 7.x releases. Contact Genesys Technical Support for additional information.
- The *Genesys 7 Events and Models Reference Manual*, which contains the T-Library API, information on `TEvents,` and an extensive collection of call models.

- The *Genesys 7 Technical Publications Glossary,* which ships on the Genesys Documentation Library DVD and which provides a comprehensive list of the Genesys and CTI terminology and acronyms used in this document.
- The Release Notes and Product Advisories for this product, which are available on the Genesys Technical Support website at `http://genesyslab.com/support`.

Information on 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*

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`.

Making Comments on This Document

If you especially like or dislike anything about this document, please 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 information in this document only and to the way in which the information is presented. Speak to 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.

Part

1

Part One: SIP Server Deployment

Part One of this *SIP Server Deployment Guide* contains deployment information specific to your SIP Server. The information in Part One is divided into the following chapters:

- Chapter 1, "SIP Server Fundamentals," on page 25, provides information about SIP Server architectures and deployment considerations.
- Chapter 2, "SIP Server General Deployment," on page 33, presents configuration and installation procedures for SIP Server.
- Chapter 3, "High-Availability Deployment," on page 51, describes how to implement a high-availability (HA) configuration for SIP Server.
- Chapter 4, "Starting and Stopping SIP Server," on page 69, describes how, and in what order, to start up T-Server among other Framework components. It also provides possible stopping commands.
- Chapter 5, "SIP Devices Support," on page 77, describes compatibility and configuration information specific to SIP Server, including instructions for setting the DN properties and recommendations for the device configuration.
- Chapter 6, "SIP Server Feature Support," on page 97, describes which features SIP Server supports and how to configure them.
- Chapter 7, "T-Library Functionality Support," on page 177, describes the T-Library functionality that SIP Server supports, known limitations, and error messages.
- Chapter 8, "SIP Server Configuration Options," on page 197, describes configuration options specific to SIP Server.

New in SIP Server for 7.6

The following new features are now available in the initial 7.6 release of SIP Server:

- **Silence Treatment in conference.** SIP Server now provides a silent treatment for conference call participants when one of them places the call on hold. See “Silence Treatment in Conference” on page 125 for details.
- **Trunk optimization for multi-site support.** SIP Server now supports trunk optimization for multi-site transfers, to ensure required information is properly reported across SIP Server instances. See “Trunk Optimization for Multi-Site Transfers” on page 166 for details.
- **Providing call participant info.** SIP Server can now distribute information about call participants to logged-in agents. See “Providing Call Participant Info” on page 164 for details.
- **Enhanced call transfer and conference functionality.** SIP Server now supports additional call transfer and conference functionality, including: routing to external destination using REFER, consultation transfer for calls on a Routing Point, and the ability to alternate between main and consultation calls. See “Call Transfer and Conference” on page 123 for details.
- **Enhanced instant messaging support.** SIP Server now supports the following call supervision modes for instant messaging (IM) calls: Silent Monitoring, Whisper Coaching, Open Supervisor Presence, and Intrusion. SIP Server also supports multiple IM sessions and delivery of Instant Messages transcripts. See “Instant Messaging” on page 143 for details.
- **Class of Service support.** SIP Server now supports Class of Service. See “Class of Service” on page 127 for details.
- **Remote Supervision support.** The Remote Supervision feature enables supervisors to monitor agent calls from outside the contact center. See “Remote Supervision” on page 115 for details.
- **Multi-Site Supervision support.** The Multi-Site Supervision feature enables supervisors at a local site to monitor agents located at remote sites. See “Multi-Site Supervision” on page 112 for details.
- **Enhanced SIP headers mapping support.** SIP Server can now extract data from a REFER message, and map it to either the Extension or UserData attribute of T-Library event messages. See “Mapping SIP Headers and SDP Messages” on page 149 for details
- **TSendDTMF request support.** SIP Server now supports TSendDTMF requests for devices that conform to RFC 2976. To support this functionality, a new DN-level configuration option, rfc-2976-dtmf, has been added. This option applies to a DN of type Trunk or Extension.

- **Enhanced single-step transfer to a Routing Point support**. SIP Server can now perform a single-step call transfer to a Routing Point DN when the call is silently monitored by a supervisor, or when the call is in emergency recording mode.
- **TAlternateCall request support.** SIP Server now supports `TAlternateCall` requests for DNs configured with the `dual-dialog-enabled` configuration option set to either `false` or `true`.
- **Blocking SIP headers support.** SIP Server can now filter out specific headers during the `INVITE` message propagation. To support this functionality, a new application-level configuration option, `sip-block-headers`, has been added.
- **Trunk capacity support**. The number of calls per trunk can now be controlled with new DN-level configuration options: `capacity` and `capacity group`. See the option descriptions on page 229.
- **Stream Manager overloading scenario support.** SIP Server supports the load-sharing mechanism implemented in Stream Manager. In an environment with distributed Stream Managers, if a dialog is rejected solely because the rejecting Stream Manager is overloaded, SIP Server tries to re-send that dialog to another Stream Manager. See the *Framework 7.6 Stream Manager Deployment Guide* for details.
- **OPTIONS messages support.** SIP Server now supports processing of `OPTIONS` messages.
- **DTMF tones generation support.** SIP Server can now send a request to Stream Manager to generate DTMF tones using the `TApplyTreatment` request with `TreatmentType` set to `PlayApplication`. See "DTMF Tones Generation" on page 137.
- **Enhanced high-availability support.** When operating in a high-availability environment, SIP Server can now synchronize calls that are in `ringing` state.

Notes: For a list of new features common to all T-Servers, see "New for All T-Servers in 7.6" on page 254 of this document.

For a complete list of supported features and their configuration, see Chapter 6, "SIP Server Feature Support," on page 97.

For a list of configuration option changes that apply to SIP Server, see "Changes from 7.5 to 7.6" on page 249.

Chapter

1 SIP Server Fundamentals

This chapter provides more in-depth information about SIP Server and contains the following sections:

Overview

SIP Server has the same position in the Genesys Media Layer as all Genesys T-Servers. SIP Server is a combined T-Server and a call-switching component, in which the call-switching element functions as a SIP (Session Initiation Protocol) Back-to-Back User Agent (B2BUA). In concrete terms, this means that call switching and control is performed by Genesys—no third-party PBX or ACD system is required. A call's audio signal and its associated data travel on a single network, which eliminates the problems associated with synchronizing separate voice and data networks. Because SIP Server supports the Internet Engineering Task Force (IETF) SIP RFC 3261 suite, it is compatible with the most popular SIP-compatible, off-the-shelf hardware or software.

SIP Server can operate with or without a third-party softswitch. Genesys SIP Server gives the entire Genesys line of products access to SIP networks, offering a standards-based, platform-independent means of taking full advantage of the benefits of voice/data convergence.

SIP Server Architecture

Figure 1 presents a generalized architecture of the SIP Server network.

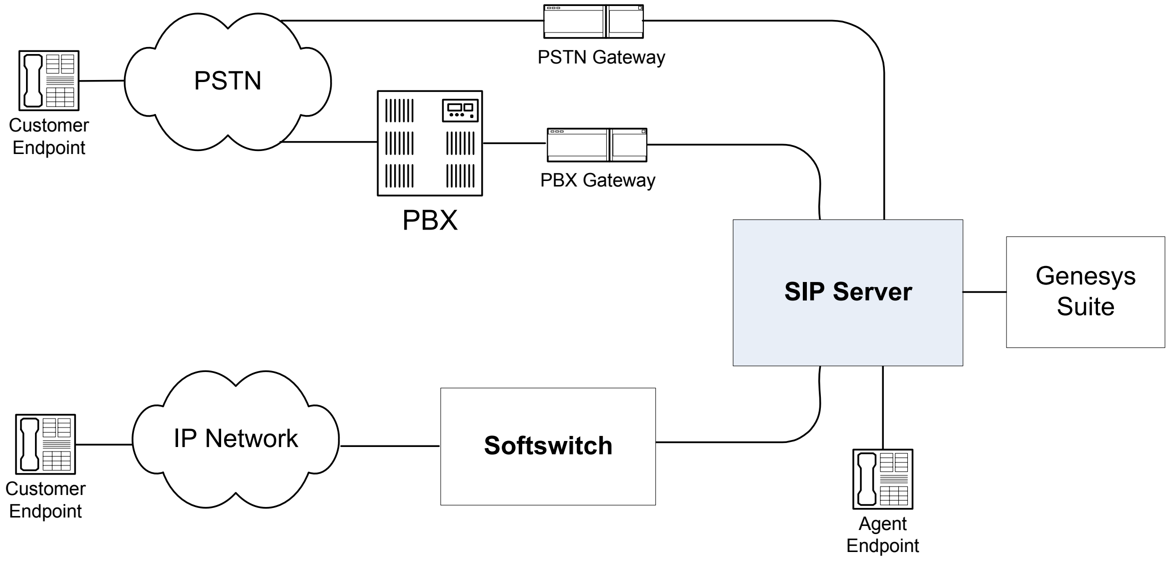


Figure 1: SIP Network Architecture

SIP Server provides all SIP signaling and T-Server functions. Stream Manager is an optional component that is used to play music-on-hold, music-in-queue and announcements, and to collect DTMF digits. A third-party music server is an optional component that is used as an external music source for music-on-hold. A Multipoint Conference Unit (MCU) is an optional component that is used for third-party call control (3pcc) conference calls. It is also used for silent voice monitoring, whisper coaching, intrusion monitoring, and agent-initiated call recording.

The SIP messages that SIP Server sends or receives are very similar in all configurations, but the destination to which SIP Server sends the SIP requests differs according to the deployment configuration. This mostly applies to the routing of `INVITE` messages. Other messages follow the path established by `INVITE`.

See Chapter 5, “SIP Devices Support,” on page 77 for full details on how to configure the elements of such a network.

SIP Server Deployment Modes

The following SIP Server deployment modes are currently supported:

- Stand-alone mode
- Application Server mode

- Customer-side proxy mode

Stream Manager is used as an MCU in these scenarios. For more information, see the *Framework 7.6 Stream Manager Deployment Guide*.

Stand-alone Mode

Figure 2 illustrates SIP Server in the Stand-alone mode.

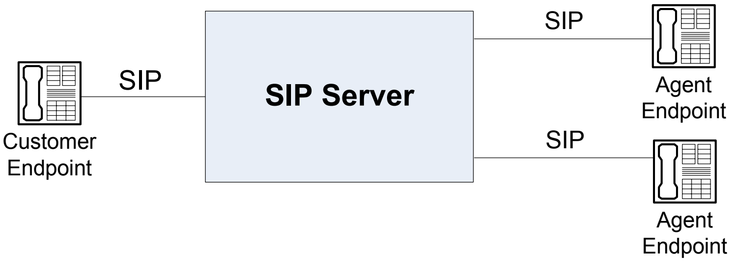


Figure 2: Stand-alone Mode

In this configuration, SIP Server sends all messages to the addresses of the customer and agent endpoints. The IP addresses in this scenario are determined by SIP Server from either of the following sources:

- Configuration Manager.

  When, for each agent DN, an IP address is configured on the DN `Annex` tab. For example, for DN `1077` on the `Annex` tab in the `TServer` section, you set the `contact` option to the `1077@192.168.2.55` value. This is useful for agent endpoints.

- Lookup in the local registrar.

  When the agent DN is defined with the registrar as `agent1@company.com`, and its SIP endpoint has registered this SIP URI (Uniform Resource Indicator) with the registrar as `1077@192.168.2.55`, the `INVITE` message is sent to the IP address `192.168.2.55`.

- SIP Server resolves the name as it was dialed.

  For example, if the dialed name is `customer@somedomain.com`, the request is sent to the address `somedomain.com`.

Application Server Mode

In the configuration shown in Figure 3, SIP Server is deployed as an Application Server behind a softswitch. This is the most common deployment of SIP Server.

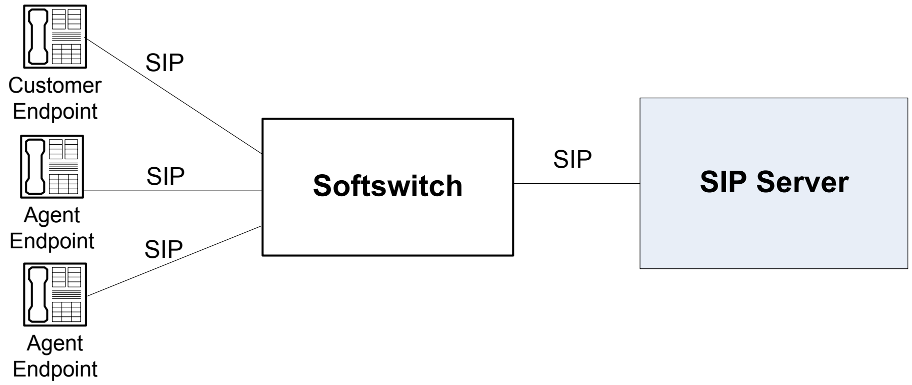


Figure 3: Application Server Mode

In the Application Server mode, SIP Server communicates with a single softswitch. SIP Server is configured to send all `INVITE` requests to the IP address of the softswitch.

In this configuration, the softswitch bypasses SIP Server for direct agent-to-agent calls. As a consequence, agent-to-agent calls are not visible to SIP Server, and it cannot provide any control over these calls.

Customer-Side Proxy Mode

In the configuration shown in Figure 4, a softswitch is deployed between SIP Server and agent endpoints, but customer endpoints communicate directly with SIP Server.

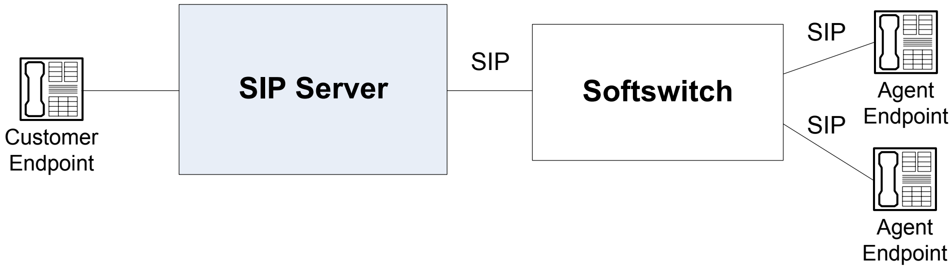


Figure 4: Customer-Side Proxy Mode

All inbound calls (from customers to agents) are routed by SIP Server to a softswitch, the IP address of which is configured in Configuration Manager.

For outbound calls (from agents to customers), the IP addresses are determined from the `Request URI` message, or they are configured in Configuration Manager as gateways (DNs of type `Trunk`). Alternatively, SIP Server can resolve the name as it was dialed. For example, if the dialed name is `customer@somedomain.com`, the request is sent to the address `somedomain.com`.

In this configuration, the softswitch bypasses SIP Server for direct agent-to-agent calls. As a consequence, agent-to-agent calls are not visible to SIP Server, and it cannot provide any control over these calls.

Media Server Deployment Architecture

Figure 5 illustrates one possible deployment architecture for a third-party media server (such as a music server or MCU), or for Genesys Stream Manager used in conjunction with SIP Server and Genesys business applications.

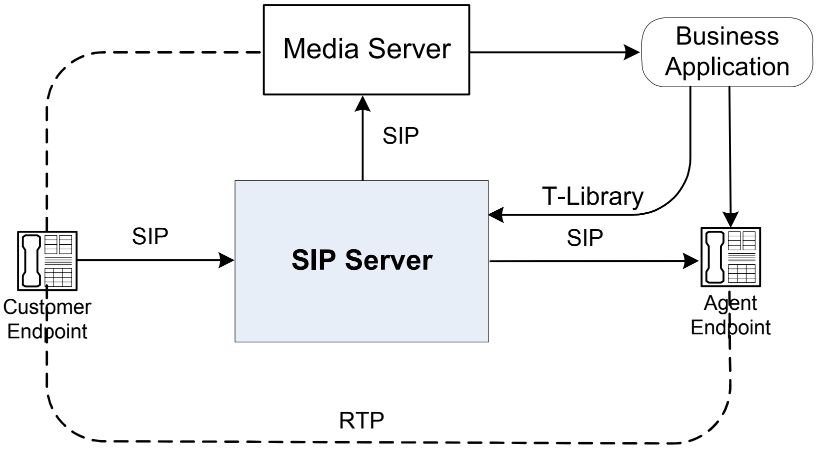


Figure 5: Media Server Deployment Architecture

Call Scenario

1. A call arrives and is established with a contact center agent. SIP Server operates as a SIP B2BUA and maintains two separate SIP dialogs, one for the customer and one for the agent. The RTP stream is negotiated between the customer and agent endpoints directly, using the codec. SIP Server provides flexibility with manipulation of SDP information.
2. The agent invokes a call-hold operation either by using a `THoldCall` request to SIP Server, or by pressing the `Hold` button on the endpoint.
3. SIP Server selects a media server in sequence from among all configured servers with the same priority, and then establishes a new SIP dialog to the music server. SIP Server then sends the `INVITE` message to the customer session to connect the RTP stream to the music server, and then re-`INVITEs` the agent session to stop RTP traffic from it.
4. When the agent invokes a call-retrieve operation either by using the `TRetrieveCall` request, or by pressing the `Retrieve` button on the endpoint, SIP Server terminates the music dialog by sending a `BYE` message, and then re-`INVITEs` the customer session to connect the RTP stream with the agent endpoint.

Redundant SIP Servers

SIP Servers can operate in a high-availability (HA) environment, providing you with redundant systems. One basic principle of redundant SIP Servers is the standby redundancy type, which dictates how quickly a backup SIP Server steps in when the primary SIP Server goes down. The Framework Management Layer currently supports two types of redundant configurations: `warm standby` and `hot standby`.

SIP Server in an HA configuration differs from most Genesys T-Servers in the role it performs in the SIP network. It is not a switch, but it does have traditional switching capabilities.

Instructions for configuring SIP Server redundancy are available in Chapter 3, “High-Availability Deployment,” on page 51.

Load Balancing

Figure 6 illustrates a load-balancing architecture for situations in which the call rate exceeds the capacity of a single SIP Server.

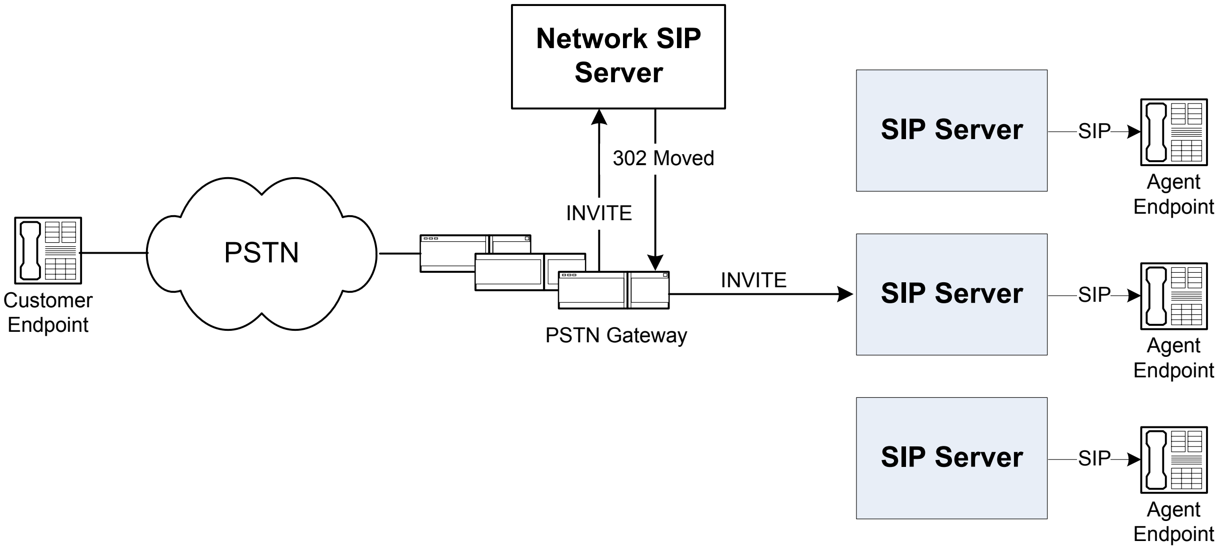


Figure 6: SIP Server Sample Load-Balancing Configuration

In this configuration, all inbound calls first arrive at the Network SIP Server, which performs all initial routing. The routing on the Network SIP Server is either direct to agents, or to the second tier of Routing Points on the SIP Server.

Routing inbound calls directly to agents assumes the following:

- Multiple TDM-to-VoIP gateways (or incoming SIP firewalls for pure VoIP calls) are deployed to provide sufficient call capacity.

- Multiple SIP Servers are deployed. Each can serve up to 1000 agents.
- One or more Network SIP Servers are deployed. Given the high throughput of a Network SIP Server, it is very likely that a single SIP Server will be sufficient for most deployments.
- Gateways are configured to send all calls to the Network SIP Servers. If multiple Network SIP Servers are required, you can partition gateways, so that they send calls to different SIP Servers; or you can configure each gateway to send calls to one of the Network SIP Servers, based, for example, on call origination or destination.

As a result, the following provides a generalized call flow:

1. Network SIP Servers communicate with Universal Routing Server (URS) (not shown in Figure 6 on page 30). URS selects an agent on one of the SIP Servers by using real-time state information available from the SIP Servers via Stat Servers (not shown). URS responds to the Network SIP Server with the agent's DN and the location name of the SIP Server.
2. Network SIP Server communicates the `ConnID` and attached data to the selected SIP Server. It then responds to the PSTN gateway with a `302` message containing the IP address and External Routing Point number on the destination SIP Server.
3. The gateway processes the `302` message, and then sends a new `INVITE` message to the selected SIP Server.
4. SIP Server receives the `INVITE` message from the External Routing Point, matches the call, and reroutes it to the selected agent by using Inter Server Call Control (ISCC).

Multi-Site Support

SIP Server, like any conventional T-Server, is built with the T-Server Common Part that contains the ISCC component responsible for call data transfer between multiple sites. Currently, SIP Server supports the following ISCC transaction types: `route`, `direct-notoken`, `direct-uui`, `pullback`, and `reroute`. However, `direct-uui` is supported only in a pure SIP environment.

For instructions on installing and configuring a multi-site environment, including information on the ISCC features, please see Chapter 10, "Multi-Site Support," on page 267.

Next Steps

Now that you have gained a general understanding of the roles and features available in SIP Server, you are ready to learn how SIP Server is installed and configured. That information is presented in the next few chapters of this *Deployment Guide.* So unless you are already familiar with SIP Server deployment and operation procedures, continue with Chapter 2, “SIP Server General Deployment,” on page 33. Otherwise, you may want to proceed to Chapter 6, “SIP Server Feature Support,” on page 97, where you will find information about feature configurations that SIP Server supports.

Chapter

2 SIP Server General Deployment

This chapter contains general information for the deployment, configuration, and installation of your SIP Server. You may have to complete additional configuration and installation steps specific to your SIP Server and devices. You will find these steps in Part One of this document.

This chapter contains the following sections:

Note: You *must* read the *Framework 7.6 Deployment Guide* before proceeding with this SIP Server guide. That document contains information about the Genesys software you must deploy before deploying SIP Server.

Prerequisites

SIP Server has a number of prerequisites for deployment. Read through this section before deploying your SIP Server.

Software Requirements

Framework Components

You can only configure SIP Server after you have deployed the Configuration Layer of Genesys Framework. This layer contains DB Server, Configuration Server, Configuration Manager, and, at your option, Deployment Wizards. If you intend to monitor or control SIP Server through the Management Layer, you must also install and configure components of this Framework layer, such as Local Control Agent (LCA), Message Server, Solution Control Server (SCS), and Solution Control Interface (SCI), before deploying SIP Server.

Refer to the *Framework 7.6 Deployment Guide* for information about, and deployment instructions for, these Framework components.

Media Layer and LCA

To monitor the status of components in the Media Layer through the Management Layer, you must load an instance of LCA on every host running Media Layer components. Without LCA, Management Layer cannot monitor the status of any of these components. If you do not use the Management Layer, LCA is not required.

Supported Platforms

Refer to the *Genesys Supported Operating Environment* white paper for the list of operating systems and database systems supported in Genesys releases 7.x. You can find this document on the Genesys Technical Support website at `http://genesyslab.com/support/dl/retrieve/default.asp?item=B6C52FB62DB42BB229B02755A3D92054&view=item`.

For UNIX-based (UNIX) operating systems, also review the list of patches Genesys uses for software product builds, and upgrade your patch configuration if necessary. A description of patch configuration is linked to installation `read_me.html` files for the Genesys applications that operate on UNIX, and is available within the installation packages.

Hardware and Network Environment Requirements

Hosting

Genesys recommends that you or your IT specialist assign host computers to Genesys software before you start Genesys installation. Remember the following restrictions:

- Do not install all the Genesys server applications on the same host computer.

- When installing a few server applications on the same host computer, prevent them (except for Configuration Server) from using the swap area.

Installation Privileges

During deployment, be sure to log in with an account that will permit you to perform administrative functions—that is, one that has root privileges.

Server Locations

Refer to the "Network Locations for Framework Components" chapter of the *Framework 7.6 Deployment Guide* for recommendations on server locations.

Supported Platforms

Refer to the *Genesys Supported Media Interfaces* white paper for the list of supported switch and PABX versions. You can find this document on the Genesys Technical Support website at `http://genesyslab.com/support/dl/retrieve/default.asp?item=A9CB309AF4DEB8127C5640A3C32445A7&view=item`.

Licensing Requirements

All Genesys software is licensed—that is, it is not shareware. Genesys products are protected through legal license conditions as part of your purchase contract. However, the level of technical license-control enforcement varies across different solutions and components.

Before you begin to install SIP Server, remember that, although you may not have had to use technical licenses for your software when you deployed the Configuration and Management Layers in their basic configurations, this is not the case with the Media Layer.

SIP Server requires seat-related DN technical licenses to operate even in its most basic configuration. Without appropriate licenses, you cannot install and start SIP Server. If you have not already done so, Genesys recommends that you install License Manager and configure a license file at this point. For complete information on which products require what types of licenses, and on the installation procedure for License Manager, refer to the *Genesys 7 Licensing Guide* available on the Genesys Documentation Library DVD.

The sections that follow briefly describe the T-Server/SIP Server license types.

Note: Starting with release 7.2, the licensing requirements for T-Server (including SIP Server) have changed from previous releases. Please read this section carefully and refer to the *Genesys 7 Licensing Guide* for complete licensing information.

Licensing Basic Implementations

A stand-alone SIP Server serving a single site requires licenses to register all DNs it monitors. DNs that agents use in day-to-day contact center operations, such as Extensions and ACD Positions, have to be registered using licenses that control agent seats.

Note: Configure all seat DNs that agents use (Extensions and ACD Positions) in the Configuration Layer. This enables detailed call monitoring through Genesys reporting, and generally allows you to control access to individual DNs.

Licensing HA Implementations

SIP Servers operating with the `hot standby` redundancy type require a special CTI HA technical license, which allows for high-availability implementations, in addition to regular SIP Server licenses. Neither SIP Server in a redundant pair configured for `hot standby` starts if this license is unavailable. Moreover, the primary and backup SIP Servers must use the same licenses to control the same pool of DNs. If your SIP Servers are configured with the `hot standby` redundancy type, order licenses for CTI HA support.

Licensing Multi-Site Implementations

SIP Servers performing multi-site operations require licenses that allow for such operations, in addition to regular SIP Server licenses. If some of your SIP Servers are configured for multi-site routing while others are not, either order licenses for multi-site support for all SIP Servers or install an additional License Manager to handle the SIP Servers involved in multi-site routing.

Note: You do not need licenses for multi-site support if some SIP Server clients include the local location as the `Location` attribute value in their requests for routing within the same site.

Configuring License Files

You need a license to configure and install Media Layer components. Genesys recommends that, if you have not already done so, at this point you:

1. Install License Manager.
2. Configure license files.

Note: If you use the `<port>@<server>` format when entering the name of the license server during installation, remember that some operating systems use `@` as a special character. In this case, the installation routine is unable to write license information for SIP Server to the Configuration Layer or the `run.sh` file. Therefore, when you use the `<port>@<server>` format, you must manually modify the command-line license parameter after installing SIP Server.

For information about which products require what types of licenses and for the installation procedure for License Manager, refer to the *Genesys 7 Licensing Guide* available on the Genesys Documentation Library DVD.

About Configuration Options

Configuring SIP Server is not a onetime operation. It is something you do at the time of installation and then in an ongoing way to ensure the continued optimal performance of your software. You must enter values for SIP Server configuration options on the relevant Wizard pages or on the `Options` tab of your SIP Server `Application` object in Configuration Manager. The instructions for configuring and installing SIP Server that you see here are only the most rudimentary parts of the process. You must refer extensively to the configuration options chapters located in Part One and Part Two of this book. Pay particular attention to the configuration options specific to SIP Server.

Configuration options common to all T-Servers, independent of switch type, are described in Chapter 12, “T-Server Common Configuration Options,” on page 343. SIP Server-specific configuration options are described in Chapter 8, “SIP Server Configuration Options,” on page 197. SIP Server also supports unified Genesys log options, as described in the Chapter 11, “Common Configuration Options,” on page 321.

Options that configure values for the TSCP (T-Server Common Part) software in your SIP Server are common to all T-Servers. Options based on the SIP custom features apply to your SIP Server only. Familiarize yourself with both types of options. You will want to adjust them to accommodate your production environment and the business rules that you want implemented there.

Network Considerations

This section is for system administrators, contact center operations heads, and developers who are planning to deploy Genesys SIP Server.

Deploying SIP Server is similar in many ways to deploying other components of the Genesys Framework, with the significant exception that the voice signal is carried over the data network. This has serious implications for network planning and server sizing. The primary purpose of this section is to highlight the major planning and resource concerns you face in rolling out SIP Server, and to explain how it overlaps with the underlying data network. However, this section is not intended to be an exhaustive guide to network planning. Refer to the *Framework 7.6 Deployment Guide* for further help with Framework rollout.

The performance of SIP Server is directly linked to that of the underlying data network. It is essential that you perform a proper network audit to ensure that the data network has been properly sized and tuned for real-time (voice) packet transport. This section discusses the factors that affect overall performance of an IP-based configuration, and provides some general rules to follow when deploying SIP Server.

Voice Quality

The following factors have a direct impact on voice quality:

- Network latency—Overall network delay.

  In a system in which packets are emitted at 20-millisecond (msec) intervals, some packets actually arrive at intervals ranging from 0 to 32 msec. To minimize network latency and ensure acceptable voice quality, you need to tune the network to prioritize real-time voice packets. There are various available schemes for prioritizing voice packets, depending on the IP router vendor.

- Packet loss—Voice packets that are dropped for various reasons (physical media error, timeout due to network congestion, and so on).

  Packet loss is a function result of several factors, including network bandwidth. As a general rule, the maximum sustained packet loss must not exceed 5 percent. Zero (0) percent packet loss is easily achievable on local-area networks (LANs) that use full-duplex Ethernet connections and this must be your goal for reliable high-quality voice operations.

- Packet jitter—Variation in voice packet arrival times.

  You can minimize packet jitter by using a jitter buffer at the endpoint device. As a general rule, you must set the buffer size to the maximum anticipated deviation from the typical interpacket emission time.

Other factors that influence voice quality include:

- Packet misordering—Packets arrive in the wrong order (similar to packet loss).
- Type of codec used—Codecs that do not compress the audio signal produce better voice quality but use greater bandwidth.
- Silence suppression—Silence suppression can save bandwidth, but it can also impact voice quality.

Bandwidth Requirements

Determining the bandwidth requirements for the underlying data network is another critical step in achieving proper performance and voice quality. Bandwidth requirements for a video connection are, of course, much higher. Genesys recommends that you verify network performance and voice quality by conducting performance tests and measurements in a lab environment prior to production rollout.

For an IP/Ethernet network, two factors that affect bandwidth requirements are:

- Codec used.
- Protocol headers.

Genesys has found that a full-duplex voice conversation using the G.723.1 codec requires approximately 20 Kbps (kilobits per second). When estimating actual network bandwidth needs, you must also consider such factors as network efficiency and utilization.

Note: Genesys recommends that LANs not have a throughput higher than 30 percent of the theoretical maximum (10 Mbps/100 Mbps/1 Gbps). Exceeding the 30-percent level often leads to packet loss, because the IP switches and routers are overloaded.

Remote Agent Configuration

SIP Server's remote agent capabilities range from a single remote agent, to a group of remote agents in a branch office environment. The distributed nature of branch office and remote agent architectures adds to the complexity of network sizing and tuning.

Note: Remote agent deployment behind media gateways on circuit-switching networks (such as PSTN) is not fully supported in the current release of SIP Server. If inbound calls are answered by, or transferred to, such remote agents, the `Annex` tab > `Tserver` section of the corresponding `DN` object in Configuration Manager must have the `refer-enabled` option set to `false`. This value limits call-transfer functionality to `Consult Transfer` only. In addition, your media gateway must be configured as a DN of type `Trunk`, with the `refer-enabled` option set to `false`.

Bandwidth and Network Tuning

Just as for local network deployment of a VOIP-based system, you must, if at all possible, allot proper bandwidth for voice communication and tune the underlying network for real-time media. Remote agents using a dial-up connection require greater bandwidth (at least 33 Kbps, with 56 Kbps recommended) because of the extra network overhead. This assumes use of the G.723.1 codec, although some dial-up connections may accommodate G.729. A Digital Subscriber Line (DSL) connection is a better alternative than a dial-up connection. The choice of remote access method is important—avoid sending voice communication over an unmanaged data network, such as the public Internet, where voice quality cannot be guaranteed.

For a branch office environment, network bandwidth requirements depend on the number of agents. Again, wide-area network (WAN) connectivity to the corporate LAN must be tuned for real-time voice communications. You need to ensure that service-level agreements from your virtual private network (VPN) provider give details of such requirements. End-to-end network latency must not exceed 250 msec.

Firewalls

This release of SIP Server provides no explicit support for Network Address Translation (NAT). Genesys recommends using virtual private networks (such as PPTP) and ensuring that all agents are on the same network, without NAT translators between the agents and SIP Server.

Deployment Sequence

Genesys recommends deploying SIP Server by using the SIP Server Configuration Wizard. However, if for some reason you must manually deploy SIP Server, you will also find instructions for doing that in this chapter.

The recommended sequence to follow before deploying SIP Server is described below. Steps 1 through 3 apply for both Wizard-based and manual deployment. For Wizard deployment, Steps 4 and 5 take place within the Wizard deployment process itself.

Wizard or Manual Deployment

1. Deploy Configuration Layer objects and ensure Configuration Manager is running (see the *Framework 7.6 Deployment Guide*).
2. Deploy Network objects (such as Host objects).
3. Deploy the Management Layer (see the *Framework 7.6 Deployment Guide*).

When manually deploying SIP Server, you must continue with the next two steps. If you are deploying SIP Server with the Configuration Wizard, the next two steps take place within the Wizard deployment process itself, where you can create and configure all the necessary objects for SIP Server deployment.

Manual Deployment

4. Configure Telephony objects (see “Manual Configuration of Telephony Objects” on page 44):
   - Switching Offices
   - Switches

- Agent Logins
- DNs

5. Deploy the Media Layer:
   - SIP Server (beginning with "Manual Configuration of SIP Server" on page 46).

If, during the installation procedure for any of the Genesys applications, the script warns you that Configuration Server is unavailable and that the configuration cannot be updated, continue with the installation. Following the installation, you must complete the information on the Start Info tab to ensure that SIP Server will run.

Verifying Starting Parameters

When installation is complete, verify the information on the Start Info tab to ensure that SIP Server will run. See "Verifying the manual installation of SIP Server" on page 49.

Wizard Deployment of SIP Server

Configuration wizards facilitate component deployment. SIP Server configuration and installation involves many steps, and Genesys strongly recommends that you set up SIP Server using the Wizard rather than manually. The SIP Server Configuration Wizard guides you through a series of steps and options to customize your deployment of SIP Server.

Wizard Configuration of SIP Server

The first step to take for a Wizard-based configuration is to install and launch Genesys Wizard Manager. (Refer to the *Framework 7.6 Deployment Guide* for instructions.) When you first launch Genesys Wizard Manager, it suggests that you set up the Management Layer and then the Framework. The Framework setup begins with configuring and creating the objects related to SIP Server, starting with the Switch and Switching Office objects, and the SIP Server's Application object itself.

Note: With the wizard, you create your SIP Server Application object in the course of creating your Switch object.

During creation of the Switch object, you also have an opportunity to run the Log Wizard to set up SIP Server logging. Then, you can specify values for the most important SIP Server options. Finally, you can create contact center objects related to SIP Server, such as DNs, Agent Logins, and some others.

Note: During configuration of a `Switch` object, the Wizard prompts you to copy a SIP Server installation package to an assigned computer. After that package is copied to the destination directory on the SIP Server host, complete the last steps of the SIP Server configuration. Then, install SIP Server on its host.

After you complete the Framework configuration, the Genesys Wizard Manager screen no longer prompts you to set up the Framework. Instead, it suggests that you set up your solutions or add various contact center objects to the Framework configuration, including the `Switch`, `DNs` and `Places`, `Agent Logins`, `Agent Groups`, `Place Groups`, and, in a multi-tenant environment, a `Tenant`. In each case, click the link for the object you wish to create. Again, you create a new SIP Server `Application` object in the course of creating a new `Switch` object.

Wizard Installation of SIP Server

After creating and configuring your SIP Server and its related components with the wizard, you can proceed to SIP Server installation. That installation process closely mimics that of previously installed components.

Note: Certain Wizard-related procedures are not described in this document. Refer to the *Framework 7.6 Deployment Guide* for general instructions.

Warning! Genesys does not recommend installation of its components via a Microsoft Remote Desktop connection. The installation should be performed locally.

Procedure: Installing SIP Server on UNIX using the wizard

Start of procedure

1. In the directory to which the SIP Server installation package was copied during Wizard configuration, locate a shell script called `install.sh`.
2. Run this script from the command prompt by typing `sh` and the file name. For example: `sh install.sh`.
3. When prompted, confirm the host name of the computer on which SIP Server is to be installed.
4. When prompted, confirm the application name of the SIP Server that is to be installed.

5. Specify the destination directory into which SIP Server is to be installed, with the full path to it.
6. If asked which version of the product to install, the 32-bit or the 64-bit, choose the one appropriate to your environment.
7. Specify the license information that SIP Server is to use.
8. As soon as the installation process is finished, a message appears announcing that installation was successful. The process places SIP Server in the directory with the name specified during the installation.

End of procedure

Next Steps

- To test your configuration and installation, go to Chapter 4, "Starting and Stopping SIP Server," on page 69, and try it out.
- To configure and install redundant SIP Servers, see Chapter 3, "High-Availability Deployment," on page 51.
- To install T-Servers for a multi-site environment, proceed to Chapter 10, "Multi-Site Support," on page 267.

Procedure:
Installing SIP Server on Windows using the wizard

Start of procedure

1. Open the directory to which the SIP Server installation package was copied during Wizard configuration.
2. Locate and double-click `Setup.exe` to start the installation. The `Welcome` screen launches.
3. When prompted, specify the connection parameters to the Configuration Server associated with this SIP Server.
4. Identify the SIP Server `Application` object in the Configuration Layer to be used by this SIP Server.
5. Specify the license information that SIP Server is to use.
6. Specify the destination directory into which SIP Server is to be installed.
7. Click `Install` to begin the installation.
8. Click `Finish` to complete the installation.

By default, SIP Server is installed as a Genesys service (Windows `Services`) with `Automatic` startup type.

End of procedure

Next Steps

- To test your configuration and installation, go to Chapter 4, "Starting and Stopping SIP Server," on page 69, and try it out.
- To configure and install redundant SIP Servers, see Chapter 3, "High-Availability Deployment," on page 51.
- To install SIP Servers for a multi-site environment, proceed to Chapter 10, "Multi-Site Support," on page 267.

Manual Deployment of SIP Server

Deploying SIP Server manually requires that you configure a number of different objects in the Configuration Layer prior to setting up your SIP Server objects and then installing SIP Server. This section describes the manual deployment process.

Manual Configuration of Telephony Objects

This section describes how to manually configure SIP Server telephony objects if you are using Configuration Manager.

Recommendations

Genesys recommends registering (configuring) only those entities you plan to use in the current configuration. The more data there is in the Configuration Database, the longer it takes for the CTI setup to start, and the longer it will take to process configuration data. Remember that adding configuration objects to the Genesys Configuration Database does not cause any interruption in contact center operation.

Depending on how much work is required to manually configure all applications and objects, consider registering more Person objects first, with a set of privileges that lets them perform configuration tasks.

Switching Offices

Your telephony network may contain many switching offices, but you should only configure those that are involved with customer interactions.

Using Configuration Manager, be sure to register a Switching Office object of type SIP Switch that accommodates your Switch object under Environment. Until you have done this, you cannot register a Switch object under Resources (single-tenant environment) or a Tenant (multi-tenant environment).

Note: The value for the switching office name must not have spaces in it.

Switches

1. Configure a `Switch` object for each switch on your telephony network. Assign each `Switch` object to the appropriate SIP Server `Application` object.
2. If implementing the multi-site configuration, specify access codes for all switches on the network so that the call-processing applications can route and transfer calls between switches.

   Two types of access codes exist in a Genesys configuration:

   - Default access codes that specify how to reach this switch from any other switch in the Genesys environment.
   - Switch-to-switch access codes that specify how to reach a particular switch from any other switch. Use this type when either a nondefault dial number or routing type is required between any two locations. When a switch-to-switch access code is configured, its value has a higher priority than that of a default access code.

   See Chapter 10, "Multi-Site Support," on page 267, for step-by-step instructions.

Note: When the numbering plan uses unique directory number (DN) assignment across sites and multi-site routing is not used, you do not have to configure access codes.

DNs and Agent Logins

For each SIP Server for which you are configuring DNs, you must configure all DNs that agents and their supervisors use in day-to-day contact center operation—so-called *seat-related DNs*—such as Extensions and ACD Positions. Otherwise, SIP Server does not register such DNs.

1. To configure telephony objects within each switch, consult the switch documentation. For configuration information specific to your SIP devices, see Chapter 5, "SIP Devices Support," on page 77.
2. Check the numbering plan for different types of DNs, to see if you can save time by registering Ranges of DNs. Usually, DNs of the same type have consecutive numbers, which will make an otherwise tedious configuration task easy. `Agent Login` objects almost always have consecutive numbers, which means you can register them through the Range of Agent Logins feature as well.
3. If you plan to use Virtual Queues and Virtual Routing Points in the contact center operation, Genesys recommends registering them after you have outlined the call-processing algorithms and identified your reporting needs.

Note: Remember that CTI applications, not the switch, generate telephony events for DNs of these types.

Multi-Site Operations

See the section, “Configuring Multi-Site Support” on page 307, for information on setting up DNs for multi-site operations.

Manual Configuration of SIP Server

Note: Use the *Framework 7.6 Deployment Guide* to prepare accurate configuration information. You may also want to consult *Configuration Manager Help,* which contains detailed information about configuring objects.

Recommendations

Genesys recommends using an Application Template when you are configuring your SIP Server application. The Application Template for SIP Server contains the most important configuration options set to the values recommended for the majority of environments. When modifying configuration options for your SIP Server application later in the process, you can change the values inherited from the template rather than create all the options by yourself.

Procedure: Configuring SIP Server manually

Start of procedure

1. Follow the standard procedure for configuring all `Application` objects to begin configuring your SIP Server `Application` object. Refer to the *Framework 7.6 Deployment Guide* for instructions.
2. In a multi-tenant environment, specify the `Tenant` to which this SIP Server belongs on the `General` tab of the `Properties` dialog box.
3. On the `Connections` tab, add all Genesys applications to which SIP Server must connect.

Note: For multi-site deployments, you should also specify SIP Server connections on the `Connections` tab for any SIP Servers that may transfer calls directly to each other.

4. On the `Options` tab, specify values for configuration options as appropriate for your environment.

Note: For SIP Server option descriptions, see Chapter 8, "SIP Server Configuration Options," on page 197. The configuration options common to all T-Servers are described in the Chapter 12, "T-Server Common Configuration Options," on page 343 chapter. SIP Server also uses common Genesys log options, described in the Chapter 11, "Common Configuration Options," on page 321.

5. In a multi-site environment, you must complete additional SIP Server configuration steps to support multi-site operations; see Chapter 10, "Multi-Site Support," on page 267.

End of procedure

Next Steps

- See "Manual Installation of SIP Server" on page 47.

Manual Installation of SIP Server

The following directories on the Genesys 7.6 SIP Server product CD contain SIP Server installation packages:

- `SIP_Server/<component>/<platform>` for UNIX installations, where `<component>` is SIP Server or DMX, and `<platform>` is your operating system.
- `SIP_Server\<component>\windows` for Windows installations, where `<component>` is SIP Server or DMX.

Procedure:
Installing SIP Server on UNIX manually

Note: During installation on UNIX, all files are copied into the directory you specify. No additional directories are created within this directory. Therefore, do not install different products into the same directory.

Start of procedure

1. In the directory to which the SIP Server installation package was copied, locate a shell script called `install.sh`.
2. Run this script from the command prompt by typing `sh` and the file name. For example: `sh install.sh`.

3. When prompted, confirm the host name of the computer on which SIP Server is to be installed.
4. When prompted, specify the host and port of Configuration Server.
5. When prompted, enter the user name and password to access Configuration Server.
6. When prompted, select the SIP Server application you configured in "Configuring SIP Server manually" on page 46 from the list of applications.
7. Specify the destination directory into which SIP Server is to be installed, with the full path to it.
8. If asked which version of the product to install, the 32-bit or the 64-bit, choose the one appropriate to your environment.
9. Specify the license information that SIP Server is to use: either the full path to, and the name of, the license file, or the license server parameters.
10. As soon as the installation process is finished, a message appears announcing that installation was successful. The process places SIP Server in the directory with the name specified during the installation.

End of procedure

Next Steps

- To verify manual installation, go to "Verifying the manual installation of SIP Server" on page 49.
- To test your configuration and installation, go to Chapter 4, "Starting and Stopping SIP Server," on page 69, and try it out.
- To configure and install redundant SIP Servers, see Chapter 3, "High-Availability Deployment," on page 51.
- To install SIP Servers for a multi-site environment, proceed to Chapter 10, "Multi-Site Support," on page 267.

Procedure: Installing SIP Server on Windows manually

Start of procedure

1. In the directory to which the SIP Server installation package was copied, locate and double-click `Setup.exe` to start the installation.
2. When prompted, specify the connection parameters to the Configuration Server associated with this SIP Server.

3. When prompted, select the SIP Server Application object you configured in "Configuring SIP Server manually" on page 46 from the list of applications.
4. Specify the license information that SIP Server is to use: either the full path to, and the name of, the license file, or the license server parameters.
5. Specify the destination directory into which SIP Server is to be installed.
6. Click Install to begin the installation.
7. Click Finish to complete the installation.

By default, SIP Server is installed as a Genesys service (Windows Services) with Automatic startup type.

End of procedure

Next Steps

- To verify manual installation, go to "Verifying the manual installation of SIP Server" on page 49.
- To test your configuration and installation, go to Chapter 4, "Starting and Stopping SIP Server," on page 69, and try it out.
- To configure and install redundant T-Servers, see Chapter 3, "High-Availability Deployment," on page 51.
- To install SIP Servers for a multi-site environment, proceed to Chapter 10, "Multi-Site Support," on page 267.

Procedure: Verifying the manual installation of SIP Server

Purpose: To verify the completeness of the manual installation of SIP Server to ensure that SIP Server will run.

Prerequisites

- Installing SIP Server on UNIX manually, page 47
- Installing SIP Server on Windows manually, page 48

Start of procedure

1. Open the Properties dialog box for a corresponding Application object in Configuration Manager.
2. Verify that the State Enabled check box on the General tab is selected.
3. Verify that the Working Directory, command-line, and Command-Line Arguments are specified correctly on the Start Info tab.

4. Click `Apply` and `OK` to save any configuration updates.

End of procedure

Next Steps

At this point, you have either used the wizard to configure and install SIP Server, or you have done it manually, using Configuration Manager. In either case, if you want to test your configuration and installation, go to Chapter 4, “Starting and Stopping SIP Server,” on page 69, and try it out. Otherwise, if you want to configure and install redundant SIP Servers, see Chapter 3, “High-Availability Deployment,” on page 51. If you want to install SIP Server for a multi-site environment, proceed to Chapter 10, “Multi-Site Support,” on page 267.

Chapter

3

High-Availability Deployment

This chapter describes the general steps for setting up a high-availability (HA) environment for SIP Server. The high-availability architecture implies the existence of redundant applications, a primary and a backup. These are monitored by a management application so that, if one application fails, the other can take over its operations without any significant loss of contact center data.

The Framework Management Layer currently supports two types of redundant configurations: `warm standby` and `hot standby`. This chapter describes the redundant architecture and how to configure SIP Server so that it operates with either type. This chapter contains the following sections:

Overview

A SIP Server high-availability deployment utilizes the concept of a *Virtual IP address*, hiding two hosts of primary and backup SIP Servers behind one Virtual IP address. SIP endpoints and gateways are configured to send all SIP messages to SIP Server using this single Virtual IP address. The Virtual IP address is preserved during the switchover, and SIP messages are delivered only to the host with SIP Server running in *primary* mode. When the Management Layer detects a SIP Server failure, it disables the SIP messages traffic to the backup SIP Server and enables it to the primary server. This method effectively hides the switchover (and switchback) from the endpoints and gateways.

Note: The traffic switchover from one host to another may take a few seconds, during which some SIP messages could be lost and not delivered to the primary SIP Server.

The Management Layer and Configuration Layer components and T-Library clients must use a unique IP address for the host for communication to SIP Server and Local Control Agent (LCA).

SIP Server supports two types of redundant configurations: `warm standby` and `hot standby` using the Network Load Balancing (NLB) cluster (Windows) or the Virtual IP address (UNIX).

The configuration option `sip-address` identifies the SIP interface address of SIP Server: the Virtual IP address of the NLB cluster for Windows, and the Virtual IP interface for UNIX. This option must be configured for both primary and backup servers, and the option setting must be the same.

Note: If the NLB cluster (Windows) is configured to operate in `unicast` mode (the default), two network adapters are required on each host to enable communication between the cluster hosts. One adapter handles the network traffic for cluster operations using the Virtual IP address. The second adapter handles the traffic from the Management Layer and Configuration Layer components, from T-Library clients, and from primary and backup SIP Servers communication using the unique host IP address.

Figure 7 illustrates SIP Server HA deployment.

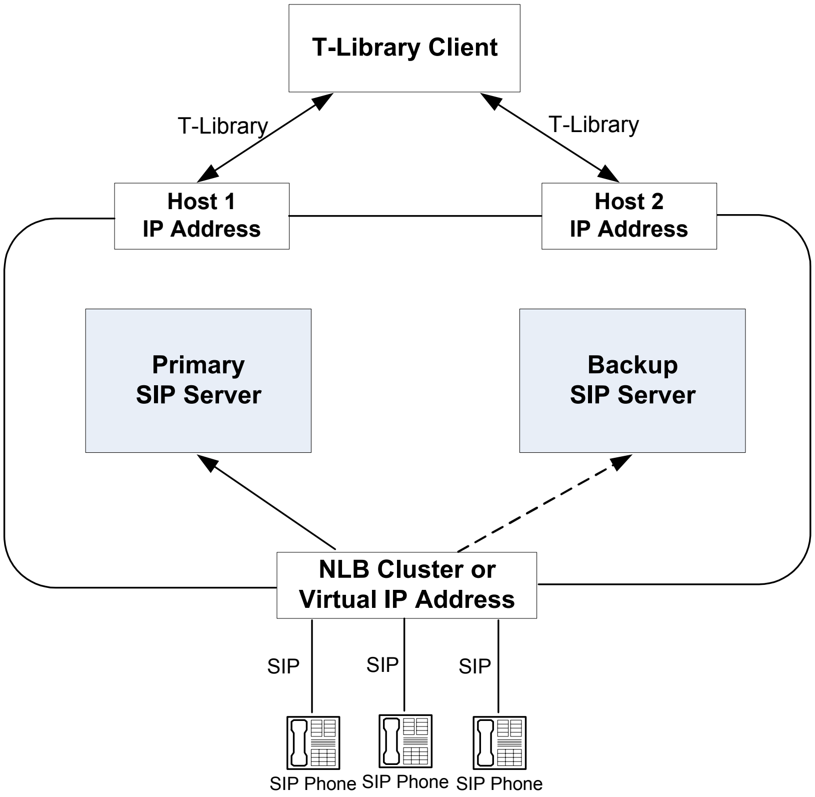


Figure 7: SIP Server High-Availability Deployment

Warm Standby Redundancy Type

Genesys uses the expression *warm standby* to describe the redundancy type in which a backup server application remains initialized and ready to take over the operations of the primary server. The `warm standby` redundancy type reduces to a minimum the inability to process interactions that may have originated during the time it took to detect the failure. It also eliminates the need to bring a standby server online, thereby increasing solution availability.

SIP Server supports `warm standby` the same way as others T-Servers do. There is no propagation of information from a primary SIP Server about calls, devices, monitoring subscriptions, and agent states to a backup SIP Server.

Hot Standby Redundancy Type

Genesys uses the expression *hot standby* to describe the redundancy type in which a backup server application remains initialized, clients connect to both the primary and backup servers at startup, and the backup server data is synchronized from the primary server.

The `hot standby` SIP Server deployment is similar to `warm standby`. It is based on the architecture and design currently used in other Genesys T-Servers.

Primary and backup SIP Servers must be deployed either using the NLB cluster (Windows) or the Virtual IP address (UNIX).

Data synchronization and existing client connections to the backup guarantee higher availability of a component. Data synchronization comprises information about calls, device states, monitoring subscriptions, and agent states.

SIP Server supports `hot standby` mode for established calls, for calls in the `ringing` state, and for calls that are parked on a Routing Point. Any telephony function can now be performed on all synchronized calls after a switchover.

Known Limitations

The following `hot standby` limitations exist in this implementation:

- Client requests sent during the failure and switchover may be lost.
- SIP requests sent by the endpoints during the failure and switchover may be lost.
- SIP Server does not synchronize interactions that begin before it starts.
- Some T-Library events might be duplicated or lost.
- Reference IDs from client requests can be lost in events.

Prerequisites

This section presents basic requirements and recommendations for configuring and using redundant SIP Servers.

Requirements

You must install the Management Layer if you are installing redundant SIP Server applications. In particular, install LCA on each computer that runs SIP Server.

Warning! Genesys strongly recommends that you install the backup and primary SIP Servers on different host computers.

Synchronization Between Redundant SIP Servers

When SIP Servers operate in a high-availability environment, the backup SIP Server must be ready to take on the primary role when required. For this purpose, both SIP Servers must be running and must have the same information. When you configure redundant SIP Servers to operate with the `hot standby` type, the primary SIP Server uses the connection to the backup to deliver synchronization updates. Genesys recommends that you enable the

Advanced Disconnect Detection Protocol (ADDP) for this connection. Do so using the configuration options in the “Backup-Synchronization Section” section. See Chapter 12, “T-Server Common Configuration Options,” on page 343 for option descriptions.

Note: The option `internal-registrar-persistent` must be set to `true` to enable Configuration Server to propagate changes of the `contact` information to the backup SIP Server when an endpoint registers at SIP Server.

Configuration Warnings

When configuring SIP Servers to support either the `warm standby` or `hot standby` redundancy type, remember:

- When at least one of the two SIP Servers that operate in a redundant mode is running, do not change a redundancy type, host, or port in either SIP Server configuration.
- When both the primary and backup SIP Servers are running, do not remove the backup SIP Server `Application` object from the configuration.

You are responsible for the option synchronization in the configuration of the primary and backup SIP Servers; Configuration Server does not synchronize either options or their values in different SIP Server `Application` objects. That is, you must configure both SIP Servers to have the same options with the same values. If you change a value in one SIP Server configuration, you must manually change it in the other SIP Server configuration. However, the log options in the primary SIP Server can differ from those in the backup SIP Server configuration.

Warm Standby Configuration

This section describes how to configure redundant SIP Servers to work with the `warm standby` redundancy type, including details on their connections and settings.

General Order of Deployment

The general guidelines for SIP Server `warm standby` configuration are:

Wizard Deployment

If you used wizards to configure SIP Servers and selected the `warm standby` redundancy type, no additional configuration is required for your SIP Servers.

Manual Deployment

If you did not use wizards to configure SIP Servers:

1. Manually configure two SIP Server `Application` objects as described in “Manual Configuration of SIP Server” on page 46.

2. Make sure the Switch object is configured for the switch these SIP Servers should serve, as described in "Manual Configuration of SIP Server" on page 46.
3. Modify the configuration of the primary and backup SIP Servers as instructed in the following sections.

After completing the configuration steps, ensure that both SIP Servers are installed.

Manual Modification of SIP Servers for Warm Standby

Modify the configuration of both the primary and backup SIP Server Application objects as described in the following sections.

Note: Starting with release 7.5, you can configure multiple ports for any application of type server. When multiple ports are configured for a server in a warm standby redundancy pair, the primary and backup servers must match with respect to the number of ports, their Port IDs, and the Listening Mode settings.

Procedure: Modifying the primary SIP Server configuration for warm standby

Start of procedure

1. Stop both the primary and backup SIP Servers if they are already running.
2. Open the Configuration Manager main window.
3. Open the Properties dialog box of the Application object for the SIP Server that you want to configure as a primary server.
4. Click the Switches tab.
5. Ensure that it specifies the Switch object that this SIP Server Application object should serve. If necessary, select the correct Switch object using the Browse button.
6. Click Apply to save the configuration changes.
7. Click the Server Info tab.
8. Specify the SIP Server Application object you want to use as the backup server. Use the Browse button next to the Backup Server text box to locate the backup SIP Server Application object.
9. Select Warm Standby as the Redundancy Type.
10. Click Apply to save the configuration changes.

11. Click the Start Info tab.
12. Select Auto-Restart.
13. Click Apply and OK to save the configuration changes.

End of procedure

Next Steps

- Modifying the backup SIP Server configuration for warm standby, page 57

Procedure: Modifying the backup SIP Server configuration for warm standby

Start of procedure

1. Make sure the two SIP Server are *not* running.
2. Open the Configuration Manager main window.
3. Open the Properties dialog box of the Application object for the SIP Server that you want to configure as a backup server.
4. Click the Switches tab.
5. Using the Browse button, select the same Switch object you associated with the primary SIP Server Application object.
6. Click Apply to save the configuration changes.
7. Click the Start Info tab.
8. Select Auto-Restart.
9. Click Apply and OK to save the configuration changes.

End of procedure

Hot Standby Configuration

This section describes how to configure redundant SIP Servers to work with the hot standby redundancy type, including details on their connections and settings.

General Order of Deployment

The general guidelines for SIP Server hot standby configuration are:

Wizard Deployment

- If you used wizards to configure SIP Servers and selected the hot standby redundancy type, no additional configuration is required for your SIP Servers.

Manual Deployment

- If you did not use wizards to configure SIP Servers:
  - **a.** Manually configure two SIP Server Applications objects as described in “Manual Configuration of Telephony Objects” on page 44.
  - **b.** Make sure the Switch object is configured for the switch these SIP Servers should serve, as described in “Manual Configuration of Telephony Objects” on page 44.
  - **c.** Modify the configuration of the primary and backup SIP Servers as instructed in the following sections.

After completing the configuration steps, ensure that both SIP Servers are installed.

Manual Modification of SIP Servers for Hot Standby

Modify the configuration of both the primary and backup SIP Server Application objects for hot standby redundancy as described in the following sections.

Note: Starting with release 7.5, you can configure multiple ports for any application of type server. When multiple ports are configured for a server in a hot standby redundancy pair, the primary and backup servers must match with respect to the number of ports, their Port IDs, and the Listening Mode settings.

Procedure:
Modifying the primary SIP Server configuration for hot standby

Start of procedure

1. Stop both primary and backup SIP Servers if they are already running.
2. Open the Configuration Manager main window.
3. Open the Properties dialog box of the Application object for the SIP Server that you want to configure as a primary server.
4. Click the Switches tab.

5. Ensure that it specifies the `Switch` object that this SIP Server `Application` object should serve. If necessary, select the correct `Switch` object using the `Browse` button.
6. Click `Apply` to save the configuration changes.
7. Click the `Server Info` tab.
8. In the `Ports` section, select the port to which the backup server will connect for HA data synchronization, and click `Edit Port`.
   a. In the `Port Properties` dialog box, on the `Port Info` tab, select the `HA sync` check box.
   b. Click `OK`.

Note: If the `HA sync` check box is not selected, the backup SIP Server will connect to the *default* port of the primary SIP Server.

9. Specify the SIP Server `Application` object you want to use as the backup server. Use the `Browse` button next to the `Backup Server` field to locate the backup SIP Server `Application` object.
10. Select `Hot Standby` as the `Redundancy Type`.
11. Click `Apply` to save the configuration changes.
12. Click the `Start Info` tab.
13. Select `Auto-Restart`.
14. Click `Apply` to save the configuration changes.
15. To enable ADDP between the primary and backup SIP Servers, click the `Options` tab. Open or create the `backup-sync` section, and configure corresponding options.

Note: For a list of options and valid values, see the “Backup-Synchronization Section” section in Chapter 12, “T-Server Common Configuration Options,” on page 343.

16. Click `Apply` and `OK` to save the configuration changes.

End of procedure

Next Steps

- Modifying the backup SIP Server configuration for hot standby, page 60

Procedure:
Modifying the backup SIP Server configuration for hot standby

Start of procedure

1. Make sure the two SIP Servers are *not* running.
2. Open the Configuration Manager main window.
3. Open the `Properties` dialog box of the `Application` object for the SIP Server that you want to configure as a backup server.
4. Click the `Switches` tab.
5. Using the `Browse` button, select the same `Switch` object you associated with the primary SIP Server `Application` object.
6. Click the `Start Info` tab.
7. Select `Auto-Restart`.
8. Click the `Options` tab.
9. Modify the values for all necessary configuration options. Genesys recommends that you set all configuration options for the backup SIP Server to the same values as for the primary SIP Server; the only exceptions are the log options and the `server-id` option.
10. Click `Apply` and `OK` to save the configuration changes.

End of procedure

Warm/Hot Standby Deployment on Solaris

The `warm/hot standby` redundancy on Solaris is achieved by hiding two hosts, on which primary and backup servers are running, behind the same IP address. This method effectively hides the switchover (and switchback) from the endpoints and gateways.

Unique and Common Interfaces

Two Sun hosts on the same network subnet (“host1” and “host2”) are used. Each host has two logical IP interfaces assigned to a single Ethernet interface. The first IP interface (called “unique” in this example) has the unique IP address of the host on the subnet. The second IP interface (called “common” in this example, though it is sometimes referred to as “virtual”) has an address on the same subnet and this address is shared between two hosts.

The unique interface should always be activated on each host. T-Library clients, and management and configuration software also use the unique interface of the hosts. The common interface should be activated only if SIP Server is working in primary role on that host. Otherwise, the common interface should be deactivated. SIP User Agents (endpoints) are only aware of the common interface.

The address and hostname of the common interface, as well as addresses and hostnames of the unique interfaces, must be known to the DNS server.

On each host, the `/etc/hosts` file must contain the record about the common interface, in the form:

```
<IP_address> <common_name_of_the_host>
```

On each host, inside the `/etc` directory, you must create the file:

```
hostname.name_of_ethernet_interface:1
```

Where `name_of_ethernet_interface` is the actual name of the Ethernet interface on that machine—for example; `hostname.dmfe0:1`.

This file must contain the hostname of the common interface as it is known to the DNS server and is recorded inside the `/etc/hosts` file.

Management of the state of the common interface is provided by the Sun administrative command `ifconfig`.

To bring up the common interface, you must issue the command:

```
ifconfig name_of_ethernet_interface:1 up
```

To bring it down, you must issue the command:

```
ifconfig name_of_ethernet_interface:1 down
```

This command should be wrapped by shell batch files. Each host must contain two such shell files, one to bring up the common interface, and one to bring it down.

The Configuration Layer contains four applications, which are referenced by these shell files as third-party servers.

Alarm Reaction Scripts

Four corresponding Alarm Reaction Scripts, which are responsible for the start of each of the third-party servers mentioned above, should be created in order to execute these shell files:

1. Virtual IP Down on host1
2. Virtual IP Up on host1
3. Virtual IP Down on host2
4. Virtual IP Up on host2

Alarm Conditions

You should also create four corresponding Alarm Conditions:

1. Primary for host1

   For Warm Standby: Alarm Condition 1 is linked with Log Event `00-04562 Warm Standby (Primary) mode activated` for the server that is running on host1. It starts Alarm Reaction Scripts 2 and 3.

   For Hot Standby: Alarm Condition 1 is linked with Log Event `00-04563 Hot Standby (Primary) mode activated` for the server that is running on host1. It starts Alarm Reaction Scripts 2 and 3.

2. Primary for host2

For Warm Standby: Alarm Condition 2 is linked with Log Event `00-04562 Warm Standby (Primary) mode activated` for the server that is running on host2. It starts Alarm Reaction Scripts 1 and 4.

For Hot Standby: Alarm Condition 2 is linked with Log Event `00-04563 Hot Standby (Primary) mode activated` for the server that is running on host2. It starts Alarm Reaction Scripts 1 and 4.

3. Backup for host1

   For Warm Standby: Alarm Condition 3 is linked with Log Event `00-04560 Warm Standby (backup) mode activated` for the server that is running on host1. It starts Alarm Reaction Script 1.

   For Hot Standby: Alarm Condition 3 is linked with Log Event `00-04561 Hot Standby (backup) mode activated` for the server that is running on host1. It starts Alarm Reaction Script 1.

4. Backup for host2

   For Warm Standby: Alarm Condition 4 is linked with Log Event `00-04560 Warm Standby (backup) mode activated` for the server that is running on host2. It starts Alarm Reaction Script 3.

   For Hot Standby: Alarm Condition 4 is linked with Log Event `00-04561 Hot Standby (backup) mode activated` for the server that is running on host2. It starts Alarm Reaction Script 3.

Notes: The `Cancel timeout` parameter for all Alarm Conditions must be set to `1` in Configuration Manager.

If your host machines have more than one Ethernet interface, Genesys server clients should use the unique IP interface of the same Ethernet interface where the common IP interface is assigned.

Warm/Hot Standby Deployment on Windows

The `warm/hot standby` redundancy on Windows is achieved by using the Network Load Balancing (NLB) technology included in Microsoft Windows Server 2003. The unique and fully distributed architecture of NLB enables it to deliver failover protection without requiring any special hardware.

Network Load Balancing

A Network Load Balancing cluster uses the concept of a *Virtual IP address*. The SIP switch and/or all endpoints are configured to send all SIP requests to SIP Server using this single address. The cluster software delivers the requests to only one of the servers in the cluster, and switches over to another SIP Server if it detects a SIP Server failure.

The advantages of a Network Load Balancing solution are:

- It requires no special configuration on the SIP switch or SIP endpoints—you need to configure only one IP address.
- The switch or endpoints do not require any special actions during switchover.
- Orderly primary/backup switchover can be invoked from the Genesys Management Layer (for maintenance purposes).
- No special hardware is required on the NLB cluster machines.

Configuration Procedures

This section provides detailed procedures for configuring the various elements required for configuring the `warm/hot standby` redundancy on Windows.

Procedure: Configuring cluster hosts

Start of procedure

1. On each cluster host, (see Figure 7 on page 53), install Windows Server 2003 and enable the NLB feature.

   Notes: Genesys recommends that you install the Management and Configuration Layers outside the NLB cluster. Genesys Universal Router Servers can work outside or inside the cluster.

   The NLB architecture uses the subnet switch to deliver incoming network traffic to all cluster hosts. As result the entire incoming network traffic for the Virtual IP address distributed by the subnet switch to all ports. Genesys recommends that you create a separate Virtual LAN for all cluster adapters from the same cluster, or use a dedicated switch for cluster adapters.

2. On each cluster host, install SIP Servers and LCAs.
3. On each cluster host, configure the NLB parameters as follows:
   - `Port Range`: Must include a port as configured in the `sip-port` option.
   - `Protocols`: `UDP` and `TCP`, if required.
   - `Filtering mode`: `Multiple` (for multiple hosts).
   - `Affinity`: `None` or `Single`.

- `Load weight: Equal.`

A datagram from the SIP switch arrives at both hosts of the cluster, but the NLB Driver lets traffic go to the Application Layer of only one cluster host.

For more information about how to administer Network Load Balancing technology, see the Windows 2003 Server documentation or online help.

Note: You must configure the softswitch or endpoints to send datagrams to the virtual address of the NLB cluster—that is, to the port as it is configured in the `sip-port` option.

End of procedure

Next Steps

- Creating a batch file

Procedure: Creating a batch file

Prerequisites

- Configuring cluster hosts, page 63

Start of procedure

- On the host where SIP Server is installed, create a batch file containing the following commands (see "Recommended Sample Batch File" on page 65):
  - `wlbs enable` *`XXXX`* `<cluster_name>:<ID_Primary>`
  - `wlbs disable` *`XXXX`* `<cluster_name>:<ID_Backup>`

  Where:
  - `wlbs` is name of the Windows utility used to control Network Load-Balancing.
  - `enable` is the command to enable traffic handling for the host on which the primary SIP Server is running.
  - *`XXXX`* is the decimal value number of the port as it is configured in the `sip-port` option.
  - `cluster_name` is the Virtual IP address of the cluster.
  - `ID_Primary` is the unique host ID of the host where the primary SIP Server is currently running.
  - `disable` is the command to disable traffic handling for the host on which the backup SIP Server is running.

- ID_Backup is the unique host ID of the host where the backup SIP Server is currently running.

If these commands are issued at the moment when SIP Server changes its roles, the primary SIP Server always handles the traffic.

End of procedure

Next Steps

- Configuring primary/backup SIP Servers

Recommended Sample Batch File

This section contains a sample of the batch file you should use to monitor the state of the host as a member of the NLB cluster. This script should be configured as a third-party server on host1 of the cluster.

```
title Tserver and cluster monitor
wlbs enable 5060 clustername:1
wlbs disable 5060 clustername:2
:alive
@sleep 1
@pulist |findstr /C:"sip_server.exe" >dump.log
@if errorlevel 1 goto ex
@wlbs query testcluster:2 |findstr /C:"Host 1 converg" >dump.log
@if not errorlevel 1 goto alive
@date /T>>monitor.log
@TIME /T>>monitor.log
@echo Host not in cluster anymore,t-server will be killed>>
monitor.log
tskill sip_server*
exit
:ex
@date /T>>monitor.log
@TIME /T>>monitor.log
@echo t-server is not running anymore>> monitor.log
exit
```

For host2 of the cluster, use the same script but, with host numbers reversed.

For a technical overview of Network Load Balancing and Windows Server 2003 clustering technologies, go to the Microsoft website.

Procedure:
Configuring primary/backup SIP Servers

Prerequisites

- Configuring cluster hosts, page 63
- Creating a batch file, page 64

Start of procedure

1. In Configuration Manager, configure SIP Server Application objects in warm standby mode (that is, in the primary/backup relationship).
2. On the Options tab in the TServer section, configure the following options:
   - sip-port: Set the same value for both SIP Servers.
   - sip-address: Set to the SIP interface address of SIP Server, which is the Virtual IP address of the NLB cluster for Windows. This option must be configured for both primary and backup servers, and the option setting must be the same.
3. Create a new Application object that uses the Third Party Server Application Template.
4. On the Server Info tab of the new Application object, set the host to the name of the host on which SIP Server is installed. Set a valid value for the communication port. Use the default values for all other parameters on this tab.
5. On the Start Info tab, set the working directory to the location of the batch file you created in “Creating a batch file” on page 64. For the command-line parameter, use the name of that batch file. Use the default values for all other parameters on this tab.
6. In the Configuration Database, for the host name (TCP/IP addresses) for each SIP Server, do *not* use the virtual TCP/IP address of the cluster, but use the actual TCP/IP address of the host on which SIP Server will be running.

End of procedure

Next Steps

- Configuring Alarm Conditions

Procedure: Configuring Alarm Conditions

Purpose: To create and set Alarm Conditions using the Solution Control Interface (SCI) for both the primary and backup SIP Servers. For each SIP Server two alarm conditions should be created.

Prerequisites

- Configuring cluster hosts, page 63
- Creating a batch file, page 64
- Configuring primary/backup SIP Servers, page 66

Start of procedure

1. Create a new Alarm Condition.
2. For Warm Standby: Set the Alarm Detection to Detection Log Event `00-04562 (Warm-Standby Primary mode activated).`
3. For Hot Standby: Set the Alarm Detection to Detection Log Event `00-04563 Hot Standby (Primary) mode activated.`
4. Set the `Alarm Source` to the `Application` object corresponding to the relevant SIP Server.
5. On the `Assigning Alarm Reactions` tab, click `Add.` Right-click to create a new script.
6. On the `Alarm Reaction` tab, select `Start Specified Application.`
7. For the `application to start` parameter, choose the Third Party Server `Application` object you created in “Configuring primary/backup SIP Servers” on page 66.
8. Create a new Alarm Condition.
9. For Warm Standby: Set the Alarm Detection to Detection Log Event `00-04560 (Warm-Standby Backup mode activated).`
10. For Hot Standby: Set the Alarm Detection to Detection Log Event `00-04561 Hot Standby (backup) mode activated.`
11. Set the `Alarm Source` to the specific `Application` object corresponding to the relevant SIP Server.
12. On the `Assigning Alarm Reactions` tab, click `Add.` Right-click to create a new script.
13. On the `Alarm Reaction` tab, select `Stop Specified Application.`

14. For the `application to stop` parameter, choose the Third Party Server `Application` object you created in “Configuring primary/backup SIP Servers” on page 66.

End of procedure

Next Steps

At this point, you have learned how to configure and install redundant SIP Servers. Go to Chapter 4, “Starting and Stopping SIP Server,” on page 69, to test your configuration and installation, or continue with Chapter 10, “Multi-Site Support,” on page 267, for more possibilities.

Chapter

4

Starting and Stopping SIP Server

This chapter describes methods for stopping and starting SIP Server, focusing on manual startup for SIP Server. It contains the following sections:

Command-Line Parameters

You can start and stop Framework components using the Management Layer, a startup file, a manual procedure, or the Windows Services Manager.

With all these methods, command-line parameters are usually required for a server application in addition to an executable file name.

Common command-line parameters are as follows:

`-host` The name of the host on which Configuration Server is running.

`-port` The communication port that client applications must use to connect to Configuration Server.

`-app` The exact name of an Application object as configured in the Configuration Database.

`-l`

The license address. Use for the server applications that check out technical licenses. Can be either of the following:

- The full path to, and the exact name of, the license file used by an application. 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 `port@host` format. For example, `-l 7260@ctiserver`.

Note: Specifying the License Manager's host and port parameter eliminates the need to store a copy of a license file on all computers running licensed applications.

`-V`

The version of a Framework component. Note that specifying this parameter does not start an application, but returns its version number instead. You can use either uppercase or lowercase.

`-nco X/Y`

The Nonstop Operation feature is activated; `X` exceptions occurring within `Y` seconds do not cause an application to exit. If the specified number of exceptions is exceeded within the specified number of seconds, the application exits or, if so configured, the Management Layer restarts the application. If the `-nco` parameter is not specified, the default value of `6` exceptions handled in `10` seconds applies. To disable the Nonstop Operation feature, specify `-nco 0` when starting the application.

`-lmspath`

The full path to log messages files (the common file named `common.lms` and the application-specific file with the extension `*.lms`) that an application uses to generate log events. This parameter is used when the common and application-specific log message files are located in a directory other than the application's working directory, such as when the application's working directory differs from the directory to which the application is originally installed.

Note that if the full path to the executable file is specified in the startup command-line (for instance, `c:\gcti\multiserver.exe`), the path specified for the executable file is used for locating the `*.lms` files, and the value of the `lmspath` parameter is ignored.

Note: In the command-line examples in this document, angle brackets indicate variables that must be replaced with appropriate values.

Starting and Stopping with the Management Layer

Procedure: Configuring SIP Server to start with the Management Layer

Start of procedure

1. Open the SIP Server `Application's Properties` dialog box.
2. Click the `Start Info` tab.
3. Specify the directory where the application is installed and/or is to run as the `Working Directory`.
4. Specify the name of the executable file as the `command-line`.
5. Specify command-line parameters as the `Command–Line Arguments`.

   The command-line parameters common to Framework server components are described on page 69.
6. When you are finished, click `Apply`.
7. Click `OK` to save your changes and exit the `Properties` dialog box.

End of procedure

Note: Before starting an application with the Management Layer, make sure the startup parameters of the application are correctly specified in the application's `Properties` dialog box in Configuration Manager.

After its command-line parameters are correctly specified in the `Properties` dialog box, you can start and stop SIP Server from Solution Control Interface (SCI), which is the graphical interface component of the Management Layer. (The starting procedure for SCI is described in the *Framework 7.6 Deployment Guide.*) *Framework 7.6 Solution Control Interface Help* provides complete instructions on starting and stopping applications.

You can also use the Management Layer to start a SIP Server that has failed. To enable SIP Server's auto-restart functionality, select the corresponding check box in the `Application's Properties` dialog box.

Note that when you start (or restart) an application via the Management Layer, the application inherits environment variables from Local Control Agent (LCA), which executes the startup command. Therefore, you must also set the environment variables required by the application for the account that runs LCA.

Warning! *Stopping* an application via the Management Layer is not considered an application failure. Therefore, the Management Layer does not restart applications that it has stopped unless an appropriate alarm condition and alarm reaction are configured for these applications.

Starting with Startup Files

Startup files are files with the extension `run.sh` (on UNIX) or `startServer.bat` (on Windows), which installation scripts create and place into the applications' directories during the installations. These files are created for all Framework server applications except:

- Configuration Server (primary or backup) running on Windows.
- Backup Configuration Server running on UNIX.
- DB Server running on Windows.
- LCA running on either Windows or UNIX.

When using a startup file, verify that the startup parameters the installation script inserted in the startup file are correct. Use the following instructions for UNIX and Windows to start those application for which startup files are created. See the appropriate sections in "Starting Manually" on page 73 to identify which applications should be running for a particular application to start.

Procedure:
Starting SIP Server on UNIX with a startup file

Start of procedure

1. Go to the directory where an application is installed.
2. Type the following command line:

   ```
   sh run.sh
   ```

End of procedure

Procedure: Starting SIP Server on Windows with a startup file

Start of procedure

To start SIP Server on Windows with a startup file, use either of these methods:

- Go to the directory where an application is installed and double-click the `startServer.bat` icon.

Or

- From the `MS-DOS` window, go to the directory where the application is installed and type the following command-line:

  ```
  startServer.bat
  ```

End of procedure

Starting Manually

When starting an application manually, you must specify the startup parameters at the command prompt, whether you are starting on UNIX or Windows. At the command prompt, command-line parameters must follow the name of the executable file. On the `Shortcut` tab of the Program `Properties` dialog box, command-line parameters must also follow the name of the executable file.

The command-line parameters common to Framework server components are described on page 69.

If an `Application` object name, as configured in the Configuration Database, contains spaces (for example, `T-Server Nortel`), the `Application` name must be surrounded by quotation marks in the command-line:

```
-app "T-Server Nortel"
```

Before starting SIP Server, be sure that the following components are running:

- DB Server that provides access to the Configuration Database
- Configuration Server
- License Manager

Procedure:
Starting SIP Server on UNIX manually

Start of procedure

- Go to the directory where SIP Server is installed, and type the following command-line:

```
sip_server -host <Configuration Server host>
-port <Configuration Server port> -app <SIP Server Application>
-l <license address> -nco [X]/[Y]
```

End of procedure

Procedure:
Starting SIP Server on Windows manually

Start of procedure

- Start SIP Server from either the `Start` menu or the `MS-DOS` window. If you use the `MS-DOS` window, go to the directory where SIP Server is installed, and type the following command-line parameters:

```
sip_server.exe -host <Configuration Server host>
-port <Configuration Server port> -app <T-Server Application>
-l <license address> -nco [X]/[Y]
```

End of procedure

Verifying Successful Startup

After executing the startup command, you might want to check whether it was successful.

If you used the Management Layer to start SIP Server, check whether Solution Control Interface displays `Started` or `Service Unavailable` status for the corresponding application. Refer to the "Troubleshooting" section of the *Framework 7.6 Management Layer User's Guide* if the startup command does not result in either `Started` or `Service Unavailable` status for some period of time.

If you start your SIP Server with startup files or manually, and if you have configured logging to console or a log file, check the log for messages similar to the following:

- SIP Server log file: `Link connected`

Stopping Manually

The following stopping procedures apply to Genesys server applications, such as DB Server, Configuration Server, Message Server, Local Control Agent, Solution Control Server, HA Proxy, T-Server, SIP Server, and Stat Server.

Procedure: Stopping SIP Server on UNIX manually

Start of procedure

To stop a server application from its console window on UNIX, use either of these commands:

- `Ctrl+C`
- `kill <process number>`

End of procedure

Procedure: Stopping SIP Server on Windows manually

Start of procedure

To stop a server application on Windows, use either of these commands:

- To stop a server application from its console window on Windows, use the `Ctrl+C` command.
- To stop a server application on Windows, use the `End Task` button on the Windows `Task Manager`.

End of procedure

Starting and Stopping with Windows Services Manager

When starting an application installed as a Windows Service, make sure the startup parameters of the application are correctly specified in the `ImagePath` in the `Application` folder in the `Registry Editor`. The `ImagePath` must have the following value data:

```
<full path>\<executable file name> -service <Application Name as
Service> -host <Configuration Server host>
```

```
-port <Configuration Server port> -app <Application Name>
-l <license address>
```

where the command-line parameters common to Framework server components are described on page 69 and

`-service` The name of the Application running as a Windows Service; typically, it matches the Application name specified in the `-app` command-line parameter.

Framework components installed as Windows Services with the autostart capability are automatically started each time a computer on which they are installed is rebooted.

You can start Framework components installed as Windows Services with the manual start capability with the `Start` button in `Services Manager`.

Note: Use the `Windows Services` window to change the startup mode from `Automatic` to `Manual` and vice versa.

Regardless of a component's start capability, you can stop Framework components installed as Windows Services with the `Stop` button in `Services Manager`.

Next Steps

This chapter concludes SIP Server general deployment. Refer to subsequent chapters in this guide for detailed reference information and any special procedural instructions that pertain to SIP Server.

Chapter

5 SIP Devices Support

This chapter presents reference information for configuring devices and the switch elements of SIP Server. It contains the following sections:

Overview

SIP devices that represent SIP endpoints are configured in Configuration Manager as the following types of DN:

- `Extension` (or `ACD Position`)—An agent's endpoint (SIP Phone)
- `Trunk`—Any external number (for example, a gateway access number)
- `Voice over IP Service`—SIP services (Stream Manager, Music-On-Hold server, and so on)
- `Routing Point`—Used internally by SIP Server
- `ACD Queue`—Used internally by SIP Server

Note: DNs of type `External Routing Point` are also supported by SIP Server. They are not specific to SIP Server and are used by the T-Server Common Part component of SIP Server in a multi-site environment.

Table 1 contains cross-reference information on SIP devices and Genesys DN types. Use this information to configure SIP devices properly in the Configuration Layer.

Table 1: Device Type Cross Reference

SIP Device Type	Genesys DN Type
Endpoints (SIP phones)	Extension (or ACD Position)
Routing Points	Routing Point Routing Queue ACD Queue
Gateway SIP Proxy SIP Server in a multi-site deployment Voice Mail Service (Asterisk only)	Trunk
MCU	Voice over IP Service, with service type set to mcu
Softswitch	Voice over IP Service, with service-type set to softswitch
Music servers	Voice over IP Service, with service-type set to music
Treatment service	Voice over IP Service, with service-type set to treatment
Recording service	Voice over IP Service, with service-type set to recorder
Application service	Voice over IP Service, with service-type set to application

Configuring Devices and Services

This section describes how to configure the SIP device types for SIP Server environments. It contains the following sections:

- "Configuring ACD Queues" on page 79
- "Configuring MCUs" on page 79
- "Configuring Endpoints" on page 81
- "Configuring Gateways" on page 83
- "Configuring Music Servers" on page 85
- "Configuring Routing Points" on page 87
- "Configuring Softswitches" on page 88

- "Configuring an Application Service" on page 89
- "Configuring a Recording Service" on page 90
- "Configuring a Treatment Service" on page 91

Configuring ACD Queues

To configure ACD Queues, follow the provided procedure using Configuration Manager.

Procedure: Configuring an ACD Queue

Start of procedure

1. Under a configured `Switch` object, select the `DNs` folder. From the `File` menu, select `New` > `DN` to create a new `DN` object.
2. In the `New DN Properties` dialog box, on the `General` tab, specify the following properties:
   a. `Number`: Enter the number of the configured DN. This value must be a dialable number on the switch. You must not use the @ symbol or a computer name when configuring this property.
   b. `Type`: Select `ACD Queue` from the drop-down box.
3. When you are finished, click `Apply`.

End of procedure

Configuring MCUs

To configure a Multipoint Conference Unit (MCU), follow the provided procedure using Configuration Manager.

Procedure: Configuring an MCU

Start of procedure

1. Under a configured `Switch` object, select the `DNs` folder. From the `File` menu, select `New` > `DN` to create a new `DN` object.

2. In the New DN Properties dialog box, on the General tab, specify the following properties:
   a. Number: Enter the MCU name. This name is used during SIP registration only if the MCU registers with the SIP registrar. If the MCU does not register with the registrar, enter a short description of MCU for this property.
   b. Type: Select Voice over IP Service from the drop-down box.
3. Click the Annex tab.
4. Create a section named TServer. In the TServer section, create options as specified in Table 2.

Table 2: Configuring an MCU

Option Name	Option Value	Description
contact	SIP URI	Specifies the contact URI (Uniform Resource Indicator) that SIP Server uses for communication with the MCU. See the URI format and option description on page 230.
oos-check	0–300	(Optional) Specifies how often (in seconds) SIP Server checks a device for out-of-service status. See the option description on page 236.
oos-force	0–30	(Optional) Specifies the time interval (in seconds) that SIP Server waits before placing a device that does not respond in out-of-service state when the oos-check option is enabled. See the option description on page 236.
prefix	A string	(Optional) Specifies the starting digits of the number that are used when sending calls to MCU. See the option description on page 238.
recovery-timeout	0–86400 seconds	(Optional) Specifies whether an MCU is taken out of service when an error is encountered, and for how long it is out of service. See the option description on page 239.
service-type	mcu	Set this option to mcu.

5. When you are finished, click Apply.

End of procedure

You can configure multiple MCUs. In this case, SIP Server distributes the load for all MCUs in a round-robin fashion.

Configuring Endpoints

To configure SIP endpoints, follow the provided procedure using Configuration Manager.

Note: In order to update the DN object, SIP Server must have Full Control permission for it. By default, it does not have this permission. You must grant the System account Full Control permission by changing the Permissions on the DNs folder object in Configuration Manager.

Procedure: Configuring endpoints

Start of procedure

1. Under a configured Switch object, select the DNs folder. From the File menu, select New > DN to create a new DN object.
2. In the New DN Properties dialog box, on the General tab, specify the following properties:
   a. Number: Enter the username part of the endpoint's Address of Record (AOR) as an alphanumeric string. You must not use the @ symbol or a computer name when configuring this property.
   b. Type: Select Extension (or ACD Position) from the drop-down box.
3. Click the Annex tab.
4. Create a section named TServer. In the TServer section, create options as specified in Table 3.

Table 3: Configuring Endpoints

Option Name	Option Values	Description
authenticate-requests	register, invite	Specifies whether incoming SIP requests are treated with an authentication procedure under the following conditions: 1. The name of the incoming SIP message exists in the list of the authenticate-requests parameter. 2. The option password is configured on the same DN object. See the option description on page 228.
contact	SIP URI	(Optional, depends on the phone registration) Specifies the contact URI that SIP Server uses for communication with the endpoint. See the URI format and option description on page 230.
dual-dialog-enabled	true, false	Set the value to false for endpoints that accept only one active SIP dialog, or cannot provide remote CTI control by the NOTIFY message to answer, hold, or retrieve call operations. Set the value to false for Siemens optiPoint phones that are used in re-INVITE mode for third-party call control (3pcc) operations. See the option description on page 232.
make-call-rfc3725-flow	1, 2	Specifies which SIP call flow will be used when a call is initiated by the TMakeCall request. Only flow 1 and flow 2 from RFC 3725 are currently supported. See the option description on page 233.
password	A string	Specifies the password for SIP endpoint registration with the local registrar. If it is present, registration attempts are challenged, and the password is verified. If it is not present, the registration is not challenged. The realm for password authentication is configured globally; there is one realm per SIP Server. See the option description on page 238.

Table 3: Configuring Endpoints (Continued)

Option Name	Option Values	Description
refer-enabled	true, false	Specifies whether the REFER method is sent to an endpoint. The recommended setting is true. See the option description on page 240.
reinvite-requires-hold	true, false	(Optional, for Genesys SIP Endpoints only) Specifies whether the endpoint is placed on hold by re-inviting it with the hold SDP. See the option description on page 240.
request-uri	SIP URI	Specifies the value of the Request-URI address to be used in the INVITE message, if that address is different from the address where the message will be sent. See the option description on page 241.
sip-cti-control	talk, hold	Specifies the behavior of the DN representing the SIP endpoint that supports the BroadSoft SIP Extension Event Package. See the option description on page 244.

5. When you are finished, click Apply.

End of procedure

Configuring Gateways

To configure gateways, follow the provided procedure using Configuration Manager.

Procedure: Configuring a gateway

Start of procedure

1. Under a configured Switch object, select the DNs folder. From the File menu, select New > DN to create a new DN object.
2. In the New DN Properties dialog box, on the General tab, specify the following properties:
   a. Number: Enter the gateway name. This name is used only during SIP registration when the gateway registers with the SIP registrar. If the gateway does not register with the registrar, enter a short description of the gateway for this property.
   b. Type: Select Trunk from the drop-down box.
3. Click the Annex tab.
4. Create a section named TServer. In the TServer section, create options as specified in Table 4.

Table 4: Configuring a Gateway

Option Name	Option Value	Description
contact	SIP URI	Specifies the contact URI that SIP Server uses for communication with the gateway. See the URI format and option description on page 230.
oos-check	0–300	(Optional) Specifies how often (in seconds) SIP Server checks a device for out-of-service status. See the option description on page 236.
oos-force	0–30	(Optional) Specifies the time interval (in seconds) that SIP Server waits before placing a device that does not respond in out-of-service state when the oos-check option is enabled. See the option description on page 236.
password	A string	(Optional) Specifies the password for gateway registration with the local registrar. This is used for incoming REGISTER requests, not for outgoing INVITE requests.

Table 4: Configuring a Gateway (Continued)

Option Name	Option Value	Description
prefix	A string	(Optional) Contains the initial digits of the number that must match a particular gateway for that gateway to be selected. If multiple gateways match the prefix, the gateway with the longest prefix that matches is selected.
priority	Any non-negative integer	(Optional) Specifies a gateway priority when deciding a route—a smaller number designates higher priority. If more than one gateway with the same prefix is selected, the gateway with highest priority is normally selected. This option is used to control primary-backup gateway switchover during a failure, and to provide lowest-cost routing.
refer-enabled	true, false	Specifies whether the REFER method is sent to an endpoint. The recommended setting is true. See the option description on page 240.
recovery-timeout	0–86400 seconds	(Optional) Specifies whether a gateway is taken out of service when an error is encountered, and for how long it is out-of-service. See the option description on page 239.
replace-prefix	A digit string	(Optional) Specifies the digits that are inserted in the DN instead of the prefix for the gateway. If this annex is empty or absent, the number is not modified. See the option description on page 241.

5. When you are finished, click Apply.

End of procedure

Configuring Music Servers

To configure external music servers, follow the provided procedure using Configuration Manager.

Procedure:
Configuring a Music Server

Start of procedure

1. Under a configured Switch object, select the DNs folder. From the File menu, select New > DN to create a new DN object.
2. In the New DN Properties dialog box, on the General tab, specify the following properties:
   a. Number: Enter the Music Server name. This name is used only during SIP registration if the Music Server registers with the SIP registrar. If the music server does not register with the registrar, enter a short description of the Music Server for this property.
   b. Type: Select Voice over IP Service from the drop-down box.
3. Click the Annex tab.
4. Create a section named TServer. In the TServer section, create options as specified in Table 5.

Table 5: Configuring a Music Server

Option Name	Option Values	Description
contact	SIP URI	Specifies the contact URI that SIP Server uses for communication with the music server. See the URI format and option description on page 230.
oos-check	0–300	(Optional) Specifies how often (in seconds) SIP Server checks a device for out-of-service status. See the option description on page 236.
oos-force	0–30	(Optional) Specifies the time interval (in seconds) that SIP Server waits before placing a device that does not respond in out-of-service state when the oos-check option is enabled. See the option description on page 236.
recovery-timeout	0–86400 seconds	(Optional) Specifies whether a music server is taken out of service when an error is encountered, and for how long it is out of service. See the option description on page 239.

Table 5: Configuring a Music Server (Continued)

Option Name	Option Values	Description
request-uri	SIP URI	Specifies the value of the Request-URI address to be used in the INVITE message, if that address is different from the address where the message will be sent. See the option description on page 241.
service-type	music or moh	Set this option to music or moh.

5. When you are finished, click Apply.

End of procedure

Configuring Routing Points

To configure routing points, follow the provided procedure using Configuration Manager.

Procedure: Configuring Routing Points

Start of procedure

1. Under a configured Switch object, select the DNs folder. From the File menu, select New > DN to create a new DN object.
2. In the New DN Properties dialog box, on the General tab, specify the following properties:
   a. Number: Enter the numeric-only DN number that is easily dialed directly from a phone keypad. You must not use the @ symbol or a computer name when configuring this property.
   b. Type: Select either Routing Point or Routing Queue from the drop-down box.
3. When you are finished, click Apply.

End of procedure

Configuring Softswitches

If you deploy proxies or softswitches between SIP Server and any internal DNs or agent endpoints, configure the proxies or softswitches using the provided procedure in Configuration Manager.

Procedure: Configuring softswitches

Start of procedure

1. Under a configured `Switch` object, select the `DNs` folder. From the `File` menu, select `New` > `DN` to create a new `DN` object.
2. In the `New DN Properties` dialog box, on the `General` tab, specify the following properties:
   a. `Number`: Enter the softswitch server name. This name is currently not used for any messaging, but it must still be unique. Enter a short description for this property.
   b. `Type`: Select `Voice over IP Service` from the drop-down box.
3. Click the `Annex` tab.
4. Create a section named `TServer`. In the `TServer` section, create options as specified in Table 6.

Table 6: Configuring Softswitches

Option Name	Option Value	Description
`contact`	SIP URI	Specifies the contact URI that SIP Server uses for communication with the softswitch. On some softswitches this is the same as the public IP address used by endpoints to contact the softswitch. However, other softswitches require a separate port. See the URI format and option description on page 230.

Table 6: Configuring Softswitches (Continued)

Option Name	Option Value	Description
public-contact	An IP address	Contains the public host:port pair for a softswitch. This is the public IP address of the softswitch. SIP Server uses this address to fill the destination (Refer-To) address in REFER requests. On some switches, this is the same as the contact address; if this is the case, you do not need to specify this parameter.
service-type	softswitch	Set this option to softswitch.

5. When you are finished, click Apply.

End of procedure

Additional Information

You can configure multiple softswitches in either an active load-balancing configuration or in a primary-standby configuration. For load-balancing, define services with the same priority to each service. For the primary-standby configuration, give higher priority to the primary service entry.

Configuring an Application Service

To configure an application service, follow the provided procedure using Configuration Manager.

Procedure: Configuring an application service

Start of procedure

1. Under a configured Switch object, select the DNs folder. From the File menu, select New > DN to create a new DN object.
2. In the New DN Properties dialog box, on the General tab, specify the following properties:
   a. Number: Enter the treatment server name.
   b. Type: Select Voice over IP Service from the drop-down box.
3. Click the Annex tab.
4. Create a section named TServer. In the TServer section, create options as specified in Table 7.

Table 7: Configuring an Application Service

Option Name	Option Values	Description
`contact`	SIP URI	Specifies the contact URI that SIP Server uses for communication with the treatment server. See the URI format and option description on page 230.
`service-type`	`application`	Set this option to `application`.

5. When you are finished, click `Apply`.

End of procedure

Configuring a Recording Service

To configure a recording service, follow the provided procedure using Configuration Manager.

Procedure: Configuring a recording service

Start of procedure

1. Under a configured `Switch` object, select the `DNs` folder. From the `File` menu, select `New` > `DN` to create a new `DN` object.
2. In the `New DN Properties` dialog box, on the `General` tab, specify the following properties:
   a. `Number:` Enter the recorder server name.
   b. `Type:` Select `Voice over IP Service` from the drop-down box.
3. Click the `Annex` tab.
4. Create a section named `TServer`. In the `TServer` section, create options as specified in Table 8.

Table 8: Configuring a Recording Service

Option Name	Option Value	Description
contact	SIP URI	Specifies the contact URI that SIP Server uses for communication with the recorder server. See the URI format and option description on page 230.
request-uri	SIP URI	Specifies the value of the Request-URI address to be used in the INVITE message, if that address is different from the address where the message will be sent. See the option description on page 241.
service-type	recorder	Set this option to recorder.

5. When you are finished, click Apply.

End of procedure

Additional Information

SIP Server can also record a file name when emergency recording is initiated by an agent. See the emergency-recording-filename configuration option (page 206) for more information.

Configuring a Treatment Service

To configure a treatment service, follow the provided procedure using Configuration Manager.

Procedure:
Configuring a treatment service

Start of procedure

1. Under a configured Switch object, select the DNs folder. From the File menu, select New > DN to create a new DN object.
2. In the New DN Properties dialog box, on the General tab, specify the following properties:
   a. Number: Enter the treatment server name.
   b. Type: Select Voice over IP Service from the drop-down box.
3. Click the Annex tab.

4. Create a section named TServer. In the TServer section, create options as specified in Table 9.

Table 9: Configuring a Treatment Service

Option Name	Option Values	Description
contact	SIP URI	Specifies the contact URI that SIP Server uses for communication with the treatment server. See the URI format and option description on page 230.
service-type	treatment	Set this option to treatment.

5. When you are finished, click Apply.

End of procedure

Configuring Out-of-Service Detection

The VOIP devices can be deployed in either of these modes:

- Load-balancing mode (for the MCU, music server, treatment service, recording service, and application service)
- Primary-standby mode

Load-Balancing Mode

When multiple devices are configured in this mode, SIP Server chooses a device in a round-robin fashion. When it encounters a failure of a device that has a non-zero recovery timeout, SIP Server will disable the device and only use the remaining ones.

SIP Server will re-enable a device when the following occurs:

- A timeout configured by the recovery-timeout option expires.
- The device entry in Configuration Manager is changed.

The recovery-timeout configuration option enables an automatic periodic switchback attempt by SIP Server.

Primary-Standby Mode

When two or more devices of the same type are configured in this mode, SIP Server chooses a device with a higher priority as configured by the priority option to direct all traffic to it during normal operation. A failure of the primary device will move it to out-of-service status. The secondary device will then be selected on future requests.

If active out-of-service detection is not configured, switchback to the primary device will occur when the `recovery-timeout` option value has expired, despite the in-service or out-of-service status of the device. Therefore, the `recovery-timeout` option can be enabled for an automatic periodic switchback attempt.

If active out-of-service detection is configured, this mechanism checks the device status after the `recovery-timeout` option has expired, and will force the device back into service only when its recovery has been detected.

Active Out-of-Service Detection

SIP Server supports active out-of-service detection that can be enabled for the following types of DNs:

- `Voice over IP Service` (MCU, treatment, softswitches, and so on)
- `Trunk`

To enable out-of-service detection, configure the following options in the `TServer` section of the `Annex` tab of the corresponding DN:

- `oos-check` (see page 236)
- `oos-force` (see page 236)
- `recovery-timeout` (see page 239)

The `oos-check` option specifies how often (in seconds) SIP Server checks a device for out-of-service status. When no response is received, and the `oos-force` option is configured, SIP Server waits the specified `oos-force` timeout expires before placing a device that does not respond in out-of-service state. For `Voice over IP Service` DNs, SIP Server generates the following log message stating that the specified device is out of service based on active out-of-service detection:

```
52000|STANDARD|GCTI_DEVICE_OUT_OF_SERVICE|Device [the name of the
device] is out of service
```

The DN will be set back in service automatically when the `recovery-timeout` timer expires. For `Voice over IP Service` DNs, SIP Server generates the following log message stating that the specified device is back in service based on active out-of-service detection:

```
52001|STANDARD|GCTI_DEVICE_BACK_IN_SERVICE|Device [the name of the
device] is back in service
```

Notes: Active out-of-service detection on an MCU does not lead to switching to another MCU when an out-of-service device is detected. When an out-of-service MCU is detected, it is marked as unavailable and will not be used for further conferencing until it is back in service.

All treatments are restarted on another treatment DN when SIP Server detects that the DN of type `Voice over IP Service` with the `service-type` option set to `treatment` is out of service.

In a High-Availability Environment

When operating in a high-availability environment, the states of devices are synchronized to the backup SIP Server. As a result, in-service or out-of-service states are preserved during a switchover. After the switchover, the new primary SIP Server will restart the `oos-check`, `oos-force`, and `recovery-timeout` timers for those devices that are out of service, and that have the corresponding options set in their configuration.

Configuring Agent Logins

SIP Server can work either with softswitches or in stand-alone mode, in which the SIP endpoint communicates directly with SIP Server. In both scenarios, you must configure the `Switch` object in the Configuration Layer. The manner in which you configure your SIP Server must reflect the properties of all the objects that your SIP Server monitors. If a client issues a `TRegisterAddress` request for a DN that is not configured in Configuration Manager, SIP Server will generate an `EventError` message.

Because only SIP Server uses agent logins, they do not need to match user information on the softswitch. SIP Server manages the status of agents who use these logins, and allows these agents to log in to the SIP addresses.

Configuring Stream Manager

Stream Manager is a Genesys client application that streams media files in order to provide announcements and music to callers queued on Routing Points and ACD queues. It can also serve as a music server or as an MCU. SIP Server does not need Stream Manager for normal call handling.

You configure Stream Manager as a `DN` object of type `Voice over IP service`, with the `service-type` option set in the `TServer` section on the `Annex` tab. The values are: `music`, `mcu`, `treatment`, `conference`, and `recorder`.

To use Stream Manager as a music server, it must be configured as described in “Configuring Music Servers” on page 85.

Stream Manager must be able to access the audio files that SIP Server requests to play. These files are located in subdirectories of the installed Stream Manager root directory. The files must be in the appropriate codec format, with the filename suffix corresponding to the codec type. All Stream Managers integrated with SIP Server must contain the same announcement/music files in the same directory structure. The treatment will fail if the file does not exist in the specified directory.

If you are using a gateway, you must determine the codec it uses before you determine which codec is specified in Genesys.

See the *Framework 7.6 Stream Manager Deployment Guide* for more information about Stream Manager.

Chapter

6 SIP Server Feature Support

This chapter describes the advanced functionality that SIP Server supports. It contains the following sections:

Associating an ACD Queue with a Routing Point

SIP Server is able to associate an ACD Queue with a Routing Point by specifying the `Routing Point` DN in the `Association` field in the `Properties` dialog box of the `ACD Queue` DN object in Configuration Manager.

The call flow for this functionality is as follows:

- Agents log into the ACD Queue.
- An inbound call arrives at the ACD Queue and at the associated Routing Point. The call is not auto-distributed to an agent in that ACD Queue.
- A Universal Routing Server (URS) strategy on the Routing Point selects an available agent in the ACD Queue.
- The call is routed to an agent's DN, which responds with a SIP Ringing message. As a result, an `EventDiverted` message is distributed against the ACD Queue and `EventRouteUsed` and `EventDiverted` messages are distributed against the Routing Point.
- The agent answers the call.

Notes: The inbound call will be treated as a regular call to the ACD Queue if no URS application has registered for the Routing Point associated with the ACD Queue.

The inbound call will be treated as a regular call to the ACD Queue if a URS application has registered for the Routing Point associated with the ACD Queue, but the routing timeout expires.

Asterisk Voice Mail Integration

SIP Server exchanges SIP messaging with an Asterisk-based Voice Mail system using a SIP trunk. voice mail boxes with Asterisk have the same names as their corresponding DNs, but you do not need to configure voice mail boxes as DNs in Configuration Manager. You can access these voice mail boxes by dialing a user-defined prefix and a DN number. It is also possible to dial a prefix and a special extension so that the call will be delivered to the Voice Mail Main Menu.

SIP Server configuration options are used to organize the delivery of calls to the Voice Mail system when certain conditions are met. For example, when an incoming call is not answered within a specified timeout. Universal Routing Server strategies are used to organize more flexible control over delivery of the call to a voice mail box.

Note: Voice mail for DNs is handled by an Asterisk softswitch when using Asterisk as a softswitch with SIP Server. You do not need to configure voice mail and Message Waiting Indicator (MWI) in SIP Server for any DNs. However, voice mail is configured for agents and Agent Groups in this scenario.

Genesys Voice Mail Configuration Adapter for Asterisk

The Genesys Voice Mail Configuration Adapter for Asterisk (GVMA) is a utility that allows you to synchronize the Asterisk Voice Mail configuration file (`voicemail.conf`) with data from the Configuration Server. By default, GVMA creates a voice mail box in Asterisk for each extension DN configured in Configuration Manager. GVMA can also be configured to use DNs of other types for the same purpose.

GVMA obtains the necessary Asterisk Voice Mail configuration properties from the following objects in Configuration Manager:

- DNs are used to configure voice mail boxes for extensions.
- Agent Logins are used to configure voice mail boxes for agents.
- Agent Groups are used to configure of voice mail boxes for agent groups.

When it has retrieved this information, GVMA performs the following tasks:

1. Connects to Configuration Server.
2. Makes a backup copy of the current Asterisk configuration.
3. Loads DNs from Configuration Server.
4. Updates the Asterisk Voice Mail configuration file based on the retrieved data.
5. Instructs Asterisk to reload configuration files.

GVMA is run manually or scheduled to run periodically using your operating system’s scheduling tools.

Message Waiting Indicator Functionality

SIP Server supports Message Waiting Indicator (MWI) notifications by registering with Asterisk and processing MWI notifications sent by Asterisk within `NOTIFY` messages. SIP Server sends to a DN an `EventUserEvent` message with detailed information about the message waiting status attached in the `UserData` fields.

MWI Processing for Extension Voice Mail Boxes

SIP Server registers (using the `REGISTER` request) all extensions as voice mail boxes with Asterisk when the option `mwi-extension-enable` (see page 214) has a value of true. Asterisk sends NOTIFY SIP messages to SIP Server that contain MWI information. SIP Server stores MWI information received from Asterisk, and distributes an EventUserEvent message with the latest MWI information to each registered extension voice mail box.

MWI Processing for Agent Voice Mail Boxes

SIP Server registers each agent voice mail box with Asterisk immediately after the EventAgentLogin message is received and when the option mwi-agent-enable (see page 214) has a value of true. SIP Server stores MWI information received from Asterisk for each agent voice mail box and distributes an EventUserEvent message with the latest MWI information to the DN that the agent has logged into. SIP Server will terminate this registration immediately after an EventAgentLogout message is received.

MWI Processing for Agent Group Voice Mail Boxes

SIP Server registers the agent group voice mail box with Asterisk immediately after the first member's EventAgentLogin message is received and when the option mwi-group-enable (see page 215) has a value of true. SIP Server stores MWI information received from Asterisk for each agent group voice mail box and distributes an EventUserEvent message with the latest MWI information to all DNs associated with agents that are logged in and configured to use this voice mail box. SIP Server will terminate this registration immediately after an EventAgentLogout message is received from the last agent associated with this agent group's voice mail box.

Note: SIP Server associates an agent with an agent group voice mail box by using the option gvm_group_mailbox located in the TServer section on the Annex tab of the Agent Login configuration object. Any agent configuration object that is configured with the same voice mail box specified in this option is associated with this agent group's voice mail box.

Feature Configuration

MWI functionality is configured on the Asterisk side and on the SIP Server side. The following SIP Server configuration options support MWI functionality:

- mwi-host (page 215)
- mwi-port (page 215)
- mwi-domain (page 214)
- mwi-extension-enable (page 214)
- mwi-agent-enable (page 214)

- mwi-group-enable (page 215)
- mwi-mode (page 215) (for backward compatibility)

For Asterisk configuration and integration with SIP Server, see the *Framework 7.6 SIP Server Integration Reference Manual.*

Busy Lamp Field Feature

This feature enables SIP Server to interoperate with the BroadWorks softswitch and to receive notifications about BroadWorks line usage.

See the *Framework 7.6 SIP Server Integration Reference Manual* for more information about Busy Lamp Feature (BLF) support in SIP Server.

Calls Outside the Premise

Participants within VoIP conversations are divided into two groups:

- Internal parties that represent agents or supervisor SIP endpoints. They communicate directly with each other by signaling directly with SIP Server and by RTP streams. In Configuration Manager, they are configured as DNs of type Extension.
- External parties that represent customers or agents at remote sites. They communicate with call center devices using a media gateway or other proxy services. External parties do not have a direct representation at the Configuration Layer, but must be represented as a Trunk DN or a “media gateway” that is associated with a SIP Server Switch object.

You need to choose a gateway at the same premise where the agent SIP endpoint is located to minimize network load for RTP traffic and for VoIP media services, such as music on hold, central mixing conferencing, or voice recording. Devices in the same premise must be configured with the same value of the geo-location option.

- An internal party has the same option value as the corresponding Extension DN object when a call is established.
- An external party has the same option value as the corresponding Trunk DN object when a call is established.
- A media server has the same option value as the corresponding Voice over IP Service DN object when a call is established.

SIP Server determines which gateway or trunk to choose for the outbound call, based on the settings of the find-trunk-by-location option.

To determine the gateway for an external party of an inbound call, the IP address from the Via header of the incoming INVITE message must match with the host address of the contact option of the Trunk DN. If a match is successful, the geo-location label for the matched trunk will be used as the geo-location label for the external party. In order to make this match work, the contact of the corresponding trunk must be the same because it is expected to be inside the Via header of the incoming INVITE message (most likely a decimal IP address).

For other services, such as music, treatment, recorder, or mcu, SIP Server searches for the same geo-location label as the party requesting such service.

Call Recording

SIP Server supports both regular call recording and emergency call recording.

Regular Call Recording

Call recording is performed by passing an RTP stream through Stream Manager. Stream Manager acts as a media stream proxy, recording all media packets into a file. Depending on the configuration, Stream Manager may perform media mixing, or it may save the RTP packets as is, thus improving call recording performance. (See the *Framework 7.6 Stream Manager Deployment Guide* for details.) Call recording is always enabled on a single call leg, such as a leg with a gateway or a leg with a SIP phone.

Call recording starts after a call becomes established. It does not result in any changes to the call itself, to event processing, or to any other generated TEvents.

When call recording starts, SIP Server creates two new SIP dialogs with Stream Manager. SIP Server sends re-INVITE requests to all call participants with the SDP from Stream Manager. As a result, the RTP stream between call participants is passed via Stream Manager.

Call recording has the highest priority compared to other operations that can be performed on the call when it is established. That is, when the EventEstablished message is generated on the destination DN, the operations on the call are performed in the following order:

1. Call recording, if enabled
2. Personal greeting, if enabled
3. Supervisor monitoring, if enabled

Recording is not available for consultation calls. If configuration enables recording on a DN, and a consultation call is made to this DN, recording will not start.

Recording is only available for calls with audio media. If a call contains video or IM media, recording will not start.

Only one recording is allowed on a call. If configuration enables recording on both devices in the call, recording will only start on the device that first signals EventEstablished.

Feature Configuration

Table 10 provides an overview of the main steps required to configure the call recording functionality.

Table 10: Task Flow—Configuring Call Recording

Objective	Related Procedures and Actions
1. Configure a DN.	To enable call recording on a particular DN, in the `TServer` section on the `Annex` tab of the `DN` object in Configuration Manager, set the configuration option `record` to `true`. To record all inbound calls coming for a particular media gateway, set the `record` option to `true` on the `Trunk` DN that represents this gateway.
2. Configure a SIP Server `Application` object.	In the `TServer` section on the `Options` tab of the SIP Server `Application` object in Configuration Manager, set the configuration option `recording-filename` to the name of the recorded file—for example: `call-$ANI$-$DNIS$-$DATE$-$TIME$-$CONNID$-$UUID$-$AGENTDN$-$AGENTID$`
3. Configure an `Extension` attribute.	Specify an `Extension` attribute with key `record` in the `TRouteCall` request. See the key values in Table 24, “Use of the Extensions Attribute,” on page 186. The routing strategy will determine whether call recording is needed.
4. Configure a Stream Manager `Application` object.	Set configuration options in the Stream Manager `Application` object for precise control of how recording is performed. See the *Framework 7.6 Stream Manager Deployment Guide* for more information. It is recommended to use `pcap` recording mode on Stream Manager for best performance.
5. Configure a recording service.	Configure a DN of type `Voice over IP Service` with the following configuration options: • `contact`: Set to the device’s IP address used for recording. • `request-uri`: Set to the SIP URI. • `service-type`: Set to `recorder`. See “Configuring a recording service” on page 90 for details.

Emergency (Manual) Call Recording

SIP Server performs emergency call recording when processing a single-step conference call request that specifies `AttributeOtherDN` as a `Trunk` DN specifying the `gcti::record` number. When this attribute is set, SIP Server recognizes this special request and initiates call recording as follows:

- Selects one of the available call recording units that are configured in Configuration Manager. See "Configuring a Recording Service" on page 90 for more information.
- Performs a single-step conference call and adds the selected call recording unit to the call.
- Creates the file name as configured in the `emergency-recording-filename` option that is described on page 206.

To stop emergency call recording, the agent must issue the `TDeleteFromConference` request using the `gcti::record` number.

Note: Refer to the *SIP Server 7.5.0 Call Recording White Paper* for more information about call recording. This document is available from Genesys Technical Support or Genesys Professional Services.

Feature Limitations

Emergency call recording cannot be activated on a consultation call if it has already been activated from the same DN on the primary call. Emergency call recording can only be activated on both primary and consultation calls if initiated from different DNs.

Call Supervision

Call supervision functionality is designed to enable contact center managers to monitor agents, and it also enables agents to invite their supervisors to the call when dealing with a customer.

SIP Server supports the following call supervision scenarios:

- Standard Call Supervision—Enables supervisors to monitor agents where supervisors and agents are located on the same site.
- Multi-Site Supervision—Enables supervisors at a local site, from an endpoint controlled by a local SIP Server, to monitor remote agents, whose endpoints are controlled by another SIP Server. See "Multi-Site Supervision" on page 112.
- Remote Supervision feature—Enables supervisors to monitor agents from outside the contact center—for example, from an off-premise cell phone. See "Remote Supervision" on page 115.

Overview

There are two types of call supervision that SIP Server supports:

- *Subscription* monitoring enables supervisors to subscribe and monitor one agent. If the subscription is active, SIP Server automatically invites the supervisor to all calls where the agent participates. SIP Server stops working in this mode when the subscription is cancelled.
- *Assistance* monitoring is activated by an agent by issuing an assistance request sent to the supervisor. The agent can issue this while he or she is on a call with a customer.

Supervision Modes

Call supervision is performed in three different modes:

- *Silent monitoring* hides the supervisor's presence from all call participants, including the monitored agent who is the target of supervisor's attention.
- *Whisper coaching* hides the supervisor's presence from all call participants but the monitored agent. Only the agent can hear the supervisor.
- *Open supervisor presence* invites the supervisor to the call through subscription or assistance call supervision scenarios, but all call participants are aware of the supervisor's presence and can hear him or her.

The supervisor can choose any of these three modes for the call supervision subscription, but the agent can only use the last two modes for an assistance request.

Supervision Scopes

The call supervision scope specifies the time frame when the supervisor must participate in the call. There are two scopes available in SIP Server:

- *Agent scope* allows the supervisor to monitor the agent. The supervisor joins the call when the call is established on the monitored agent's DN. The supervisor leaves the call immediately after the agent leaves the call.
- *Call scope* allows the supervisor to control the customer's experience. The supervisor joins the call when the call is established on the monitored agent's DN, or when the supervisor receives the assistance request from the agent. SIP Server keeps the supervisor as part of the call as long as either a customer or monitored agent remains in the call.

The supervisor can choose either of these scopes for the monitoring subscription.

An assistance request issued by the agent does not specify the supervision scope, so the scope always contains the `call` value. Therefore, if a supervisor is invited to a call through an assistance request, he or she will stay on the call until the call is finished.

Supervision Types

The call supervision type specifies the number of calls to be monitored—either one call or all calls.

- If *one call* is chosen for the subscription, the subscription is cancelled automatically when the supervisor finishes monitoring the first call on the monitored agent.
- If *all calls* is chosen for the subscription, the supervisor must cancel the subscription manually when he or she want to stop monitoring the agent's calls.

The call supervision type cannot be specified for an assistance request. The `one call` type is always used when call supervision is initiated through an assistance request. The type cannot be changed through the configuration settings.

Monitoring Session

A *monitoring session* is the process in which a supervisor listens to an agent-customer conversation. There are two types of monitoring sessions that are defined by the session creation scenario:

- A *subscription session* is created by SIP Server automatically when a call is delivered to an agent's DN, using the existing call supervision subscription.
- An *assistance session* is created as a result of the assistance request sent by an agent to a supervisor.

A monitoring session of any type must be initialized with the following three parameters when it is created:

- Supervision type
- Supervision mode
- Supervision scope

These parameters in the subscription session are initialized with the values of the corresponding parameters in the subscription from which this session was derived. An assistance session uses information passed in the assistance request and includes some configuration parameters for the initialization purpose. See "Feature Configuration" on page 107 for more information.

A monitoring session begins when a supervisor joins a call, and it ends when the supervisor disconnects from the call.

One call can have multiple monitoring sessions of both types, and all are active at the same time. Each monitoring session is uniquely identified by the supervisor involved. As a result, the supervisor can participate in only one monitoring session at a time, but one agent can be part of multiple monitoring sessions.

The following example demonstrates how multiple monitoring sessions are created in one call:

- Agent1 answers an incoming call, and Supervisor1 is invited to the call based on the existing subscription.
- Agent1 sends an assistance request to Supervisor2, who also joins the call.

This call has two monitoring sessions active at the same time: the first session has a subscription type, and the second session is an assistance session.

Intrusion

Intrusion occurs when a supervisor activates a new call supervision subscription to monitor an agent who is currently on a call. SIP Server creates the requested subscription and immediately invites the supervisor to join the existing call.

Feature Configuration

This section describe how to configure call supervision. It covers the following topics:

- Subscription, page 107
- Assistance Request, page 109
- Supervisor Auto-release, page 109
- Hiding Supervisor Presence, page 111
- Configuration Options, page 111

Subscription

Call supervision subscription is controlled by two T-Library requests:

- `TMonitorNextCall`
- `TCancelMonitoring`

The supervisor's desktop must be able to process these two requests in order to perform call supervision.

The first request creates a new subscription, and the second request cancels the existing subscription. These requests use `AttributeThisDN` to identify the supervisor and `AttributeOtherDN` to identify the monitored agent.

Subscription Creation

SIP Server creates a new subscription based on the `TMonitorNextCall` request from the supervisor. The request is either accepted or rejected.

SIP Server rejects the request in the following scenarios:

- The supervisor or the monitored agent already has an active subscription.

However, if the TMonitorNextCall request tries to activate a monitoring subscription that is already active (for example, the supervisor who submitted this request is already set up to monitor the agent), SIP Server responds with the standard EventMonitoringNextCall messages sent to the agent and supervisor DNs. This request is not rejected, because it does not create multiple subscriptions on one DN.

- The supervisor or the agent DN is not configured in Configuration Manager.

If the request is accepted, SIP Server creates a new subscription and initializes it with the type, mode, and scope information that was defined in the request.

This information is part of the request as the following attributes:

- AttributeMonitorNextCallType, which defines the type of call supervision. Its possible values are MonitorOneCall and MonitorAllCalls.
- AttributeExtensions/MonitorMode, which defines the mode of call supervision. Its possible values are normal, mute, coach, and connect.
- AttributeExtensions/MonitorScope, which defines the scope of call supervision. Its possible values are call and agent.

If one or both of the monitoring extensions are missing or incorrect, the following values are used:

- default-monitor-scope for MonitorScope
- default-monitor-mode for MonitorMode

SIP Server confirms the new subscription for both the supervisor and the agent by sending an EventMonitoringNextCall message to both destinations. This event always contains AttributeExtensions that include both monitoring extensions. These extensions represent the monitoring configuration for a new subscription.

See “Using the Extensions Attribute” on page 186 for more information.

Note: SIP Server identifies the agent for the call supervision by the agent DN specified in the OtherDN attribute of the TMonitorNextCall request. The agent’s login ID is not used for this purpose. In particular, this means that SIP Server does not try to identify the agent who is logged in on the monitored DN, or to analyze the agent’s state to decide if supervision should be activated for a call. SIP Server monitors calls made to or from the specified DN, regardless of the person using this DN, until supervision scope expires (see “Supervision Scopes” on page 105).

Subscription Cancellation

SIP Server can cancel active subscriptions using the following methods:

- Manual, where a supervisor submits a TCancelMonitoring request.

- Automatic, where SIP Server cancels the subscription when a `MonitorOneCall`-type monitoring session is terminated.

A supervisor can submit a `TCancelMonitoring` request at any time. SIP Server identifies a subscription by the pair of supervisor and agent DNs. If this subscription exists, then it will be cancelled. Otherwise, SIP Server returns an `EventError` message.

SIP Server generates `EventMonitoringCancelled` events for both the supervisor and the agent, to inform them that the subscription was cancelled.

Assistance Request

An assistance request is a `TSingleStepConference` request containing the `AssistMode` parameter in the extensions. SIP Server creates a new monitoring session based on the assistance request, but a monitoring subscription is not created.

The `AssistMode` extension is identical to the `MonitorMode` extension used in the `TMonitorNextCall` request. The difference is that `AssistMode` can contain only the `connect` and `coach` values.

There are no parameters to define the scope and type of the monitoring in an assistance request, so the following monitoring parameters are used:

- `MonitorScope` set to `call`
- `MonitorType` set to `MonitorOneCall`

These two settings are hard-coded and cannot be changed.

Supervisor Auto-release

Depending on the type of monitoring scope and mode, SIP Server determines whether to release a supervisor from the call. If the monitoring scope is `agent`, SIP Server releases the supervisor from the call at the same time that the monitored agent leaves the call. If the monitoring scope is `call` and the other party of the call is aware of the supervisor's presence on the call and can hear this supervisor, SIP Server does not release the supervisor from the call.

Call Scenarios

This section presents two-party and three-party call scenarios to demonstrate how auto-release rules work.

Example 1 Three-party call, `MonitorScope=call`:

1. A call is established with three parties: a caller, a supervisor, and Agent 1 (a monitored target of the supervisor).
2. Agent 1 transfers the call to Agent 2 (who is not monitored by the supervisor).

3. The call now has the following parties: the caller, the supervisor, and Agent 2.

   The supervisor is not released in this step, because `MonitorScope` is set to `call`, and the call is not finished yet (the monitor scope has not expired).

4. The caller hangs up. Now this call contains only two parties.

5. One of the following happens:
   - If `MonitorMode` is set to `mute` or `coach`, SIP Server will release the supervisor and the call, because the supervisor is on the call with the agent (Agent 2) who is not the monitoring target of this supervisor, and the agent is not aware of the supervisor's presence.
   - If `MonitorMode` is set to `connect`, SIP Server will not release the supervisor, so Agent 2 can hear the supervisor.

Example 2 Three-party call, `MonitorScope=agent`:

1. A call is established with three parties: a caller, a supervisor, Agent 1 (a monitored target of the supervisor).
2. Agent 1 transfers a call to Agent 2 (who is not monitored by a supervisor).
3. SIP Server releases the supervisor from the call. The caller and Agent 2 remain on the call.

Example 3 Three-party call with recording, `MonitorScope=call`, `MonitorMode=mute`:

1. A call is established with three parties and a recorder: a caller, a supervisor, Agent 1 (a monitored target of the supervisor), and the recorder.
2. The caller hangs up. Now this call contains three parties: Agent 1, the supervisor, and the recorder.
3. SIP Server releases the supervisor and the call, because `MonitorMode` is set to `mute` and the agent cannot talk to the supervisor.

Example 4 Three-party call with recording, `MonitorScope=call`, `MonitorMode=connect`:

1. A call is established with three parties and a recorder: a caller, a supervisor, Agent 1 (a monitored target of the supervisor), and the recorder.
2. Agent 1 transfers the call to Agent 2 (who is not monitored by the supervisor).

   Now the call has the following parties: the caller, the supervisor, Agent 2, and the recorder. The supervisor is not released in this scenario, because `MonitorScope` is set to `call`, and the call is not finished yet (the monitor scope is not expired).

3. The caller hangs up.
4. SIP Server does not auto-release the call, but will enable the supervisor to continue talking to Agent 2.

Hiding Supervisor Presence

A supervisor who is performing silent monitoring or whisper coaching must be hidden from other call participants. If the scenario involves whisper coaching, only the monitored agent (who can hear the supervisor) must be aware of his or her presence on the call.

Call participants receive information about other participants joining or leaving the call from the corresponding T-Library events distributed by SIP Server. The T-Library desktop applications used by call center employees must be able to process the T-Library events and indicate the recent changes in a call status. For example, they can show that new participant has just joined or left the call.

Hiding a supervisor's presence means filtering out any events that inform other participants about the supervisor's activity. SIP Server inserts specific information into the T-Library events that allow T-Library clients to decide if a particular event must be shown to the customer or it must be suppressed. SIP Server make modifications to the events if at least one monitoring session is active on a call. The following attributes support this functionality:

- `AttributeCallState`
- `AttributeOtherDNRole`
- `AttributeThirdPartyDN`
- `AttributeThirdPartyDNRole`

The details on how those attributes are modified are found in the *Genesys 7 Events and Models Reference Manual*.

Configuration Options

The following SIP Server Application-level options support call supervision functionality:

- `cancel-monitor-on-disconnect` (page 203)
- `default-monitor-mode` (page 203)
- `default-monitor-scope` (page 204)
- `intrusion-enabled` (page 212)
- `monitor-internal-calls` (page 213)

Feature Limitations

The following known limitations currently apply to call supervision:

- SIP Server does not monitor consultation calls that are made either from or to a DN that is under call supervision. No monitoring is activated in this scenario, and the supervisor will not be invited to monitor the agent.
- Call supervision functionality is disabled for video calls.

- A supervisor participating in a monitoring session cannot initiate a 1pcc or 3pcc call transfers or conference calls because it can change the supervisor's status in the conference call.
- If a supervisor is already engaged in a call when an agent DN that it is targeting joins a new call (which requires monitoring), SIP Server does not invite the supervisor to monitor the new agent conversation. Even if the supervisor disconnects from its current call, the monitoring session for the new agent conversation will not start. SIP Server will activate monitoring for the next call on the targeted DN.
- Call supervision functionality is supported only when Stream Manager is used as an MCU. Stream Manager is required because SIP Server sends proprietary information in the SIP messages to set up a specific conference mode on the Stream Manager.
- When two agents are monitored by two different supervisors, and one agent calls the other agent, SIP Server invites only one supervisor to the call.

Multi-Site Supervision

When SIP Servers operate in a multi-site environment, a supervisor at a local site, from an endpoint controlled by a local SIP Server, can monitor remote agents, whose endpoints are controlled by another SIP Server.

Feature Configuration

To enable this feature, both the supervisor's SIP Server and the agent's SIP Server must be configured for mutual multi-site access with the ISCC transaction type `route` or `direct-uui` (see Chapter 10, "Multi-Site Support," on page 267). Additionally, a special `Routing Point` DN, dedicated for multi-site supervision, must be configured under the agent's `Switch` object. The `Routing Point` number must be specified in the `observing-routing-point` option of the agent's SIP Server `Application` object. A special routing strategy must be loaded on the observing Routing Point to route the observing call leg to the supervisor. (See "Routing Strategy Design Sample" on page 113.)

A multi-site monitoring session can be initiated by a T-Library client, connected to the supervisor's SIP Server, by issuing a `TMonitorNextCall` request. The request must contain:

- The `Location` parameter with the `remote` value
- (Optional) The `MonitorMode` parameter
- (Optional) The `MonitorScope` parameter

If optional parameters are not specified in the `TMonitorNextCall` request, the values will be taken from the `default-monitor-mode` and `default-monitor-scope` configuration options of the agent's SIP Server `Application` object.

The `TMonitorNextCall` request, issued by a T-Library client to the supervisor's SIP Server, is transmitted through the ISCC connection to the agent's SIP Server and registered on both servers.

After a call has been answered by an agent, the agent's SIP Server initiates the observing service by creating a call leg to the Routing Point, specified by the `observing-routing-point` option.

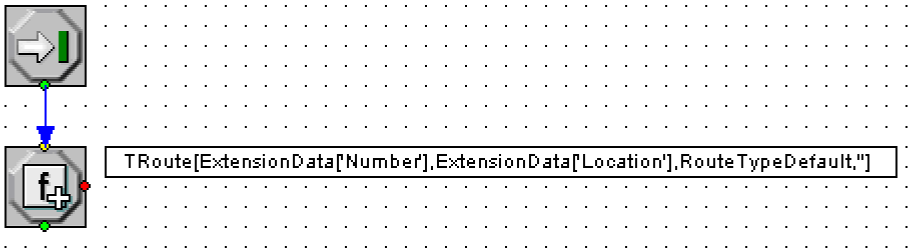
The `EventRouteRequest` message generated by the agent's SIP Server reports the supervisor's switch name and the number in the `Location` and `Number` extensions respectively. The routing strategy, loaded on the observing Routing Point, must use this information to route the observing leg of the call to the supervisor's endpoint.

When the supervisor answers, he or she will be connected to the call in the mode defined by the `MonitorMode` parameter of the `TMonitorNextCall` request.

During a multi-site supervision session, the supervisor's connection to the monitored call can be changed between the initial `MonitorMode` and an open supervisor presence, with the `TSetMuteOff` and `TSetMuteOn` requests containing the supervisor's DN in the `dn` parameter. A supervision session can be canceled with the `TCancelMonitoring` request.

Routing Strategy Design Sample

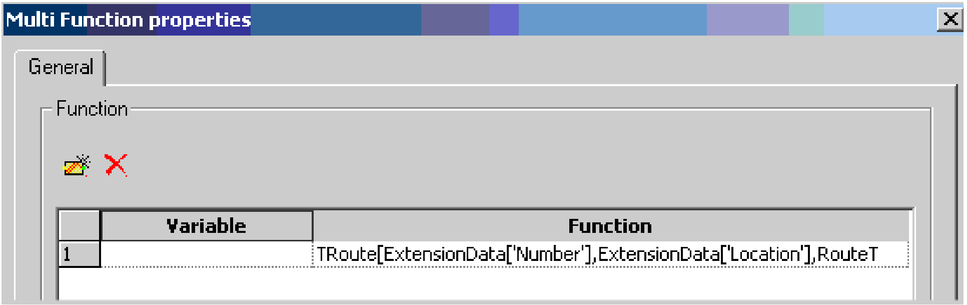
This section provides a routing strategy design sample (see Figure 8), which should be loaded on the observing Routing Point at the agent's SIP Server to support multi-site supervision.



```
TRoute[ExtensionData['Number'],ExtensionData['Location'],RouteTypeDefault,'']
```

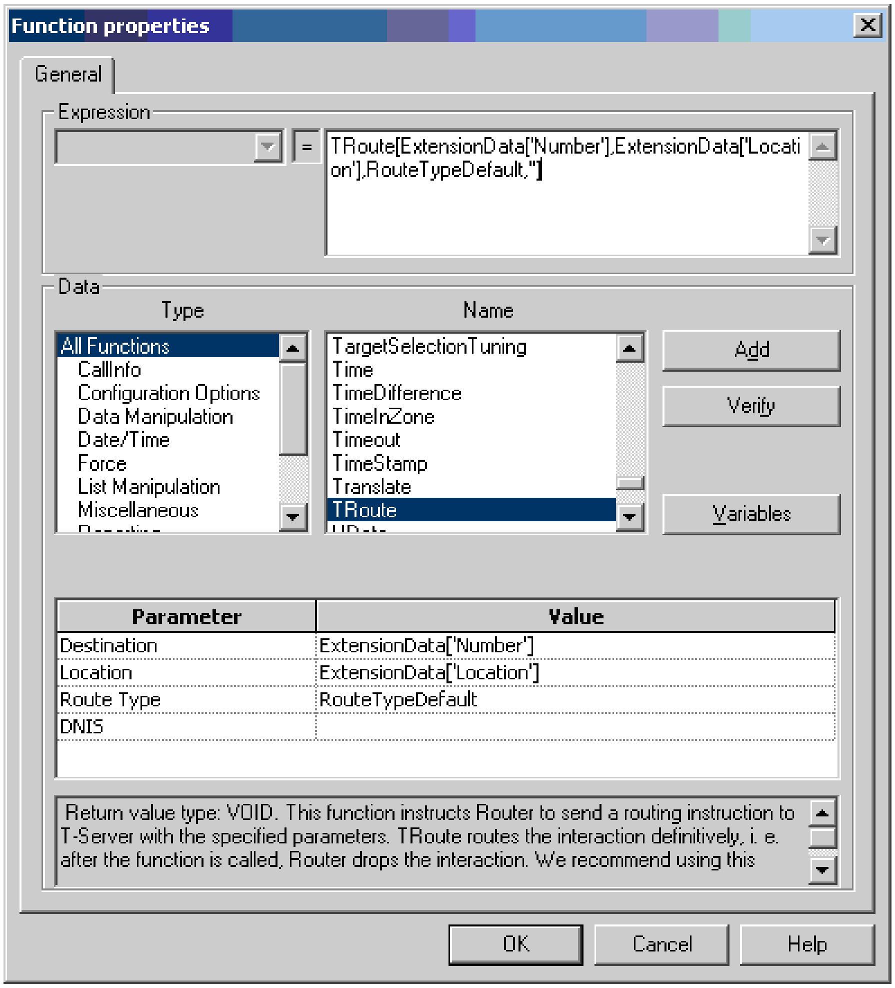
Figure 8: A Routing Strategy Design Sample

The sample strategy uses a single the `Multi Function` routing object (see Figure 9). The supervisor's number and switch name are retrieved from the `EventRouteRequest` extensions by the `ExtensionData` function. These values are passed to the `TRoute` function in the `Destination` and `Location` parameters.



	Variable	Function
1		TRoute[ExtensionData['Number'],ExtensionData['Location'],RouteT

Figure 9: The Multi Function Object



Expression: TRoute[ExtensionData['Number'],ExtensionData['Location'],RouteTypeDefault,'']

Type: All Functions

Name: TRoute

Parameter	Value
Destination	ExtensionData['Number']
Location	ExtensionData['Location']
Route Type	RouteTypeDefault
DNIS	

Return value type: VOID. This function instructs Router to send a routing instruction to T-Server with the specified parameters. TRoute routes the interaction definitively, i. e. after the function is called, Router drops the interaction. We recommend using this

Figure 10: The Function Properties

Feature Limitations

The following known limitation currently applies to multi-site supervision:

- SIP Server will report a correct `DNRole Observer (10)` parameter in corresponding events for a supervisor's DN only if SIP Server operates in a pure SIP environment.

Remote Supervision

The Remote Supervision feature enables supervisors to monitor agent calls from outside the contact center—for example, from an off-premise cell phone. Call prompting when the supervisor first dials into the contact center is used to determine whether the supervisor is authorized to access the service, what target they want to monitor, and for how long.

The Remote Supervision feature includes the following functionality:

- Credentials check—SIP Sever can check login and password credentials to verify that the supervisor is authorized to access the service.
- Targeted monitoring—The supervisor can choose to monitor either an individual agent or calls distributed to agents from a particular Routing Point or ACD Queue.
- Session persistence—The session can continue after the first monitored call ends, and for all consecutive calls (for the selected target), until the supervisor decides to hang up. In between calls, the supervisor's call is parked.
- DN translation—If configured for it, SIP Server can translate the supervisor's external DN to an internal DN, so that Reporting can monitor the call.
- Standard Call Supervision supported—Remote Supervision also supports the following Call Supervision functions: Supervision Modes and Supervision Scopes. For a description of these functions, see "Call Supervision" on page 104.

Feature Configuration

When using this feature, a remote supervisor dials from outside the contact center to a Routing Point with a special URS routing strategy. The strategy collects a caller's login information. Additionally, the strategy may collect the following information from the caller (or otherwise specify):

- A desired monitoring target number
- A supervision type (`AllCalls`), mode, and scope
- An associated internal DN, used for reporting purposes
- A post-feature destination DN

The strategy places these parameters as `Extensions` attributes in the `TRouteCall` request.

Monitoring session starts by routing a remote supervisor's call to the special pre-defined DN with the number `gcti::park`. This DN is used to park the supervisor's call before call monitoring starts, and between calls, when several calls are monitored.

While the call is parked, the supervisor hears silence or a music file, specified by the `parking-music` option.

Procedure: Configuring remote supervision

Start of procedure

1. In Configuration Manager, select the SIP Server `Application` object and add the `parking-music` option in the `TServer` section on the `Options` tab. This option specifies the music file, which will be played to a remote party parked on the `gcti::park` DN.
2. Plan the Remote Supervision routing strategy to meet your specific needs.

   The sample strategy described below is a simplified prototype. You may design your own strategy to include any custom logic available in the URS, implement credential verification, based on accessing enterprise databases, and utilize custom prompts. As a result, the strategy should park a call on the `gcti::park` device.
3. Prepare the recordings of the voice prompts, which are used in the routing strategy to collect the caller's login information and optional feature selection. In this sample configuration, Stream Manager is used for announcement service and caller input collection.

   a. Prepare the voice prompts in 8 KHz mono audio format, encoded with one of the codecs, supported by Stream Manager.

   b. Assign a numeric ID, unique within the Stream Manager's `announcement` directory, to each announcement.

   c. Name each recording file using the following format:

   ```
   <ID>_<codec_suffix>.wav
   ```

   Refer to the *Stream Manager 7.6 Deployment Guide* for the list of supported codecs and corresponding filename suffixes.

   For efficiency, to avoid real-time trans-coding, you may also use the SMzip utility, supplied with Stream Manager to trans-code your prompts off-line into zip files, containing the prompts encoded with all supported audio codecs.

   To trans-code each prompt, use a command similar to the following:

   ```
   >smzip -ac all 9000.zip 9000_pcmu.wav
   ```

   d. Place the resulting audio files (either `.wav` or `.zip`) into the Stream Manager's `announcement` directory. If you use multiple Stream Managers, make sure that the files are replicated through all Stream Managers' `announcement` directories.

The following sample strategy collects the caller's Agent Login and password, requests the monitoring target number, and then initiates multiple-call monitoring, using the collected caller's input.

The strategy uses the following prompts.

Table 11: Strategy Prompts

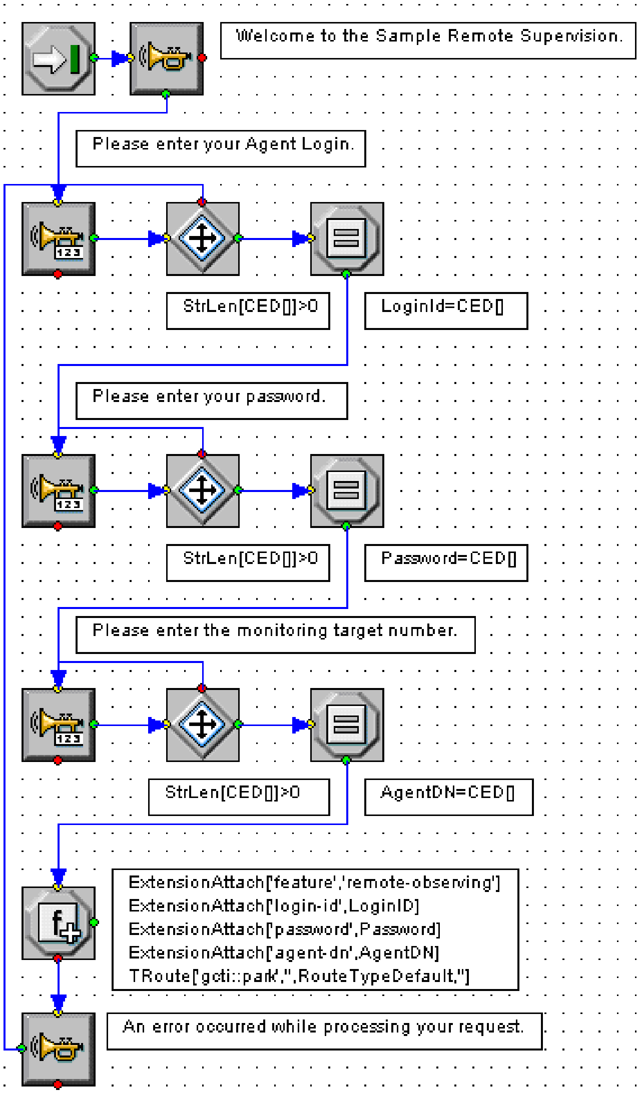
Prompt text	ID	File Name
Welcome to the Sample Remote Supervision.	9000	9000_pcmu.wav
Please enter your Agent login.	9010	9010_pcmu.wav
Please enter your password.	9020	9020_pcmu.wav
Please enter the monitoring target number.	9030	9030_pcmu.wav
An error occurred while processing your request.	9090	9090_pcmu.wav

4. In Interaction Routing Designer (IRD), design your routing strategy. See "Routing Strategy Design Sample" on page 117.
5. Save the complete strategy in IRD.
6. Load the strategy into a Routing Point by using the `Loading` tab in IRD.
7. Test your strategy by placing a call from an external phone to the Routing Point number. Calls from internal DNs are not allowed.

End of procedure

Routing Strategy Design Sample

This section describes a strategy design sample (see Figure 11), followed by the explanation of each block, called in the IRD terms *routing object.*

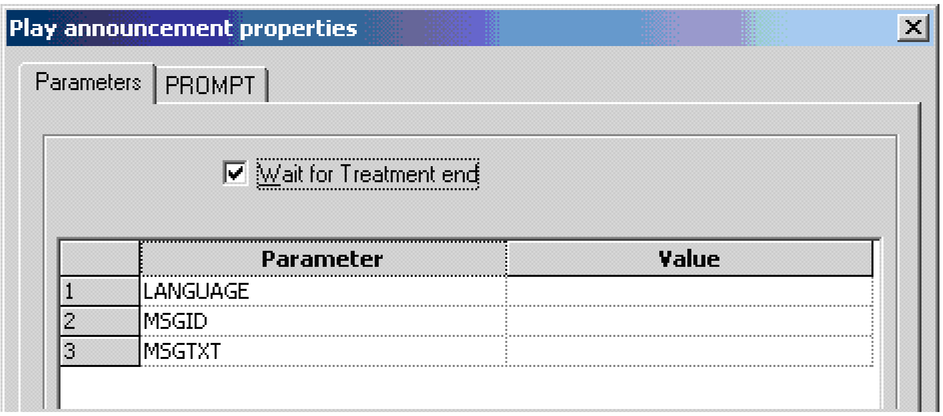

Welcome to the Sample Remote Supervision.
Please enter your Agent Login.
StrLen[CED[]]>0
LoginId=CED[]
Please enter your password.
StrLen[CED[]]>0
Password=CED[]
Please enter the monitoring target number.
StrLen[CED[]]>0
AgentDN=CED[]
An error occurred while processing your request.

```
ExtensionAttach['feature','remote-observing']
ExtensionAttach['login-id',LoginID]
ExtensionAttach['password',Password]
ExtensionAttach['agent-dn',AgentDN]
TRoute['gcti::park','',RouteTypeDefault,'']
```

Figure 11: A Routing Strategy Design Sample

Initial Greeting

The first `Play announcement` routing object is used to play back the initial greeting (see Figure 12).



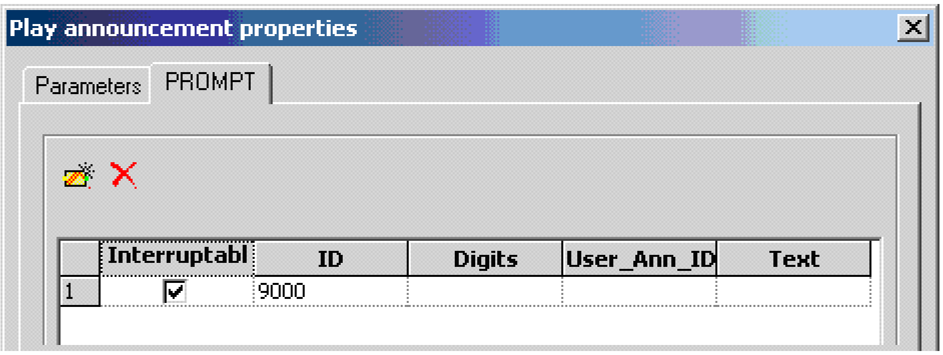
☑ Wait for Treatment end

	Parameter	Value
1	LANGUAGE	
2	MSGID	
3	MSGTXT	

Figure 12: Initial Greeting: Parameters Tab

The LANGUAGE, MSGID and MSGTXT parameters are not used by the SIP Server implementation of the Announcement treatment. The Wait for treatment end check box specifies to the URS that it should wait for the treatment to complete, before proceeding to the next strategy step.

On the PROMPT tab (see Figure 13), the ID specifies the prompt to be played. The Interruptible flag allows the caller to skip the greeting message by pressing any key on the phone keypad.



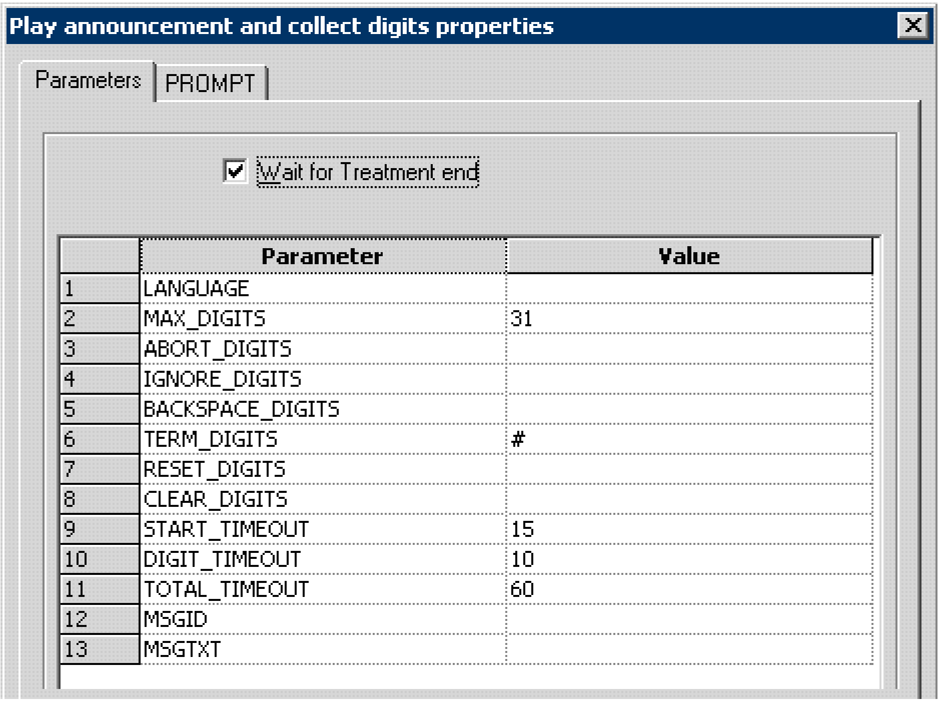
	Interruptabl	ID	Digits	User_Ann_ID	Text
1	☑	9000			

Figure 13: Initial Greeting: PROMPT Tab

Collecting Agent Login Code

Although your strategy may implement an arbitrary approach to perform caller verification, this sample strategy demonstrates the SIP Server's built-in functionality. SIP Server verifies the login-id and password information provided in corresponding extensions parameters, against the Agent Login Code and Password, specified in the Agent Login object in the Configuration Layer. Therefore, for the purpose of this strategy, ensure that you have configured Agent Login objects with the numeric-only Agent Login Codes and Passwords, so that they could be entered through the phone keypad.

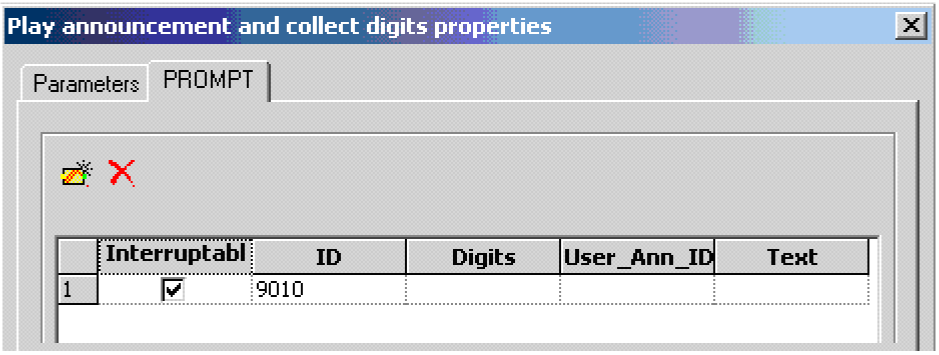
After the initial greeting, the strategy uses the Play Announcement and collect digits routing object to request the caller's Agent Login (see Figure 14). It plays the "Please enter your Agent login" prompt and retrieves the caller's digits input. The caller is expected to enter a numeric login code, up to 31 digits long, terminated by the "#" key.



☑ Wait for Treatment end

	Parameter	Value
1	LANGUAGE	
2	MAX_DIGITS	31
3	ABORT_DIGITS	
4	IGNORE_DIGITS	
5	BACKSPACE_DIGITS	
6	TERM_DIGITS	#
7	RESET_DIGITS	
8	CLEAR_DIGITS	
9	START_TIMEOUT	15
10	DIGIT_TIMEOUT	10
11	TOTAL_TIMEOUT	60
12	MSGID	
13	MSGTXT	

Figure 14: Collecting Digits: Parameters Tab

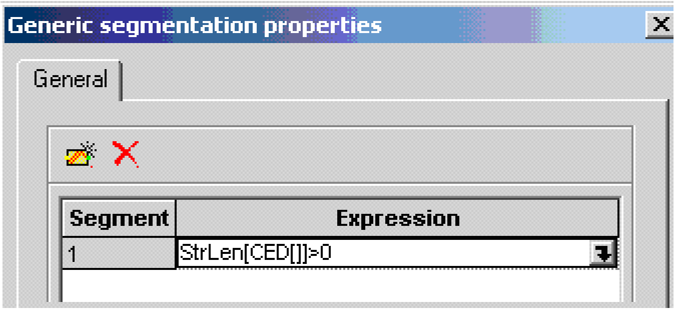


	Interruptabl	ID	Digits	User_Ann_ID	Text
1	☑	9010			

Figure 15: Collecting Digits: PROMPT Tab

Verifying Caller Input

The next `Generic segmentation` object verifies that the caller's input (returned by the `CED[]` function) is not empty (see Figure 16).



Generic segmentation properties

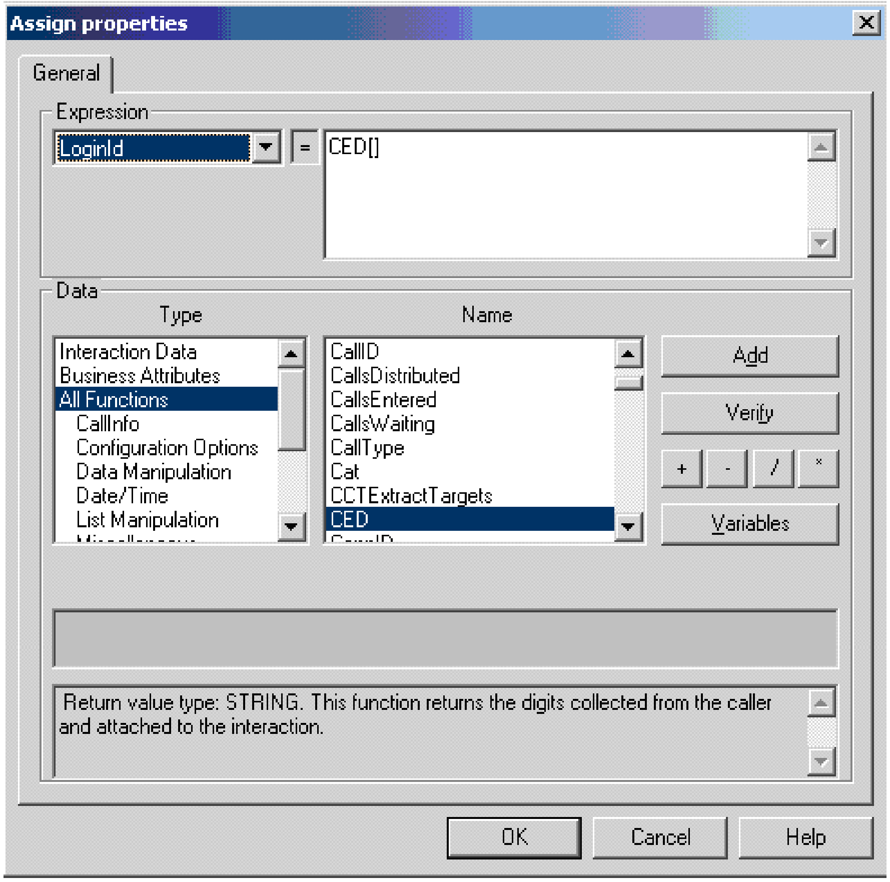
Segment	Expression
1	StrLen[CED[]]>0

Figure 16: The Generic Segmentation Object

If the caller did not enter any digits within 15 seconds (as specified by the `START_TIMEOUT` parameter in the preceding `Play Announcement and collect digits` object, the segmentation object will repeat the previous prompt.

Storing the Entered Agent Login Code in a Variable

The next `Assign` routing object defines the `LoginId` internal strategy variable (use `Variables` button to define the variable) and assigns the caller's input value to this variable (see Figure 17).



Assign properties

Expression: LoginId = CED[]

Data

Type: All Functions

Name: CED

Return value type: STRING. This function returns the digits collected from the caller and attached to the interaction.

Figure 17: The Assign Object

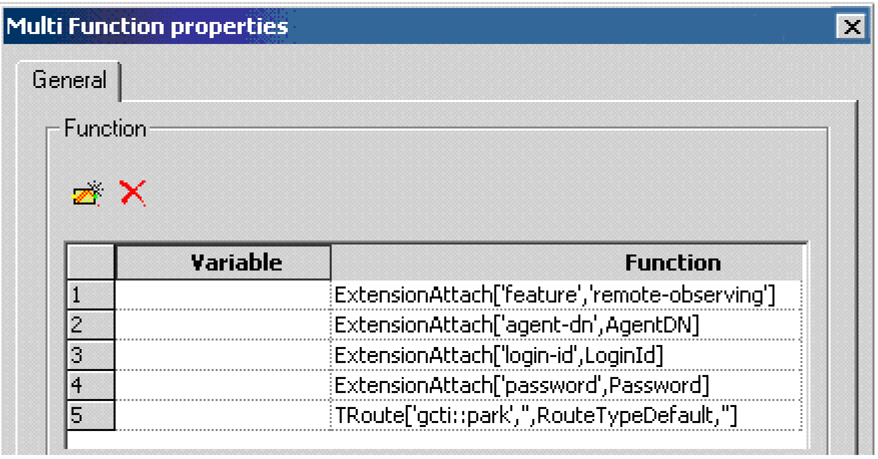
Collecting Password and Monitoring Target

Subsequent routing strategy blocks prompt the caller to enter a password and the desired monitoring target DN number, an a sequence, similar to the one used to collect the Agent Login information.

Collected input is placed into the `Password` and `AgentDN` internal variables.

Invoking Remote Monitoring

The key point of the strategy is the `Multi Function` object (see Figure 18).



	Variable	Function
1		ExtensionAttach['feature','remote-observing']
2		ExtensionAttach['agent-dn',AgentDN]
3		ExtensionAttach['login-id',LoginId]
4		ExtensionAttach['password',Password]
5		TRoute['gcti::park','',RouteTypeDefault,'']

Figure 18: The Multi Function Object

The `Multi Function` object attaches the extensions parameters to the `TRouteCall` request, as the request extensions, and routes the call to the `gcti::park` DN.

SIP Server determines the desired feature—remote observing from the value of the `feature` extension parameter—and verifies the user's login information.

If the supplied information is correct, SIP Server starts Remote Monitoring session. The supervisor hears silence or a music file until a call comes to the specified target. After that, the caller connects to the monitored call.

It the parameters are incorrect (for example, wrong login, password, or monitoring target information), SIP Server responds with the `EventError` message to the `TRouteCall` request. This will trigger the default branch of the `Multi Function` object.

Verifying the Result

If an error occurs for any reason, the strategy uses the subsequent `Play announcement` object to notify the caller about the error and return the control flow to the point, where the user is requested to enter the Agent Login again.

Use of the Extensions Attribute

The following Extensions attribute parameters can be used to configure the application that starts and defines the remote supervision session:

- feature—This key with the remote-observing value triggers registration of the routed party as a supervisor with parameters specified in additional Extensions described in this section.
- dn—An optional DN number that can be used during the monitoring session, as a substitute for the external PSTN number that the supervisor used to dial in. If you do not include this parameter, no TEvents will be distributed for this DN.
- login-id and password—These optional parameters are used to establish that the supervisor is authorized for remote access to the feature. The treatment can prompt for the login-id alone, or for both the login-id and the password.
- agent-dn—The target that the supervisor wants to monitor. This can be a Routing Point, ACD Queue, or an agent DN.
- monitor-type (AllCalls)—An optional parameter that enables the supervisor to monitor all consecutive calls for the selected target, until the supervisor decides to hang up and end the monitoring session. In between monitored calls, the supervisor's call is parked.
- post-feature-dn—An optional parameter that specifies a Routing Point, to which the supervisor will be connected after the supervision session.

Feature Limitations

The following known limitations currently apply to remote supervision:

- MonitorMode of remote supervision session cannot be changed during active supervision.
- One Call supervision type is not supported.

Call Transfer and Conference

SIP Server supports the following call transfers:

- First-party call control (1pcc) transfers: single-step and two-step transfers.
- Third-party call control (3pcc) transfers: single-step and two-step transfers.

In these scenarios the REFER request method is used. If an endpoint does not support the REFER method, the re-INVITE method can be configured for use in two-step transfers.

Note: First-party call control calls are not supported when SIP Server is integrated with the Siemens HiPath 8000 switch.

Starting with release 7.5, SIP Server can send a REFER message to the transferred party when the following scenarios occur:

- A REFER message was received from an endpoint.
- A single-step call transfer was received from a client.

This removes SIP Server from the SIP signalling loop.

SIP Server analyzes the destination specified in either scenario and then determines if a different contact is specified in the outgoing REFER message based on the following criteria:

- The destination is unknown to SIP Server (no regular DN and no Trunk DN contains the prefix that matches the specified destination).
- The destination refers to a DN of type Trunk that contains the oosp-transfer-enabled option set to true (see page 236).

If either of the scenarios is true, SIP Server prepares a Contact for the Refer-To header of the outgoing REFER message, based on the following conditions:

- If there is no DN, and no DN of type Trunk is specified as the destination, the Contact information from the caller DN or the Trunk DN will be specified in the Refer-To header of the outgoing REFER message. It is the responsibility of the caller to determine where to transfer the call.
- If a Trunk DN is specified as the destination, and it contains the oosp-transfer-enabled option set to true, the contact information from this trunk will be specified in the Refer-To header of the outgoing REFER message.

Note: If the caller DN or the Trunk DN in either scenario contains the override-domain option specified (page 237), the value of this option will be specified in the Refer-To header of the outgoing REFER message.

Routing to External Destination Using REFER

SIP Server supports the routing of an inbound call to an external destination by using the REFER method. When the feature is activated, SIP Server places itself in the Out Of Signaling Path. This feature applies to the following scenarios:

- An inbound call is routed from a Routing Point to an external destination.
- An agent transfers an inbound call by using the single-step transfer to a Routing Point, and then the call is routed to an external destination.
- An agent transfers an inbound call by using the blind transfer to a Routing Point, and then the call is routed to an external destination.

Note: This feature is not applicable for scenarios (the second and the third, above) where a conference (supervision or emergency recording) is involved.

To support this feature, configure the following options for a DN of type `Trunk`:

- `oosp-transfer-enabled`—Set the value for this option to `true` (page 236).
- `refer-enabled`—Set the value for this option to `true` (page 240).

Single-Step Transfer Using re-INVITE

Scenarios in which single-step transfers use the re-`INVITE` request method require that the originating DN be configured with the `refer-enabled` option set to `false` (see page 240).

Conference Calls

SIP Server supports third-party call control conferences with central mixing, using MCUs that support more than three participants.

Silence Treatment in Conference

SIP Server can provide a silent treatment for conference call participants when one of them places the call on hold. This will allow conference call participants to continue the conference without interruption (or hearing the music-on-hold treatment). This feature is applicable to conference calls where participants are located in single-site or multi-site environments.

For this feature to work, the `music-in-conference-file` option (see page 213) must be set to the valid name of the silent audio file to be played in applicable conferences (more than two active participants).

Consultation Transfers and Conferences

SIP Server provides the ability for parties participating in a consultation call to initiate 3pcc (third-party call control) transfers or conferences. A typical supported scenario would be:

1. An inbound call is routed to Agent A.
2. Agent A originates a consultation call with Agent B.
3. Agent B originates a consultation call with Agent C.

Consultation Transfers For Calls on a Routing Point

SIP Server allows an agent to complete a consultation transfer of a call that is located on a Routing Point. This transfer operation is supported in single-site and multi-site environments.

In a single-site environment, the call transfer can be completed both when a treatment is playing for the call on a Routing Point, or when the call is just parked on a Routing Point.

In a multi-site environment, when a consultation call is made to a Routing Point located on another site, the call transfer can be completed only when a treatment is playing for the call on the Routing Point. If a call is just parked on a Routing Point, the complete transfer operation will not be successful and SIP Server will generate an `EventError` (`Call in invalid state`) message.

Alternating Between Main and Consultation Calls

SIP Server enables agents to handle up to three 3pcc calls on their SIP endpoint. This functionality supports alternate call operation between the main call and an answered consultation call, as well as the main call and a consultation call that is queued on a Routing Point, as described in the following scenario:

1. A call is routed to Agent A.
2. Agent A places the call on hold and initiates a consultation call by dialing to a Routing Point.
3. Agent A is placed in a queue at the Routing Point, waiting for another agent to become available (a treatment is played).
4. Agent A places the consultation call on hold and retrieves the main call from hold.

For the alternate call operation to work transparently in a multi-site environment, a treatment must be applied to a call on a Routing Point at the earliest possible time. If a treatment is not applied, the alternate call operation will not be successful and SIP Server will generate an `EventError (Call in invalid state)` message.

Feature Limitations

The following known limitation currently applies to call conferences:

- Three-way conference on the phone is not reported properly. Call participants can talk to each other, but such a call is not reported as conference.

Class of Service

Class of Service (COS) is the functionality that defines telephony capabilities for a device or an agent. In SIP Server, COS telephony capabilities are defined by configuring the following:

- Outbound dialing rules
- Ring-through rules

Class of Service can be assigned to the device (a `DN` object in SIP Server `Switch` configuration) or to the agent (an `Agent Login` object in SIP Server `Switch` configuration).

The COS assigned to the agent takes precedence over the COS assigned to the device. That is, when different COSs are assigned to the device and to the agent, SIP Server will use the COS assigned to the agent.

Outbound Dialing Rules

Outbound dialing rules define whether an agent or device is able to make outbound calls, and they also translate dialed numbers according to the specified patterns.

An outbound dialing rule is specified as a pair of patterns. The first pattern, `in-pattern,` is used to match the dialed number. The second pattern, `out-pattern,` defines how to transform the dialed number, when matched, to the `in-pattern`.

It is possible to specify several outbound dialing rules within a single COS definition.

When an outbound call is initiated either from a device (1pcc) or from a softphone (3pcc), SIP Server attempts to match the dialed number with one of the outbound dialing rules. If the matching rule is not found, SIP Server will proceed with the outbound call as it is dialed. If the matching rule is found (that is, `in-pattern` in one of the outbound dialing rules matched the dialed number), SIP Server will apply this rule to the call according to the following procedure:

1. `out-pattern` from the outbound dialing rule is applied to the dialed number to produce the resulting dialed number.
2. If the resulting dialed number is empty, SIP Server rejects an attempt to make the outbound call.
3. If the resulting dialed number is not empty, SIP Server attempts to place the outbound call to the resulting dialed number.

If there are multiple outbound dialing rules configured within a single COS, and `in-pattern` in more than one rule matches the dialed number, SIP Server will choose the outbound dialing rule with the most matched digits in it.

Example

Here is an example of the outbound dialing rule:

```
in-pattern=411;out-pattern=9411
```

Dialing Rule Format

Dialing rules must conform to the following syntax, represented using Augmented Backus-Naur Form (ABNF) notation.

```
dialing-plan-rule      = [name] in-pattern [out-pattern]
name                   = *( ALPHA / DIGIT / "-")
in-pattern             = 1*(digit-part / abstract-group)
out-pattern            = 1*(symbol-part / group-identifier) *param-part
digit-part             = digits / range / sequence
symbol-part            = digits / symbols
range                  = "[" digits "-"  digits "]" group-identifier
sequence               = "[" 1*(digits [","] ) "]" group-identifier
abstract-group         = fixed-length-group / flexible-length-group
fixed-length-group      = 1*group-identifier
flexible-length-group   = "*"  group-identifier
param-part             = ";" param-name "=" param-value
param-name             = "ext" / "phone-context" / "dn"
param-value            = 1*ANYSYMBOL
group-identifier       = ALPHA
digits                 = 1*DIGIT
symbols                = 1*("-" / "+" / ")" / "(" / ".")
```

Common Syntax Notations

Syntax notations common to many of these rules include:

- `*`—Indicates that 0 to an infinite number of the item following this symbol are acceptable.
- `1*`—Indicates that one repetition is required. For T-Server, only one instance is acceptable.
- `/`—Indicates that any of the items mentioned, or a combination of those items, is acceptable.

Component Notations

Component notations include:

- `dialing-plan-rule = [name] in-pattern [out-pattern]`

  Where:

  - `[name]` is the name of the rule option—for example, `rule-01`. In ABNF notation, the brackets (`[)]`) indicate that 0 or 1 instance of the component is required. However, for SIP Server, a name is required.

  - in-pattern is the part of the rule to which SIP Server looks when attempting to match the input number.
  - [out-pattern] is the part of the rule that instructs SIP Server how to translate the input number into the required format. The brackets indicate that either 0 or 1 instance is required. You must create an out-pattern for number translation rules.

- name = *(ALPHA / DIGIT / "-")

  Where:

  - ALPHA indicates that letters can be used in the name for the rule option.
  - DIGIT indicates that numbers can be used in the name for the rule option.
  - "-" indicates that a dash (-) can also be used in the option name—for example, rule-01.

- in-pattern = 1*(digit-part / abstract-group)

  Where:

  - digit-part represents numbers. SIP Server uses this when selecting the most appropriate rule from the entire dialing plan.
  - abstract-group represents one or more letters with each letter representing one or more numbers. SIP Server uses this when transforming a dial string.

  For example, [1-9] is the digit-part (representing a range of numbers) and ABBB is the abstract-group for in-pattern=[1-9]ABBB.

- out-pattern = 1*(symbol-part / group-identifier) *param-part

  Where:

  - symbol-part represents digits, symbols, or a combination. Symbols are rarely used. They are not used in the United States.
  - group-identifier are letters that represent groups of numbers. A letter in the out-pattern represents one or more digits, based on the number of times the letter is used in the in-pattern.
  - *param-part represents an additional parameter, such as phone-context. Remember that an asterisk (*) means that 0 to an infinite number of these are acceptable.

  For example, in rule-04; in-pattern=1AAABBBCCC;out-pattern=91ABC, 91 is the symbol-part; A, B, and C are group-identifiers in the out-pattern, each representing three digits, since there are three instances of each in the in-pattern.

  Note: Prefix an out-pattern value with a plus sign (+) for the inbound rule when the output must be in a global form (E.164 format).

- digit-part = digits / range / sequence

  Where:

  - digits are numbers 0 through 9.

  - `range` is a series of digits—for example, 1–3.
  - `sequence` is a set of digits.
- `symbol-part = digits / symbols`

  Where:
  - `digits` are numbers 0 through 9.
  - `symbols` include such characters as `+`, `-`, and so on.
- `range = "[" digits "-" digits "]" group-identifier`

  Where:
  - `"[" digits "-" digits "]"` represents the numeric range—for example, `[1-2]`.
  - `group-identifier` represents the group to which the number range is applied.

    For example, `[1-2]` applies to group identifier `A` for `in-pattern=[1-2]ABBB`. When SIP Server evaluates the rule to determine if it matches the number, it examines whether the first digit of the number, identified as `group-identifier A`, is `1` or `2`.
- `sequence = "[" 1*(digits [","] ) "]" group-identifier`

  Where:
  - `"[" 1*(digits [","] ) "]"` represents a sequence of digits, separated by commas, and bracketed. SIP Server requires that each digit set have the same number of digits. For example, in `[415,650]` the sets have three digits.
  - `group-identifier` represents the group to which the number sequence is applied.

    For example, in `in-pattern=1[415,650]A*B`, `[415,650]` applies to `group-identifier A`. When SIP Server evaluates the rule to determine if it matches the number, it examines whether the three digits (`group-identifier A`) following the `1` in the number are `415` or `650`.
- `abstract-group = fixed-length-group / flexible-length-group`

  Where:
  - `fixed-length-group` specifies a group composed of a specific number of digits, and determined by the number of times the group identifier is included in the `in-pattern`. For example, for `in-pattern=1AAABBBCCCC`, there are three digits in group `A` and `B`, but four in group `C`.

    When you create an `out-pattern`, you include the group identifier only once, because the `in-pattern` tells SIP Server how many digits belong in that group. For example, `rule-04` (see page 132) is `in-pattern=1AAABBBCCCC; out-pattern=91ABC`.
  - `flexible-length-group` specifies a group composed of 0 or more digits in the group represented by the `group-identifier`.

    For example, in `in-pattern=1[415,650]A*B`, `*B` represents the flexible length group containing the remaining digits in the number.

The component `abstract-group` is used only for the `in-pattern`.

- `fixed-length-group = 1*group-identifier`

  See the earlier explanation under `abstract-group`.

- `flexible-length-group = "*"  group-identifier`

  See the earlier explanation under `abstract-group`.

- `param-part = ";" param-name "=" param-value`

  Where:

  - `";"` is a required separator element.
  - `param-name` is the name of the parameter.
  - `"="` is the next required element.
  - `param-value` represents the value for `param-name`.

- `param-name = "ext" / "phone-context" / "dn"`

  Where:

  - `"ext"` represents the extension.
  - `"phone-context"` represents the value of the `phone-context` option configured on the switch.
  - `"dn"` represents the directory number.

- `param-value = 1*ANYSYMBOL`

  Where:

  - `ANYSYMBOL` represents any number, letter, or symbol with no restrictions.

- `group-identifier = ALPHA`
- `digits = 1*DIGIT`
- `symbols = 1*("-" / "+" / ")" / "(" / ".")`

Transformation

The transformation algorithm consists of two phases:

1. Input number parsing based on `in-pattern`.
2. Output number construction based on `out-pattern` and the parsed input string.

The first phase consists of selecting all group elements from `in-pattern` (the `abstract-group`, `range`, and `choice` parameters), and assigning them a corresponding group of digits from the input number. There are three kinds of such assignment:

- Explicit—For ranges, sets, and fixed-length-groups where the length and position of the group is set explicitly in `in-pattern`.
- Positional—For flexible-length-groups; the digits are selected exclusively based on their position with respect to other groups.
- Analytical—For entities; the digits are selected based on some kind of analytical processing.

Examples

These are examples of how SIP Server applies the rules configured above to various input numbers.

Example 1

Input Number	in-pattern	out-pattern	Output Number
914159131472	9*A	+A	+14159131472
011441581234567	011#CABBB*D	+A-(B)-D	+44-(158)-1234567
1472	[1-8]ABBB	AB	1472
911	AAA	9A	9911
14159131472	*A9131472	80409131472	80409131472
16503570622	1[415,650]A*B	B	3570622

Example 2

```
rule-01=in-pattern=[1-8]ABBB;out-pattern=AB
rule-02=in-pattern=AAAA;out-pattern=A
rule-03=in-pattern=1[415,650]A*B;out-pattern=B
rule-04=in-pattern=1AAABBBCCCC;out-pattern=91ABC
rule-05=in-pattern=*A913BBBB;out-pattern=80407913B
rule-06=in-pattern=011#CA*B;out-pattern=9011AB
```

rule-01 SIP Server receives input number 2326.

As a result of the rule selection process, SIP Server determines that the matching rule is rule-01:

```
name=rule-01;in-pattern=[1-8]ABBB;out-pattern=AB
```

The matching count for this rule is 1, because Group A matches the digit 2.

As a result of the parsing process, SIP Server detects two groups: Group A = 2 and Group B = 326.

SIP Server formats the output string as 2326.

rule-02 SIP Server receives input number 9122.

As a result of the rule selection process, SIP Server determines that the matching rule is rule-02:

```
name=rule-02;in-pattern=AAAA;out-pattern=A
```

The matching count for this rule is 0; however, the overall length of the input number matches that of the in-pattern configuration.

As a result of the parsing process, SIP Server detects one group: Group `A = 9122`.

SIP Server formats the output string as `9122`.

rule-03 SIP Server receives input number `16503222332`.

As a result of the rule selection process, SIP Server determines that the matching rule is `rule-03`:

```
name=rule-03;in-pattern=1[415,650]A*B;out-pattern=B
```

The matching count for this rule is `4`, because the first digit matches and all three digits in Group `A` match.

As a result of the parsing process, SIP Server detects two groups: Group `A = 650` and Group `B = 3222332`.

SIP Server formats the output string as `3222332`.

rule-04 SIP Server receives input number `19253227676`.

As a result of the rule selection process, SIP Server determines that the matching rule is `rule-04`:

```
name=rule-04;in-pattern=1AAABBBCCCC;out-pattern=91ABC
```

The matching count for this rule is `1`, because the first digit matches.

As a result of parsing process, SIP Server detects three groups: Group `A = 925`, Group `B = 322`, and Group `C = 7676`.

SIP Server formats the output string as `919253227676`.

rule-05 SIP Server receives input number `4089137676`.

As a result of the rule selection process, SIP Server determines that the matching rule is `rule-05`:

```
name=rule-05;in-pattern=*A913BBBB;out-pattern=80407913B
```

The matching count for this rule is `3`, because three digits match.

As a result of the parsing process, SIP Server detects two groups: Group `A = 408` and Group `B = 7676`.

SIP Server formats the output string as `804079137676`.

rule-06 SIP Server receives input number `011441112223333`.

As a result of the rule selection process, SIP Server determines that the matching rule is `rule-06`:

```
name=rule-06;in-pattern=011#CA*B;out-pattern=9011AB
```

The matching count for this rule is `3`, because three digits match.

As a result of the parsing process, SIP Server detects two groups: Group `A = 44` and Group `B = 1112223333`.

SIP Server formats the output string as `9011441112223333`.

Call Rejection by Out-Dialing Rules

A call attempt can be rejected by the out-dialing rules. To indicate this condition, SIP Server generates an EventError response to the corresponding request, with the reason code Invalid Destination DN (415).

Ring-Through Rules

The ring-through rules define whether a call is sent to an agent or a device. The following ring-through rules are supported by SIP Server:

- Reject call when a device is already in a call

  This rule is enforced by the Switch object-level configuration option reject-call-incall (page 241) within COS.

- Reject call when an agent is not ready on a device

  This rule is enforced by the Switch object-level configuration option reject-call-notready (page 241) within COS.

Call Rejection by COS Ring-Through Rules

A call attempt can be rejected by the COS ring-through rules. To indicate this condition, SIP Server generates an EventError response to the corresponding request, with the reason code Invalid Destination DN (93).

Feature Configuration

Table 12 provides an overview of the main steps required to configure Class of Service.

Table 12: Task Flow—Configuring Class of Service

Objective	Related Procedures and Actions
1. Configure a COS DN.	Configure a DN of type Voice over IP Service to represent the COS entity itself. This DN should have outbound dialing rules and ring-through rules specified in the options. See “Configuring a COS DN” on page 135.
2. Assign COS to a device.	Assign the COS DN to one or multiple DNs of type Extension or ACD Position within the same Switch configuration object. See “Assigning COS to a Device” on page 136
3. Assign COS to an agent.	Assign the COS DN to one or multiple Agent Logins. See “Assigning COS to an Agent” on page 136.

Procedure: Configuring a COS DN

Start of procedure

1. Create a COS DN under the SIP Server `Switch` object with a type of `Voice Over IP Service`.
2. On the `Annex` tab of the COS `DN` object, in the `TServer` section, set the configuration option `service-type` to `cos`.
3. On the `Annex` tab of the COS `DN` object, in the `TServer` section, specify the outbound dialing rules. Each rule is defined using the configuration option named `out-rule-<n>` (page 237)—for example:

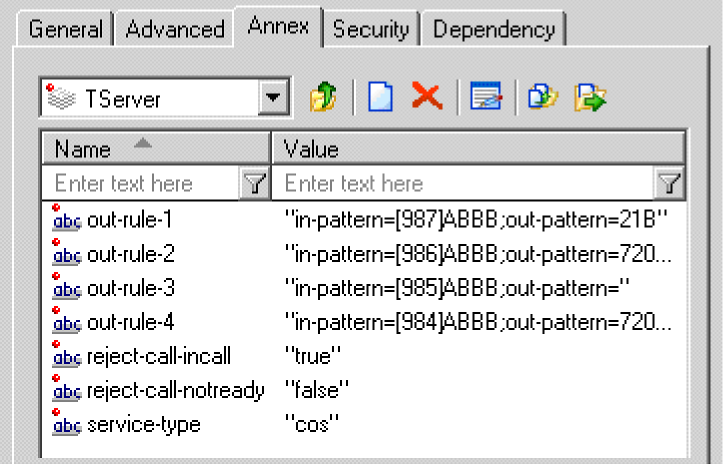
   `out-rule-1 = in-pattern=411;out-pattern=9411`

   Rules must be numbered sequentially—for example, `out-rule-1`, `out-rule-2`, and so on.
4. On the `Annex` tab of the COS `DN` object, in the `TServer` section, specify the ring-through rules—for example:

   `reject-call-incall = true`

   `reject-call-notready = true`

   Figure 19 illustrates a sample configuration for the COS DN with outbound dialing rules and ring-through rules specified.



Name	Value
out-rule-1	"in-pattern=[987]ABBB;out-pattern=21B"
out-rule-2	"in-pattern=[986]ABBB;out-pattern=720...
out-rule-3	"in-pattern=[985]ABBB;out-pattern="
out-rule-4	"in-pattern=[984]ABBB;out-pattern=720...
reject-call-incall	"true"
reject-call-notready	"false"
service-type	"cos"

Figure 19: Configuring COS DN: Sample Configuration

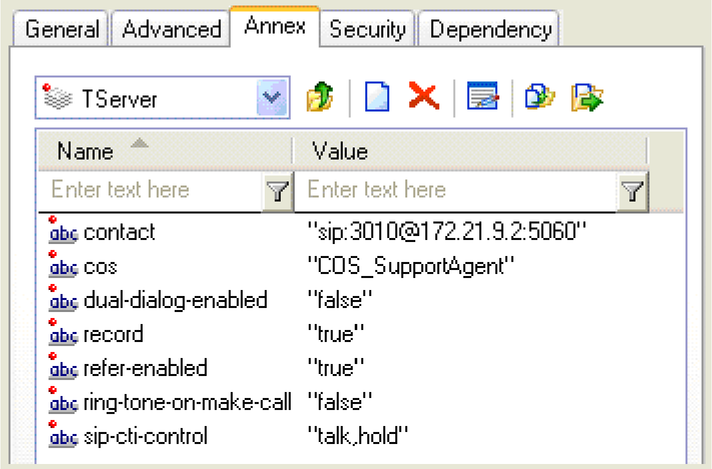
5. When you finished, click `Apply` to save your configuration.

End of procedure

Assigning COS to a Device

COS is assigned to a device by associating the COS DN with the device DN of type Extension or ACD Position. In the TServer section on the Annex tab, add the cos configuration option, with the value set to the name of the COS DN.

Figure 20 illustrates a sample configuration of the device DN with COS assigned.



Name	Value
contact	"sip:3010@172.21.9.2:5060"
cos	"COS_SupportAgent"
dual-dialog-enabled	"false"
record	"true"
refer-enabled	"true"
ring-tone-on-make-call	"false"
sip-cti-control	"talk,hold"

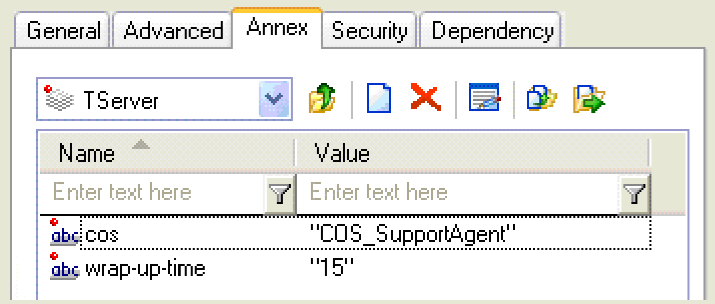
Figure 20: Assigning COS to a Device: Sample Configuration

COS may be assigned to multiple DNs by using the Manage Options command in Configuration Manager.

Assigning COS to an Agent

COS is assigned to an agent by associating the COS DN with the Agent Login object. In the TServer section on the Annex tab of the Agent Login object, add the cos configuration option, with the value set to the name of the COS DN.

Figure 21 illustrates a sample configuration of the Agent Login object with COS assigned.



Name	Value
cos	"COS_SupportAgent"
wrap-up-time	"15"

Figure 21: Assigning COS to an Agent Login: Sample Configuration

Checking the Destination Availability

SIP Server uses COS to analyze the availability of the destination in the following order:

1. SIP Server checks if an agent is logged in on the extension.
2. If the agent is logged in, the Agent Login COS is applied.
3. If the agent is not logged in on the device, SIP Server checks if COS is configured for the device.
4. If COS is configured for the device, SIP Server applies this device's COS.
5. If COS is not configured for the device, SIP Server checks if options `reject-call-incall` and `reject-call-notready` are specified for the device directly (without using COS).
6. If the `reject-call-incall` and `reject-call-notready` options are specified, SIP Server uses these options.
7. If none of the preceding apply, SIP Server considers the destination available.

DTMF Tones Generation

SIP Server can send a request to Stream Manager to generate DTMF tones using the `TApplyTreatment` request with the `PlayApplication` treatment.

The following key-value pairs (see Table 13) are used in the attribute `parameters` for the `PlayApplication` treatment:

Table 13: Key-Value Pairs for TreatmentPlayApplication

Key	Type	Value
GSIP_APP_ID	Integer	Specifies the application, which tells SIP Server to send a request to Stream Manager to generate DTMF tones.
GSIP_DTMF_TO_DIAL	String	The DTMF string to be generated.
GSIP_DTMF_DURATION	Integer	The duration of the DTMF tone in msec. This parameter is optional with the default value of `100 msec`.

Feature Configuration

Table 14 provides an overview of the main steps required to configure DTMF tones generation support.

Table 14: Task Flow—Configuring DTMF Tones Generation Support

Objective	Related Procedures and Actions
1. Configure a SIP Server Application object.	In the TServer section on the Options tab of the SIP Server Application object in Configuration Manager, set the configuration option sip-dtmf-send-rtp to true.
2. Configure an application service.	Configure a DN of type Voice over IP Service with the following configuration options: • contact: Set to the device's IP address used for sending DTMF tones. • service-type: Set to application. See "Configuring an application service" on page 89 for details.

Emulated Agents

SIP Server fully emulates business-call–handling functions. It also performs agent emulation for any agent who logs in using a request, when the device provided in the ThisQueue attribute is defined as a Routing Point or ACD Queue in the SIP Server configuration.

SIP Server provides a fully functional agent model that enables full agent support for SIP Server desktop applications as well as for other Genesys solutions.

Business-Call Handling

This section describes how SIP Server handles different types of calls.

SIP Server Call Classification

SIP Server automatically assigns every call to one of three categories—*business, work-related,* or *private*. Based on this assignment, SIP Server applies the appropriate business-call handling after the call is released.

Business Calls

SIP Server automatically categorizes as a *business call* any call distributed to an agent either from a Queue or from a Routing Point. Use the following configuration options to define what additional calls to or from an agent are classified as business calls:

- `inbound-bsns-calls` (page 210)
- `outbound-bsns-calls` (page 216)
- `inherit-bsns-type` (page 210)
- `internal-bsns-calls` (page 211)
- `unknown-bsns-calls` (page 225)

Work-Related Calls

SIP Server categorizes as a *work-related call* any non-business call that an agent makes while in After Call Work (ACW). SIP Server does not apply any automatic business-call handling after a work-related call.

Because emulated agents can make or receive a direct work-related call while in wrap-up time, SIP Server pauses the emulated wrap-up timer for the duration of such a call.

If an agent receives a direct work-related call during legal-guard time, SIP Server cancels the legal-guard timer and reapplies it at the end of the work-related call.

Private Calls

SIP Server categorizes as a *private call* any call that does not fall into the business or work-related categories. SIP Server does not apply any automatic business-call handling after a private call. If emulated agents receive a direct private call while in wrap-up or legal-guard time, the emulated wrap-up or legal-guard timer is not interrupted.

Emulated Agents Support

SIP Server provides a fully functional emulated-agent model that you can use either in addition to agent features available on the PBX, or in place of them where they are not available on the PBX.

When this feature is used, SIP Server emulates the following functionality:

- Login and logout
- Agent set Ready
- Agent set Not Ready (using various work modes)
- Automatic after-call work
- After call work in idle

- Automatic legal-guard time to provide a minimum break between business-related calls

Emulated Agent Login/Logout

You can configure SIP Server to perform emulated login either always, never, or on a per-request basis. Use the following SIP Server configuration options to configure emulated agent login:

- `emulate-login` (page 248)
- `emulated-login-state` (page 206)
- `agent-strict-id` (page 200)

Emulated Agent Ready/NotReady

Emulated agents can perform an emulated `Ready` or `NotReady` request regardless of whether they are on a call, subject to the rules governing work modes.

SIP Server also reports any change in agent mode requested by the agent while remaining in a `NotReady` state (*self-transition*).

Note: Note that the *Genesys 7 Events and Models Reference Manual* and the *Voice Platform SDK 7.6 .NET* (or *Java*) *API Reference* define which agent state/agent mode transitions are permissible.

Emulated After-Call Work

SIP Server can apply emulated wrap-up (ACW) for agents after a business call is released, unless the agent is still involved in another business call (see “Business Calls” on page 139).

Timed and Untimed ACW

SIP Server applies emulated ACW for an agent after any business call is released from an established state. SIP Server automatically returns the agent to the `Ready` state at the end of a *timed* ACW period. The agent must return to the `Ready` state manually when the ACW period is *untimed*.

Events and Extensions

SIP Server indicates the expected amount of ACW for an agent in `EventEstablished`, using the extension `WrapUpTime`. It is not indicated in `EventRinging`, because the value may change between call ringing and call answer. Untimed ACW is indicated by the string value `untimed`; otherwise, the value indicates the expected ACW period in seconds.

SIP Server reports ACW using `EventAgentNotReady` with `workmode` = `3` (`AgentAfterCallWork`), and it indicates the amount of ACW it will apply using the extension `WrapUpTime`.

SIP Server sends EventNotReady(ACW) before EventReleased at the end of the business call.

Emulated ACW Period

The amount of emulated ACW that SIP Server applies (when required) after a business call is determined by the value in configuration option wrap-up-time.

Configuration option untimed-wrap-up-value determines which specific integer value of wrap-up-time indicates *untimed* ACW. To specify untimed ACW in request extensions or user data, you should use the string untimed instead. All positive integer values are treated as indicating timed ACW (in seconds). For backward compatibility, the default value of untimed-wrap-up-value is 1000.

Note: Changing the value of untimed ACW should be done with care, because it may affect the interpretation of all integer values of the option wrap-up-time in Configuration Manager. If lowered, it may change timed ACW to untimed ACW, or disable ACW altogether. If raised it may change untimed or disabled ACW to timed ACW. The use of the option (string) value untimed is encouraged where possible to minimize the impact of any future changes to the value of option untimed-wrap-up-value.

ACW in Idle

An agent can activate wrap-up time on request when idle, by issuing a TAgentNotReady request with workmode = 3 (AgentAfterCallWork).

You can configure this feature using the following options:

- timed-acw-in-idle (page 225)
- acw-in-idle-force-ready (page 198)

Extending ACW

An agent can request an extension to the amount of emulated ACW for a call while in emulated ACW or in the legal-guard state.

The agent requests an extension to ACW by sending RequestAgentNotReady with workmode = 3 (AgentAfterCallWork). SIP Server determines the period of the extended ACW from the extension WrapUpTime, as follows:

- Value = 0—There is no change to the ACW period, but SIP Server reports how much ACW time remains.
- Value greater than 0—SIP Server adds the given number of seconds to the timed ACW period. Untimed ACW remains unaffected.
- Value = untimed—SIP Server applies untimed ACW.

SIP Server sends EventAgentNotReady with workmode = 3 (AgentAfterCallWork), reporting the newly extended amount of ACW using the extension WrapUpTime. If the agent was in the emulated legal-guard state, SIP Server places the agent back into the emulated ACW state.

The agent may extend the period of ACW as many times as desired. At the end of the extended timed ACW period, SIP Server applies legal-guard time, if any is configured. No legal-guard time is applied if the emulated ACW was untimed.

Calls While in Emulated ACW

SIP Server's handling of an agent making or receiving a call while in emulated ACW is governed by the configuration option backwds-compat-acw-behavior (see page 201).

Emulated Legal-Guard Time

SIP Server applies emulated legal-guard time for agents before they are about to be automatically set Ready after any period of timed ACW, or after the last business call is released where there is no ACW to be applied. It is a regulatory requirement in many countries that agents have a break of a few seconds before the next call can arrive. No legal-guard time is applied if the ACW period was not timed, or if the agent is not being placed into the Ready state.

SIP Server reports legal-guard time using EventAgentNotReady with workmode = 2 (LegalGuard). If an agent requests to be logged out during emulated legal-guard time, SIP Server immediately logs the agent out.

If the agent requests to go to a Not Ready or Ready state during legal-guard time, SIP Server terminates legal-guard time and transitions the agent to the requested state. If the agent requests to return to the ACW state, SIP Server reapplies legal-guard time at the end of ACW, provided that the agent still requires it according to the preceding criteria.

The period of legal-guard time is determined by the configuration option legal-guard-time (page 212).

Endpoint Service Monitoring

When SIP Server starts up, it considers that all DNs configured in Configuration Manager are in the In Service state.

Note: Where SIP Server functions as an application server behind a softswitch, a DN is considered to be in Out of Service state if the softswitch responds with a 408 Request Timeout message to an INVITE message during creation of a new call.

DNs are considered out of service in two scenarios:

- The SIP endpoint fails to respond to the incoming INVITE message during the creation of a new call.

- The SIP endpoint fails to respond to the re-INVITE message during an already established telephone call.

In both scenarios—when the SIP endpoint fails to respond to the incoming INVITE message during the creation of a new call, or when it fails to respond to the re-INVITE message during an already established call—SIP Server generates an EventDNOutOfService message.

DNs are considered back in service in several scenarios:

- When a SIP REGISTER message comes from the endpoint
- When an endpoint initiates a call by sending an INVITE message
- When an endpoint responds to an INVITE message with a Ringing (OK) message

An EventDNBackInService message is generated in all scenarios.

Instant Messaging

SIP Server supports the Instant Messaging (IM) media type as follows:

- A special SDP message for the IM media type is supported within the SIP INVITE dialog.

  The IM SDP message must be in a form that is generated and accepted by Microsoft Live Communication Server and Office Communicator. SIP Server supports standard SIP call flows for IM, and the SIP INVITE messages are assumed to be the same for the IM sessions. SIP Server produces the same TEvents for the IM sessions as it does for voice calls.
- A SIP MESSAGE request method is supported within an established INVITE dialog to exchange instant messages within a SIP session.
- The content of an instant message can be distributed via a EventUserEvent message to a DN of type Communication DN with the name gcti::im.

  Any application that need to receive the content of any instant message must register this DN. When a SIP Server MESSAGE request is received by the SIP Server IM session, SIP Server informs its clients by distributing an EventUserEvent message. The client applications can then perform additional tasks in response to the IM content.

Instant Messaging Transcript

SIP Server supports exchanging instant messages via T-Library within the established call context. This enables applications, such as Agent Desktop, to display instant messages that were sent or received during a conversation, including the chat transcript, and to send an instant message to other parties of the call.

The instant message is delivered to a T-Library client via `EventPrivateInfo` messages. A T-Library client can send an instant message using a T-Library request `TPrivateService`.

SIP Server distributes the `EventPrivateInfo` message when one of the participants in the call sends an instant message. The `EventPrivateInfo` message is sent to all other participants in the call. `AttributeExtensions` of the `EventPrivateInfo` message contains information about the instant message.

SIP Server distributes `EventPrivateInfo` messages with the Instant Messages Transcript when a new participant is added to the call as a result of a transfer or conference operation. The Instant Messages Transcript contains all instant messages that were previously exchanged between the participants in the call. `EventPrivateInfo` with the transcript is sent to the new participant who was added to the call.

The following keys are supported:

- `im`—Contains the text of the instant message.
- `im-content-type`—Contains the value provided in the `Content-Type` header of the SIP message that delivered IM.
- `im-transcript`—Contains the transcript of the IM call. The value of this extension is a string containing an XML document that complies with the Genesys Multimedia Chat Transcript Schema.

Note: Transcript data delivery is only supported via T-Library.

Supported Call Operations

SIP Server supports the following call operations within an IM session.

Direct Calls

- Direct 1pcc calls—SIP Server processes direct 1pcc calls with IM media the same way as voice calls. SIP Server will specify `AttributeMediaType = 5` (`TMediaChat`) in TEvents for such calls.
- Direct 3pcc calls—SIP Server processes a `TMakeCall` request for a call with IM media, when it is specified in the `Extensions` attribute with the `chat` key containing a value of `true`. If it is not specified, SIP Server will process the `TMakeCall` request in the normal fashion.

Hold

SIP Server supports the Hold operation in the same manner as for voice calls. The `Hold` request has no effect on the SIP signaling level for IM calls. SIP Server will not re-`INVITE` SIP endpoints with different SDP dialogs when performing the Hold operation, because no changes in the SDP dialog are necessary.

Note: SIP Server does not support Music On Hold (MOH) for IM calls.

Transfer

SIP Server supports transfer in the same manner as for voice calls. However there are some exceptions:

- SIP endpoints that are already part of a call are not considered for re-INVITE requests.
- The IM SDP dialog is used to INVITE the SIP endpoint that is the call transfer destination.

Single-step transfers are supported for non-conference calls. Single-step transfer of the conference is not supported.

Two-step transfers are supported for all calls. Agents can exchange IM text with each other during the consultation call. These IM text messages are not visible to the customer.

Conference

SIP Server supports conference for calls made within the IM session, and it is performed in the same manner as for voice calls. However, SIP endpoints that are already part of a call are not considered for re-INVITE requests. Also, unlike regular voice calls, SIP Server does not use Stream Manager to establish conference calls. SIP Server establishes an IM conference between the SIP endpoints itself, by issuing SIP MESSAGE requests to all conference participants.

Routing

SIP Server supports routing of IM calls using Universal Routing Server (URS) in the same manner as for voice calls. Multiple IM calls can be routed to the same agent. To achieve that, SIP Server distributes AttributeMediaType set to 5 (TMediaChat) in TEvents for IM calls. URS and Stat Server can distinguish IM calls by this attribute and, according to the configuration, route many calls with IM media to the same agent.

Treatments

SIP Server supports treatments for calls with IM media. However, unlike voice calls, SIP Server does not use Stream Manager to apply treatments. Instead, SIP Server executes treatments for IM calls itself by using the SIP MESSAGE request in accordance with the treatment request parameters.

SIP Server supports the following treatments for the IM calls:

- `PlayAnnouncement`
- `CollectDigits`
- `PlayAnnouncementAndDigits`

The `PlayAnnouncement` treatment is performed by SIP Server when it sends the `MESSAGE` request to the caller. The `MESSAGE` request is sent for each prompt specified in the `TreatmentPlayAnnouncement` request. The content of the `MESSAGE` request is created using the `TEXT` parameter in the prompt specified by the `TreatmentPlayAnnouncement` request. When processing the `TreatmentPlayAnnouncement` request for IM calls, SIP Server supports the `TEXT` parameter only.

The `CollectDigits` treatment is performed by SIP Server when it receives the `MESSAGE` request from the caller, and it then sends the complete content of the `MESSAGE` request as collected digits to URS. When processing the `TreatmentCollectDigits` request for IM calls, SIP Server supports the `TOTAL_TIMEOUT` parameter only. All other parameters are not supported.

The `PlayAnnouncementAndDigits` treatment is performed by SIP Server by performing `TreatmentPlayAnnouncement` and then `TreatmentCollectDigits`.

Note: It is recommended that you start a routing strategy with `TreatmentCollectDigits` when applying treatments to an IM call because the strategy can collect the details of the initial IM and store it as `UserData`.

Supervision

SIP Server supports supervision functionality for IM calls, as described in "Call Supervision" on page 104. T-Library messaging for supervisor monitoring scenarios for IM calls is the same as for voice calls, with the only exception that, for IM calls, `AttributeMediaType` is set to `5` (`TMediaChat`).

SIP Server supports the following call supervision modes for IM calls:

- Silent monitoring
- Whisper coaching
- Open supervisor presence

SIP Server supports the following call supervision scopes for IM calls:

- Agent
- Call

SIP Server supports the following call supervision types for IM calls:

- One call
- All calls

Silent Monitoring for IM Calls

When silent monitoring is applied to the IM call, SIP Server uses the following algorithm to distribute instant messages:

- Instant messages sent by a caller or an agent are visible to all participants: the caller, the agent, and the supervisor.
- Instant messages sent by a supervisor are not visible to the caller or the agent.

Whisper Coaching for IM Calls

When whisper coaching is applied to the IM call, SIP Server uses the following algorithm to distribute instant messages:

- Instant messages sent by a caller or an agent are visible to all participants: the caller, the agent, and the supervisor.
- Instant messages sent by a supervisor are visible to the agent only.

Open Supervisor Presence for IM Calls

When open supervisor presence is applied to the IM call, SIP Server uses the following algorithm to distribute instant messages:

- Instant messages sent by caller or agent are visible to all participants: the caller, the agent, and the supervisor
- Instant messages sent by supervisor are also visible to the all participants: the caller, the agent, and the supervisor

This mode is the same as the normal IM conference mode.

Multiple Instant Messaging Sessions

SIP Server supports the handling of several simultaneous IM sessions by one agent. The maximum number of simultaneous sessions for an agent is defined by a capacity rule. The agent can also handle one voice call and several IM sessions.

Feature Configuration

Processing UserData

The following options are used to configure how `UserData` is processed:

- `user-data-im-enabled` (see page 225)
- `user-data-im-format` (see page 226)

Configuring Microsoft Live Communication Server

Microsoft Office Communicator is the client of Microsoft Live Communication Server (LCS). There is no direct communication link between SIP Server and Office Communicator; LCS is the bridge in this scenario. As such, SIP Server is configured to register with LCS, or not to register with LCS.

In either scenario, there is a common configuration:

- SIP Server's IP address must be configured as a trusted server with LCS.
- The following options must be configured for each DN that is using Office Communicator:
  - override-domain—The value of this option must be equal to the computer name from the SIP account of Office Communicator.
  - request-uri—The value of this option must be equal to the computer name from the SIP account of Office Communicator.
  - contact—The value must specify the corresponding LCS IP address, the SIP port, and the TCP transport protocol. For example:

    ```
    192.168.14.121:5060;transport=tcp
    ```

SIP Server Registers with LCS

In this scenario, LCS performs presence monitoring using a special account created in LCS. SIP Server registers with LCS using this account and subscribes for the Office Communicator presence statuses on behalf of this account. The Trunk DN configuration must contain the force-register option (see page 232).

SIP Server Does Not Register with LCS

The subscribe-presence-from option is used to specify the SIP URI in the From: header of the SUBSCRIBE message sent to LCS from SIP Server.

If SIP Server does not register with LCS, this option contains a username in the SIP URI that is not configured on LCS.

In order for non-LCS users to subscribe to the presence state of LCS users (such as Office Communicator users), each LCS user must be configured to allow subscriptions to all users, or to a special user that is specified in the subscribe-presence-from option of the Trunk DN that represents LCS. You must add all users or a special user to the Allow list in the LCS user configuration.

Configuring the Instant Messaging Solution

There is a number of ways to implement the Instant Messaging solutions in Genesys. This includes enabling DNs in your contact center to handle instant messages after they arrive at SIP Server. Deploying the Instant Messaging solution requires configuring various Genesys components. For detailed information, see the *Genesys 7.6 Instant Messaging Solution Guide,* which consolidates possible Instant Messaging solutions and configuration information for each of them.

Mapping SIP Headers and SDP Messages

SIP Server can now extract data from some incoming SIP messages and map it to either an `Extensions` or `UserData` attribute in T-Library event messages. SIP Server can map T-Library request attributes (passed in the `TRouteCall` message) to SIP parameters in the outgoing `INVITE` message. SIP Server can also map the whole SDP message body, or any particular line in it as `Extensions` or `UserData` attributes.

From SIP Messages to T-Library Messages

SIP Server processes SIP messages and can map related data to T-Library event attributes as described in this section. This information becomes further available to other Genesys Framework components. This functionality is supported for following SIP messages:

- `INVITE`
- `INFO`
- `UPDATE`
- `REFER`

INVITE Messages

SIP Server can extract data from an incoming `INVITE` message and send the data to Universal Routing Server if the call is made at a Routing Point. Data is retrieved as values from the headers and parameters of the `INVITE` message and then populated into the `Extensions` or `UserData` attributes in the `EventRouteRequest` message. You can configure which headers and parameters to extract data from by creating a section that corresponds to a SIP method name on the `Options` tab of the SIP Server `Application` object.

For example, the `INVITE` section lists the values of the headers and the header parameters that are extracted from the SIP message. The names of the options within this section contain a prefix, a dash (-), and a suffix, in the following format:

```
<prefix><-><suffix>
```

The prefix is the name of the TEvent attribute, and begins with either `extensions` or `userdata`. The prefix `extensions` instructs SIP Server to put the SIP header or parameter into the `Extensions` attribute. The prefix `userdata` instructs SIP Server to put the SIP header or parameter into the UserData attribute.

The suffix value is a number, and must be unique for all option names containing the same prefix.

The option value determines which header in the header parameter of the SIP message is processed.

The value of the header or header parameter is added as a key-value pair into the attribute, using the form <header_name>=<header_value>. Use the colon character (:) to address the parameter name of a header. The SIP method (INVITE) must be used instead of the header name if you want to populate the parameter from the SIP Request-line parameter.

Mapping occurs only if both the following conditions are true:

- The header or header parameter is contained within the incoming INVITE message.
- The header or header parameter is configured within the INVITE section.

As a result of mapping, the following key-value pair will be created within the EventRouteRequest message:

- The key in the attribute will be equal to the value of the configuration option.
- The value of the attribute will be equal to the value of the header or header parameter within the SIP INVITE message.

INFO and UPDATE Messages

SIP Server can generate an EventAttachedDataChanged message if it receives SIP INFO or UPDATE messages. The UserData attribute in the event and the corresponding call can contain information from the SIP message.

The INFO and UPDATE configuration sections must be used to configure mapping from these SIP messages to the corresponding T-Library events. The rules for this configuration are the same as the rules for configuring the INVITE section. Only mapping to the UserData attribute is supported for the INFO and UPDATE messages. Therefore, the INFO and UPDATE sections can contain userdata-<n> options, but they may not contain extensions-<n> options.

REFER Messages

The SIP REFER method provides single-step transfer functionality. SIP Server retrieves data from headers and parameters of the REFER message, and then populates it into the Extensions or UserData attributes in all events associated with a transfer transaction. This feature also enables Genesys Voice Platform (GVP) to transfer a call to Genesys Framework with attached data.

When a REFER message arrives, SIP Server analyzes the user part of the REFER TO URI to determine if the destination of the call is an internal DN or an external destination. SIP Server checks if any configuration mapping is provided in the REFER section of SIP Server Application object. If such mappings exist, SIP Server extracts the values of the appropriate SIP headers into the attribute Extensions or UserData, according to the configuration. If the destination is a Routing Point, after this processing, SIP Server generates an EventRouteRequest message containing the necessary attribute values.

You can configure which headers and parameters to extract data from by creating the REFER section on the Options tab of the SIP Server Application object. The rules for this configuration are the same as the rules for configuring the INVITE section (see page 149).

To map data from the REFER message into the Extensions and/or UserData attributes, configure userdata-<n> options and extensions-<n> option in the REFER section.

It is possible to configure SIP Server to extract data from custom headers added in the REFER message, and to process the data from the custom SIP headers.

Known Limitations

This feature is applicable only to scenarios where a call is made to a Routing Point using the REFER method. To pass attached data in other scenarios, use the mapping configuration of the INFO and UPDATE messages instead.

From T-Library Messages to SIP Messages

SIP Server can map headers or header parameters passed in the RequestRouteCall message to the outgoing INVITE message that is sent as a result of the call routing process. There are two ways to specify the values of the headers or header parameters to be mapped:

- Use the extensions-<n> SIP Server configuration option. This method maps only headers.
- Use the SIP_HEADERS and SIP_REQUEST_PARAMETERS extension of the RequestRouteCall message.

Both methods can work simultaneously; for example—you can create a mapping list using extensions-<n> options in the SIP Server configuration, and also specify the names in the SIP_HEADERS extension of the TRouteCall request.

Using the extensions-<n> Option

To use extensions-<n> to configure T-Library request attributes to the SIP mapping, you configure extensions-<n> options in a SIP Server Application object using Configuration Manager. Those options are specified in the section named after the SIP request used to route the call, which is INVITE.

Configuration Example

This example demonstrates how to map a TRouteCall request extension called InfoToSendInInvite to the outgoing INVITE message.

The following steps show how to configure SIP Server:

1. Create the INVITE section on the Options tab of the SIP Server Application object.
2. In the INVITE section, create an option named extensions-1, and set the value to InfoToSendInInvite.

Note: This configuration example is based on the assumption that this is the first extensions-<n> option in the INVITE section.

SIP Server uses this configuration when it receives a TRouteCall request from the URS with either InfoToSendInInvite or InfoToSendInRefer extensions defined.

The following log excerpt provides the details:

```
message RequestRouteCall
    AttributeThisDN             '5000'
    AttributeConnID             006e01886c3d7001
    AttributeOtherDN            '21101'
    AttributeExtensions         [371] 00 0B 00 00..
        'InfoToSendInInvite'    'INVITE from SIP Server'
    AttributeDNIS               '5000'
    AttributeRouteType          1 (RouteTypeDefault)
    AttributeReferenceID        9
```

SIP Server adds a new header InfoToSendInInvite to the outgoing INVITE message:

```
INVITE sip:21101@DestinationHost:21101 SIP/2.0
From: <sip:7102@SourceHost:7102>;tag=28B10B44
To: <sip:21101@ DestinationHost >
Call-ID: 931E620E-F3F9-4D72-A451-36B1BB259532-1
CSeq: 1 INVITE
Content-Length: 145
Content-Type: application/sdp
Contact: <sip: SourceHost:5060>
InfoToSendInInvite: INVITE from SIP Server
Max-Forwards: 70
Session-Expires: 1800;refresher=uac
Min-SE: 90
Supported: timer
```

Using SIP_HEADERS and SIP_REQUEST_PARAMETERS

This method does not require any changes to the configuration of the SIP Server `Application` object. In this case, all mapping information is provided in the `TRouteCall` request that is created by the URS routing strategy and sent to SIP Server.

The `TRouteCall` request should contain two specialized extensions to trigger T-Library-to-SIP mapping in SIP Server:

- `SIP_HEADERS`—Contains a list of extension names of `TRouteCall` to be mapped to the outgoing SIP message as headers.
- `SIP_REQUEST_PARAMETERS`—Contains a list of extension names to be mapped to the outgoing SIP message as Request URI parameters.

The values of headers and header parameters must be also specified in the `TRouteCall` request.

Example

This example demonstrates how mapping works in SIP Server when it receives the following `TRouteCall` request:

```
message RequestRouteCall
    AttributeThisDN                 '5000'
    AttributeConnID                 006e01886c3d7001
    AttributeOtherDN                '21101'
    AttributeExtensions             [371] 00 0B 00 00..
        'SIP_HEADERS'               'hdr-host1,hdr-host2'
        'SIP_REQUEST_PARAMETERS'    'prm-host1,prm-host2'
        'hdr-host1'                 'host1'
        'hdr-host2'                 'host2'
        'prm-host1'                 'local1'
        'prm-host2'                 'local2'
    AttributeDNIS                   '5000'
    AttributeRouteType              1 (RouteTypeDefault)
    AttributeReferenceID            9
```

This message contains both `SIP_HEADERS` and `SIP_REQUEST_PARAMETERS` extensions, which means that both new headers and new Request URI parameters should be added to the outgoing SIP message:

```
INVITE sip:21101@ DestinationHost:21101;prm-host1=local1;
prm-host2=local2 SIP/2.0
From: <sip:7102@ SourceHost:7102>;tag=28B10B44
To: <sip:21101@ DestinationHost >
Call-ID: 931E620E-F3F9
CSeq: 1 INVITE
Content-Length: 145
```

```
Content-Type: application/sdp
Contact: <sip: SourceHost >
hdr-host1: host1
hdr-host2: host2
Max-Forwards: 70
Session-Expires: 1800;refresher=uac
Min-SE: 90
Supported: timer
```

SDP Message Mapping

SIP Server can map the whole SDP message body, or any particular line in it as `AttributeExtensions` or `AttributeUserData`. Configuring this type of mapping is similar to configuring mapping of SIP Server headers or parameters. The option names in the `INVITE` section must use the same rules described for SIP Server message mapping; however, the option name must be `SDP`, and it must be followed by a colon and a letter that indicates the type of mapped SDP message. For example, the `userdata-1=SDP:m` option is mapped using the following parameters from `AttributeUserData`:

```
m=audio 18234 RTP/AVP 8 101
```

Note: If the `AttributeUserData` parameter in the `EventRouteRequest` message contains a pair such as:

```
SDP:m audio 18234 RTP/AVP 8 101
```

You must use the `SDP` value to configure the mapping of the whole SDP message.

Music and Announcements

SIP Server is able to control the playing of announcements by using Stream Manager. Stream Manager plays files using a codec negotiated with the SIP Server switch. The list of possible codecs is configured in SIP Server with the `audio-codecs` option (page 201). For information about how to configure Stream Manager refer to “Configuring Stream Manager” on page 94.

Announcement Treatments on Routing Points

The two announcement treatments, PlayAnnouncement and PlayAnnouncementAndDigits, include the parameters listed in Table 15.

Table 15: Announcement Treatment Parameters

Parameter	Description
LANGUAGE	Ignored.
MSGID	Ignored.
MSGTXT	Ignored.
PROMPT	Contains up to 10 subprompts. Each contains a music file, and these are played in order.
INTERRUPTABLE	When this check box is selected, the caller can interrupt the announcement with a DTMF keystroke.
ID	Contains an integer, that refers to the Stream Manager announcement/<integer> file. For example, the value 1 refers to the file announcement/1_alaw.wav, if the G.711 A-law codec is used.
DIGITS	Ignored.
USER_ID	Supported by Stream Manager using the users/<customer id>_<ann.id> file.
USER_ANN_ID	Supported by Stream Manager using the users/<customer id>_<ann.id> file.
TEXT	Ignored.

Music-on-hold is not available for these treatment types. Stream Manager is the only source for the announcements.

If the treatment is terminated early because of a problem with Stream Manager or SIP Server, SIP Server sets the Extension data fields ERR_CODE and ERR_TEXT. To determine whether these fields and their values exist, from a routing strategy, use the function ExtensionData. Place this function on a normal completion branch (not the error branch) after the treatment.

Refer to the *Universal Routing 7.6 Reference Manual* for more information on the use and configuration of strategies.

Note: Leave the Wait For Treatment End check box selected, to allow the treatment to play until completion.

Music Treatments on Routing Points

Table 16 describes the parameters for music treatments:

Table 16: Music Treatment Parameters

Parameter	Description
MUSIC_DN	Specifies the music source that Stream Manager plays. The format is: `<directory>/<music file name>` Where `<directory>` is a sub-directory of the Stream Manager root directory, and `<music file name>` refers to the name of the file—without the codec extension. For example, `music/in_queue` refers to the file `music/in_queue_alaw.wav` if the G.711 A-law codec is used. To specify the number of repetitions, the parameter `repeat=<N>` must be used, where `<N>` is any positive integer. If no repetition is specified, the music file loops forever. The valid formats are: • `<directory>/<music file name>`—The specified file loops endlessly. • `<directory>/<music file name>;repeat=<N>`—The specified file is repeated `<N>` times. The `default-music` option is used if the value of the `MUSIC_DN` parameter is not specified.
DURATION	Specifies the duration of the music (in seconds). **Note:** This parameter is ignored if `MUSIC_DN` is blank. This treatment ends before music is played. To continue playing music after the treatment terminates, consider creating one of the following strategies: • Execute the treatment inside a route-selection treatment block. In this case, the treatment continues until a route target is selected. • Follow the treatment with the `SuspendForTreatmentEnd` function. In this case, the treatment plays music until terminated after the delay specified in option `DURATION`. • Follow the treatment with the `delay` function. In this case, the treatment plays music for the period specified in option `delay`. If `DURATION` is less than `delay`, silence is played for the time difference.

Refer to the *Universal Routing 7.6 Reference Manual* for more information on use and configuration of strategies.

Other Treatments on Routing Points

These treatments in Table 17 continuously loop a pre-defined audio file to a call. You must configure Stream Manager to use these treatments. Music-on-hold cannot be the source of the audio file. The treatment types are as follows:

Table 17: Other Treatments on Routing Points

Treatment Type	Description
Busy	Plays a busy tone. To define the busy tone audio file, configure the SIP Server busy-tone option (page 202).
Fast Busy	Plays a fast busy tone. To define the fast busy tone audio file, configure the SIP Server fast-busy-tone option (page 209).
Silence	Plays no sound. To define the silence audio file, configure the SIP Server silence-tone option (page 220).
Ringback	Plays ringback tone. To define the ringback audio file, configure the SIP Server ring-tone option (page 219).
CollectDigits	Collects customer-entered digits.
RecordUser-Announcement	Records a user's announcement and saves into a users folder.

Refer to the *Universal Routing 7.6 Reference Manual* for more information about the use and configuration of strategies.

No-Answer Supervision

This section describes SIP Server's No-Answer Supervision feature and its configuration.

Business and Private Calls

No-Answer Supervision can be applied to business and private calls.

Business Calls

SIP Server automatically categorizes as a *business call* any call distributed to an agent either from a Queue or from a Routing Point. Use the following configuration options to define what additional calls to or from an agent are classified as business calls:

- inbound-bsns-calls (page 210)

- `outbound-bsns-calls` (page 216)
- `inherit-bsns-type` (page 210)
- `internal-bsns-calls` (page 211)
- `unknown-bsns-calls` (page 225)

Private Calls

SIP Server categorizes as a *private call* any call that does not fall into the business or work-related categories. SIP Server does not apply any automatic business-call handling after a private call. If an agent receives a direct private call while in wrap-up or legal-guard time, the emulated wrap-up or legal-guard timer is not interrupted.

You can apply No-Answer Supervision to private calls, using the configuration option `nas-private` (page 216).

Agent No-Answer Supervision

This feature provides the following functionality:

- If an agent does not answer a call within a specified timeout, SIP Server can divert the call to a sequence of overflow destinations. Alternatively, you can configure SIP Server to return calls automatically to the last distribution device.
- If an agent fails to answer a call within a specified timeout, you can configure SIP Server to either log out the agent or set the agent to `NotReady` to prevent further calls from arriving.

Configuration Options

SIP Server provides the following configuration options for defining the behavior of the Agent No-Answer Supervision feature:

- `agent-no-answer-action` (page 199)
- `agent-no-answer-overflow` (page 199)
- `agent-no-answer-timeout` (page 200)
- `nas-private` (page 216)

Extension No-Answer Supervision

The No-Answer Supervision feature includes devices of type `Extension`. If a call is not answered on an extension within a specified timeout, SIP Server can divert the call to a sequence of overflow destinations. Alternatively, you can configure SIP Server to return calls automatically to the last distribution device.

Configuration Options

SIP Server provides the following configuration options for defining the behavior of No-Answer Supervision with devices of type Extension:

- extn-no-answer-overflow (page 208)
- extn-no-answer-timeout (page 208)

Position No-Answer Supervision

The No-Answer Supervision feature includes devices of type ACD Position. If a call is not answered on a position within a specified timeout, SIP Server can divert the call to a sequence of overflow destinations. Alternatively, you can configure SIP Server to return calls automatically to the last distribution device.

Configuration Options

SIP Server provides two configuration options for defining the behavior of No-Answer Supervision with devices of type ACD Position:

- posn-no-answer-overflow (page 217)
- posn-no-answer-timeout (page 218)

Device-Specific Overrides

SIP Server provides three configuration options with which you can configure device-specific overrides for individual devices. You set the values for these options in the TServer section on the Annex tab of the individual device. The options are:

- no-answer-action (page 234)
- no-answer-overflow (page 234)
- no-answer-timeout (page 235)

Extension Attributes for Overrides for Individual Calls

For all of the No-Answer Supervision options, you can specify the corresponding Extension attribute in the TRouteCall request, to override the configured value for individual calls. This method allows the no-answer behavior to be determined in a routing strategy. The three extensions are:

- NO_ANSWER_TIMEOUT
- NO_ANSWER_OVERFLOW
- NO_ANSWER_ACTION

See “Using the Extensions Attribute” on page 186 for more information.

Feature Limitations

No-Answer Supervision functionality will not be activated if the divert-on-ringing configuration option is set to false.

Personal Greeting

Personal greeting functionality enables Stream Manager to play a media file to a customer and an agent when the agent answers the call. It is possible to play the same file or different files to the customer and agent.

Feature Configuration

To configure personal greetings, follow the provided procedure using Configuration Manager.

Procedure:
Configuring a personal greeting

Start of procedure

1. Under a configured Switch object, select the Agent Logins folder. From the File menu, select New > Agent Login to create a new Agent Login object.
2. In the New Agent Login Properties dialog box, on the General tab, specify the Agent Login name in the Code text box.
3. Click the Annex tab.
4. Click or create a section named TServer. In the TServer section, create options as specified in Table 18.

Table 18: Configuring a Personal Greeting

Option Name	Option Value	Description
agent-greeting	A file name	Specifies the name of the media file that will be used as a greeting for the agent. See the option description on page 228.
customer-greeting	A file name	Specifies the name of the media file that will be used as a greeting for the customer. The customer greeting plays continuously until the agent greeting finishes playing. See the option description on page 230.

5. When you are finished, click `Apply`.

End of procedure

Additional Information

You can also attach these values from a routing strategy. The `agent-greeting` and `customer-greeting` keys enable personal greetings. These keys are processed by SIP Server using the `AttributeExtension` parameters of the `TRouteCall` request.

Notes: The keys that are contained in attribute `Extension` have a higher priority than the options specified in the `Agent Login` object.

The customer greeting plays continuously until the agent greeting finishes playing.

Presence Subscription

Presence is an indicator of an agent's status regarding possible communication. An agent's client application provides presence information (or state) to SIP Server, which distributes this information to it's clients. SIP Server supports "accepting" presence subscriptions and "subscribing" presence subscriptions. These subscriptions are used with DNs of type `Extension` and DNs of type `Routing Point`.

When subscribing to DNs of type `Extension`, SIP Server notifies subscribers that the status of the endpoint is `open` when that endpoint registers with SIP Server. SIP Server will notify any subscribers that the status of the endpoint is closed when the SIP endpoint is not registered or when the registration has expired.

When subscribing to DNs of type `Routing Point`, SIP Server always notifies subscribers that the status of the endpoint is `open`.

Feature Configuration

Procedure:
Enabling presence subscription

Purpose: To enable presence subscription for a particular DN. This procedure can also be used when subscribing to the presence of any endpoints that are located behind any presence agent or a softswitch supporting presence notifications.

Start of procedure

1. Create a DN of type `Trunk` in Configuration Manager. Parameters for all presence subscriptions from the SIP Server to a particular softswitch are configured in this `Trunk` DN.
2. Configure these options in the `TServer` section on the `Annex` tab of the `Trunk` DN:
   - `contact` (page 230)
   - `subscribe-presence-domain` (page 245)
   - `subscribe-presence-from` (page 245)
   - `subscribe-presence-expire` (page 246)
3. Create a DN of type `Extension` in Configuration Manager.
4. Configure these options in the `TServer` section on the `Annex` tab of the `Extension` DN:
   - `contact` (page 230)
   - `request-uri` (page 241)
   - `subscribe-presence` (page 246)
   - `enable-agentlogin-presence` (page 232)
5. Create an `Agent Login` object in Configuration Manager for each DN that will have subscription enabled. The `Agent Login` name must be equal to the `DN` object name. Each `Agent Login` object must be associated with an agent.

End of procedure

Note: Any internal calls that are made with this softswitch will not be monitored by SIP Server, and the agent state will be changed by SIP Server to `Not Ready`.

Updating Agent State

SIP Server changes the agent state in response to any notifications about presence state changes by using the `PUBLISH` request method. SIP Server accepts the `PUBLISH` request and provides automatic agent state updates based on any presence updates received within the `PUBLISH` request. SIP Server distributes notifications about presence updates to all subscribers based on the presence update received within the `PUBLISH` request.

SIP Server accepts `PUBLISH` requests when they are received for a DN in an internal domain. SIP Server processes the presence update from the `PUBLISH` request and distributes presence update notification to all subscribers for this DN.

The `PUBLISH` request functionality is enabled at the DN level in Configuration Manager by specifying the `subscribe-presence` option (page 246). The value must be set to `publish` to indicate that presence change notifications are issued from the `PUBLISH` request.

Note: This functionality has been verified with the Eyebeam SIP endpoint when it is configured to work in `Presence Agent` mode. This mode enables `PUBLISH` request processing.

SIP Server updates the agent state when the agent login name matches the DN name. Agent updates are processed as follows:

- When SIP Server receives a presence notification with an `open` status, it performs the following steps:
  - Confirms if the agent is logged in. If the agent is not logged in, SIP Server sends an `EventAgentLogin` message.
  - Confirms if any activity is indicated in the presence notification.
    - If there is no activity, and if the agent is in a `NotReady` state, SIP Server sends an `EventAgentReady` message.
    - If there is activity, and if the agent is in a `Ready` state, SIP Server sends an `EventAgentNotReady` message and attaches the activity from the presence notification as the `ReasonCode` attribute.
- When SIP Server receives a presence notification with a `closed` status, it confirms that the agent is logged in. If the agent is logged in, SIP Server then sends an `EventAgentLogout` message.
- All notifications about the changes of an agent state are ignored when the agent is in the `NotReady` (`AfterCallWork`) state. The requested agent state is applied when the ACW time is over. For example, if an agent completes the call, SIP Server transfers the agent into the `ACW` state, and the `PUBLISH` request with an `open` status comes from the agent's SIP phone, then SIP Server does not change the agent state immediately. It waits for the ACW time to expire, and then places this agent into the `Ready` state. If in the same scenario SIP Server receives the `PUBLISH` request with a `busy` status from the agent's SIP phone, SIP Server will not change the agent state until the ACW timer is over, meaning that the agent remains in the `NotReady` state.

Notes: When configuring presence subscription for Microsoft's Live Communication Server (LCS), you must also configure the `Trunk` DN as specified in "Remote Server Registration" on page 166.

When Microsoft's Office Communicator is integrated with Microsoft Outlook and the presence state is set to `In a Meeting` or `Vacation`, Microsoft's Live Communications Server sends a presence notification with the `Busy` presence state to SIP Server. However, SIP Server is unable to provide `In a Meeting` or `Vacation` presence states to any subscriber. Instead, a notification with the `Busy` presence state is generated by SIP Server.

Preview Interactions

Preview interactions allow agents to preview desktop interactions before receiving a call. SIP Server sends Preview Interaction messages to the desktop applications using the `EventPrivateInfo` message. The desktop application sends preview interaction messages using the `TPrivateService` request.

SIP Server sends a `previewInteractionRequest` message to the desktop application when it receives a `RouteCall` request to a DN that is configured with the `preview-interaction` option (page 238) set to `true`.

The desktop application responds with a `previewInteractionResponse` message to SIP Server. The `previewInteractionResponse` message provides SIP Server with information regarding the agent's ability to process the incoming interaction. The `status` field contains an `accepted` value or a `rejected` value that specifies if the agent will accept the interaction.

SIP Server sends a `previewInteractionAcknowledge` message to the desktop application after it receives the `previewInteractionResponse` message from it. This message informs the desktop application that the `previewInteractionResponse` message was successfully processed by SIP Server.

The `previewInteractionCancel` message is sent by SIP Server to an application in the following scenarios if there was an unsuccessful completion of a preview interaction:

- The preview timeout expired. SIP Server sends the `previewInteractionCancel` message with the `status` field set to `expired` to an application when the `previewInteractionRequest` message was issued but SIP Server did not receive a `previewInteractionResponse` message within the specified timeout value for the `preview-expired` option (page 218).
- The call was abandoned. SIP Server sends the `previewInteractionCancel` message to an application with the `status` field set to `canceled`.

Providing Call Participant Info

SIP Server can distribute information about all call participants—except the trunks allocated for communication between SIP Servers, distribution devices (such as Routing Points or ACD), and supervisors' DNs—to logged-in agents by using the SIP `NOTIFY` method and `EventUserEvent` messages.

The information about the call participants is reported in the `Extensions` attribute of the relevant event using the following key-value pairs:

- `LCTPartiesLength`—An integer that specifies how many parties are involved in a single call.

- LCTParty<n>—An integer that represents a party of the call, where n is an integer value starting from 0.

Feature Configuration

Table 19 provides an overview of the steps required to configure call info for agents.

Table 19: Task Flow—Configuring Call Info for Agents

Objective	Related Procedures and Actions
1. Configure the Trunk DNs.	Set the sip-server-inter-trunk option to true for DNs of type Trunk that are allocated for direct signaling between SIP Servers. The NOTIFY method will be sent only to sessions that are established through such trunks. For more information, see “Trunk Optimization for Multi-Site Transfers” on page 166.
2. Configure the SIP Server Application object.	Set the sip-enable-call-info option to true.

Providing a Caller ID

SIP Server supports providing caller ID information that is displayed on a destination party’s phone, and replacing the caller ID with another number if necessary. This feature is supported using either of the following methods:

- The Extensions attribute with the CPNDigits key in the following messages: TMakeCall, TMakePredicitiveCall, TInitiateConference, and TInitiateTransfer.

  If the CPNDigits key is set in the Extensions attribute, the value of this key overrides the username provided in the URI in the From header of the INVITE message. This Extension is not applicable when performing a TMakeCall request using the REFER method.
- The cpn option (page 231) at the Trunk DN level. In this case, the caller ID information will be replaced by the SIP URI setting in this option for all outgoing calls through this Trunk DN.

Remote Server Registration

SIP Server supports registering with a remote server under a specified account. The remote server registration is enabled on a per-Trunk DN basis. SIP Server registers Trunk DNs at a remote server when the force-register option (page 232) is configured.

SIP Server also uses the values of the following options when registering with a remote server:

- contact, when determining where to send the REGISTER request.
- password, when the REGISTER request is challenged.

See "Configuring endpoints" on page 81 for more information about this options.

Remote Talk

The Remote Talk feature enables the answering of an incoming call remotely by a T-Library client, by sending the TAnswerCall request to SIP Server. For this feature to work, the sip-cti-control option must be set to talk (see page 244).

The SIP endpoint must support the BroadSoft Application Server interface in order to use the Remote Talk feature for remote call control.

Trunk Capacity Support

SIP Server enables control of the number of calls to be handled by a specific Voice over IP device represented in the SIP Server configuration as Trunk. The following DN-level configuration options support this feature: capacity and capacity-group. See the option descriptions and configuration examples on page 229.

Trunk Optimization for Multi-Site Transfers

SIP Server supports trunk optimization for multi-site transfers. When the trunk optimization functionality is in use, the OtherDN attribute contains correct information and is reported properly in EventPartyChanged messages in the following scenarios:

Scenario 1

Figures 22 and 23 show the state of the call before and after the multi-site transfer.

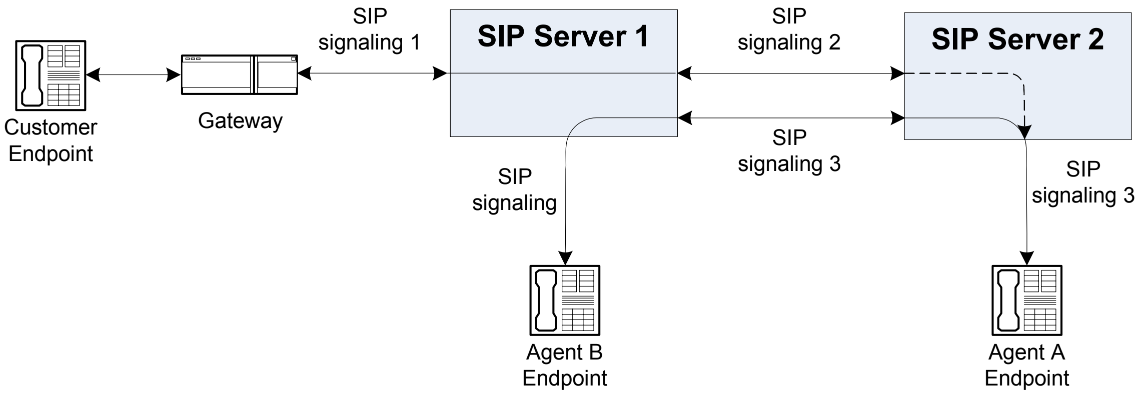


Figure 22: Call Before REFER with Replaces Transfer

1. An inbound call is routed to Agent A at the SIP Server 2 site.
2. Agent A initiates a two-step transfer to Agent B at the SIP Server 1 site.

In this scenario, SIP Server uses a SIP `REFER` request with the `Replaces` header to report call data for Agent B.

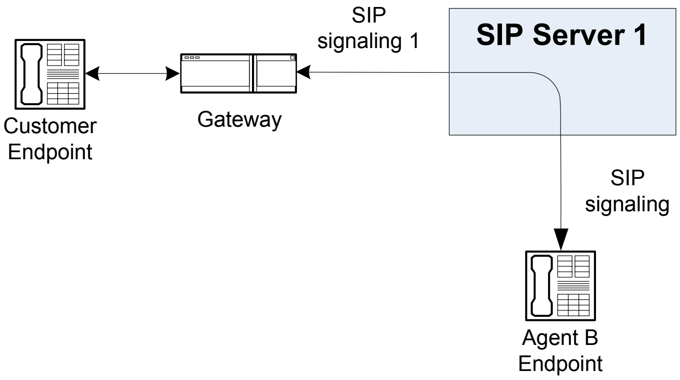


Figure 23: Call After REFER with Replaces Transfer

After the transfer is completed, both the transferring agent (Agent 1) and secondary SIP Server (SIP Server 2) are released from the call.

Scenario 2

Figures 24 and 25 show the state of the call before and after the multi-site transfer.

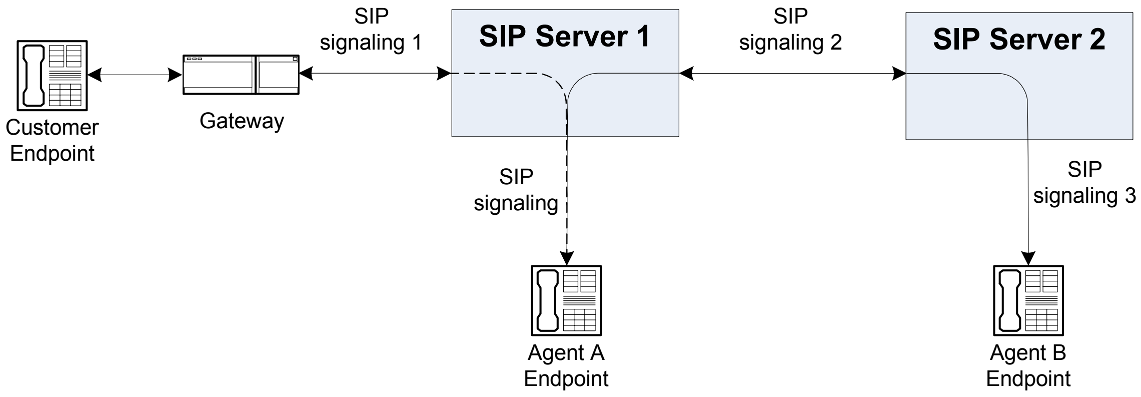


Figure 24: Call Before INVITE with Replaces Transfer

1. An inbound call is routed to Agent A at the SIP Server 1 site.
2. Agent A initiates a two-step transfer to Agent B at the SIP Server 2 site.

In this scenario, SIP Server uses a SIP `INVITE` request with the `Replaces` header to report call data for Agent B.

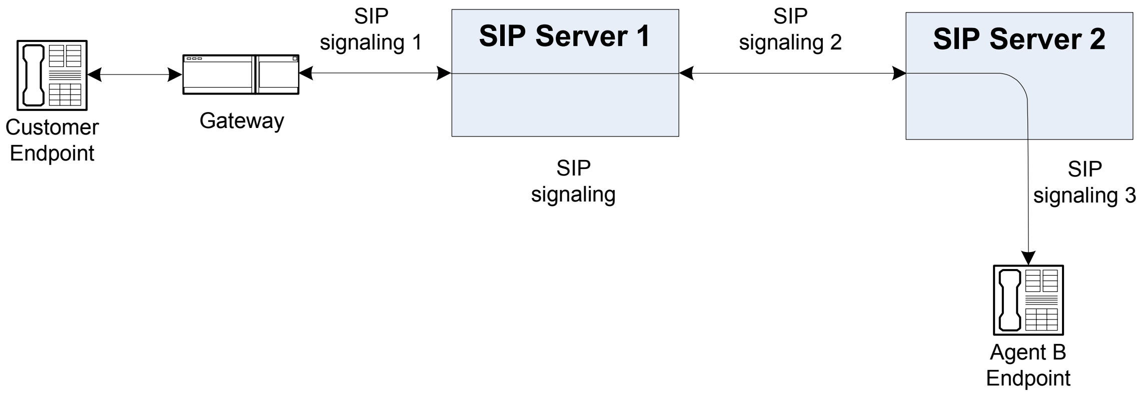


Figure 25: Call After INVITE with Replaces Transfer

In this case, the consultation call between the agents are merged on SIP Server 1, with user data propagated to the destination SIP Server (SIP Server 2). After the transfer is completed, SIP Server 1 remains in the signaling path—only the transferring agent (Agent A) is released from the call.

Scenario 3

Figures 26 and 27 show the state of the call before and after the multi-site transfer.

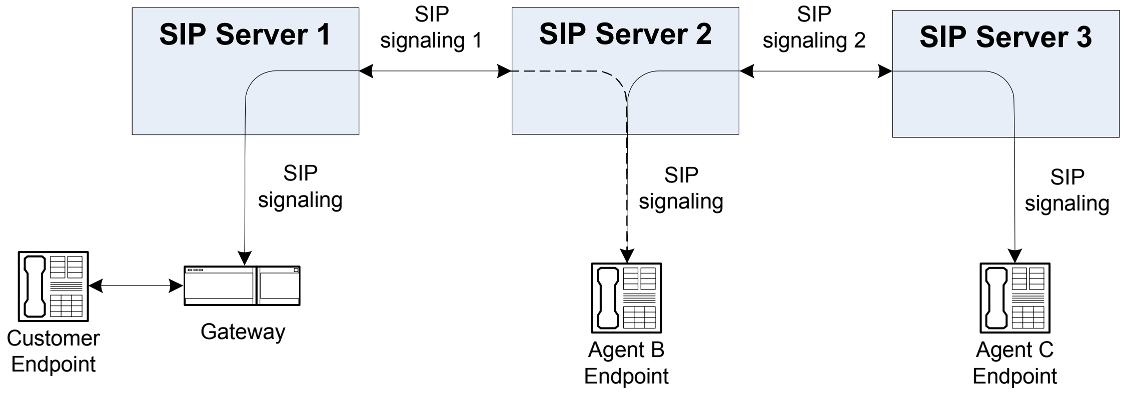


Figure 26: Call Before INVITE with Replaces Transfer

1. From a SIP Server 1 site, a call arrives to Agent B at the SIP Server 2 site.
2. Agent A initiates a two-step transfer to Agent C at the SIP Server 3 site.

In this scenario, SIP Server uses a SIP `INVITE` request with the `Replaces` header to report call data for Agent C.

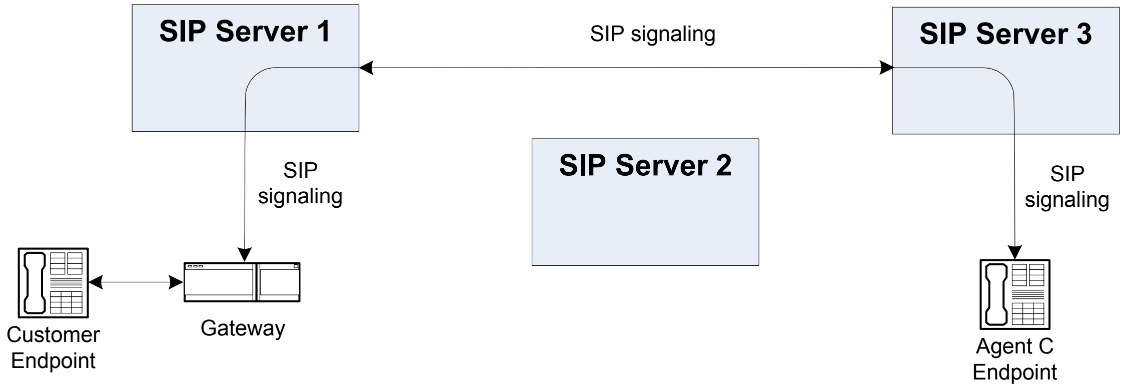


Figure 27: Call After INVITE with Replaces Transfer

After the transfer is completed, SIP Server 2 is removed from the signaling path. An `EventPartyChanged` message is generated for Agent C on SIP Server 3, based on information received in the `INVITE` request with the `Replaces` header.

Feature Configuration

Table 20 provides an overview of the main steps required to configure trunk optimization.

Table 20: Task Flow—Configuring Trunk Optimization

Objective	Related Procedures and Actions
1. Create Trunk DNs.	In each SIP Server configuration (origination and destination), in the corresponding SIP Switch, configure a DN of type Trunk. These Trunk DNs will be used for direct signaling between SIP Servers. For each Trunk DN, in the TServer section of the Annex tab, configure the following options: • refer-enabled—Set this option to true (see page 240). • oosp-transfer-enabled—Set this option to true (see page 236). • sip-server-inter-trunk—Set this option to true (see page 244).
2. Configure the SIP Server Application object.	In multi-site routing, to avoid reporting an access resource as AttributeOtherDN in related events, in the extrouter section of the SIP Server Application object, set the cast-type option to an ISCC direct transaction type (such as direct-uui).

Feature Limitation

This functionality requires direct signaling (no media gateways or session border controllers) between any two SIP Server instances, with no alteration of the SIP attributes (CALL-ID, to header, from header) as these are used for unique call context matching.

Video Support

SIP Server supports the following scenarios related to Video Call functionality:

- Push Video
- Video Call on Hold
- Video Call Transfer
- Video Call Treatment

- Outbound Video Call

Push Video

Push Video functionality enables a person to play a video file to another call participant during a call. SIP Server can support video streams using Stream Manager and T-Library functions.

Start Video

To start playing a video file, SIP Server uses the `TSingleStepConference` function to push video from an agent to a customer. This function must contain the following attributes:

- `OtherDN`—Represents a video source. It is always defined as the `gcti::video` string.
- `Extensions`—Must contain the following key-value pairs:
  - `VideoFile`—A string that contains the name of the video file that will be played for the customer. If this key-value pair is not specified, the default video file will be played. The default video file is configured in the SIP Server `Application` object, using the `default-video-file` configuration option (page 204).
  - `AgentVideo`—A string that identifies the origin of the video stream played to the agent. The values are as follows:
    - `from-third-party`—The agent receives video from a third party—that is, the party that participated in the call before the operation started.
    - `to-third-party`—The agent receives the same video stream as played to the third party—that is, video from the file specified by the `VideoFile` parameter. (The `from-video-file` value can be used as an alias.)

    In either case, both the customer and agent hear each other and the audio that comes with the video file. The customer, the agent, and the audio source from the video file are three participants in the audio conference. When the pushed video ends, the customer and the agent continue a regular two-party conversation.

    If the `AgentVideo` key is not specified, or if it is empty, the `to-third-party` value will be used.

Stop Video

There are several ways to stop playing a video file:

- By deleting a party from a conference
- By releasing the `gcti::video` device

- When the video file ends

Deleting a Conference Call

SIP Server uses the TDeleteFromConference function to stop a video stream from a conference call. In this scenario, the OtherDN attribute is always defined as the gcti::video string.

Releasing the Device

SIP Server uses the TReleaseCall function to stop a video stream by releasing the gcti::video device. In this scenario, the ThisDN attribute is always defined as the gcti::video string.

When the File Ends

Stream Manager will end the SIP dialog for the gcti::video device when the video file ends, but it will not end the other SIP dialogs that belong to the conference.

Other Supported Scenarios

Video Call on Hold

The video call can be put on hold by using the THoldCall function. Stream Manager analyzes the endpoint capabilities submitted inside the endpoint's SDP message, and when supported, plays a video file.

Video Call Transfer

SIP Server supports a video call transfer by providing a regular offer/answer SDP message exchange between an endpoint during 1pcc operation.

Video Call Treatment

SIP Server supports a video call treatment in which a video file can be played to a customer when their call is on a Routing Point. The treatment prompts can be defined in a URS strategy that points to video files. For the video calls, Stream Manager plays both video and audio when a video prompt is specified.

Outbound Video Call

An agent can initiate an outbound video call using the TMakeCall function. In the SIP Server configuration, the refer-enabled option must be set to true, or the make-call-rfc3725-flow option must be set to 1. The INVITE message to an external destination will contain SDP information with the agent's endpoint video capabilities.

When the agent initiates an outbound video call, and a recipient accepts it, the video file starts playing. If the recipient's endpoint does not have video capabilities or refuses the video connection, only an audio connection—without the video—is established.

Feature Configuration

Table 21 provides an overview of the main steps required to configure video support.

Table 21: Task Flow—Configuring Video Support

Objective	Related Procedures and Actions
1. Install a PC video camera.	Follow the instructions in the video camera documentation.
2. Configure a SIP endpoint to support video functionality.	Follow the instructions specific to the SIP endpoint you are using. If using the Genesys SIP Endpoint, launch the Audio and Video Tuning Wizard. Complete the wizard steps, and select the installed video camera on the corresponding wizard page.
3. Configure a SIP Server `Application` object.	In the SIP Server `Application` object, in the `TServer` section on the `Options` tab, specify the `default-video-file` configuration option.This option contains the name of the video file that is played to the caller if a single-step conference to the `gcti::video` device does not contain a `VideoFile` key in the `Extensions` attribute.
4. Configure a `gcti::video` device.	For Push Video, complete the following procedure: • Configuring a gcti::video device, page 174
5. Configure a video service.	For Push Video, complete the following procedure: • Configuring a video service, page 174

Follow the provided procedures using Configuration Manager.

Procedure: Configuring a gcti::video device

Start of procedure

1. Under a configured Switch object, select the DNs folder. From the main menu, select File > New > DN to create a new DN object.
2. In the New DN Properties dialog box, click the General tab, and then specify the following properties:
   a. Number: Enter gcti::video.
   b. Type: Select Trunk from the drop-down box.
3. When you are finished, click Apply.

End of procedure

Next Steps

- Configuring a video service

Procedure: Configuring a video service

Start of procedure

1. Under a configured Switch object, select the DNs folder. From the main menu, select File > New > DN to create a new DN object.
2. In the New DN Properties dialog box, click the General tab, and then specify the following properties:
   a. Number: Enter the DN name. This name is currently not used for any messaging, but it must still be unique.
   b. Type: Select Voice over IP Service from the drop-down box.
3. Click the Annex tab, and create the TServer section. Under the TServer section, set the options as specified in Table 22.

Table 22: Configuring a Video Service

Option Name	Option Values	Description
contact	SIP URI	Specifies the value using the Stream Manager application settings in the following format: IP address::SIP port
request-uri	SIP URI	Specifies the value to be used as a template for the source of the video stream and as the value of the Request-URI parameter in the INVITE message: annc@<stream_manager_hostport>;play=<file>);repeat=<number>)
service-type	video	Set this option to video.

4. When you are finished, click Apply.

End of procedure

Feature Limitations

The Video Conference functionality is not supported.

Genesys Voice Platform Integration

For detailed information about SIP Server integration with the Genesys Voice Platform (GVP), see the following documents:

- The *Genesys 7.5 GVP–SIP Server Integration Guide*—This guide provides an overview of the GVP–SIP Server integration in its various modes—In-Front, Behind, and Stand-Alone—as well as the relevant procedures for completing the integration. This document applies to the 7.5 release of SIP Server, and the 7.5 and 7.6 releases of GVP.
- The *Genesys 8.0 Voice Platform Solution 8.0 Integration Guide*—This guide provides an overview of the Voice Platform Solution (VPS), with the aim of integrating the various components that make up the solution. This document applies to the 7.6 release of SIP Server and the 8.0 release of GVP.

Chapter

7 T-Library Functionality Support

This chapter describes the T-Library functionality that SIP Server supports. It contains the following sections:

Using T-Library Functions

Table 23 presents the T-Library functionality supported in SIP Server. The table entries use these notations:

N—Not supported

Y—Supported

E—Event only is supported

I—Supported, but reserved for Genesys Engineering

In Table 23, when a set of events is sent in response to a single request, the events are listed in an arbitrary order. An asterisk (*) indicates the event that contains the same `Reference ID` as the request. For more information, refer to the *Genesys 7 Events and Models Reference Manual* and the *Voice Platform SDK 7.6 .NET (or Java) API Reference* for complete information on the T-Server events, call models, and requests.

Table 23 reflects only the switch functionality that Genesys software supports and might not include the complete set of events that the switch offers.

Certain requests listed in Table 23 are reserved for Genesys Engineering and are listed here merely for completeness of information.

Table 23: Supported Functionality

Feature Request	Request Subtype	Corresponding Event(s)	Supported
General Requests			
TOpenServer		EventServerConnected	Y
TOpenServerEx		EventServerConnected	Y
TCloseServer		EventServerDisconnected	Y
TSetInputMask		EventACK	Y
TDispatch		Not Applicable	Y
TScanServer		Not Applicable	Y
TScanServerEx		Not Applicable	Y
Registration Requests			
TRegisterAddress[a]		EventRegistered	Y
TUnregisterAddress[a]		EventUnregistered	Y
Call-Handling Requests			
TMakeCall[b]	Regular	EventDialing	Y
	DirectAgent		N
	SupervisorAssist		N
	Priority		N
	DirectPriority		N
TAnswerCall		EventEstablished	Y[c]
TReleaseCall		EventReleased	Y
TClearCall		EventReleased	N
THoldCall		EventHeld	Y
TRetrieveCall		EventRetrieved	Y
TRedirectCall		EventReleased	Y
TMakePredictiveCall[d]		EventDialing*, EventQueued	Y

Table 23: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
Transfer/Conference Requests			
TInitiateTransfer[b]		EventHeld, EventDialing*	Y
TCompleteTransfer		EventReleased*, EventPartyChanged	Y
TInitiateConference[b]		EventHeld, EventDialing*	Y
TCompleteConference		EventReleased*, EventRetrieved, EventPartyChanged, EventPartyAdded	Y
TDeleteFromConference		EventPartyDeleted*, EventReleased	Y
TReconnectCall		EventReleased, EventRetrieved*	Y
TAlternateCall		EventHeld*, EventRetrieved	Y
TMergeCalls	ForTransfer	EventHeld, EventReleased*, EventRetrieved EventPartyChanged	N
	ForConference	EventHeld, EventReleased*, EventRetrieved, EventPartyChanged, EventPartyAdded	N
TMuteTransfer[b]		EventHeld, EventDialing*, EventReleased, EventPartyChanged	N
TSingleStepTransfer[b]		EventReleased*, EventPartyChanged	Y
TSingleStepConference		EventRinging*, EventEstablished	Y

Table 23: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
Call-Routing Requests			
TRouteCall[b]	Unknown	EventRouteUsed	Y
	Default		Y
	Label		N
	OverwriteDNIS		N
	DDD		N
	IDDD		N
	Direct		N
	Reject		Y
	Announcement		N
	PostFeature		N
	DirectAgent		N
	Priority		N
	DirectPriority		N
	AgentID		N
	CallDisconnect		N
Call-Treatment Requests			
TApplyTreatment	Unknown	(EventTreatmentApplied + EventTreatmentEnd)/ EventTreatmentNotApplied	N
	IVR		N
	Music		Y
	RingBack		Y
	Silence		Y
	Busy		Y
	CollectDigits		Y
	PlayAnnouncement		Y

Table 23: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
	PlayAnnouncementAnd-Digits		Y
	PlayApplication		Y
	VerifyDigits		N
	RecordUserAnnouncement		Y
	DeleteUserAnnouncement		N
	CancelCall		N
	SetDefaultRoute		N
	TextToSpeech		N
	TextToSpeechAndDigits		N
	FastBusy		Y
TGiveMusicTreatment		EventTreatmentApplied	N
TGiveRingBackTreatment		EventTreatmentApplied	N
TGiveSilenceTreatment		EventTreatmentApplied	N
DTMF (Dual-Tone MultiFrequency) Requests			
TCollectDigits		EventDigitsCollected	N
TSendDTMF		EventDTMFSent	Y
Voice-Mail Requests			
TOpenVoiceFile		EventVoiceFileOpened	N
TCloseVoiceFile		EventVoiceFileClosed	N
TLoginMailBox		EventMailBoxLogin	N
TLogoutMailBox		EventMailBoxLogout	N
TPlayVoice		EventVoiceFileEndPlay	N
Agent & DN Feature Requests			
TAgentLogin		EventAgentLogin	Y
TAgentLogout		EventAgentLogout	Y

Table 23: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
TAgentSetIdleReason		EventAgentIdleReasonSet	N
TAgentSetReady		EventAgentReady	Y
TAgentSetNotReady		EventAgentNotReady	Y
TMonitorNextCall	OneCall	EventMonitoringNextCall	Y
	AllCalls		Y
TCancelMonitoring		EventMonitoringCanceled	Y
TCallSetForward	None	EventForwardSet	N
	Unconditional		N
	OnBusy		N
	OnNoAnswer		N
	OnBusyAndNoAnswer		N
	SendAllCalls		N
TCallCancelForward		EventForwardCancel	N
TSetMuteOff		EventMuteOff	Y
TSetMuteOn		EventMuteOn	Y
TListenDisconnect		EventListenDisconnected	N
TListenReconnect		EventListenReconnected	N
TSetDNDOn		EventDNDOn	Y
TSetDNDOff		EventDNDOff	Y
TSetMessageWaitingOn		EventMessageWaitingOn	N
TSetMessageWaitingOff		EventMessageWaitingOff	N
Query Requests			
TQuerySwitch[a]	DateTime	EventSwitchInfo	N
	ClassifierStat		N

Table 23: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
TQueryCall[a]	PartiesQuery	EventPartyInfo	N
	StatusQuery		Y
TQueryAddress[a]	AddressStatus	EventAddressInfo	Y
	MessageWaitingStatus		N
	AssociationStatus		N
	CallForwardingStatus		N
	AgentStatus		Y
	NumberOfAgentsInQueue		Y
	NumberOfAvailableAgents-InQueue		Y
	NumberOfCallsInQueue		Y
	AddressType		Y
	CallsQuery		Y
	SendAllCallsStatus		N
	QueueLoginAudit		Y
	NumberOfIdleClassifiers		N
	NumberOfClassifiersInUse		N
	NumberOfIdleTrunks		N
	NumberOfTrunksInUse		N
	DatabaseValue		N
	DNStatus		Y
	QueueStatus		Y

Table 23: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
TQueryLocation[a]	AllLocations	EventLocationInfo	I
	LocationData		I
	MonitorLocation		I
	CancelMonitorLocation		I
	MonitorAllLocations		I
	CancelMonitorAllLocations		I
	LocationMonitorCanceled		I
	AllLocationsMonitor-Canceled		I
TQueryServer[a]		EventServerInfo	Y
User-Data Requests			
TAttachUserData		EventAttachedDataChanged	Y
TUpdateUserData		EventAttachedDataChanged	Y
TDeleteUserData		EventAttachedDataChanged	Y
TDeleteAllUserData		EventAttachedDataChanged	Y
ISCC (Inter Server Call Control) Requests			
TGetAccessNumber[b]		EventAnswerAccessNumber	I
TCancelRegGetAccess-Number		EventReqGetAccess-NumberCanceled	I
Special Requests			
TReserveAgent		EventAgentReserved	I
TSendEvent		EventACK	I
TSendEventEx		EventACK	I
TSetCallAttributes		EventCallInfoChanged	I
TSendUserEvent		EventACK	Y
TPrivateService		EventPrivateInfo	Y

Table 23: Supported Functionality (Continued)

Feature Request	Request Subtype	Corresponding Event(s)	Supported
Network Attended Transfer/Conference Requests[e]			
TNetworkConsult		EventNetworkCallStatus	N
TNetworkAlternate		EventNetworkCallStatus	N
TNetworkTransfer		EventNetworkCallStatus	N
TNetworkMerge		EventNetworkCallStatus	N
TNetworkReconnect		EventNetworkCallStatus	N
TNetworkSingleStep-Transfer		EventNetworkCallStatus	N
TNetworkPrivateService		EventNetworkPrivateInfo	N
ISCC Transaction Monitoring Requests			
TTransactionMonitoring		EventACK	Y
		EventTransactionStatus	E

a. Only the requestor receives a notification of the event associated with this request.

b. This feature request may be made across locations in a multi-site environment. However, if the location attribute of the request contains a value relating to any location other than the local site—except when the response to this request is `EventError`—there will be a second event response that contains the same reference `ID` as the first event. This second event will be either `EventRemoteConnectionSuccess` or `EventRemoteConnectionFailed`.

c. Supported for SIP endpoints that have the Remote Talk feature activated.

d. SIP Server does not use the `extensions` parameter. Any data in this parameter is ignored.

e. All T-Servers support NAT/C requests with `AttributeHomeLocation` provided that this attribute identifies a network location that is capable of processing such requests. Refer to the *Network T-Server Deployment Guides* to determine whether a specific Network T-Server can process these requests.

Using the Extensions Attribute

SIP Server supports the use of the `Extensions` attribute as documented in the *Genesys 7 Events and Models Reference Manual* and the *Voice Platform SDK 7.6 .NET (or Java) API Reference.* See those documents for complete information on the T-Server events, call models, and requests. Additionally, the `Extensions` described in Table 24 are also supported.

Table 24: Use of the Extensions Attribute

Extension		Used In	Description
Key	**Type**		
Call Recording			
record	String	TRouteCall	• When set to `destination`, call recording is initiated on the routing destination DN (agent), and will continue until the agent leaves the call. • When set to `source`, call recording is initiated on the DN that sent a call to the Routing Point (customer), and will continue until the customer leaves the call. See “Call Recording” on page 102 for details.
Call Supervision			
MonitorMode	String	TMonitorNextCall	Specifies the monitor mode: • `mute, normal`—A mute connection. • `connect`—A three-party conference call. • `coach`: Only the agent can hear the supervisor (whisper coaching). See “Call Supervision” on page 104 for details.

Table 24: Use of the Extensions Attribute (Continued)

Extension		Used In	Description
Key	Type		
MonitorScope	String	TMonitorNextCall	Specifies the required intrusion/observation scope: • agent—The monitoring is initiated for a specific agent. The supervisor is disconnected when the call is transferred or released, but will be connected to the next call that is routed to the same agent. • call—The monitoring is initiated to track an entire customer call. If the call is transferred to another agent, queue, or VRU, the monitoring function continues with the call until the customer disconnects the call.
AssistMode	String	TSingleStep-Conference	Specifies the required assistance mode: • connect—This is the default value - a three-party conference call. • coach—Only the agent can hear the supervisor (whisper coaching).
No Answer Supervision			
NO_ANSWER_TIMEOUT	String	TRouteCall	If set, the value of this Extension overrides any value set in any of the following configuration options for the current call: • no-answer-timeout • agent-no-answer-timeout • extn-no-answer-timeout • posn-no-answer-timeout See "No-Answer Supervision" on page 157 for details.

Table 24: Use of the Extensions Attribute (Continued)

Extension		Used In	Description
Key	Type		
NO_ANSWER_ACTION	String	TRouteCall	If set, the value of this Extension overrides any value set in any of the following configuration options for the current call: • `no-answer-action` • `agent-no-answer-action`
NO_ANSWER_OVERFLOW	Comma-separated list	TRouteCall	If set, the value of this Extension overrides any value set in any of the following configuration options for the current call: • `no-answer-overflow` • `agent-no-answer-overflow` • `extn-no-answer-overflow` • `posn-no-answer-overflow`
Providing Call Participant Info			
LCTPartiesLength	Integer	EventUserEvent	If set, the value of this Extension specifies the how many parties are involved in a single call. See "Providing Call Participant Info" on page 164 for details.
LCTParty<n>	Integer	EventUserEvent	If set, the value of this Extension represents a party of the call, where *n* is an integer value starting from `0`. See "Providing Call Participant Info" on page 164 for details.

Table 24: Use of the Extensions Attribute (Continued)

Extension		Used In	Description
Key	Type		
Providing Caller ID			
CPNDigits	String	TMakeCall, TMakePredictiveCall, TInitiateConference, TInitiateTransfer	If set, the value of this Extension overrides the username provided in the URI in the `From` header of the `INVITE` message. This Extension is not applicable when performing a `TMakeCall` request using the `REFER` method. See "Providing a Caller ID" on page 165 for details.
Remote Supervision			
feature	String "remote-observing"	TRouteCall	This extension triggers registration of a routed party as a supervisor with parameters specified in additional `Extensions` described in this subsection. See "Remote Supervision" on page 115 for details.
dn	String	TRouteCall	An optional DN number that can be used during the monitoring session, as a substitute for the external PSTN number that the supervisor used to dial in.
agent-dn	String	TRouteCall	The target that the supervisor wants to monitor. This parameter can be a Routing Point, ACD Queue, or an agent DN.
login-id	String	TRouteCall	(Optional) A Login ID, which will be used for remote client authorization.

Table 24: Use of the Extensions Attribute (Continued)

Extension		Used In	Description
Key	Type		
monitor-type	AllCalls	TRouteCall	(Optional) The supervisor will monitor all consecutive calls for the selected target, until the supervisor decides to hang up and end the monitoring session. In between monitored calls, the supervisor’s call is parked.
password	String	TRouteCall	(Optional) A password, which will be used for remote client authorization for a specified Login ID.
post-feature-dn	String	TRouteCall	(Optional) A Routing Point, to which the remote supervisor will be connected after the supervision session.

Table 24: Use of the Extensions Attribute (Continued)

Extension		Used In	Description
Key	Type		
Video Support			
VideoFile	String	TSingleStep-Conference	A string that contains the name of the video file that will be played for the customer. If this key-value pair is not specified, the default video file will be played. The default video file is configured in the SIP Server `Application` using the `default-video-file` configuration option. See "Video Support" on page 170 for details.
AgentVideo	String	TSingleStep-Conference	A string that identifies the origin of the video stream played to the agent. The values are as follows: • `from-third-party`—The agent receives video from a third party—that is, the party that participated in the call before the operation started. • `to-third-party`—The agent receives the same video stream played to the third party—that is, video from the file specified by the `VideoFile` parameter. See "Video Support" on page 170 for details.

Error Messages

Table 25 presents the complete set of error messages SIP Server distributes in EventError, which SIP Server generates when it cannot execute a request because of an error condition.

Table 25: Error Messages for SIP Server

Code	Symbolic Name	Description
40	TERR_NOMORE_LICENSE	No more licenses are available.
41	TERR_NOT_REGISTERED	Client has not registered for the DN.
42	TERR_RESOURCE_SEIZED	Resource is already seized.
43	TERR_IN_SAME_STATE	Object is already in requested state.
50	TERR_UNKNOWN_ERROR	Unknown error code. Request cannot be processed.
51	TERR_UNSUP_OPER	Operation is not supported.
52	TERR_INTERNAL	Internal error.
53	TERR_INVALID_ATTR	Attribute in request operation is invalid.
54	TERR_NO_SWITCH	No connection to the switch.
55	TERR_PROTO_VERS	Incorrect protocol version.
56	TERR_INV_CONNID	Connection ID in request is invalid.
57	TERR_TIMEOUT	Switch or T-Server did not respond in time.
58	TERR_OUT_OF_SERVICE	Switch or T-Server is out of service.
59	TERR_NOT_CONFIGURED	DN is not configured in the Configuration Database.
61	TERR_INV_CALL_DN	DN in request is invalid.
93	TERR_DEST_INV_STATE	Invalid destination state.
96	TERR_CANT_COMPLETE_CONF	Call cannot add new conference party.
119	TERR_BAD_PASSWD	Password was invalid. May occur when SIP Server receives a 404 Not Found message from the Media Gateway after an unsuccessful attempt to route a call
122	TERR_CANT_REG_DNS	Cannot register DNs on the switch.

Table 25: Error Messages for SIP Server (Continued)

Code	Symbolic Name	Description
128	TERR_BAD_DN_TYPE	Invalid DN type for DN registration.
166	TERR_RES_UNAVAIL	(JTAPI object) resource is not available.
168	TERR_INV_ORIG_ADDR	Originating address in request was invalid.
177	TERR_TARG_DN_INV	DN target (in route call) was invalid.
195	TERR_CFW_DN_INV	Call forwarding address is invalid.
243	TERR_CLNT_NOT_MON	Internal error—client corrupted in T-Server.
259	TERR_INV_PASSWD	Invalid credentials (login_id or password).
302	TERR_INV_DTMF_STRING	DTMF string invalid.
410	TERR_INAPPR_TRTM	Invalid treatment type.
415	TERR_INV_DEST_DN	The destination DN in the request is invalid.
470	TERR_PARTY_NOT_ON_CALL	Party in request is not involved in a call.
496	TERR_INV_CALL_STATE	Party in request is in the `call` state.
506	TERR_RECVD_INV_STATE	Call/Party is in invalid state for this time
700	TERR_INV_LOGIN_REQ	Agent cannot log in at this time.
701	TERR_INV_LOGOUT_REQUEST	Agent cannot logout.
702	TERR_INV_READY_REQ	Agent cannot go to `ready` state.
1605	TERR_INVALIDPARTY	Party in request was invalid on switch.
1183	TERR_CSTA_SUBRES_OUTST_LIMIT_EXC	Rejects the second consecutive call party control request if it comes in less than one second after the first one.
3002	TERR_PRIVVIOLATION	User doesn’t have security privilege on the switch.
3005	TERR_UNSUCC_ROUTECALL	`Routecall` request was unsuccessful.

Known Limitations

Several known limitations result from the current SIP Server and softswitches/gateways interface:

1. The Stuck Calls Cleanup functionality of T-Server Common Part is not supported.
2. Due to the specifics of gateway behavior in performing SIP `REFER` methods, support for remote agents has some limitations. In order to use remote agents, you must perform one of the two following steps:
   - Provision customers and remote agents to use physically separate gateways (otherwise, calls from agents to customers take shortcuts within gateways, which means that SIP Server loses track of the call and therefore cannot perform call control). Even in this configuration, direct calls between two remote agents on the same gateway are not visible to SIP Server.

   Or,

   - Disable the SIP `REFER` method for the gateways where the remote agents are located. This enables SIP Server to see agent-to-customer and agent-to-agent calls.
3. You must configure the Outbound Contact Solution to use a single-step transfer when you use `Transfer` Mode. SIP Server does not support consultation calls when it is working with the Outbound Contact Solution.
4. A single-step conference from an existing call or an existing conference (including monitored calls) is supported to an internal DN destination only. A single-step conference from an existing call or an existing conference to a Routing Point, an ACD Queue, or an external destination is not supported.
5. A single-step transfer cannot be established from a DN in a `Ringing` state.
6. An `EventReleased (switch::)` message is issued when the last internal party leaves a call.
7. A Network Attended Transfer and Conference is not supported when SIP Server is the originating server.
8. SIP Server allows an agent to complete a consultation transfer of a call only if the call is located on a Routing Point. Third-party call control (3pcc) blind conference calls are not supported.
9. The `TDeleteFromConference` request is not supported for first-party call control (1pcc) conference calls.
10. SIP Server does not report a first-party call control (1pcc) conference with mixing on an endpoint.

Third-Party Equipment—Known Limitations

The known limitations when SIP Server is operating with a third-party equipment are as follows:

1. The Siemens HiPath 8000 switch is supported with the following limitations:
   - 1pcc (first-party call control) calls are not supported.
   - The remote answer feature (`TAnswerCall`) is not available with version 2.0. However, the remote answer feature is supported with version 2.2.
   - Call scenarios containing the `REFER` message are not supported with this switch. SIP Server must be configured to use `re-INVITE`-based call control methods.
   - Genesys recommends setting the `dual-dialog-enabled` configuration option to `false` if Siemens optiPoint phones are used in `re-INVITE` mode for third-party call control (3pcc) operations.
   - Sometimes SIP Server cannot retrieve a call within a mixed phone environment. To avoid this problem, set the `sip-hold-rfc3264` option with a proper value on the `Annex` tab of the DN.
2. The following media gateways support `re-INVITE`-based call transfers only:
   - Alcatel 7515
   - Cisco A5350
   - Cisco A5400
   - Asterisk
   - Sonus

Chapter

8 SIP Server Configuration Options

This chapter describes the configuration options that are unique to SIP Server and contains the following sections:

SIP Server also supports common log options described in Chapter 11 on page 321 and options common to all T-Servers described in Chapter 12 on page 343. Please refer to the *Framework 7.6 Stream Manager Deployment Guide* if you need to configure Stream Manager for use with SIP Server.

Application-Level Options

Configuration options specific to SIP Server functionality are set in Configuration Manager, in the corresponding sections on the `Options` tab of the SIP Server `Application` object.

T-Server Section

This section must be called `TServer`.

For ease of reference, the options have been arranged in alphabetical order.

acw-in-idle-force-ready

Default Value: `true`
Valid Values: `true`, `false`
Changes Take Effect: Immediately
Related Feature: "Emulated Agents" on page 138

Specifies whether, after timed manual wrap-up (when option `timed-acw-in-idle` is set to `true`), SIP Server forces the agent to the `Ready` state. With value `false`, SIP Server returns the agent to the state he or she was in prior to requesting manual wrap-up.

Note: For compatibility with the previous SIP Server releases, you can use the name `cwk-in-idle-force-ready` for this option as an alias.

after-routing-timeout

Default Value: `10`
Valid Values: Any integer
Changes Take Effect: Immediately

Specifies the length of time (in seconds) that SIP Server waits before diverting the call from the Routing Point DN to the destination DN after `RequestRouteCall` was processed. When the call is not diverted before the specified value, the `EventError` message is issued. It will contain the `Reference ID` of the `TRouteCall` request.

Notes: Set the value of the `after-routing-timeout` option less than the value of the `rq-expire-tmout` option.

The `after-routing-timeout` option is also dependent on the `divert-on-ringing` option:

- When the `divert-on-ringing` option is set to `true`, the call is considered as "diverted" when the `180 Ringing` message arrives from the destination DN.
- When the `divert-on-ringing` option is set to `false`, the call is considered as "diverted" when the `200 OK` message arrives from the destination DN.

agent-group

Default Value: None
Valid Value: Any agent group value
Changes Take Effect: At the next agent login session

Specifies a value for an agent group that will be used for SIP Server reporting.

SIP Server obtains the value for this option in the following order of precedence:

1. In the `TServer` section of the `Annex` tab of the `DN` object.

2. In the TServer section of the Options tab of the SIP Server Application object.

agent-no-answer-action

Default Value: none
Valid Values:

none	SIP Server takes no action on agents when calls are not answered.
notready	SIP Server sets agents to NotReady when calls are not answered.
logout	SIP Server automatically logs out agents when calls are not answered.

Changes Take Effect: Immediately
Related Feature: "No-Answer Supervision" on page 157

Defines SIP Server's default action if a logged-in agent fails to answer a call within the time defined in the agent-no-answer-timeout option. See also the NO_ANSWER_ACTION extension in section "Using the Extensions Attribute" on page 186 for more information about how this option is used.

SIP Server obtains the value for this option in the following order of precedence:

1. In the TServer section of the Annex tab of the Agent Login object.
2. In the TServer section of the Options tab of the SIP Server Application object.

agent-no-answer-overflow

Default Value: No default value
Valid Values:

none	SIP Server does not attempt to overflow a call on an agent desktop when the agent-no-answer-timeout option expires.
recall	SIP Server returns the call to the last distribution device (the device reported in the ThisQueue attribute of the call) when the agent-no-answer-timeout option expires.
release	SIP Server releases the call.
Any valid overflow destination	SIP Server returns the call to the specified destination when the value set when the agent-no-answer-timeout option expires.

Changes Take Effect: Immediately
Related Feature: "No-Answer Supervision" on page 157

Specifies a sequence of overflow destinations (separated by comma) that SIP Server attempts to overflow to when the time specified in the agent-no-answer-timeout option expires. SIP Server attempts to overflow in the order specified in the list.

- When all overflow attempts fail, SIP Server abandons overflow. See also the NO_ANSWER_OVERFLOW extension in section “Using the Extensions Attribute” on page 186 for more information about how this option is used.
- When the list of overflow destinations contains the recall value and the call was not distributed, SIP Server skips to the next destination in the list.

SIP Server obtains the value for this option in the following order of precedence:

1. In the TServer section of the Annex tab of the Agent Login object.
2. In the TServer section of the Options tab of the SIP Server Application object.

agent-no-answer-timeout

Default Value: 15
Valid Value: Any integer from 0–600
Changes Take Effect: Immediately
Related Feature: “No-Answer Supervision” on page 157

Defines the default time (in seconds) that SIP Server allows for a logged-in agent to answer a call before executing the actions defined in the agent-no-answer-overflow and agent-no-answer-action options.

When set to 0, the Agent No-Answer Supervision feature is disabled. See the NO_ANSWER_TIMEOUT extension in section “Using the Extensions Attribute” on page 186 for more information about how this option is used.

SIP Server obtains the value for this option in the following order of precedence:

1. In the TServer section of the Annex tab of the Agent Login object.
2. In the TServer section of the Options tab of the SIP Server Application object.

agent-strict-id

Default Value: false
Valid Values: true, false
Changes Take Effect: Immediately
Related Feature: “Emulated Agents” on page 138

Specifies whether, for agents, SIP Server enables any AgentID to be used during login (value false), or only those configured in the Configuration Layer (value true).

am-detected

Default Value: drop
Valid Values:

drop	The call is released.
connect	The connected call stays connected.

Changes Take Effect: Immediately

Specifies the behavior of SIP Server where CPD is operating and an answering machine is detected on an Outbound call. SIP Server provides the CPD result in `UserData` attached to the call as a key-value pair with key `AnswerClass` containing the value `AM`. This `UserData` in `EventRouteRequest` provides extra information to the strategy, so that the strategy can decide to drop the `AM` call if required.

audio-codecs

Default Value: `telephone-event, PCMU, PCMA, G723, G729, GSM`
Valid Values: Any from the list of `telephone-event, PCMU, PCMA, G723, G729,` and `GSM` words, delimited by commas. Unrecognized words are ignored.
Changes Take Effect: Immediately

Specifies the list of audio codec files that Stream Manager release 7.0.2 (or earlier) uses to play treatments.

Note: This option is obsolete. It can be used for backward compatibility with the previous SIP Server releases.

backwds-compat-acw-behavior

Default Value: `false`
Valid Value: `true, false`
Changes Take Effect: Immediately
Related Feature: “Emulated Agents” on page 138

Specifies whether pre-7.5 behavior after-call work is enabled (value = `true`) or disabled (value = `false`), for backward compatibility.

Calls While in Emulated ACW

With value `true,` if an agent receives or makes a business call while in emulated ACW, SIP Server does the following:

1. Stops the ACW timer.
2. Forces the agent to the `Ready` state.
3. Restarts ACW (and the legal-guard timer) after the new business call is released.

If an agent makes or receives a work-related call while in ACW, SIP Server does the following:

1. Suspends the ACW, but leaves the agent in the ACW state.
2. Resumes the ACW timer once the work-related call is released.

SIP Server categorizes as a *work-related call* any call that an agent makes while in the `NotReady` state with `workmode` set to `AfterCallWork` or `AuxWork`.

After the ACW and any configured legal-guard time have been completed, the agent is forced to the `Ready` state. If an agent makes or receives a private call during ACW, no action is taken and the ACW timer keeps running.

With value `false,` if an agent receives or makes a business call while in emulated ACW, SIP Server does the following:

1. Stops the ACW timer and adds the remaining amount of ACW to the ACW period for the new call. If either of the ACW periods is untimed, the resulting ACW will also be untimed.
2. Forces the agent to the `NotReady (ManualIn)` state.
3. Restarts ACW after the business call is released.

If an agent makes or receives a work-related or private call while in ACW, SIP Server does the following:

1. Suspends the ACW timer.
2. Forces the agent to the `NotReady (ManualIn)` state.
3. Returns the agent to the ACW state, and resumes the ACW timer once the call is released.

After the ACW and any configured legal-guard time have been completed, the agent is forced to the `Ready` state.

Business Call While Not Ready

With value `true,` if an agent receives a business call while in an emulated `NotReady` state, except for ACW or legal-guard time, SIP Server sets the agent state to `Ready` for the duration of the business call.

With value `false,` if an agent receives a business call while in emulated `NotReady` state, except for ACW or legal-guard time, SIP Server will maintain the current agent state for the duration of the business call. After the call and any associated wrap-up are completed, SIP Server will return to his or her previous `NotReady` state. Note that no legal-guard time is applied, because the agent does not go into the `Ready` state.

busy-tone

Default Value: `music/busy_5sec`
Valid Values: Name and path of any valid audio file
Changes Take Effect: Immediately for all new calls

Specifies the audio file to be played for the `Busy` treatment.

call-rq-gap

Default Value: `0`
Valid Value: Any integer from `0–1000`
Changes Take Effect: Immediately

Specifies (in milliseconds) the length of delay applied to a request issued against a busy call (a call that has another request working on it already). This prevents race conditions on the different call legs.

Note: Genesys recommends that you set this option to `0`.

cancel-monitor-on-disconnect

Default Value: `true`
Valid Values:

`true`	Call supervision subscription is canceled.
`false`	Call supervision subscription is not canceled.

Changes Take Effect: Immediately
Related Feature: "Call Supervision" on page 104

Specifies whether call supervision subscription is canceled when the client that requested it disconnects from SIP Server.

collect-tone

Default Value: `music/collect`
Valid Values: Name and path of any valid audio file
Changes Take Effect: Immediately for all new calls

Specifies the audio file that SIP Server uses to produce the noncompletion tone played during DTMF digit collection.

cpd-info-timeout

Default Value: `3`
Valid Values: `0-30`
Changes Take Effect: Immediately

Specifies the time interval, in seconds, during which SIP Server waits for SIP `INFO` messages with the results of call progress detection from a gateway. This timeout starts right after the `200 OK` messages is received from the gateway.

default-dn

Default Value: `NULL`
Valid Value: Any valid DN
Changes Take Effect: Immediately

Specifies the DN to which calls are sent when URS is nonoperational, or when the timeout specified in the `router-timeout` option expires.

Note: You can also use this option for emergency ACD routing.

This option can be set as the SIP Server Application level and at the Switch/DN level. The setting at the Switch/DN level takes precedence over the Application level setting.

default-monitor-mode

Default Value: `mute`

Valid Values:

mute (or normal)	Silent monitoring is used (supervisor connection is mute)
coach	Whisper coaching is used (only the monitored agent can hear the supervisor)
connect	The open supervisor presence is used

Changes Take Effect: Immediately
Related Feature: "Call Supervision" on page 104

Initializes a new call supervision subscription monitor mode if the MonitorMode extension is not provided (or if its value is specified incorrectly) in the TMonitorNextCall request.

default-monitor-scope

Default Value: call
Valid Values:

call	The supervisor remains on the call until it is finished.
agent	SIP Server disconnects the supervisor from the call automatically when the monitored agent leaves the call.

Changes Take Effect: Immediately
Related Feature: "Call Supervision" on page 104

Initializes a new call supervision subscription monitor scope if the MonitorScope extension is not provided (or its value is specified incorrectly) in the TMonitorNextCall request.

default-music

Default Value: music/on_hold
Valid Value: Name and path of any valid audio file
Changes Take Effect: Immediately for all new calls

Specifies the name of the file that is played for the music treatment, if none is specified in TApplyTreatment, or if the specified file is missing.

default-video-file

Default Value: NULL
Valid Values: Any valid video file codec and path
Changes Take Effect: Immediately
Related Feature: "Video Support" on page 170

Contains the name of the video file that will be played to the caller if a single-step conference to the gcti::video device does not contain a VideoFile key in the Extensions attribute.

delay-between-refresh-on-switchover

Default Value: 0
Valid Values: A standard timeout value format

Changes Take Effect: Immediately

Specifies the delay timeout between dialogs re-invites that SIP Server performs after the `delay-to-start-refresh-on-switchover` option setting expires. When this option is set to `0` (the default), the dialogs are not updated. Recommended values between `10` and `50 msec.`

delay-to-start-refresh-on-switchover

Default Value: `10000 (10 seconds)`
Valid Values: A standard timeout value format
Changes Take Effect: Immediately

Specifies the time interval that SIP Server waits after a switchover before re-inviting previously connected dialogs. The timeout must be long enough to account for possible network switching delays.

divert-on-ringing

Default Value: `true`
Valid Values:

`true`	SIP Server generates `EventRouteUsed` and `EventDiverted` messages when the SIP `180 Ringing` response arrives for the `INVITE` request at the routing destination.
`false`	SIP Server postpones `EventRouteUsed` and `EventDiverted` messages until the call is answered by the routing destination with a SIP `200 OK` message. If the call is not answered within the value specified by the `rq-expire-tmout` option, the destination SIP dialog is canceled and an `EventError` message is generated.

Changes Take Effect: Immediately

Determines SIP Server behavior when routing calls.

dtmf-payload

Default Value: `101`
Valid Values: Any integer from `0–128`
Changes Take Effect: Immediately

Specifies the value to be used in negotiations for the DTMF payload type, when Stream Manager release 7.0.2 (or earlier) treatments are used.

Note: This option is obsolete. It can be used for backward compatibility with the previous SIP Server releases.

emergency-recording-cleanup-enabled

Default value: `false`
Valid values:

`true`	SIP Server automatically terminates emergency recording.
`false`	SIP Server does not terminate emergency recording.

Changes Take Effect: Immediately

Specifies whether SIP Server automatically terminates emergency recording when no internal parties remain on a call.

emergency-recording-filename

Default Value: NULL
Valid Values: Any valid file name using the variables specified below
Changes Take Effect: When the next emergency call recording is initiated

Specifies the recorded file name when emergency call recording is initiated by an agent. When this option contains a value, the generated emergency call recording file name is added as UserData to the call with the GSIP_EMRGREC_FN key. When this option does not contain a value, the recorded file name will be the UUID of the call.

The following variables are used when creating the file:

$ANI$:	The calling number.
$DNIS$:	The called number.
$DATE$:	The current date (GMT) in the Y-M-D format.
$TIME$:	The current time (GMT) in the H-M-S format.
$CONNID$:	The Connection ID of the call.
$UUID$:	The UUID of the call.
$AGENTID$:	The Agent Login ID, if the agent is logged in on the device where the emergency call recording is initiated.
$AGENTDN$:	The DN where the emergency call recording is initiated.

emulated-login-state

Default Value: ready
Valid Values:

ready	SIP Server distributes EventAgentReady after EventAgentLogin.
not-ready	SIP Server distributes EventAgentNotReady after EventAgentLogin.

Changes Take Effect: Immediately
Related Feature: “Emulated Agents” on page 138

When SIP Server performs an emulated agent login and the client specifies an agent work mode other than ManualIn or AutoIn, SIP Server uses this option to determine which event to distribute.

When the client specifies the agent work mode ManualIn, SIP Server distributes EventAgentNotReady after EventAgentLogin, and places the agent in the NotReady state.

When the client specifies the agent work mode AutoIn, SIP Server distributes EventAgentReady after EventAgentLogin, and places the agent in the Ready state.

This option can be set in a number of places, and SIP Server processes it in the following order of precedence, highest first. If the value is not present at the higher level, SIP Server checks the next level, and so on.

1. In the `Agent Login` object, on the `Annex` tab in the `TServer` section.
2. In the `DN` object which represents the device, on the `Annex` tab in the `TServer` section.
3. In the `DN` object which represents the Agent Group, such as an ACD Queue, on the `Annex` tab in the `TServer` section.
4. In the SIP Server `Application` object, on the `Options` tab in the `TServer` section.

enforce-external-domains

Default Value: `NULL`
Valid Values: A list of computer names or IP addresses that are external to SIP Server. The list can be separated by semicolons (;).
Changes Take Effect: Immediately

When a value is configured, SIP Server checks the list of computer names or IP addresses against the computer names or IP addresses specified in the URI of the `From` header. If there is a match, then the DN is considered external.

When a value is not configured, SIP Server uses the user part of the URI only to find the device.

event-ringing-on-100trying

Default Value: `false`
Valid Values:

`true`	SIP Server generates `EventRinging`.
`false`	SIP Server does not generate `EventRinging`.

Changes Take Effect: Immediately

Specifies whether SIP Server generates an `EventRinging` message for a DN when it receives a `100 Trying` SIP message. Normally, the `EventRinging` message is generated on `180 Ringing` SIP message, but this option allows for GVP integration when the IVR Server is configured in Behind-the-Switch mode.

Note: This option must be set at both the `Application` and at the `Switch/DN` level because it is used for proper synchronization with the I-Server `Application`.

external-registrar

Default Value: `NULL`
Valid Values: String conforming to the SIP-URI syntax of RFC 3261, defined as:

```
sip:[userinfo]hostport uri-parameters[headers]
```

Changes Take Effect: Immediately

Specifies the location of an external registrar service. SIP Server implements very limited registrar functionality in order to support clients that can only register dynamically (for example, Microsoft Messenger 4.7–5.1). Such clients must be configured in the Configuration Layer as DNs. Depending on the state of the internal SIP Server registrar, registration subscriptions from either all, or only unconfigured, clients are forwarded to the external registrar.

For example: `sip:192.168.8.100:5090;transport=tcp`

If no external registrar is specified, a `503 Service Unavailable` error is returned for the `REGISTER` method.

extn-no-answer-overflow

Default Value: None
Valid Values:

`none`	SIP Server does not attempt to overflow a call on an extension when `extn-no-answer-timeout` expires.
`recall`	SIP Server returns the call to the last distribution device (the device reported in the `ThisQueue` attribute of the call) when `extn-no-answer-timeout` expires.
`release`	SIP Server releases the call.
Any valid overflow destination	SIP Server returns the call to the specified destination when the value set for the `extn-no-answer-timeout` option expires.

Changes Take Effect: Immediately
Related Feature: "No-Answer Supervision" on page 157

Specifies a sequence of overflow destinations (separated by comma) that SIP Server attempts to overflow to when the time specified in option `extn-no-answer-timeout` has expired. SIP Server attempts to overflow in the order specified in the list.

- When all overflow attempts fail, SIP Server abandons overflow. See also the `NO_ANSWER_OVERFLOW` extension in section "Using the Extensions Attribute" on page 186 for more information about how this option is used.
- When the list of overflow destinations contains the `recall` value and the call was not distributed, T-Server skips to the next destination in the list.

SIP Server obtains the value for this option in the following order of precedence:

1. In the `TServer` section of the `Annex` tab of the `DN` object of type `Extension`.
2. In the `TServer` section of the `Options` tab of the SIP Server `Application` object.

extn-no-answer-timeout

Default Value: `15`
Valid Value: Any integer from `0–600`
Changes Take Effect: Immediately

Related Feature: "No-Answer Supervision" on page 157

Defines the default no-answer timeout (in seconds) that SIP Server applies to any device of type extension. When the timeout ends, SIP Server executes the actions defined in option extn-no-answer-overflow.

When set to 0, the No Answer Supervision feature for DNs of type Extension is disabled. See the NO_ANSWER_TIMEOUT extension in section "Using the Extensions Attribute" on page 186 for more information about how this option is used.

SIP Server obtains the value for this option in the following order of precedence:

1. In the TServer section of the Annex tab of the DN object of type Extension.
2. In the TServer section of the Options tab of the SIP Server Application object.

fast-busy-tone

Default Value: music/atb_5sec
Valid Values: Name and path of any valid audio file
Changes Take Effect: Immediately for all new calls

Specifies the audio file to be played for the FastBusy treatment.

fax-detected

Default Value: drop
Valid Values:

drop	The call is released.
connect	The connected call stays connected.

Changes Take Effect: Immediately

Specifies the behavior of SIP Server where CPD is operating and a fax machine is detected on an outbound call. SIP Server provides the CPD result in UserData attached to the call as a key-value pair with key AnswerClass containing the value Fax. This UserData in EventRouteRequest provides extra information to the strategy, so that the strategy can decide to drop the Fax call if required.

find-trunk-by-location

Default Value: false

Valid Values:

`true`	SIP Server chooses a gateway or trunk for the outbound call by matching the value of the `geo-location` option of the DN with the value of the `geo-location` option for the `Trunk` device.
`false`	SIP Server chooses a gateway or trunk from the pool of all configured trunks which are in service based on the prefix match. (The value of the `geo-location` option will be ignored.) If there is more than one trunk in the pool, SIP Server chooses the trunk in a round-robin algorithm that provides equal gateway load. However, if an external party is transferred to an outbound destination, the same gateway that connected the external party to the call is used for the outbound transfer.

Changes Take Effect: Immediately

Determines SIP Server behavior for choosing a gateway or trunk for the outbound call.

forced-notready

Default Value: `true`
Valid Values:

`true`	The desktop is forced into the `Not Ready` state.
`false`	The desktop is not forced into the `Not Ready` state.

Changes Take Effect: Immediately for all future calls

Determines whether the desktop is forced into a `Not Ready` state when it does not respond after a `Preview Interaction` dialog box has been displayed on the desktop.

Note: This option works with the `preview-interaction` and `preview-expired` options to determine what action to take when a desktop does not respond to a preview interaction before the time expires.

inbound-bsns-calls

Default Value: `false`
Valid Values: `true, false`
Changes Take Effect: Immediately
Related Feature: “Business-Call Handling” on page 138

Specifies whether SIP Server considers all established inbound calls on an agent as business calls.

inherit-bsns-type

Default Value: `false`
Valid Values: `true, false`
Changes Take Effect: Immediately
Related Feature: “Business-Call Handling” on page 138

Determines whether a consultation call that is made from a business primary call contains the business call attribute.

internal-bsns-calls

Default Value: false
Valid Values: true, false
Changes Take Effect: Immediately
Related Feature: "Business-Call Handling" on page 138

Determines whether SIP Server considers internal calls made to any agent as business calls.

internal-registrar-domains

Default Value: NULL
Valid Values: Any valid computer names, separated by a semi-colon (;)
Changes Take Effect: Immediately. Existing subscriptions remain valid.

Specifies the list of logical computer names, registration subscriptions from the endpoints of which are handled by the internal registrar. For example, if DN 4813 is configured in Configuration Manager, DN 4814 is not configured, and the internal registrar is enabled, then:

- REGISTER from 4813@world is accepted.
- REGISTER from 4813@galaxy is forwarded to the external registrar.
- REGISTER from 4814@world is rejected with 404 Not Found.

internal-registrar-enabled

Default Value: true
Valid Values:

true	SIP Server's internal registrar is enabled.
false	All registration subscriptions are proxied to external registrar (see the external-registrar option).

Changes Take Effect: Immediately. Existing subscriptions remain valid.

Specifies whether the internal registrar is enabled. When this option is set to false, a 503 Service Unavailable error is returned for the REGISTER method.

internal-registrar-persistent

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

Enables SIP Server to update the DN attribute contact in the configuration database. When an endpoint registers, SIP Server takes the contact information from the REGISTER request and updates or creates a key called contact in the Annex tab of the corresponding DN.

In hot standby configuration, set the internal-registrar-persistent option to true to enable Configuration Server to propagate changes of the contact information to the backup SIP Server.

Note: SIP Server must have Full Control permission for the DN objects in order to update a configuration object. By default, it does not have this permission. You need to grant Full Control permission for the System account for the all DNs on the corresponding switch. It is done for all DNs at once by changing the permissions for the System account on the DN folder in the Switch object. Or, you can start SIP Server under another account that has Change permission on the necessary DNs.

intrusion-enabled

Default Value: true
Valid Values:

true	SIP Server invites the supervisor to the current call.
false	SIP Server does not invite the supervisor to the current call. Instead, SIP Server will wait for the next call on the monitored agent's DN to invite the supervisor.

Changes Take Effect: Immediately
Related Feature: "Call Supervision" on page 104

Determines SIP Server behavior when a TMonitorNextCall request is submitted while the monitored agent is on a call.

legal-guard-time

Default Value: 0
Valid Value: Any integer from 0–30
Changes Take Effect: Immediately
Related Feature: "Emulated Agents" on page 138

Specifies a legal-guard time (in seconds) for agents to postpone the transition to the Ready state after a business call or after timed ACW. SIP Server always considers a routed call as a business call. The default value of 0 (zero) disables the functionality of this option.

logout-on-disconnect

Default Value: true
Valid Values:

true	The EventLogout message is distributed as soon as the client that requested the login disconnects from SIP Server or unregisters the DN in question. The EventLogout message is distributed when SIP Server distributes EventOutOfService.
false	The EventLogout message is not distributed.

Changes Take Effect: Immediately

Specifies how the EventLogout message is distributed.

make-call-alert-info

Default Value: NULL
Valid Value: Any string
Changes Take Effect: Immediately

The contents of this field are passed in the Alert-Info header of the INVITE message sent to the origination party in response to a TMakeCall request. This is used to enable a distinctive ringtone or auto-answer on the originating party's endpoint.

For example, setting this field to <file://Bellcore-dr3> turns on a triple ring on Cisco 7940 endpoints.

max-legs-per-sm

Default Value: 0
Valid Values: Any integer from 0–65000
Changes Take Effect: Immediately for new legs. Previously created legs are not dropped.

Specifies the maximum number of legs that are created on each connected Stream Manager. If all connected Stream Managers already have the specified number of legs, no new leg is created.

monitor-internal-calls

Default Value: true
Valid Values:

true	SIP Server starts monitoring sessions for all calls on the DN where call supervision subscription is active.
false	SIP Server starts monitoring sessions only if external parties participate in the call.

Changes Take Effect: Immediately
Related Feature: "Call Supervision" on page 104

Specifies SIP Server behavior to start monitoring sessions.

music-in-conference-file

Default Value: The value is taken from the default-music option
Valid Values: A string containing the valid name of the music file
Changes Take Effect: Available for next 3pcc or 1pcc hold operation
Related Feature: "Silence Treatment in Conference" on page 125

Specifies the silent audio file to be played in applicable conferences (more than two active participants). If the conference has only two active participants, then the music file defined in the default-music option will be played for the other party when the call is placed on hold. For conferences with more than two active participants, the music-in-conference-file option is used for a silent MOH treatment instead. Recommended value is music/silence.

For example, if a supervisor in silent monitoring mode listens in on a call between a customer and an agent, the supervisor is not considered an active participant. If the agent or customer places the call on hold, the remaining participants will hear the default-music MOH treatment. However if the supervisor places the call on hold, the music/silence file is played instead, so that the customer and agent can continue their conversation undisturbed.

music-in-queue-file

Default Value: None
Valid Values: `<default_music_directory>/<file_name>`
Changes Take Effect: Immediately for all new calls

Specifies the file name of the music file to be played when a call is queued on a particular ACD Queue.

Notes: This option is set at the SIP Server `Application` level and at the `Switch/DN` level (DN of type `ACD Queue`). The setting at the `Switch/DN` level takes precedence over the `Application` level setting.

If there is no value specified for this option, the value of the Stream Manager DN `request-uri` option is used instead. For more information about this parameter, see the *Framework 7.6 Stream Manager Deployment Guide*.

mwi-agent-enable

Default Value: `false`
Valid Values:

`true`	MWI for the agent's voice mail box is enabled.
`false`	MWI for the agent's voice mail box is disabled.

Changes Take Effect: Immediately

Related Feature: "Message Waiting Indicator Functionality" on page 99

Enables or disables MWI for the agent's voice mail box.

mwi-domain

Default Value: None
Valid Values: Any computer name
Changes Take Effect: During the next attempt to register for MWI
Related Feature: "Message Waiting Indicator Functionality" on page 99

Specifies the computer name in the URI of the `REGISTER` request. SIP Server sends this information to Asterisk in order to initiate MWI. The value of this option must be a computer name that is recognized by Asterisk.

mwi-extension-enable

Default Value: `false`

Valid Values:

`true`	MWI for the extension's voice mail box is enabled.
`false`	MWI for the extension's voice mail box is disabled.

Changes Take Effect: Immediately
Related Feature: "Message Waiting Indicator Functionality" on page 99

Enables or disables MWI for the extension's voice mail box.

mwi-group-enable

Default Value: `false`
Valid Values:

`true`	MWI for the agent groups's voice mail box is enabled
`false`	MWI for the agent group's voice mail box is disabled.

Changes Take Effect: Immediately
Related Feature: "Message Waiting Indicator Functionality" on page 99

Enables or disables MWI for the agent groups's voice mail box.

mwi-host

Default Value: None
Valid Values: Any host name or IP address
Changes Take Effect: During the next attempt to register for MWI
Related Feature: "Message Waiting Indicator Functionality" on page 99

Specifies the host name of the Voice Mail system to get MWI notification from the host where Asterisk is running. SIP Server will send a `REGISTER` request to `mwi-host:mwi-port` to initiate MWI.

mwi-mode

Default Value: `REGISTER`
Valid Values: `REGISTER, SUBSCRIBE`
Changes Take Effect: After SIP Server restart
Related Feature: "Message Waiting Indicator Functionality" on page 99

When this option is set to `SUBSCRIBE,` SIP Server activates SIP subscriptions for all voice mail box owners as configured by other `mwi-<>` options. When set to `REGISTER,` the MWI functionality is enabled using the `REGISTER` SIP request method. For backward compatibility with the previous SIP Server releases, set this option to a value of `REGISTER`.

mwi-port

Default Value: None
Valid Values: Any available port
Changes Take Effect: During the next attempt to register for MWI
Related Feature: "Message Waiting Indicator Functionality" on page 99

Specifies the port of the Voice Mail system to get MWI notification from the port where Asterisk is running. SIP Server will send a `REGISTER` request to `mwi-host:mwi-port` to initiate MWI.

nas-private

Default Value: `false`
Valid Values: `true, false`
Changes Take Effect: Immediately
Related Feature: "No-Answer Supervision" on page 157

Specifies whether No-Answer Supervision is enabled for private calls.

You can set this option at the `Application` and `Switch/Agent Login` or `Switch/DN` level (DN of type `Extension`). When set at the `Application` level, the option value is applied globally to all private calls. When set at the `Switch` level, the option value is applied to a particular DN or Agent Login.

Note: The option setting at the `Switch` level takes precedence over the `Application` level setting.

observing-routing-point

Default Value: None
Valid Values: A `Routing Point` DN
Changes Take Effect: For the next call
Related Feature: "Remote Supervision" on page 115

Specifies the service observing Routing Point used for Multi-Site Supervision of the agents, whose endpoints are controlled by this SIP Server. This option must contain a number of a valid `Routing Point` DN in order for the Multi-Site Supervision feature to work. No routing strategy is required to be loaded on the observing Routing Point.

outbound-bsns-calls

Default Value: `false`
Valid Values: `true, false`
Changes Take Effect: Immediately
Related Feature: "Business-Call Handling" on page 138

Specifies whether SIP Server considers all established outbound calls on an agent as business calls.

override-to-on-divert

Default Value: `false`
Valid Values:

`true`	The username is equal to the destination DN.
`false`	The username is equal to the Routing Point or ACD Queue number.

Changes Take Effect: Immediately

Controls the username part of the `To` header URI for outgoing `INVITE` messages when a call is diverted from a Routing Point or an ACD Queue.

parking-music

Default value: `music/silence`
Valid Values:
Changes Takes Effect: For the next parked call
Related Feature: "Remote Supervision" on page 115

Specifies the music file, which is played to the remote party parked on the `gcti::park` DN.

posn-no-answer-overflow

Default Value: None.
Valid Values:

`none`	SIP Server does not attempt to overflow a call on a position when `posn-no-answer-timeout` expires.
`recall`	SIP Server returns the call to the last distribution device (the device reported in the ThisQueue attribute of the call) when `posn-no-answer-timeout` expires.
`release`	SIP Server releases the call.
Any valid overflow destination	SIP Server returns the call to the specified destination when the value set for the `posn-no-answer-timeout` option expires.

Changes Take Effect: Immediately
Related Feature: "No-Answer Supervision" on page 157

Specifies a sequence of overflow destinations (separated by comma) that SIP Server attempts to overflow to when the time specified in option `posn-no-answer-timeout` expires. SIP Server attempts to overflow in the order specified in the list.

- When all overflow attempts fail, SIP Server abandons overflow. See also extension `NO_ANSWER_OVERFLOW` in section "Using the Extensions Attribute" on page 186 for more information about how this option is used.
- When the list of overflow destinations contains the value `recall` and the call was not distributed, SIP Server skips to the next destination in the list.

SIP Server obtains the value for this option in the following order of precedence:

1. In the `TServer` section of the `Annex` tab of the `DN` object of type `ACD Position`.
2. In the `TServer` section of the `Options` tab of the SIP Server `Application` object.

posn-no-answer-timeout

Default Value: 15
Valid Value: Any integer from 0–600
Changes Take Effect: Immediately
Related Feature: "No-Answer Supervision" on page 157

Defines the default no-answer timeout (in seconds) that SIP Server applies to any device of type position. When the timeout ends, SIP Server executes the actions defined in option posn-no-answer-overflow.

When set to 0, the No Answer Supervision feature for DNs of type ACD Position is disabled. See the NO_ANSWER_TIMEOUT extension in section "Using the Extensions Attribute" on page 186 for more information about how this option is used.

SIP Server obtains the value for this option in the following order of precedence:

1. In the TServer section of the Annex tab of the DN object of type Position.
2. In the TServer section of the Options tab of the SIP Server Application object.

predictive-call-router-timeout

Default Value: 20
Valid Value: Any non-negative integer
Changes Take Effect: After SIP Server restart

Specifies the maximum time (in seconds) that an answered predictive call can wait on a Routing Point DN for a Universal Routing Server (URS) request. If there is no request during this time, the call is dropped. This is primarily a clean-up mechanism for scenarios when URS is non-operational.

preview-expired

Default Value: 90
Valid Values: Any positive integer
Changes Take Effect: Immediately for future calls
Related Feature: "Preview Interactions" on page 164

Specifies the time (in seconds) that the Preview Interaction dialog box remains open on a desktop. After the time expires, the dialog box closes and the desktop changes to a Not Ready state.

Note: The preview-expired option works with the preview-interaction and forced-notready options to determine what action to take when a desktop does not respond to a preview interaction before the time expires.

recording-filename

Default Value: NULL

Valid Values: Any valid file name using the variables specified below
Changes Take Effect: When the next call recording is initiated
Related Feature: "Call Recording" on page 102

Specifies the file name for call recording when call recording is initiated automatically, according to the SIP Server configuration. When this option contains a value, the generated file name is added as UserData to the call with the GSIP_REC_FN key. When this option does not contain a value, the file name is the UUID of the call.

The following variables are used when creating the file:

$ANI$:	The calling number.
$DNIS$:	The called number.
$DATE$:	The current date (GMT) in the Y-M-D format.
$TIME$:	The current time (GMT) in the H-M-S format.
$CONNID$:	The Connection ID of the call.
$UUID$:	The UUID of the call.
$AGENTID$:	The Agent Login ID, if the agent is logged in on the device where the call recording is initiated.
$AGENTDN$:	The DN where the call recording is initiated.

registrar-default-timeout

Default Value: 1800
Valid Values: 1 - 3600
Changes Take Effect: Immediately

Specifies the expiration timeout for a REGISTER request as a value (in seconds) in the 200 OK response that is sent by SIP Server to the SIP endpoint.

ring-tone

Default Value: music/ring_back
Valid Values: Name and path of any valid audio file
Changes Take Effect: Immediately for all new calls

Specifies the audio file to be played for the RingBack treatment.

ringing-on-route-point

Default Value: true
Valid Values:

true	SIP Server responds with a 180 Ringing message.
false	SIP Server does not respond with a 180 Ringing message.

Changes Take Effect: Immediately

Specifies whether SIP Server responds with a 180 Ringing message when a call arrives at a Routing Point. It enables transfers for calls waiting at Routing Points. The disadvantages are:

- Possible undesirable ringback tone.

- Multiple ringing messages delivered for the same call.

router-timeout

Default Value: 10
Valid Value: Any non-negative integer
Changes Take Effect: Immediately

Specifies the maximum time (in seconds) that a call remains on a Routing Point before a timeout is triggered and the call is sent to the DN specified in default-dn.

session-refresh-interval

Default Value: 1800
Valid Values: 0, 90—1800
Changes Take Effect: Immediately

Specifies (in seconds) how often active calls are checked to see if they are still active. A 0 (zero) value disables this feature (the session refresh mechanism is turned off). Values between 1 and 89 (inclusive) are treated as value 90.

This option is used to remove stuck calls that must accumulate if endpoints terminate calls without sending the appropriate SIP message.

set-notready-on-busy

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

With this option set to true, when a call is distributed to a ready agent (that is, the agent is not previously engaged in a call), and the agent endpoint responds to the INVITE with a 4xx, 5xx, or 6xx message, SIP Server places the agent in the Not Ready state (an EventAgentNotReady message is distributed). In addition, a ReasonCode key with a value equal to a returned error will be reported in the Extensions attribute in the EventAgentNotReady message. If a call is distributed to an agent via an ACD queue, the agent is placed in the Not Ready state and the call is diverted to the same ACD queue (at the end of the queue).

silence-tone

Default Value: music/silence
Valid Values: Name and path of any valid audio file
Changes Take Effect: Immediately for all new calls

Specifies the audio file to be played for the Silence treatment.

sip-address

Default Value: NULL
Valid Values: Any valid IP address or host name
Changes Take Effect: After SIP Server restart

Specifies an IP address of the SIP Server interface. This option must be set when deploying SIP Server on a host with multiple network interfaces. SIP Server uses this value to build the `Via` and the `Contact` headers in SIP messages. When this option is not set, SIP Server attempts to detect the IP address automatically.

sip-block-headers

Default Value: An empty string
Valid Values: A comma-separated list of the headers to be filtered out during `INVITE` message propagation
Changes Take Effect: Immediately

Specifies a way to filter out headers during `INVITE` message propagation. With an empty string, no headers will be filtered out.

sip-call-retain-timeout

Default Value: `1`
Valid Values: `0–3600`
Changes Take Effect: Immediately

Defines how long (in seconds) a T-Library call is kept in SIP Server's memory after all call parties (members of the call) are released. This option does not set any delay in generating an `EventReleased` message for a particular DN when the `BYE` message is received (or sent).

sip-dtmf-send-rtp

Default Value: `false`
Valid Values:

`true`	SIP Server instructs Stream Manager to send DTMF tones to all call participants using one or both of the following DTMF generation methods: RTP packets with Named Telephone Event (NTE) payload as specified by RFC 2833, and in-band audio tones according to ITU-T Recommendation Q.23.
`false`	The feature is disabled.

Changes Take Effect: Immediately
Related Feature: "DTMF Tones Generation" on page 137

Specifies whether SIP Server instructs Stream Manager to send DTMF tones when a T-Library client issues a `TSendDTMF` request.

sip-enable-call-info

Default Value: `false`
Valid Values: `true, false`
Changes Take Effect: Immediately
Related Feature: "Providing Call Participant Info" on page 164

When set to `true`, SIP Server distributes the information about call participants to logged-in agents by using the SIP `NOTIFY` method and `EventUserEvent` messages.

sip-enable-100rel

Default Value: `true`
Valid Values:

`true`	SIP Server advertises support for `100rel`, and requires it whenever the other side indicates support.
`false`	SIP Server does not negotiate support for the reliability of provisional responses.

Changes Take Effect: Immediately

Enables processing of `100rel` parameters and messages.

Note: SIP Server does not process reliable responses in the following scenarios:

- In third-party call control (`TMakeCall`) operations
- When re-`INVITE` methods are used
- When call routing is performed to any destination

sip-enable-moh

Default Value: `false`
Valid Value:

`true`	Music-on-hold is enabled.
`false`	Music-on-hold is disabled.

Changes Take Effect: At the next `Hold`/`THoldCall` operation

Enables or disables music-on-hold.

sip-enable-sdp-codec-filter

Default Value: `false`
Valid Values:

`true`	SIP Server modifies the SDP message body during SIP renegotiation.
`false`	SIP Server does not modify the SDP message body.

Changes Take Effect: Immediately

Specifies whether SIP Server modifies the SDP message body during SIP renegotiation. All codecs that are not in the list of values for the `audio-codecs` option are deleted from the SDP. As a result, all call center audio traffic is established based on the codecs listed in the `audio-codecs` option.

sip-hold-rfc3264

Default Value: `false`

Valid Value:

true	RFC3264-compliant implementation.
false	RFC2543-compliant implementation.

Changes Take Effect: Immediately

Specifies which implementation of hold media SDP is used by SIP Server for third-party call control (3pcc) hold operations.

Note: When this option is set at the DN level in the Annex tab > TServer section, it will overrides the Application level value.

sip-invite-timeout

Default Value: 0
Valid Values: 0–34
Changes Take Effect: Immediately

Specifies the number of seconds that SIP Server waits for a response to the INVITE message. The call times out if no response is received. When set to 0, or if a value is not specified, then the default SIP call timeout of 32 seconds is used.

sip-invite-treatment-timeout

Default Value: 0
Valid Values: 0–34
Changes Take Effect: Immediately

Specifies the number of seconds that SIP Server waits for a response to the INVITE message for a treatment (such as an announcement or music-on-hold). The call times out if no response is received. When set to 0, or if a value is not specified, then the default SIP call timeout of 32 seconds is used.

sip-port

Default Value: 5060
Valid Value: Any valid TCP/IP port
Changes Take Effect: After SIP Server restart

Specifies the port on which SIP Server listens for incoming SIP requests. The same port number is used for both TCP and UDP transports.

sip-refer-to-sst-enabled

Default Value: true
Valid Values:

true	The re-INVITE method is used instead of the REFER method for single-step transfer functionality.
false	The REFER method from the endpoint is not converted to a re-INVITE method and is also rejected by SIP Server if the endpoint has the refer-enabled option set to false.

Changes Take Effect: Immediately

Specifies whether a re-INVITE method is used instead of a REFER method for single-step transfer functionality. Use this option when the switch does not support the REFER method when performing a single-step transfer.

sip-retry-timeout

Default Value: 30
Valid Values: 1-3600
Changes Take Effect: Immediately

Specifies the time interval, in seconds, after which SIP Server initiates a new subscription if the previous SUBSCRIBE dialog is terminated.

sip-ring-tone-mode

Default Value: 0
Valid Values: 0,1
Changes Take Effect: Immediately

With the option set to 0, SIP Server connects Stream Manager to a call to play an audio ring tone. With the option set to 1, SIP Server waits for a response from the called device, and connects Stream Manager to a call to play an audio ring tone, only when the returned response cannot be used as the offer to a calling device.

Note: This option can be set at both the SIP Server Application level and at the Switch/DN level. The setting at the Switch/DN level takes precedence over the Application-level setting.

sip-treatments-continuous

Default Value: false
Valid Values:

true	A routing strategy treatment is played continuously played until the routing destination has answered the call.
false	A routing strategy treatment is not played continuously.

Changes Take Effect: Immediately for all new calls

Enables or disables a routing strategy treatment to be continuously played until the routing destination has answered the call.

subscription-timeout

Default Value: 180
Valid Values: 1–3600
Changes Take Effect: Immediately

Specifies the time interval (in seconds) in the Expire header of the 200 OK response message to a subscriber.

timed-acw-in-idle

Default Value: `true`
Valid Values: `true, false`
Changes Take Effect: Immediately
Related Feature: “Emulated Agents” on page 138

Specifies whether SIP Server applies the automatic wrap-up timer (using the `wrap-up-time` parameter) when an agent sends the `TAgentNotReady` request while in idle state.

When set to `false,` SIP Server does not automatically end manual wrap-up—the agent must return manually from ACW.

Note: For compatibility with the previous SIP Server releases, you can use the name `timed-cwk-in-idle` for this option as an alias.

unknown-bsns-calls

Default Value: `false`
Valid Values: `true, false`
Changes Take Effect: Immediately
Related Feature: “Business-Call Handling” on page 138

Determines whether SIP Server considers calls of unknown call type made from or to any agent as business calls.

untimed-wrap-up-value

Default Value: `1000`
Valid Value: Any nonzero positive integer
Changes Take Effect: Immediately
Related Feature: “Emulated Agents” on page 138

Specifies the threshold (in seconds) at which the timing of ACW stops and manual intervention is required (*untimed ACW*).

user-data-im-enabled

Default Value: `NULL`
Valid Values:

`page`	The `UserData` content in the IM is sent in page mode. The `MESSAGE` request(s) are exchanged in the endpoints without dialog. You must use this value for EyeBeam SIP endpoints
`session`	When the first SIP dialog containing an IM SDP is created, the `MESSAGE` request(s) are exchanged only in this dialog. You must use this value for Microsoft Office Communicator endpoints.

Changes Take Effect: Immediately
Related Feature: “Instant Messaging” on page 143

Enables the `UserData` content in the Instant Messaging (IM) for a DN.

user-data-im-format

Default Value: NULL
Valid Values:

text	The UserData content in the IM is encoded in text (text/plain) format. You must use this value for Microsoft Office Communicator endpoints
html	The UserData content in the IM is encoded in html (text/html) format. You must use this value for Eyebeam SIP endpoints.

Changes Take Effect: Immediately
Related Feature: "Instant Messaging" on page 143

Specifies the format of the UserData IM content when different SIP endpoints support different IM formats.

userdata-map-trans-prefix

Default Value: None
Valid Values: A string
Changes Take Effect: Immediately
Related Feature: "Mapping SIP Headers and SDP Messages" on page 149

Contains a transport prefix to indicate what headers in the SIP message carry the mapped UserData. SIP Server adds this prefix to all data mapped to the outgoing INVITE message. SIP Server scans incoming INVITE or REFER messages used to place a call on the Routing Point for headers that start with this prefix, in addition to performing the normal mapping procedure.

If this option is not specified, no prefix is added to the transmitted data.

wrap-up-time

Default Value: 0
Valid Value: Any positive integer, untimed

0	ACW is disabled.
Value greater than 0 but less than untimed-wrap-up-value	The number of seconds of timed ACW, after which SIP Server returns the agent to the Ready state.
Value equal to untimed-wrap-up-value	ACW is untimed and the agent must manually return to the Ready state.
Value greater than untimed-wrap-up-value	ACW is disabled.
untimed	ACW is untimed and the agent must manually return to the Ready state.

Changes Take Effect: Immediately
Related Feature: "Emulated Agents" on page 138
Specifies the amount of ACW wrap-up time allocated to emulated agents at the end of a business call.

This option can be set in a number of places, and SIP Server processes it in the following order of precedence, highest first. If the value is not present at the higher level, SIP Server checks the next level, and so on.

SIP Server option priority processing:

1. In the call, in user data `WrapUpTime` (limited to ISCC scenarios).
2. In a `DN` configuration object of type `Routing Point`, on the `Annex` tab in the `TServer` section.
3. In a `DN` configuration object of type `ACD Queue`, on the `Annex` tab in the `TServer` section.
4. In the `TAgentLogin` request, in attribute extension `WrapUpTime` (applies to this agent only).
5. In an `Agent Login` configuration object, on the `Annex` tab in the `TServer` section.
6. In a `DN` configuration object of type `Extension`, on the `Annex` tab in the `TServer` section.
7. In a `DN` configuration object of type `ACD Queue` or `Routing Point` that represents logged-in agents (`Agent Group`), on the `Annex` tab in the `TServer` section.
8. In the SIP Server `Application` object, on the `Options` tab in the `TServer` section.
9. While in ACW, in the `TAgentNotReady` request with `WorkMode=ACW` (Extending ACW), in attribute extension `WrapUpTime` (applies to this agent only).

UPDATE, INVITE, INFO, and REFER Sections

The option names in this section are a combination of the `TEvent` attribute name (`extensions` or `userdata`), a dash, and then a numeric value.

extensions-<n>

Default Value: None
Valid Values: See description
Changes Take Effect: Immediately
Related Feature: "Mapping SIP Headers and SDP Messages" on page 149

The `extension` prefix instructs SIP Server to put the SIP header (parameter) into the `Extensions` attribute.

The value determines which header within the header parameter of the SIP message is processed. The value of the header/header parameter is added as a key-value pair into the attribute as follows:

```
<header_name>=<header_value>
```

You can use the colon character to address the parameter name of a header. For example, the `extensions-8=From:tag` option puts the `From:tag=979E46B1-0FDF-`

418F-BA3F-C03F95A4D6E0-2 key-value pair into the Extensions attribute of the EventRouteRequest message.

userdata-<n>

Default Value: None
Valid Values: See description
Changes Take Effect: Immediately
Related Feature: "Mapping SIP Headers and SDP Messages" on page 149

The userdata prefix instructs SIP Server to put the SIP header (parameter) into the UserData attribute.

Agent Login–Level and DN-Level Options

You set configuration options described in this section in the TServer section on the Annex tab of the relevant Agent Login or DN object in Configuration Manager.

agent-greeting

Default Value: NULL
Valid Values: Any file name that will be played to the agent
Changes Take Effect: Immediately
Related Feature: "Personal Greeting" on page 160

Specifies the media file name that will be used as a greeting for the agent.

authenticate-requests

Default Value: None
Valid Values: register, invite
Changes Take Effect: Immediately

Determines if incoming SIP requests (REGISTER or INVITE) are treated with an authentication procedure when the following conditions are true:

- The name of the incoming SIP message exits in the list of the authenticate-requests parameter.
- The option password is configured on the same DN object.

If the authenticate-requests option is not configured on the DN, the REGISTER request still will be treated with an authentication procedure when the password configuration option is configured on the DN. If neither authenticate-requests nor password configuration options are configured on the DN, no requests will be authenticated.

auto-redirect-enabled

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

Enables automatic processing of the redirect (3XX) response from the Sonus PSX gateway.

The following events occur when this option is set to `true`:

- SIP Server sends an `INVITE` request to the Sonus PSX gateway.
- A `3XX` response containing a new Uniform Resource Identifier (URI) target is received.
- SIP Server sends the same `INVITE` request to the new target URI (the Sonus GSX gateway).

This option must be set in the `TServer` section of the `Annex` tab on the `Trunk` DN that represents the Sonus PSX gateway.

capacity

Default Value: `0`
Valid Values: Any positive integer
Changes Take Effect: Immediately

Specifies how many calls can be handled by a specific Voice over IP device represented in the SIP Server configuration as `Trunk.`

capacity-group

Default Value: `<DN name>`
Valid Values: Any non-empty string
Changes Take Effect: Immediately

Specifies the name of the `DN` object of type `Trunk,` as configured in the Configuration Layer, that represents a physical Voice over IP device. All DNs configured with the same `capacity-group` share the device capacity defined in the `capacity` option. See the following option configuration examples.

Note: The value of the `capacity` option must be defined in only one `Trunk` DN.

Example 1 DN of type `Trunk` with the Number `Cisco-8340`

```
[TServer]
capacity=200
prefix=8340
```

With these settings, the number of calls to the `Cisco-8340` trunk will be limited to 200.

Example 2

- DN of type `Trunk` with the Number `Cisco-8340`

```
[TServer]
capacity=200
capacity-group=Cisco-GW
prefix=8340
```

- DN of type `Trunk` with the Number `Cisco-8341`

```
[TServer]
capacity-group=Cisco-GW
prefix=8340
```

With these settings, the number of calls to the `Cisco-8340` and `Cisco-8341` trunks will be limited to 200.

contact

Default Value: None
Valid Values: Any alphanumerical string
Changes Take Effect: Immediately

Contains the contact URI, specifying the device's IP address, if this address is fixed. This option is necessary only for stand-alone configurations, and only if the configured device does not register itself in the SIP Server registrar. It is part of the persistent registrar feature.

For example, if the SIP device sends a `REGISTER` request to a SIP Server, and this request is accepted, SIP Server uses the contact information from the `REGISTER` request, and updates (or creates) in Configuration Manager the option `contact` in the `TServer` section of the `Annex` tab of the corresponding `DN` object.

The URI format is:

```
[sip:][number@]hostport[;transport={tcp/udp}]
```

Where:

- `sip:` is an optional prefix.
- `number` is the DN number. This is currently ignored.
- `hostport` is a `host:port` pair, where `host` is either a dotted IP address or a DNS-resolvable hostname for the endpoint.
- `transport=tcp` or `transport=udp` is used to select the network transport. The default value is `udp`.

customer-greeting

Default Value: `NULL`
Valid Values: Any file name that will be played to the customer
Changes Take Effect: Immediately
Related Feature: "Personal Greeting" on page 160

Specifies the media file name that will be used as a greeting for the customer. The customer greeting plays continuously until the agent greeting finishes playing. The `agent-greeting` and `customer-greeting` option values are used as follows:

- When both options contain different file name values, each file will be played to the customer and the agent as specified.
- When only one option contains a value, the same file will be played to both the customer and the agent.

- When neither option contains a value, no greeting will be played to either the customer or the agent.

cpn

Default Value: None
Valid Values: The SIP URI format according to the Augmented Backus-Naur Form (ABNF) in RFC 3261
Changes Take Effect: Immediately

When the value is specified for a DN of type `Trunk`, it will be used as the user part of the SIP URI in the `From` header of the `INVITE` message sent by SIP Server through this trunk. This option must not be configured on trunks that are allocated for direct signaling between SIP Servers.

Note: If the `CPNDigits` parameter is specified in the `Extensions` attribute in `TMakeCall`, `TMakePredictiveCall`, `TInitiateConference`, or `TInitiateTransfer` requests, it takes precedence over the `cpn` option setting.

default-dn

Default Value: `NULL`
Valid Values: Any valid DN
Changes Take Effect: Immediately

This option can be configured only on DNs of type `Routing Point`. Specifies the DN to which calls are sent when URS is nonoperational, or when the timeout specified in the `router-timeout` option expires. This option does not apply to calls that are delivered to an ACD Queue associated with the Routing Point.

Note: This option can be set at the SIP Server Application level and at the Switch/DN level. The setting at the Switch/DN level takes precedence over the Application level setting.

dual-dialog-enabled

Default Value: `true`
Valid Values:

`true`	Set the option to `true` for endpoints that accept more than one SIP dialog and provide remote CTI control by the `NOTIFY` message.
`false`	Set the option to `false` for endpoints that can only accept one active SIP dialog, or cannot provide remote CTI control by the `NOTIFY` message to answer, hold, or retrieve call operations.

Changes Take Effect: Immediately

Enables the SIP dialog functionality for making consultation calls, according to the endpoint type.

enable-agentlogin-presence

Default Value: `false`
Valid Values:

`true`	This value must be used in deployments where agent desktops are not used and all information about agent states is determined by presence subscription. In this environment, SIP Server controls the agent state based on SIP-level information. `EventAgentLogin` and `EventAgentReady` messages are generated if an endpoint registers with SIP Server using the `REGISTER` request, or if the endpoint submits the `PUBLISH` request with the presence content indicating an `open` status. If the endpoint terminates the SIP registration or submits the `PUBLISH` request indicating a `closed` status, then SIP Server generates `EventAgentLogout.` All TEvents are generated on behalf of the agent with the Agent ID set to the same value as the DN name for which all SIP messages are received.
`false`	This functionality is disabled.

Changes Take Effect: Immediately

Enables an agent login using presence notification. See “Presence Subscription” on page 161 for more information.

Note: You must enable the `subscribe-presence` option before enabling this option.

force-register

Default Value: `NULL`
Valid Values: Any SIP endpoint address
Changes Take Effect: Immediately

Enables trunk registration and used as the `From` header in the `REGISTER` request.

geo-location

Default Value: None
Valid Values: Any alphanumeric string
Changes Take Effect: Immediately

Table 26 describes the possible DNs that can use this option:

Table 26: DN Configuration Objects

Device Type	Genesys DN Type
Agent SIP endpoint	`Extension`
Media Gateway	`Trunk`
Music-on-hold or Music-in-queue server (such as Stream Manager)	`VoIP Service` with `service-type=music`
Voice Treatment Server (such as Stream Manager)	`VoIP Service` with `service-type=treatment`
Voice Recorder (such as Stream Manager)	`VoIP Service` with `service-type=recorder`
Multipoint Conference Unit (such as Stream Manager)	`VoIP Service` with `service-type=mcu`

Note: Virtual resources such as Routing Points or ACD Queues must not use this option.

make-call-rfc3725-flow

Default Value: `2`
Valid Values: `1, 2`
Changes Take Effect: Immediately

Controls which SIP call flow to choose when a call is initiated by a `TMakeCall` request. The specified value is equal to the call flow number as described in RFC 3725. Only flow 1 and flow 2 from RFC 3725 are currently supported.

Note: This option is enabled only when the option `refer-enabled` is set to `false` for that DN.

no-answer-action

Default Value: `none`
Valid Values:

`none`	SIP Server takes no action on agents when business calls are not answered.
`notready`	SIP Server sets agents to `NotReady` when business calls are not answered.
`logout`	SIP Server automatically logs out agents when business calls are not answered.

Changes Take Effect: Immediately
Related Feature: "No-Answer Supervision" on page 157

Determines the action taken on an agent if the agent receives a SIP Server business call but fails to answer the call within the time defined in option `agent-no-answer-timeout`.This option is defined on any `Agent Login` object. When set, the value overrides the global `agent-no-answer-action` SIP Server configuration option for that agent.

Note: If a call is abandoned before either `agent-no-answer-action` or `no-answer-timeout` or `supervised-route-timeout` expires (depending on which timer is applicable), SIP Server performs no action on this agent.

no-answer-overflow

Default Value: None
Valid Values:

`none`	SIP Server does not attempt to overflow a call on an agent desktop when `agent-no-answer-timeout` expires.
`recall`	SIP Server returns the call to the last distribution device (the device reported in the `ThisQueue` attribute of the call) when `agent-no-answer-timeout` expires.
`release`	SIP Server releases the call.
`default`	SIP Server stops execution of the current overflow sequence and continues with the SIP Server default overflow sequence, as defined by the relevant overflow option in the main SIP Server section.
Any valid overflow destination	SIP Server returns the call to the specified destination when the value set for the `agent-no-answer-timeout` option expires.

Changes Take Effect: Immediately
Related Feature: "No-Answer Supervision" on page 157

Defines a sequence of overflow destinations (separated by comma) in the order listed:

1. When the first overflow destination fails, SIP Server attempts the next one in the list.

2. When all overflow destinations in the list fail, SIP Server abandons overflow. When the list of overflow destinations contains the value `recall` and the call is not distributed, SIP Server skips to the next destination in the list.

This option is defined in the `Switches` folder on any of the following objects:

- `Agent Login`
- `DN` of type `Extension`
- `DN` of type `ACD Position`

When set, this option overrides any of the following global SIP Server configuration options for the object where it has been set (depending on configuration object type):

- `agent-no-answer-overflow` if defined for an `Agent Login` object.
- `extn-no-answer-overflow` if defined for a `DN` of type `Extension` object.
- `posn-no-answer-overflow` if defined for a `DN` of type `ACD Position` object.

no-answer-timeout

Default Value: Same as value in corresponding global option
Valid Value: Any integer from `0–600`
Changes Take Effect: Immediately
Related Feature: “No-Answer Supervision” on page 157

Defines the time (in seconds) that SIP Server waits for a call that is ringing on the device in question to be answered.

When the timer expires, SIP Server applies the appropriate overflow, and, in the case of agents, the appropriate `Logout` or `Not Ready` action.

This option is defined in the `Switches` folder on any of the following objects:

- `DN` of type `Extension`
- `DN` of type `ACD Position`
- `Agent Login` object

When set to `0`, the NoAnswer Supervision feature for this device is disabled. When set, this option overrides any of the following global SIP Server configuration options for the object where it has been set (depending on configuration object type):

- `agent-no-answer-timeout` if defined for an `Agent Login` object.
- `extn-no-answer-timeout` if defined for a `DN` of type `Extension` object.
- `posn-no-answer-timeout` if defined for a `DN` of type `ACD Position` object.

oos-check

Default Value: `0`
Valid Values: `0–300`
Changes Take Effect: Immediately
Related Feature: "Active Out-of-Service Detection" on page 93

Specifies how often (in seconds) SIP Server checks a device for out-of-service status. This option can be used in conjunction with the `oos-force` and `recovery-timeout` options, as follows:

- When no response is received, and the `oos-force` option is configured, SIP Server will mark a device as out of service when the `oos-force` timeout expires.
- When the `recovery-timeout` option setting is less than the `oos-check` timeout, SIP Server will wait the amount of time specified as the `recovery-timeout` value before checking the DN that was previously detected as out of service.
- When the `oos-check` option is set to `0`, the feature is disabled.

Note: This option is only supported on the following DN types:

- `Voice over IP Service`
- `Trunk`

The `oos-check` option is not applicable on internal DNs (DNs of type `Extension` or `ACD Position`).

oos-force

Default Value: `0`
Valid Values: `0–30`
Changes Take Effect: Immediately
Related Feature: "Active Out-of-Service Detection" on page 93

Specifies the time interval (in seconds) that SIP Server waits before placing a device that does not respond in out-of-service state when the `oos-check` option is enabled.

oosp-transfer-enabled

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

When set to `true`, SIP Server puts itself in the Out Of Signaling Path (OOSP) after the single-step transfer or routing to the external destination has been completed.

Note: This option is configured for `Trunk` DNs only, and the caller DN or the `Trunk` DN must support the `REFER` method.

out-rule-<n>

Default Value: No default value
Valid Value: Any valid string in the following format:
`in-pattern=<input pattern value>;out-pattern=<output pattern value>`
Changes Take Effect: Immediately
Related Feature: “Class of Service” on page 127

Defines a rule to be applied to an inbound number. The two parts of the option value describe the input and output patterns in the rule. When configuring the pattern values, follow the syntax defined in “Dialing Rule Format” on page 128. See “Examples” on page 132 for examples of these rules as well as detailed instructions for creating rules for your installation. For example, a value for this configuration option might look like this:

```
out-rule-01 = in-pattern=0111#CABBB*ccD;out-pattern=ABD
```

override-domain

Default Value: `NULL`
Valid Values: Any computer name
Changes Take Effect: Immediately

Enables an override of the specified computer name in the SIP `To:` header for a DN. It is used to contact a particular DN in a domain in the `To:` header that is different than the SIP Server internal registrar computer name.

Note: This option must be specified for the DN that represents Microsoft Office Communicator behind LCS.

override-domain-from

Default Value: `NULL`
Valid Values: Any computer name string
Changes Take Effect: Immediately

When set, SIP Server substitutes the computer name in the `URI` of the `From` headers with the value of this option when it sends the initial `INVITE` message to a DN or `Trunk` DN.

override-call-type

Default Value: `0`
Valid Values:

`0`	`CallTypeUnknown`
`1`	`CallTypeInternal`
`2`	`CallTypeInbound`
`3`	`CallTypeOutbound`

Changes Take Effect: Immediately

Determines the value SIP Server will use as the `CallType` attribute for internal calls made directly to a DN of type `Routing Point`. When set to `0,` SIP Server specifies the `CallType` attribute as `Internal.`

password

Default Value: None
Valid Values: Any alphanumerical string
Changes Take Effect: Immediately

In the endpoint configuration: Specifies the password for the SIP endpoint registration with the local registrar. If it is present, registration attempts are challenged and the password is verified. If it is not present, the registration is not challenged. The realm for password authentication is configured globally; there is one realm per SIP Server.

In the gateway configuration: Contains the password for gateway registration with the local registrar. This is used for incoming `REGISTER` requests, not for outgoing `INVITE` requests.

prefix

Default Value: None
Valid Values: Any alphanumerical string
Changes Take Effect: Immediately

In the MCU configuration: Specifies the starting digits of the number that are used when sending calls to MCU. The full number is built as: `<prefix><connid>@<ipaddr>:port`. Typically, MCU servers require a prefix consisting of digits in order to identify a type of conference (for example, voice only, voice and video, and so on). Set the value as `conf=` if Stream Manager is used as the MCU.

In the gateway configuration: Contains the initial digits of the number that must match a particular gateway for that gateway to be selected. If multiple gateways match a number, the gateway with the longest prefix is selected.

preview-interaction

Default Value: `false`
Valid Values:

`true`	The protocol is enabled.
`false`	The protocol is disabled.

Changes Take Effect: Immediately for all future calls
Related Feature: “Preview Interactions” on page 164

Determines if the `Preview Interaction` protocol is enabled when incoming calls are diverted from a Routing Point.

Note: This option works with the `preview-expired` and `forced-notready` options to determine what action to take when a desktop does not respond to a preview interaction before the time expires.

priority

Default Value: `0`
Valid Values: Any non-negative integer
Changes Take Effect: Immediately

Specifies the device priority for the device selection algorithm. A smaller value designates a higher priority. SIP Server will choose a device in round-robin fashion across all devices if more than one device with the same priority is configured. This option is used to control the device switchover during a failure, and to provide lowest-cost routing.

public-contact

Default Value: None
Valid Values: Any alphanumerical string
Changes Take Effect: Immediately

Contains the public `host:port` pair for a softswitch. This is the public IP address of the softswitch. SIP Server uses this address to fill the destination (`Refer-To`) address in `REFER` requests. On some switches, this is the same as the `contact` address; if this is the case, you do not need to specify this parameter.

Note: The `public-contact` option is only applicable to Alcatel 5020.

record

Default Value: `false`
Valid Values: `true, false`
Changes Take Effect: When the next call is established on the DN
Related Feature: "Call Recording" on page 102

When set to `true,` call recording begins automatically when the call is established on the DN. Call recording stops when the DN leaves the call.

recovery-timeout

Default Value: `0`
Valid Values: `0–86400 seconds`
Changes Take Effect: Immediately

Controls whether a device is taken out of service when an error is encountered, and for how long it is out-of-service. When set to `0,` automatic `out-of-service-on-error` functionality is disabled.

rfc-2976-dtmf

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

When this option is set to `true` in a particular DN (type of `Trunk` or `Extension`) configuration, SIP Server will send DTMF tones in the RFC 2976 format to that device using the `INFO` request method when an agent issues a `TSendDTMF` request.

Notes: If a `TSendDTMF` request contains a string with multiple digits (for example, `12345#`), SIP Server issues multiple `INFO` requests (one per digit).

If a `TSendDTMF` request contains a string with multiple digits, and there are unsupported DTMF tones in this string (for example, `123a67`), SIP Server still attempts to send the `INFO` request for each digit contained in the string, ignoring possible error responses from a gateway, and continuing to send subsequent digits.

refer-enabled

Default Value: `true`
Valid Values: `true, false`
Changes Take Effect: With the next new call on this DN

Specifies whether the `REFER` method is sent to an endpoint. When set to `true,` the `REFER` method is sent to:

- The call party that originates a `TMakeCall` request.
- The call party that initiates a consultation call.
- The call party that is transferred to another destination during a single-step transfer.

When set to `false,` SIP Server uses the re-`INVITE` method instead.

reinvite-requires-hold

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

When set to `true,` SIP Server enables the endpoint to be placed on hold by re-inviting it with a hold SDP.

Note: This option prevents an audio delay during 3pcc (third-party call control) conferencing with an RTC-based endpoint.

reject-call-incall

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

When set to `true`, a call attempt to a DN that is already on a call will be rejected, and the `Invalid Destination State (93)` error message will be generated.

reject-call-notready

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

When set to `true`, a call attempt to a DN at which an agent is in a `Logout`, `NotReady`, or `AfterCallWork` state will be rejected, and the `Invalid Destination State (93)` error message will be generated.

Note: The `reject-call-incall` and `reject-call-notready` options are applicable to the following T-Library requests:

- `TMakeCall`
- `TInitiateTransfer`
- `TInitiateConference`
- `TSingleStepTransfer`
- `TSingleStepConference`

replace-prefix

Default Value: `0`
Valid Values: Any non-negative integer
Changes Take Effect: Immediately

Contains the digits that are inserted in the DN instead of the prefix for the gateway. If this option value is absent, the number is not modified.

request-uri

Default Value: None
Valid Values: Any SIP URI
Changes Take Effect: Immediately

Specifies the value of the `Request-URI` address inside the `INVITE` message that is different from the address where the message will be sent. Any DN used by SIP Server must be configured with this option if SIP Server will be using it to place a call to a DMX application.

When used with DMX, set the `contact` field in the option configured for the DN as the IP address of the DMX and the `request-uri` option as actual address of the H.323 endpoint.

In video support configuration: Creates a template for specifying the source of the video stream as the value of the Request-URI parameter in the INVITE message:

annc@<stream_manager_hostport>;play=<file>

reuse-sdp-on-reinvite

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

When a call is routed to an endpoint, and this option is set to true in the destination endpoint configuration, SIP Server generates an offer by sending a re-INVITE message to the origination party (or to the MCU). When the origination party answers the offer, SIP Server sends the INVITE message with SDP information to the destination.

Note: The value must be set to true when using EyeBeam version 1.1.

ring-tone-on-make-call

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

Affects the TMakeCall request when using the re-INVITE procedure.

When set to true, SIP Server connects the caller with an audio ringtone from Stream Manager when the destination endpoint responds with a 180 Ringing message. In addition, the following options must also be configured for these scenarios:

- The calling DN initiates a TMakeCall request must be configured with the following options:
  - refer-enabled set to false (see page 240)
  - make-call-rfc3725-flow set to 1 (see page 233)
- The calling DN initiates a consultation call must be configured with the following option:
  - dual-dialog-enabled set to false (see page 232)

When the ring-tone-on-make-call option is set to false, there is no ring tone.

service-type

Default Value: None
Valid Values: Any string
Changes Take Effect: Immediately

Specifies the configured SIP device type or service (see Table 27). See Chapter 5, “SIP Devices Support,” on page 77 for more information on using this option.

Table 27: Service-Type Settings for SIP Devices

SIP Device Type or Service	Genesys DN Type	Service-Type Setting
Conference Server / MCU	Voice over IP Service	mcu
Softswitch	Voice over IP Service	softswitch
Music-on-Hold servers	Voice over IP Service	music or moh
Treatment service	Voice over IP Service	treatment
Recording service	Voice over IP Service	recorder
Application service	Voice over IP Service	application

sip-add-primary-call-id

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

When set to true, any TMakeCall request that is initiated for a DN already on call will start with an INVITE message that has a proprietary P-gcti-primary-call-id header. The value of this header is the CallID SIP attribute from the primary SIP call dialog for that DN.

sip-busy-type

Default Value: 0
Valid Values: 0, 1, 2

Changes Take Effect: Immediately

When this option is set to 0 (the default), a busy tone is always played. When this option is set to 1, a busy tone is played for a calling party only if a treatment is previously applied to a call or a call is originated by a 3pcc make call operation, and the refer-enabled option is set to false. Otherwise, the rejected response is sent back to the calling party. When this option is set to 2, a busy tone is not applied, and if SIP Server does not accept an INVITE session from a calling party, the rejected response is sent back to the calling party.

sip-cti-control

Default Value: None

Valid Values: `talk, hold`

`talk`	The `TAnswerCall` request is issued against the DN, which means that the call is answered remotely by a T-Library client. The SIP method `NOTIFY (event talk)` is used. Otherwise, the `TAnswerCall` request is not supported.
`hold`	The `THoldCall` request is processed by a `NOTIFY` (`event hold`) message. The `TRetrieveCall` request is processed by a `NOTIFY (event talk)` message.

Changes Take Effect: Immediately

Specifies the behavior of DN which represents a SIP endpoint which supports the BroadSoft SIP Extension Event Package.

Note: Both values are used simultaneously as a list of comma-separated values.

sip-ring-tone-mode

Default Value: `0`

Valid Values: `0`,`1`

Changes Take Effect: Immediately

With the option set to `0`, SIP Server connects Stream Manager to a call to play an audio ring tone. With the option set to `1`, SIP Server waits for a response from the called device, and connects Stream Manager to a call to play an audio ring tone, only when the returned response cannot be used as the offer to a calling device.

Note: This option can be set at both the SIP Server Application level and at the Switch/DN level. The setting at the Switch/DN level takes precedence over the Application-level setting.

sip-server-inter-trunk

Default Value: `false`

Valid Values: `true, false`

Changes Take Effect: Immediately

Related Feature: “Trunk Optimization for Multi-Site Transfers” on page 166

When set to `true`, depending on the scenario, SIP Server determines whether to complete the transfer operation using the `REFER` or `INVITE` request with the `Replaces` header.

straight-forward

Default Value: `false`
Valid Values:

`true`	SIP Server sends a new `INVITE` message directly to the location specified in the `Contact` header.
`false`	SIP Server resolves the `username` parameter for the `Contact` header using information from the DN configuration.

Changes Take Effect: Immediately

Describes how SIP Server responds to the `300` or `302` SIP messages received from an endpoint or a server.

Note: This option is required for GVP integration in Stand-Alone mode. `Trunk` DNs configured for use with Resource Manager and IP Communication Server must contain the value `true`.

subscription-id

Default Value: `NULL`
Valid Values: Any valid string
Changes Take Effect: Immediately

Enables the subscription for multiple DNs with one `Subscribe` message. The value must be same as the user part of the `Subscribe URI`.

Note: For integration with GVP, this value must be set to `GVP`.

subscribe-presence-domain

Default Value: `NULL`
Valid Values: Any valid computer name on the softswitch
Changes Take Effect: Immediately
Related Feature: "Presence Subscription" on page 161

Specifies the subscription domain information for the `Trunk` DN. This option value will be used with the DN name to form the `SUBSCRIBE` request `URI` and the `To:` header.

subscribe-presence-from

Default Value: `NULL`
Valid Values: Any valid SIP URI
Changes Take Effect: Immediately
Related Feature: "Presence Subscription" on page 161

Specifies the subscription endpoint information. This option value will be used to form the `From:` header in the `SUBSCRIBE` request to the softswitch.

Note: For softswitches such as Microsoft LCS and Asterisk, the username part of this SIP URI must not be configured in the softswitch.

subscribe-presence-expire

Default Value: NULL
Valid Values: Any valid positive integer
Changes Take Effect: Immediately
Related Feature: "Presence Subscription" on page 161

Specifies the subscription renewal interval (in seconds).

subscribe-presence

Default Value: NULL
Valid Values: publish, or the name of the Trunk DN representing the softswitch
Changes Take Effect: Immediately
Related Feature: "Presence Subscription" on page 161

Enables presence subscription and mapping of a presence state to an agent state.

- When set to publish, SIP Server uses presence updates from a PUBLISH SIP request sent by a SIP Endpoint, and maps the presence state from the PUBLISH request to the agent state.
- When set to the name of a Trunk DN that contains the subscription parameters is specified, the enable-agentlogin-presence option (see page 232) must also be configured for the same Trunk DN.

userdata-map-filter

Default Value: None
Valid Values:

*:	All data is mapped.
A list of prefixes	A comma-separated list of prefixes used to identify the UserData key-value pair to be mapped.

Changes Take Effect: Immediately
Related Feature: "Mapping SIP Headers and SDP Messages" on page 149

Specifies the names of the key-value pairs to be mapped. If this option is not specified, no data will be mapped.

use-register-for-service-state

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

Determines the Extension DN service state after it has been unregistered with SIP Server.

- When set to true, the DN is set to the Out of Service state in the following scenarios:
  - The SIP REGISTER request contains an Expires header value equal to 0.
  - The SIP registration timer has expired.

  An EventDNOutOfService message is generated to indicate that the DN is currently out of service.
- When set to false (or not configured), the DN service state is not set to the Out of Service state when it has been unregistered with SIP Server.

GVP Integration Options

This section describes a configuration option specific to the Genesys Voice Platform (GVP) functionality with SIP Server. Configure this option in the extrouter section on the Options tab for the SIP Server Application object in Configuration Manager.

handle-vsp

Default Value: no
Valid Values:

requests	The ISCC component of SIP Server will attempt to translate requests related to this DN before submitting them to the service provider.
events	The ISCC component of SIP Server will attempt to process events received from the service provider before distributing them to SIP Server clients.
all	The ISCC component of SIP Server will handle both the events and requests.
no	No processing will take place.

Changes Take Effect: Immediately

Specifies the way SIP Server will handle events from, and requests to, an external service provider registered for a DN using the AddressType attribute set to VSP.

Reserved Options

Warning! The options documented in this section are reserved for Genesys Engineering and their values cannot be changed.

Table 28: Reserved Configuration Options

Option Name	Option Section
accept-dn-type	Application level > TServer section
backup-mode	Application level > TServer section
call-max-outstanding	Application level > TServer section
clid-withheld-name	Application level > TServer section
correct-rqid	Application level > TServer section
default-dn-type	Application level > TServer section
dn-del-mode	Application level > TServer section
emulate-login	Application level > TServer section
enable-ims	Application level > TServer section
expire-call-tout	Application level > TServer section
ims-default-icid-prefix	Application level > TServer section
ims-default-icid-suffix	Application level > TServer section
ims-default-orig-ioi	Application level > TServer section
init-dnis-by-ruri	Application level > TServer section
kpl-interval	Application level > TServer section
kpl-loss-rate	Application level > TServer section
kpl-tolerance	Application level > TServer section
max-pred-req-delay	Application level > TServer section
nas-indication	Application level > TServer section
override-switch-acw	Application level > TServer section

Table 28: Reserved Configuration Options (Continued)

Option Name	Option Section
prd-dist-call-ans-time	Application level > TServer section
quiet-cleanup	Application level > TServer section
quiet-startup	Application level > TServer section
recall-no-answer-timeout	Application level > TServer section
reg-delay	Application level > link-control section
reg-interval	Application level > TServer section
reg-silent	Application level > link-control section
rq-expire-tmout	Application level > TServer section
sip-proxy-headers-enabled	Application level > TServer section
sync-emu-agent	Application level > TServer section
unknown-xfer-merge-udata	Application level > TServer section
use-display-name	Application level > TServer section
wrap-up-threshold	Application level > TServer section

Changes from 7.5 to 7.6

Table 29 lists the configuration options that:

- Are new or changed in the 7.6 release of SIP Server
- Have been added or changed since the most recent 7.5 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 29: Option Changes from Release 7.5 to 7.6

Option Name	Option Values	Type of Change	Details
Application Level > Options Tab > TServer Section			
alternate-call-enabled	true, false	New in 7.5, Obsolete in 7.6	Set to true by default.

Table 29: Option Changes from Release 7.5 to 7.6 (Continued)

Option Name	Option Values	Type of Change	Details
cpd-info-timeout	0-30	New in 7.5	See the option description on page 203.
delay-between-refresh-on-switchover	Any timeout value	New in 7.5	See the option description on page 204.
delay-to-start-refresh-on-switchover	Any timeout value	New in 7.5	See the option description on page 205.
enforce-external-domains	A list of computer names or IP addresses	New in 7.5	See the option description on page 207.
mwi-mode	REGISTER, SUBSCRIBE	New in 7.6	See the option description on page 215.
music-in-conference file	A path to a music file	New in 7.6	See the option description on page 213.
notrdy-bsns-cl-force-rdy	true, false	Obsolete in 7.6	
observing-routing-point	Any valid DN	New in 7.6	See the option description on page 216
parking-music	music/silence	New in 7.6	See the option description on page 217.
sip-block-headers	A comma-separated string	New in 7.5	See the option description on page 221.
sip-dtmf-send-rtp	true, false	New in 7.5	See the option description on page 221.
sip-enable-call-info	true, false	New in 7.6	See the option description on page 221.
sip-enforce-sdp-origin-rules		Obsolete in 7.6	
sip-server-inter-trunk	true, false	New in 7.6	See the option description on page 244.
sip-sync-local-contact		Obsolete in 7.6	
sip-sync-peer-contact		Obsolete in 7.6	
userdata-map-trans-prefix	A string	New in 7.6	See the option description on page 226.

Table 29: Option Changes from Release 7.5 to 7.6 (Continued)

Option Name	Option Values	Type of Change	Details
Application Level > Options Tab > extrouter Section			
default-network-call-id-matching	sip	New in 7.6	See the option description on page 361.
handle-vsp	request, events, all, no	New in 7.6	See the option description on page 247.
Application Level > Options Tab > UPDATE, INVITE, INFO, REFER Sections			
extensions-<n>	header/header parameter	New in 7.6	See the option description on page 227.
userdata-<n>	header/header parameter	New in 7.6	See the option description on page 228.
DN Level > Annex Tab > TServer Section			
capacity	Any positive integer	New in 7.6	See the option description on page 229.
capacity-group	A non-empty string	New in 7.6	See the option description on page 229.
cpn	SIP URI format	New in 7.6	See the option description on page 231.
default-dn	Any valid DN	New in 7.6	See the option description on page 231.
out-rule-<n>	A string	New in 7.6	See the option description on page 237.
override-call-type	0-4	New in 7.5	See the option description on page 237.
rfc-2976-dtmf	true, false	New in 7.5	See the option description on page 240.
reuse-sdp-on-reinvite	true, false	Modified in 7.6	The option functionality modified in 7.6. See the option description on page 242.
sip-busy-type	0, 1, 2	New in 7.6	See the option description on page 243.
userdata-map-filter	*: or a comma-separated string	New in 7.6	See the option description on page 246.

Part

2

Part Two: T-Server Common Functions and Procedures

Part Two of this *SIP Server Deployment Guide* familiarizes the reader with T-Server in general. It addresses architectural, functional, and procedural information common to all T-Servers.

The information in Part Two is divided into the following chapters:

- Chapter 9, “T-Server Fundamentals,” on page 255, describes T-Server, its place in the Framework 7 architecture, T-Server redundancy, and multi-site issues. It does not, however, provide configuration and installation information.
- Chapter 10, “Multi-Site Support,” on page 267, describes the variations available for T-Server implementations across geographical locations.
- Chapter 11, “Common Configuration Options,” on page 321, describes log configuration options common to all Genesys server applications.
- Chapter 12, “T-Server Common Configuration Options,” on page 343, describes configuration options common to all T-Server types including options for multi-site configuration.

New for All T-Servers in 7.6

Before looking at T-Server's place in Genesys solutions and in the architecture of the Genesys Framework, note the following general changes that have been implemented in the 7.6 release of T-Server:

- **ISCC Transaction Monitoring support.** This release of T-Server supports the ISCC Transaction Monitoring that allows T-Server clients to monitor ISCC transactions of the call data transfer between T-Servers in a multi-site environment. See "ISCC Transaction Monitoring Feature" on page 306 for details.
- **ANI information distribution control.** This release introduces a new configuration option that controls the distribution of the ANI information in `TEvent` messages. See "ani-distribution" on page 344 for details.
- **Enhancement of use-data-from configuration option.** This option now includes the new valid value `active-data-original-call`. See "use-data-from" on page 354 for details.
- **Enhanced agent session ID reporting.** T-Server now generates and reports a session ID associated with each new agent login (key `AgentSessionID` in `AttributeExtensions`) in agent-state events (`EventAgentLogin`, `EventAgentLogout`, `EventAgentReady`, and `EventAgentNotReady`), and also in the `EventRegistered` and `EventAddressInfo` messages for resynchronization. The agent session IDs are not synchronized with a backup T-Server and new agent session IDs will be assigned to existing agent sessions after a T-Server switchover. See the T-Server client's documentation for agent session ID reporting. Refer to the *Genesys 7 Events and Models Reference Manual* and/or *Voice Platform SDK 7.6 .NET* (or *Java) API Reference* for details on the key `AgentSessionID` in `AttributeExtensions`.
- **Client-side port definition support.** This release of T-Server supports a new security feature that allows a client application to define its connection parameters before connecting to the server application. Refer to the *Genesys 7.6 Security Deployment Guide* for details.

Notes:

- Configuration option changes common to all T-Servers are described in "Changes from Release 7.5 to 7.6" on page 366.
- For information about the new features that are available in your T-Server in the initial 7.6 release, see Part Two of this document.

Chapter

9 T-Server Fundamentals

This chapter presents overview information that applies to all T-Servers (and Network T-Servers) and their deployment. This chapter is divided into the following sections:

Learning About T-Server

The *Framework 7.6 Deployment Guide* provides you with a high-level introduction to the role that T-Server plays in the Genesys Framework. If you have already looked through that guide, you may recall that T-Server is the most important component of the Framework Media Layer (the other two components are Load Distribution Server (LDS) and HA Proxy). The Media Layer enables Genesys solutions to communicate with various media, including traditional telephony systems, voice over IP (VoIP), e-mail, and the Web. This layer also provides the mechanism for distributing interaction-related business data, also referred to as *attached data,* within and across solutions.

Framework and Media Layer Architecture

Figure 28 illustrates the position Framework holds in a Genesys solution.

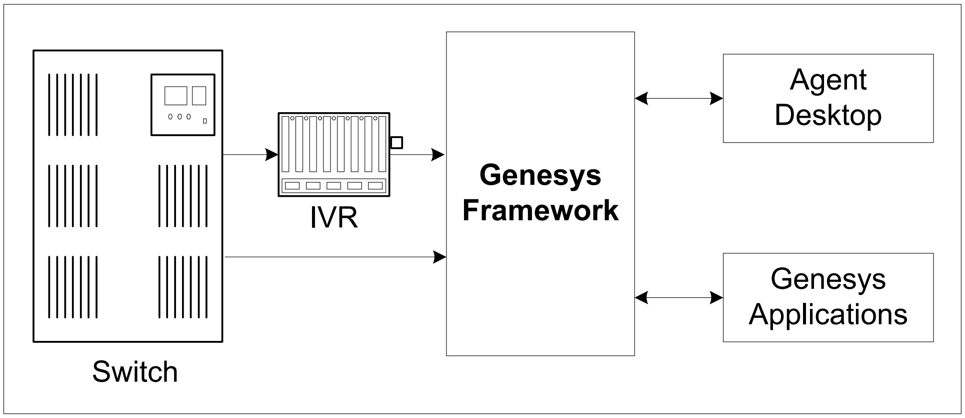


Figure 28: Framework in a Genesys Solution

Moving a bit deeper, Figure 29 presents the various layers of the Framework architecture.

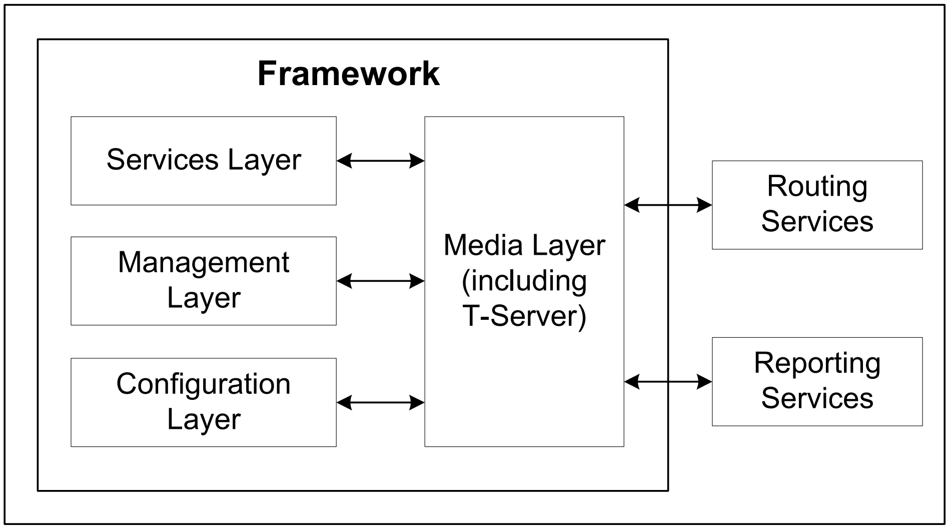


Figure 29: The Media Layer in the Framework Architecture

T-Server is the heart of the Media Layer—translating the information of the media-device realm into information that Genesys solutions can use. It enables your contact center to handle the computer-based form of the interactions that arrive and it translates the information surrounding a customer contact into reportable and actionable data.

Figure 30 presents the generalized architecture of the Media Layer.

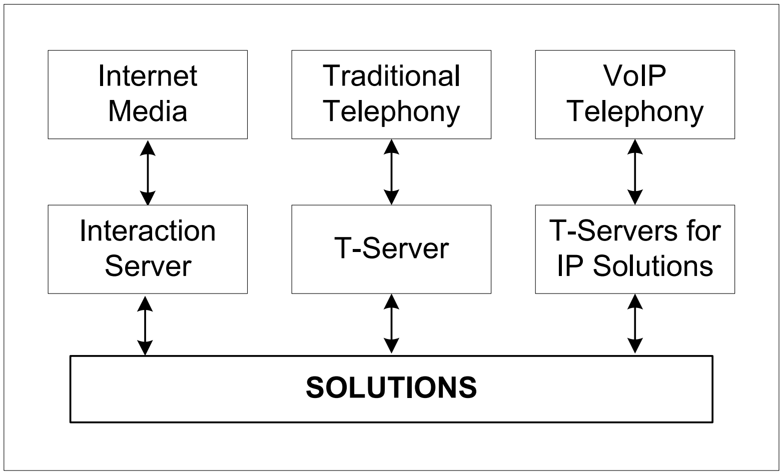


Figure 30: Media Layer Architecture

In addition to being the most important component of the Media Layer, T-Server plays the most significant role in making information about telephony traffic and its data available to Framework as a whole.

One or more components in practically every solution are T-Server clients. Solutions comprise a number of different Genesys software packages, from collections of components for various types of routing to those that allow for outbound dialing to still others. Framework in general, and T-Server in particular, enable these solutions to function in your enterprise.

T-Server has several typical clients: Stat Server, Call Concentrator, Universal Routing Server, and agent desktop applications. T-Server gets the information it needs about the enterprise from Configuration Server. Additionally, if you use the Management Layer, T-Server provides its ongoing status and various other log messages to server components of the Management Layer (for instance, allowing you to set alarms).

T-Server Requests and Events

This section outlines the roles that T-Server plays in a contact center. While it is possible to describe roles for all T-Servers, at a detailed level, T-Server's functionality depends on the hardware to which it is connected. (For example, when connected to a traditional switch, it performs CTI functions, but when connected to a VOIP-based telephony device, it controls IP traffic.) The CTI connection is only for the switch.

Details of T-Server Functionality

T-Server is a TCP/IP server that enables intelligent communication between media-specific protocols (such as the various CTI protocols, including CSTA and ASAI) and TCP/IP-based clients of T-Server. Applications that are clients to T-Server use the T-Library format to transmit requests to T-Server through a TCP/IP socket. T-Server can then either translate those requests to CTI protocol for switch use or relay them directly to other TCP/IP clients.

T-Server performs three general functions in the contact center: Bridging, Messaging, and Interaction Tracking.

Bridging

T-Server acts as a platform-independent interface between media devices and business applications. In the case of a telephony device, for instance, it receives messages from and sends commands to the telephony equipment using either CTI links provided by the switch manufacturer or interface protocols provided by telephony network vendors.

On the client-application end, T-Server offers three models (call model, agent model, and device model) unified for all switches. The core functionality (such as processing an inbound call, an agent login, or a call-forwarding request) translates into a unified application programming interface (API) called T-Library, so that applications do not need to know what specific switch model they are dealing with. On the other hand, T-Library accommodates many functions that are unique to a specific switch, so that client applications are able to derive the maximum functionality offered by a particular switch.

Refer to the *Genesys 7 Events and Models Reference Manual* for complete information on all T-Server events and call models and to the `TServer.Requests` portion of the *Voice Platform SDK 7.6 .NET* (or *Java*) *API Reference* for technical details of T-Library functions.

Messaging

In addition to translating requests and events for the client application involved in an interaction, T-Server:

- Provides a subscription mechanism that applications can use to receive notifications about interaction-related and non-interaction-related events within the contact center.
- Broadcasts messages of major importance (such as a notification that the link is down) to all clients.
- Broadcasts messages originated by a T-Server client to other T-Server clients.

The subscription mechanism consists of two parts, the DN subscription and event-type masking. Applications must register for a DN or a set of DNs to receive notifications about all events that occur in association with each registered DN. For example, when two softphone applications are registered for the same DN, and the first application initiates a call from the DN, T-Server notifies both applications that the call is initiated from the DN.

Client applications can also specify one or more types of events, and T-Server will filter out events of the non-specified types and only send events of the requested types. For example, if agent supervisors are interested in receiving agent-related events, such as AgentLogin and AgentLogout, they have to mask EventAgentLogin and EventAgentLogout, provided that a particular T-Server supports these events.

The combination of each client's subscription for DNs and masking of event types defines what messages T-Server distributes to what client.

Interaction Tracking

T-Server maintains call information for the life of the call (or other T-Server-supported media type) and enables client applications to attach user data to the call. Call information includes:

- A unique identifier, connection ID, that T-Server assigns when creating the call.
- Automatic Number Identification (ANI) and Dialed Number Identification Service (DNIS), if reported by the CTI link.
- User data that a client application (such as an Interactive Voice Response unit or Genesys Universal Routing Server) provides.

Difference and Likeness Across T-Servers

Although Figure 30 on page 257 (and other figures) depicts T-Server that works with telephony systems as a single product, this is a simplification. Because almost every traditional telephony device has its own characteristics and communication protocols, Genesys makes different T-Servers for different telephony systems. (That means T-Server you have will not work with another switch.) Thus, all T-Servers play a common role in the architecture, but their specific features differ from implementation to implementation, based on the media device in use.

Despite their switch-based differences, T-Servers for telephony systems are similar to one another in at least one important respect: they are all built with a certain amount of shared software code. This shared code is rolled into a single unit and is called T-Server Common Part (TSCP). TSCP is the central, common component for all T-Servers and has its own Release Note, which is accessible via a hyperlink from your T-Server's Release Note.

Note: This document separates common-code features based on TSCP into separate sections and chapters, such as the "T-Server Common Configuration Options" chapter. These are the options for all T-Servers that TSCP makes available for configuration.

T-Server Functional Steps During a Sample Call

The following example, Figure 31, outlines some basic steps that T-Server might take when a call arrives from outside the contact center. In this scenario, T-Server starts tracking the call even before it is delivered to the agent. T-Server then informs the selected agent that a call has arrived. When the switch delivers the call to the agent's extension, T-Server presents account information, collected at an Interactive Voice Response (IVR) unit, to the agent at the agent desktop application.

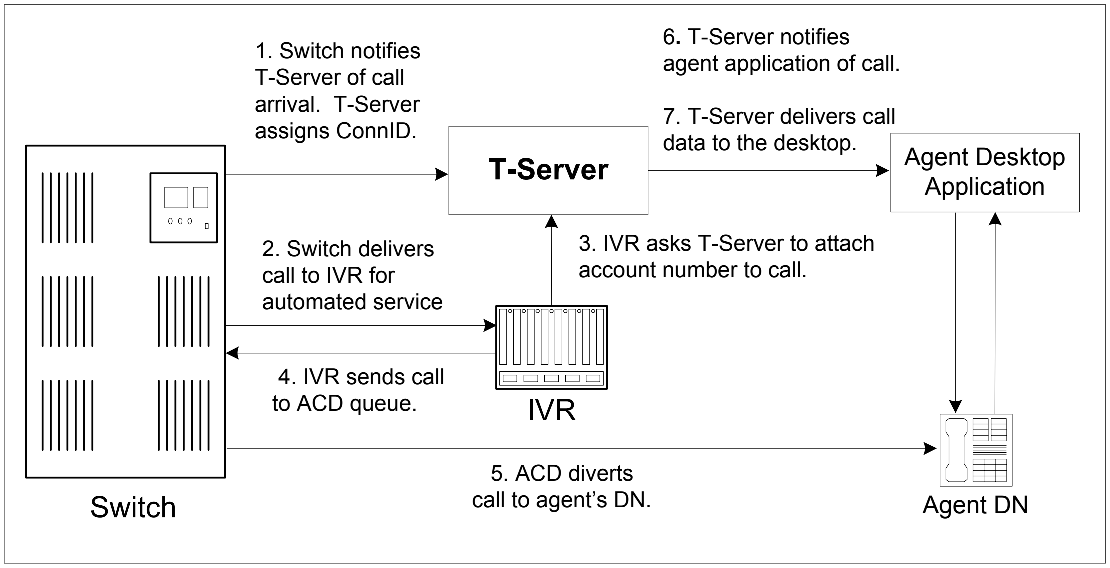

1. Switch notifies T-Server of call arrival. T-Server assigns ConnID.
6. T-Server notifies agent application of call.
7. T-Server delivers call data to the desktop.
T-Server
Agent Desktop Application
2. Switch delivers call to IVR for automated service
3. IVR asks T-Server to attach account number to call.
4. IVR sends call to ACD queue.
IVR
5. ACD diverts call to agent's DN.
Switch
Agent DN

Figure 31: Functional T-Server Steps

Step 1

When the call arrives at the switch, T-Server creates a call in its internal structure. T-Server assigns the call a unique identifier, connection ID.

Step 2

The switch delivers the call to an Interactive Voice Response (IVR) unit, which begins automated interactions with the caller.

Step 3

IVR acquires user information from the caller through prompts and requests T-Server to attach that information to the call. T-Server updates the call with the user information.

Step 4

IVR sends the call to an ACD (Automated Call Distribution) queue.

Step 5

The ACD unit distributes the call to an available agent logged in to a particular DN (directory number).

Step 6

T-Server notifies the agent desktop application that the call is ringing on the agent's DN. The notification event contains call data including `ANI`, `DNIS`, and account information that the IVR has collected.

Step 7

The agent desktop application presents the account information, including the name of the person whose account this is, on the agent's screen, so that the agent answering the call has all the relevant information.

These seven steps illustrate just a small part of T-Server's bridging, messaging, and interaction-processing capabilities.

Advanced Disconnect Detection Protocol

Since the 6.0 release of T-Server, the Advanced Disconnect Detection Protocol (ADDP) has replaced the Keep-Alive Protocol (KPL) as the method to detect failures for certain T-Server connections, including connections between two T-Servers and between a T-Server and its clients.

Notes:

- Starting with release 7.5, the KPL backward-compatibility feature is no longer supported.
- ADDP applies only to connections between Genesys software components.

With ADDP, protocol activation and initialization is made on the client's side and you can change these parameters. No additional messages are sent when there is existing activity over the connection. T-Server client applications and the remote T-Server (if any) must be listening to the socket and respond promptly to the polling signal for the connection to be preserved.

If you are going to enable ADDP, you must do it using the `protocol`, `addp-timeout`, `addp-remote-timeout`, and `addp-trace` configuration options. When configuring a timeout, consider the following issues:

- The configured timeout must be at least twice as long as the maximum network latency.
- There may be an interval when T-Server does not check for network activity.

- If the link connection fails but the client is not notified (for example, because the host is turned off, or because a network cable is unplugged), the maximum reaction time to a link-connection failure is equal to double the configured timeout plus the established network latency.

Also keep in mind that the T-Server receiving the polling signal may not respond immediately, and that a delay occurs between the polling signal and the response to travel from one T-Server to another. If you don't account for these contingencies when configuring a timeout, the connection that ADDP is monitoring will be dropped periodically.

Redundant T-Servers

T-Servers can operate in a high-availability (HA) configuration, providing you with redundant systems. The basics of each T-Server's redundant capabilities differ from T-Server to T-Server. One basic principle of redundant T-Servers is the standby redundancy type, which dictates how quickly a backup T-Server steps in when the primary T-Server goes down.

The Framework Management Layer currently supports two types of redundant configurations: `warm standby` and `hot standby`. All T-Servers offer the `warm standby` redundancy type and, starting with release 7.1, the `hot standby` redundancy type is implemented in T-Servers for most types of switches. (See Table 30.)

Instructions for configuring T-Server redundancy are available in Chapter 3, "High-Availability Configuration and Installation." Specifics on your T-Server's HA capabilities are outlined in Part Two of this document.

Notes:

- Network T-Servers use a load-sharing redundancy schema instead of `warm` or `hot standby`. Specifics on your T-Server's HA capabilities are discussed in Part Two of this document.
- IVR Server does not support simultaneous configuration of both Load Balancing functionality and `warm standby`. Only one of these is supported at a time.

Support for Hot Standby Redundancy in Various T-Servers

Use Table 30 to determine whether your T-Server supports the `hot standby` redundancy type. The table also indicates whether HA Proxy components are required for this support, and, if so, how many are required per pair of redundant T-Servers (or per link if so noted).

Table 30 only summarizes `hot standby` redundancy support in various T-Servers. For detailed, up-to-date information on the subject, see the *Genesys Supported Media Interfaces* white paper located on the Technical Support website at `http://genesyslab.com/support/dl/retrieve/default.asp?item=A9CB309AF4DEB8127C5640A3C32445A7&view=item`.

Table 30: T-Server Support of the Hot Standby Redundancy Type

T-Server Type	Hot Standby Supported	HA Proxy Required	Number of HA Proxy Components
Alcatel A4200/OXO	Yes	No	—
Alcatel A4400/OXE	Yes	No	—
Aspect ACD	Yes	No	—
Avaya Communication Manager	Yes	No[a]	—
Avaya INDeX	Yes	No	—
Cisco CallManager	Yes	No	—
DataVoice Dharma	Yes	No	—
Digitro AXS/20	Yes	No	—
EADS Intecom M6880	Yes	No	—
EADS Telecom M6500	Yes	No	—
eOn eQueue	Yes	No	—
Ericsson MD110	Yes	No	—
Fujitsu F9600	Yes	No	—
Huawei C&C08	Yes	No	—
Mitel SX-2000/MN-3300	Yes	No	—
NEC NEAX/APEX	Yes	No	—
Nortel Communication Server 2000/2100	Yes	Yes[b], No[c]	1 per link
Nortel Communication Server 1000 with SCCS/MLS	Yes	No	—
Philips Sopho iS3000	Yes	No[d]	1
Radvision iContact	No	—	—
Rockwell Spectrum	Yes	No	—

Table 30: T-Server Support of the Hot Standby Redundancy Type (Continued)

T-Server Type	Hot Standby Supported	HA Proxy Required	Number of HA Proxy Components
Samsung IP-PCX IAP	Yes	No	—
Siemens Hicom 300/HiPath 4000 CSTA I	Yes	No	—
Siemens HiPath 3000	Yes	No	—
Siemens HiPath 4000 CSTA III	Yes	No	—
Siemens HiPath DX	Yes	No	—
SIP Server	Yes	No	—
Tadiran Coral	Yes	No	—
Teltronics 20-20	Yes	Yes	1
Tenovis Integral 33/55	Yes	No	—
Network T-Servers[e]			
AT&T	No	—	—
Concert	No	—	—
CRSP	No	—	—
DTAG	No	—	—
GenSpec	No	—	—
ISCP	No	—	—
IVR Server, using network configuration	No	—	—
KPN	No	—	—
MCI	No	—	—
NGSN	No	—	—
Network SIP Server	No	—	—
Sprint	No	—	—
SR3511	No	—	—
Stentor	No	—	—

a. With release 7.1, T-Server for Avaya Communication Manager no longer uses HA Proxy for its support of `hot standby`. Earlier releases of this T-Server require two HA Proxies (for which there is a Configuration Wizard) to support `hot standby`.

b. For T-Server for Nortel Communication Server 2000/2100 in high-availability (`hot standby`) configuration, Genesys recommends that you use link version SCAI14 or above with call-progress and noncontroller-released messages enabled. See the switch-specific information in Part 2 of this *Deployment Guide* for additional information on HA configurations.

c. Starting with release 7.5, T-Server for Nortel Communication Server 2000/2100 supports HA without HA Proxy when operating in Dual CTI Links mode. See the switch-specific information in Part 2 of this *Deployment Guide* for additional information on HA configurations.

d. Starting with release 6.5.3, T-Server for Philips Sopho iS3000 supports HA both with and without HA Proxy.

e. Although they do not support high availability per se, Network T-Servers do support a load-sharing schema.

Multi-Site Support

Multi-site configuration implies the existence of two or more switches that belong to the same enterprise or service provider, and that share the Genesys Configuration Database. (In some cases this may include isolated partitions on a given switch served by different T-Servers.) The main goal of T-Server support for multi-site operations is to maintain critical information about a call as it travels from one switch to another.

For instructions on installing and configuring a multi-site environment, including information on the Inter Server Call Control (ISCC) features, please see Chapter 10, “Multi-Site Support,” on page 267.

Agent Reservation

T-Server provides support for clients to invoke the agent reservation function, `TReserveAgent()`. This function allows a server application that is a client of T-Server to reserve a DN along with an agent, a `Place`, or both, so that no other T-Server client can route calls to it during a specified reservation interval. Alternatively, when clients use the ISCC feature (see “ISCC Call Data Transfer Service” on page 269), they can use an agent reservation embedded in an ISCC request. (To do so, clients have to specify a certain `Extensions` attribute in an ISCC request when initiating an ISCC transaction. See page 274 for the list of ISCC requests.)

The reservation does not currently prevent the reserved objects from receiving direct calls or calls distributed from ACD Queues; agent reservation is intended as a way of synchronizing the operation of several clients. See `RequestReserveAgent` in the *Voice Platform SDK 7.6 .NET* (or *Java) API Reference* for more details on this function from the client’s point of view.

In addition to invoking the TReserveAgent function, you can customize the Agent Reservation feature by configuring options in the T-Server `Application` object. See“Agent-Reservation Section” on page 351 in the “T-Server Common Configuration Options” chapter in Part Two for more details.

Client Connections

The number of connections T-Server can accept from its clients depend on the operating system that T-Server runs. Table 31 illustrates the number of client connections that T-Server support.

Table 31: Number of T-Server’s Client Connections

Operating System	Number of Connections
AIX 32-bit and 64-bit modes (versions 5.1, 5.2, 5.3)	32767
HP-UX 32-bit and 64-bit modes (versions 11.0, 11.11, 11i v2)	2048
Linux 32-bit mode (versions RHEL 3.0, RHEL 4.0)	32768
Solaris 32-bit mode (versions 2.7, 8, 9)	4096
Solaris 64-bit mode (versions 2.7, 8, 9, 10)	65536
Tru64 UNIX (versions 4.0F, 5.1, 5.1B)	4096
Windows Server 2003	4096

Chapter

10 Multi-Site Support

This chapter contains general information about multi-site environments, as well as information on deploying a multi-site environment for your T-Server.

This chapter is divided into the following sections:

Note: Each switch/T-Server combination offers different multi-site options. For details describing your specific switch/T-Server environment, refer to Chapter 12, "T-Server Common Configuration Options," on page 343.

The following instructions apply to both local and remote switches and T-Servers. Because different vendor switches can be installed at the local and remote locations, this chapter covers several, but not all, possible configurations. To help determine which sections of this chapter apply to your situation, refer to Table 32 on page 283 and Table 33 on page 288.

For more information on your specific switch/T-Server environment, refer to the multi-site topics in Part Two of this guide.

Multi-Site Fundamentals

A multi-site configuration has two or more switches that belong to the same enterprise or service provider and that share the Genesys Configuration Database. (In some cases, this may include isolated partitions on a given switch served by different T-Servers.) The main goal of T-Server support for multi-site operations is to maintain critical information about a call as it travels from one switch to another.

T-Server supports multi-site operations using its *Inter Server Call Control* (ISCC; formerly called External Routing), which supports the following functions:

- **Call matching**—To link instances of a call distributed across multiple sites and to re-attach essential data associated with the call (ConnID, UserData, call history). The following T-Server features support this capability:
  - ISCC Call Data Transfer Service (active external routing)—when requested by a T-Server client by specifying the desired destination in the Location parameter, and also with various ISCC strategies performed by direct dial or by using the Transfer Connect Service. See "ISCC Transaction Types" on page 274 and "Transfer Connect Service Feature" on page 286.
  - Inter Server Call Control/Call Overflow (ISCC/COF) feature (passive external routing)—applicable when calls are overflowed to another site either directly or manually (see page 287).
  - Number Translation feature (see page 291).
  - Network Attended Transfer/Conference (NAT/C) feature (see page 299).

Note: When ISCC detects call instance reappearance on a given site, the call is assigned a unique ConnID and the user data is synchronized with the previous call instances. This ensures that ConnIDs assigned to different instances of the same call on a given site are unique.

- **Call data synchronization between associated call instances (ISCC Event Propagation)**—To provide the most current data to call instances residing on remote T-Servers. The following T-Server features support this capability:
  - User Data propagation (see page 301)
  - Party Events propagation (see page 303)

Note: ISCC automatically detects topology loops and prevents continuous updates.

Note: In distributed networks, Genesys recommends using call flows that prevent multiple reappearances of the same call instance, and call topology loops. This approach ensures that all T-Servers involved with the call report the same `ConnID`, and also optimizes telephony trunk allocation (that is, it prevents trunk tromboning).

The T-Server configuration contains information about other T-Servers with which it will communicate. T-Server uses this information to connect with the other T-Servers. During this "handshake" process, T-Servers exchange information about the following parameters:

- Protocol type
- Switch type
- Server name
- Location name (switch name)
- T-Server role (primary or backup)

To complete the handshake process, T-Servers exchange messages about the current condition of the links to their switches. After the handshake process is complete, T-Server is ready to support a multi-site operation.

ISCC Call Data Transfer Service

Because ISCC supports active external routing, T-Servers that serve different switches (usually on different sites) can exchange call data when a call is passed from one switch to another. With this functionality, T-Server provides its clients with the following additional information about each call received from another switch:

- The `ConnID` of the call
- Updates to user data attached to the call at the previous site
- Call history

Note: Load-sharing IVR Servers and Network T-Servers cannot be designated as the destination location for ISCC.

Figure 32 shows the steps that occur during a typical external routing (ISCC) transaction. Note that the location where a call is initially processed is called the *origination location,* and the location to which the call is passed is called the *destination location.*

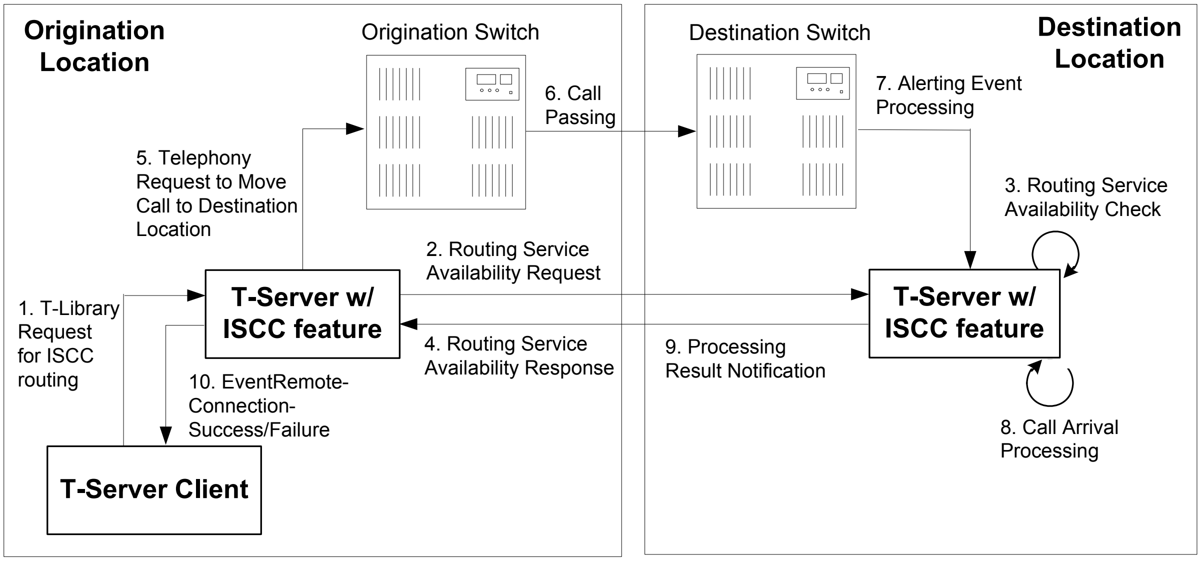

Origination Location
Origination Switch
Destination Switch
Destination Location
6. Call Passing
7. Alerting Event Processing
5. Telephony Request to Move Call to Destination Location
3. Routing Service Availability Check
2. Routing Service Availability Request
1. T-Library Request for ISCC routing
T-Server w/ ISCC feature
T-Server w/ ISCC feature
4. Routing Service Availability Response
9. Processing Result Notification
10. EventRemote-Connection-Success/Failure
8. Call Arrival Processing
T-Server Client

Figure 32: Steps in the ISCC Process

ISCC Call Flow

The following section identifies the steps (shown in Figure 32) that occur during an ISCC transfer of a call.

Step 1

A client connected to the T-Server at the origination location requests this T-Server to pass a call with call data to another location. For this purpose, the client must specify the `location` parameter (Attribute `Location`) when calling a corresponding T-Library function. ISCC processes the following T-Library requests:

- `TInitiateConference`
- `TInitiateTransfer`
- `TMakeCall`
- `TMuteTransfer`
- `TRouteCall`
- `TSingleStepTransfer`

Step 2

Upon receiving a client's request, the origination T-Server checks that the:

1. Connection to the destination T-Server is configured in the origination T-Server `Properties` dialog box.
2. The connection to the destination T-Server is active.
3. The destination T-Server is connected to its link.

4. The origination T-Server is connected to its link.

If these four conditions are met, the origination T-Server determines the transaction type that will be used for passing call data to another location in this transaction. The following possibilities exist:

- The client can request what *ISCC transaction type* (or simply *transaction type*) to use by specifying an appropriate key-value pair in the `Extensions` attribute of the request. The key-value pair must have a key equal to `iscc-xaction-type` and either an integer value as specified in the `TXRouteType` enumeration (see the *Voice Platform SDK 7.6 .NET (*or *Java) API Reference*) or a string value equal to one of the following: `default`, `route`, `direct` (or `direct-callid`), `direct-network-callid`, `direct-notoken`, `direct-ani`, `direct-uui`, `direct-digits`, `reroute`, `dnis-pool`, `pullback`, or `route-uui`.
- If the client does not specify the transaction type in the request or specifies the `default` transaction type, T-Server checks the Switch configuration for the transaction type configured in the `Access Code` (or `Default Access Code`) properties:
    - If the `Route Type` property of the `Access Code` is set to any value other than `default`, T-Server uses the specified value as the transaction type.
    - If the `Route Type` property of the `Access Code` is set to the `default` value, T-Server uses the first value from the list specified in the `cast-type` configuration option configured for the destination T-Server. If no value has been specified for the `cast-type` option, the default value of `route` is used as the transaction type.

Note: See “Switches and Access Codes” on page 308 for more information on Access Codes and Default Access Codes.

After the origination T-Server determines the requested transaction type, it determines if the destination T-Server supports this transaction type.

You must list the transaction types T-Server supports in the `cast-type` configuration option.

The origination T-Server issues a request for routing service availability and sends it to the destination T-Server. The T-Server request contains data that should be passed along with the call to the destination location. This data includes the transaction type, `ConnID`, `UserData`, and `CallHistory`.

The timer specified by the `request-tout` configuration option is set when the origination T-Server sends the request. If either the specified timeout expires or the call is abandoned before the origination T-Server receives a response from the destination T-Server, the operation is considered failed. In this scenario, the origination T-Server:

1. Generates a request to the destination T-Server to cancel the request for routing service.
2. Sends `EventError` to the client that requested the service.

3. Deletes information about the request.

Step 3

The destination T-Server receives the request for routing service availability and checks the requested type of routing. Depending on the ISCC transaction type, it stores the request information and, when appropriate, allocates access resources for the coming call. For example, an External Routing Point is allocated when the transaction type is `route`, and a DNIS number is allocated when the transaction type is `dnis-pool`.

Note: The `resource-allocation-mode` and `resource-load-maximum` configuration options determine how resources are allocated. Refer to Chapter 12, "T-Server Common Configuration Options," on page 343 for option descriptions.

If resources are unavailable, the request is queued at the destination location until a resource is free or the origination T-Server cancels the request. If the request is canceled, the destination T-Server deletes all information about the request.

If resources are unavailable because of incorrect configuration, the destination T-Server returns an error event to the origination T-Server.

Step 4

If resources are available, the destination T-Server generates a positive response and the timer is started for the interval specified by the `timeout` configuration option of the destination T-Server.

Step 5

If the origination T-Server receives a negative response, it sends an `EventError` message to the client and clears all data about the request.

If the origination T-Server receives the confirmation about routing service availability, it processes the client's request and sends a corresponding message to the switch. The timer on the origination T-Server is also started for the interval specified by the `timeout` configuration option of the destination T-Server.

Step 6

The origination switch processes the T-Server request and passes the call to the destination switch.

Step 7

If the call arrives at the destination switch, the switch generates an alerting event.

The destination T-Server waits for the call no longer than the interval specified by the timeout configured on the destination T-Server. If the call is not received at the destination location within this interval, the destination T-Server issues a failure notification to the origination T-Server, deletes all data about the request, and, when appropriate, frees the resources previously allocated for the request.

If either the specified timeout expires or the call is abandoned before the origination T-Server receives a response from the destination T-Server, the operation is considered failed. In this case, the origination T-Server:

1. Generates a request to the destination T-Server to cancel the request for routing service.
2. Responds to the client that requested the service in one of the following ways:
   - If the origination T-Server has already sent a response to the request the client sent in Step 1, the origination T-Server supplements its response with EventRemoteConnectionFailed.
   - If the origination T-Server has not yet sent a response to the client, the origination T-Server sends EventError.
3. Deletes information about the request.

Step 8

If the destination T-Server matches the arrived call, it updates the ConnID, UserData, and CallHistory attributes with the data received in the request for routing service availability. The connection ID is updated as follows:

The arrived call is assigned the ConnID that is specified in the request for routing service availability, but only if this ConnID does not coincide with the ConnID of a call that has existed at the destination site. If two such ConnIDs are identical, the arrived call is assigned a new unique ConnID.

For direct-* transaction types (where the asterisk stands for a callid, uui, ani, or digits extension), the call reaches the destination DN directly.

For the transaction types route and route-uui, the call first arrives at an External Routing Point from which it is routed to the destination DN. The call info is updated when the call reaches the External Routing Point. An External Routing Point is considered free when the first alerting event (EventQueued or EventRouteRequest) is distributed.

Please keep the following issues in mind when using the ISCC feature:

- If routing from a dedicated External Routing Point to the destination DN fails, T-Server considers the transaction failed. However, the ConnID, UserData, and CallHistory attributes are updated. Then, T-Server attempts to route the call to one of the Default DNs configured for this External Routing Point.

- If the destination T-Server did not receive a request for routing service availability, but a call arrives at an External Routing Point, T-Server considers the call to be unexpected and routes the call to the DN specified by the `dn-for-unexpected-calls` configuration option. When no alternative targets are defined, the call remains at the External Routing Point until diverted by the switch or abandoned by the caller.

For `reroute` and `pullback` transaction types, the call returns to the network location. For the `dnis-pool` transaction type, the call reaches the destination DN directly.

Step 9

If, in Step 8, the call does not arrive within the configured timeout, or the transaction fails, the destination T-Server sends a notification of failure to the origination T-Server.

Otherwise, the destination T-Server notifies the origination T-Server that the routing service was successful and deletes all information about the request.

Step 10

The origination T-Server notifies the client that the routing service was successful (or failed) and deletes all information about the request.

ISCC Transaction Types

As switches of different types provide calls with different sets of information parameters, a single mechanism for passing call data between the switches is not feasible in some cases. Therefore, the ISCC feature supports a number of mechanisms for passing call data along with calls between locations. This section describes ISCC transaction type principles, identifies which transaction types are supported for each T-Server, and defines each transaction type (beginning with “direct-ani” on page 276).

It is important to distinguish the two roles that T-Servers play in an external routing (ISCC) transaction—namely *origination T-Server* and *destination T-Server*.

- The origination T-Server initiates an ISCC transaction. It prepares to send the call to another T-Server and coordinates the process.
- The destination T-Server receives call data from an origination T-Server and matches this data to a call that will arrive at some time in the future.

The distinction between these roles is important because the range of telephony-hardware functionality often requires T-Servers to support two entirely different sets of ISCC transactions based on which of the two roles they play. For instance, it is very common for a particular T-Server to support many types of ISCC transactions when it takes on the origination role, but fewer when it takes on the role of a destination T-Server.

The ISCC transaction type `reroute` is a good example. Most T-Servers support `Reroute` as origination T-Servers, but very few support `Reroute` as destination T-Servers.

Determining and Configuring Transaction Type Support

You can find descriptions of these transaction types starting on page 276. Use Table 32 on page 283 to identify the transaction types your destination T-Server supports. A blank table cell indicates that T-Server does not support a certain transaction type.

You can configure the transaction types specific to your T-Server as values of the `cast-type` configuration option specified in the ISCC configuration section `extrouter`. Refer to Chapter 12, "T-Server Common Configuration Options," on page 343 for the option description.

ISCC Transaction Type General Principles

Generally, since most of the ISCC implementation is done at the T-Server Common Part (TSCP) code level, all T-Servers support certain ISCC transaction types. Any T-Server can act as the origination T-Server for the following transaction types:

- `direct-ani`, page 276
- `direct-notoken`, page 278
- `dnis-pool`, page 278
- `pullback`, page 280
- `reroute`, page 280
- `route` (aliased as `route-notoken`), the default transaction type, page 281

The following transaction types are unevenly supported for both the origination and destination T-Server roles:

- `direct-callid` (aliased as `direct`), page 276
- `direct-digits` (reserved for Genesys Engineering)
- `direct-network-callid`, page 277
- `direct-uui`, page 277
- `route-uui`, page 282

The `reroute` and `pullback` transaction types are supported only for selected T-Servers in the *destination* role. However, if you implement this support, other transaction types require additional configuration and testing—even those that would normally be supported by default.

direct-ani

With the transaction type `direct-ani`, the `ANI` network attribute is taken as the parameter for call matching. Properly configured switches and trunks can keep the `ANI` attribute when a call is transferred over the network. T-Server is capable of using this network feature for call matching.

Warnings!

- Depending on the switch platform, it is possible to inherit the `ANI` attribute after routing a call to a remote destination, and after performing a Single-Step Transfer and other telephone actions. However, ISCC only works properly in scenarios where the `ANI` attribute on the destination T-Server is represented by exactly the same digit string as on the origination T-Server.
- Typically, the `ANI` attribute represents the original call identifier (customer phone number), which guarantees that the attribute remains unique. However, you can use the `non-unique-ani` resource type to block ISCC from matching calls based on an ANI that is known to be non unique. (See "Configuring access resources for non-unique ANI" on page 317 for details.)

Notes:

- Some switches, such as Nortel Communication Server 2000/2100 (formerly DMS-100) and Avaya Communication Manager (formerly DEFINITY ECS (MV), may omit the `ANI` attribute for internal calls—that is, for calls whose origination and destination DNs belong to the same switch. If this is the case, do not use the `direct-ani` transaction type when making, routing, or transferring internal calls with the ISCC feature.
- When the `direct-ani` transaction type is in use, the Number Translation feature becomes active. See "Number Translation Feature" on page 291 for more information on the feature configuration.
- With respect to the `direct` transaction types, Network T-Servers and load-sharing IVR Servers are not meant to play the role of destination T-Servers for call routing. Using Network T-Server with these transaction types requires special architecture.

direct-callid

With the transaction type `direct-callid`, the call reaches the destination DN directly from another location, and the `CallID` of the call is taken as the attribute for call matching. When a call arrives at the final destination, the destination T-Server identifies its CallID, and updates the call info if the CallID matches.

Use this transaction type when the destination switch has the capability to assign to an incoming call the same network-wide unique CallID that the origination switch has already assigned to that call.

Notes:

- The direct-callid transaction type is used only in conjunction with the TRouteCall and TSingleStepTransfer function calls. They are applied only to the call that is in progress, and do not apply to functions that involve in the creation of a new call (for example, TMakeCall.)
- For T-Server for Nortel Communication Server 2000/2100, the direct-callid transaction type is also applied to the TMuteTransfer function.

direct-network-callid

With the transaction type direct-network-callid, the call reaches the destination DN directly from another location, and the NetworkCallID of the call is taken as the attribute for call matching. When a call arrives at the final destination, the destination T-Server identifies its NetworkCallID, and updates the call info if the NetworkCallID matches.

Use this transaction type when the destination switch has the capability to assign to an incoming call the same network-wide unique NetworkCallID that the origination switch has already assigned to that call.

Note: To support this transaction type, you must configure Target Type and ISCC Protocol Parameters fields of the corresponding Switch Access Code in the Configuration Layer. Refer to Part Two of this document for information about settings specific for your T-Server type.

direct-uui

With the transaction type direct-uui, so-called user-to-user information (UUI) is taken as the attribute for call matching. Some switches make it possible to send a small data packet along with a call. T-Server can use this data to recognize a call passed from one switch to another. The destination T-Server generates a local unique value for UUI, and then notifies the origination T-Server. The origination T-Server uses a provided value to mark the call coming from the origination location. The destination T-Server receives a call and checks whether it is marked with an exact UUI value. If so, the call is considered as matched.

On the Avaya Communication Manager and the Aspect ACD, UUI is referred to as “user-to-user information.” On the Siemens Hicom 300 switch with CallBridge, UUI is referred to as “Private User Data.” On the Alcatel A4400/OXE switch, UUI is referred to as “correlator data.”

Note: To support this transaction type, you must configure your switches to pass the UUI provided by your T-Server. Moreover, the trunks involved must not drop this data.

direct-notoken

With the transaction type `direct-notoken`, T-Server expects a call to arrive from another location to the destination DN specified in the request for routing service availability. When a call reaches the specified DN, T-Server processes the call as the expected externally routed call.

Notes:

- This matching criterion is weak because any call that reaches the specified DN is considered to be the expected call. Genesys recommends that you use this transaction type only in a contact center subdivision that can be reached from within the contact center only (for example, the second line of support, which customers cannot contact directly).
- With respect to the `direct` transaction types, Network T-Servers and load-sharing IVR Servers are not meant to play the role of destination T-Servers for call routing. Using Network T-Server with these transaction types requires special architecture.

dnis-pool

With the `dnis-pool` transaction type, T-Server reserves one of its DNIS access resources and waits for the call that has the same `DNIS` attribute as the name of the reserved DNIS access resource.

If the arrived call is matched successfully, the destination T-Server may update the value of the `DNIS` attribute of the call (along with `ConnID`, `UserData`, and `CallHistory`) with the value of the `DNIS` attribute of the original call. This occurs when the value of the `DNIS` attribute of the original call is specified as a value of the key-value pair `_ISCC_TRACKING_NUMBER_` in the `Extensions` attribute of the original client request.

The DNIS matching can be based on any number of digits out of all the digits that comprise the `DNIS` attribute. The number of digits that T-Server should use for DNIS matching is specified for the destination switch as the `ISCC Protocol Parameters` property of the Switch Access Code. The value syntax should be as follows:

```
dnis-tail=<number-of-digits>
```

For example, if this property is set to the `dnis-tail=7` value, ISCC matches only the last seven digits of a `DNIS`.

You must configure DNIS access resources in the switch; otherwise, ISCC fails to use this transaction type and sends `EventError` in response to the client application request.

Note: The `dnis-pool` transaction type is typically used for networks employing a "behind the SCP" architecture—network IVR. Network T-Server for GenSpec and IServer are two examples of this, but other Network T-Servers might also be used in this architecture.

In Load-Balancing Mode

When T-Server uses load balancing for call routing with the `dnis-pool` transaction type, the following processes occur:

1. A client of the origination T-Server sends a request to pass a call to the location with a DNIS access resource specified in the key-value pair `iscc-selected-dnis`.
2. The origination T-Server distributes the request for a routing service to all destination T-Servers.
3. The destination T-Servers receive the request and check that the specified DNIS is not being used by another routing service request.
4. The origination T-Server expects to receive a positive response from each destination T-Server. If the origination T-Server receives a negative response from at least one T-Server, it sends an `EventError` to the client and clears all data about the request. If the origination T-Server receives the confirmation about routing service availability from all destination T-Servers, it processes the client's request and sends a corresponding message to the switch.
5. The origination switch processes the T-Server request and passes the call to the destination switch.
6. The call arrives at the destination switch, which generates an alerting event to one of the corresponding load-balanced destination T-Servers.
7. That destination T-Server processes the call and notifies the origination T-Server that the routing service was successful and deletes all information about the request.
8. The origination T-Server sends a routing service request cancellation to all other destination T-Servers.
9. The origination T-Server notifies the client that the routing service has been successful and deletes all information about the request.

pullback

`Pullback` is used in the following scenario, for those T-Servers that support it:

1. A call arrives at Site A served by a Network T-Server.
2. At Site A, a Network T-Server client requests to pass the call by means of ISCC routing to Site B served by a premise T-Server. Any transaction type except `reroute` or `pullback` can be specified in this request.
3. The call arrives at Site B and is either answered by an agent or delivered to a routing point.
4. A client of the premise T-Server at Site B sends a `TRouteCall`, or `TSingleStepTransfer` request to transfer the call to the network.
5. The Site B premise T-Server notifies the Network T-Server about this request.
6. The network T-Server receives the notification and issues an `EventRouteRequest` to obtain a new destination.
7. After receiving the new destination information, the Network T-Server disconnects the call from its current premise location at Site B and attempts to route the call to the new destination.
8. The Site B premise T-Server stops tracking the call, which has disconnected from the premise's agent DN or routing point and is delivered to the network.
9. The network T-Server completes routing the call to its new destination.

Note: The transaction type `pullback` can be used only to return a call from a premise T-Server to the Network T-Server that serves the site from which the call was previously transferred.

reroute

`Reroute` is used in the following scenario, for those T-Servers that support it:

1. A call arrives at Site A served by a Network T-Server.
2. At Site A, a Network T-Server client requests to pass the call by means of ISCC to Site B served by a premise T-Server. Any transaction type except `reroute` or `pullback` can be specified in this request.
3. An agent at Site B answers the call.
4. A client of the premise T-Server at Site B sends a `TSingleStepTransfer` or `TRouteCall` request to transfer the call elsewhere (to a PSTN, to an agent, or to a routing point).

5. The Site B premise T-Server notifies the Network T-Server about this request and releases the call leg that resides at the agent's phone (using `TReleaseCall`) or at the Routing Point (using `TRouteCall` with the parameter `RouteTypeCallDisconnect`).
6. The Network T-Server receives the notification and reroutes the call to the requested destination—that is, it sends `EventRouteRequest` and attaches the call's user data.

Notes:

- The transaction type `reroute` can be used only to return a call from a premise T-Server to the Network T-Server that serves the site from which the call was previously transferred.
- To perform multi-site operations that are initiated with `TRouteCall` and for which the `reroute` transaction type is requested, the origination T-Server must support the `RouteTypeCallDisconnect` subtype of `TRouteCall`.

route

With the transaction type `route` (aliased as route-notoken), a call from the origination location reaches a dedicated External Routing Point, and from there, it is routed to a destination DN.

To control configured External Routing Points, T-Server must register these DNs with the switch. Failure to register implies that the External Routing Point is not available for ISCC purposes. Client applications can register External Routing Points via T-Server for monitoring purposes only.

Point-to-Point (One-to-One)

In the Point-to-Point access mode, only one trunk line is used to access an External Routing Point (for example, VDN, CDN) at the destination site. See Figure 33.

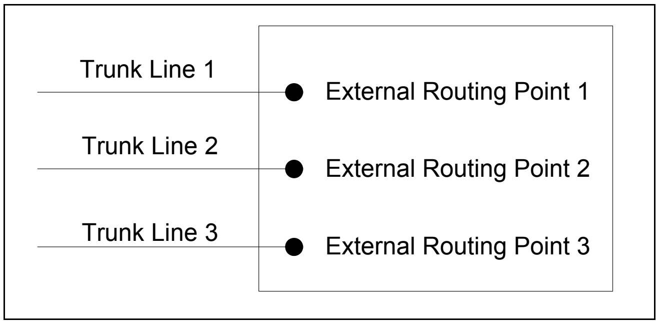


Figure 33: Point-to-Point Trunk Configuration

Note: Dedicated DNs of the `External Routing Point` type must be configured in a switch. See "Configuring Multi-Site Support" on page 307.

Multiple-to-Point (Multiple-to-One)

In the Multiple-to-Point access mode, trunk lines are assigned to the destination switch's trunk group, from which calls are routed to the final destination. See Figure 34.

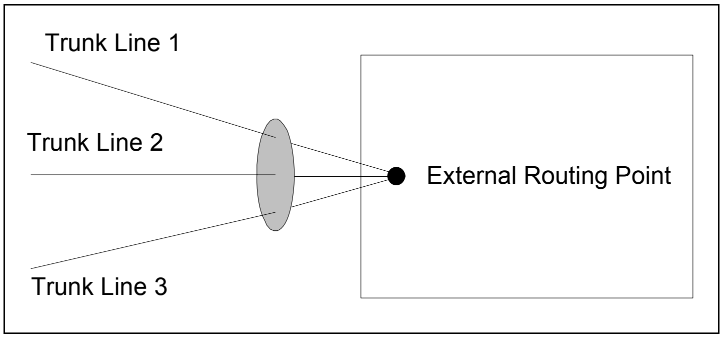


Figure 34: Multiple-to-Point Trunk Configuration

With this configuration, all calls reach the same External Routing Point. The `DNIS` attribute of a specific call differs from that of other calls and uniquely identifies the trunk from which the call arrived.

Note: To switch to this operating mode, you must configure the `route-dn` configuration option for T-Server.

route-uui

The `route-uui` transaction type employs the dedicated External Routing Point feature of the `route` transaction type (page 281) and the UUI matching feature of the `direct-uui` transaction type (page 277). This transaction type accommodates those switches that require a designated External Routing Point even though they use UUI for tracking.

Note: To support this transaction type, you must configure your switches to pass the UUI provided by your T-Server. Moreover, the trunks involved must not drop this data.

T-Server Transaction Type Support

Table 32 shows which transaction types are supported by a specific T-Server. Use this table to determine the transaction types that are available for use with your T-Server. This applies both to the `cast-type` you specify in the configuration options for your T-Server, and to any client-designated route-type requests specified for transfers of calls. A blank table cell indicates that T-Server does not support a certain transaction type.

Table 32: T-Server Support of Transaction Types

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct-uui / route-uui	direct-no-token	direct-ani	direct-digits	direct-network-callid	dnis-pool	pull-back
	one-to-one	multiple-to-one									
Alcatel A4200/OXO	Yes			Yes		Yes	Yes				
Alcatel A4400/OXE	Yes			Yes[a,b,c]	Yes[d]	Yes	Yes[a]		Yes[e]		
Aspect ACD	Yes	Yes		Yes		Yes[f]	Yes[f]				
Avaya Communication Manager	Yes				Yes	Yes	Yes				
Avaya INDeX	Yes					Yes	Yes				
Cisco CallManager	Yes			Yes		Yes	Yes				
DataVoice Dharma	Yes			Yes		Yes	Yes				
Digitro AXS/20	Yes			Yes		Yes					
EADS Intecom M6880	Yes			Yes		Yes	Yes				
EADS Telecom M6500	Yes			Yes		Yes	Yes				
eOn eQueue	Yes			Yes		Yes					
Ericsson MD110	Yes			Yes[a]		Yes	Yes[a]				
Fujitsu F9600	Yes					Yes					

Table 32: T-Server Support of Transaction Types (Continued)

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct-uui / route-uui	direct-no-token	direct-ani	direct-digits	direct-network-callid	dnis-pool	pull-back
	one-to-one	multiple-to-one									
Huawei C&C08	Yes			Yes							
Mitel SX-2000/MN3300	Yes			Yes		Yes	Yes				
NEC NEAX/ APEX	Yes			Yes		Yes	Yes				
Nortel Communication Server 2000/2100	Yes			Yes[f]		Yes[f]	Yes[f]				
Nortel Communication Server 1000 with SCCS/MLS	Yes			Yes		Yes	Yes		Yes		
Philips Sopho iS3000	Yes			Yes		Yes	Yes				
Radvision iContact	Yes		Yes								Yes
Rockwell Spectrum	Yes	Yes		Yes		Yes[f]	Yes[f]				
Samsung IP-PCX IAP	Yes			Yes		Yes					
Siemens Hicom 300/ HiPath 4000 CSTA I	Yes			Yes	Yes[b]	Yes	Yes				
Siemens HiPath 3000	Yes			Yes		Yes					
Siemens HiPath 4000 CSTA III	Yes				Yes[b]	Yes	Yes				

Table 32: T-Server Support of Transaction Types (Continued)

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct-uui / route-uui	direct-no-token	direct-ani	direct-digits	direct-network-callid	dnis-pool	pull-back
	one-to-one	multiple-to-one									
Siemens HiPath DX	Yes			Yes	Yes	Yes	Yes				
SIP Server	Yes		Yes		Yes[g]	Yes					Yes
Tadiran Coral	Yes			Yes		Yes	Yes				
Teltronics 20-20	Yes			Yes		Yes	Yes				
Tenovis Integral 33/55	Yes			Yes		Yes	Yes				
Network T-Servers											
AT&T											
Concert											
CRSP											Yes
DTAG			Yes								
GenSpec	Yes	Yes	Yes							Yes	
IVR Server, using network configuration	Yes	Yes	Yes							Yes	Yes
KPN			Yes								
ISCP											
MCI											
NGSN	Yes										Yes
Network SIP Server	Yes					Yes	Yes			Yes	
Sprint	Yes										

Table 32: T-Server Support of Transaction Types (Continued)

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct-uui / route-uui	direct-no-token	direct-ani	direct-digits	direct-network-callid	dnis-pool	pull-back
	one-to-one	multiple-to-one									
SR-3511											
Stentor											

a. Not supported in the case of function `TRequestRouteCall` on a virtual routing point: a routing point can be simulated using a hunt group with calls being deflected or transferred from the hunt-group member when routing. When a two-step (typically mute) transfer is used on such a hunt-group member, `CallID` and `ANI` usually change; thus, the `direct-callid` and `direct-ani` types do not work.

b. Not supported in the case of function `TSingleStepTransfer` when the T-Server service is simulated using a two-step transfer to the switch. In this case, `CallID` and `ANI` change; thus, the `direct-callid` and `direct-ani` types do not work.

c. Not supported if two T-Servers are connected to different nodes.

d. There are some switch-specific limitations when assigning CSTA correlator data UUI to a call.

e. Supported only on ABCF trunks (Alcatel internal network).

f. To use this transaction type, you must select the `Use Override` check box on the `Advanced` tab of the DN `Properties` dialog box.

g. SIP Server supports the `direct-uui` type.

Transfer Connect Service Feature

The Transfer Connect Service (TCS) feature supports transfer connect services available on some telephony networks. When this feature is enabled, ISCC passes user data to remote locations to which calls are transferred or conferenced using transfer connect services.

Procedure: Activating Transfer Connect Service

Start of procedure

1. Open the T-Server `Application's Properties` dialog box.
2. Click the `Options` tab.
3. Set the `tcs-use` configuration option to `always`.

4. Set the `tcs-queue` configuration option to the number of a DN on the origination switch.

   ISCC uses this DN as an intermediate step when sending calls to the remote location. The DN that is configured as `tcs-queue` receives attached data indicating the Feature Access Code (FAC) needed to reach the remote site. After a call is directed to the DN with data, a monitoring application takes the data and generates the required DTMF (dual-tone multifrequency) tones to redirect the call through the network to the remote location.
5. When you are finished, click `Apply`.
6. Click `OK` to save your changes and exit the `Properties` dialog box.

End of procedure

Note: With T-Server for Avaya Communication Manager, you can use `RequestRouteCall` with `RouteTypeOverwriteDNIS` to initiate the playing of DTMF tones. This is done through the use of another intermediate DN (typically, an announcement port configured to give the silence treatment), to which the call is routed. When the call is established on this DN, T-Server requests that the digits sent in the `DNIS` field of the `TRequestRouteCall` be played via the `ASAI-send-DTMF-single` procedure.

ISCC/COF Feature

The Inter Server Call Control/Call Overflow (ISCC/COF) feature of T-Server, that supports passive external routing, is specifically designed to handle calls delivered between sites by means other than ISCC. Such scenarios include contact center overflows and manual call transfers.

An *overflow situation* occurs when a call comes into a contact center where all agents are currently busy. In this situation, the switch can transfer (overflow) the incoming call to another site where there is an available agent.

T-Server uses two methods to handle call overflow and manual transfer scenarios. The first method is based on `NetworkCallID` matching and the second method is based on `ANI/OtherDN` matching.

When connected to each other via switch-specific networks, switches of some types can pass additional information along with transferred calls. This information may contain the `NetworkCallID` of a call, which is a networkwide unique identifier of the call.

When connected via a regular PSTN, switches of all types can send the `ANI` and/or `OtherDN` attributes to the destination switch during any call transfer operation.

While all T-Servers support the ISCC/COF feature using the ANI and/or OtherDN attributes, only a few support this feature using the NetworkCallID attribute. Table 33 shows the T-Server types that provide the NetworkCallID of a call.

Table 33: T-Server Support of NetworkCallID for ISCC/COF Feature

T-Server Type	Supported NetworkCallID Attribute
Alcatel A4400/OXE	Yes
Aspect ACD	Yes
Avaya Communication Manager	Yes
Nortel Communication Server 2000/2100	Yes
Nortel Communication Server 1000 with SCCS/MLS	Yes
Rockwell Spectrum	Yes
SIP Server	Yes

The ISCC/COF feature can use any of the three attributes (NetworkCallID, ANI, or OtherDN) as criteria for matching the arriving call with an existing call at another location. Consequently, the attribute that is used determines what ConnID, UserData, and CallHistory are received for the matched call from the call's previous location.

Warning! Depending on the switch platform, it is possible to inherit the ANI attribute after routing a call to a remote destination, and after performing a Single-Step Transfer and other telephone actions. However, ISCC/COF works properly only in scenarios where the ANI attribute on the destination T-Server is represented by exactly the same unique digit string as on the origination T-Server. Typically the ANI attribute represents the original call identifier (customer phone number), which guarantees that the attribute remains unique.

Note: When the ISCC/COF feature is in use, the Number Translation feature becomes active. See "Number Translation Feature" on page 291 for more information on the feature configuration.

ISCC/COF Call Flow

Figure 35 shows the sequence of steps that occur in an ISCC/COF scenario when a call is made or transferred by an agent at Site A to a DN at Site B, or when a call is overflowed from Site A to Site B.

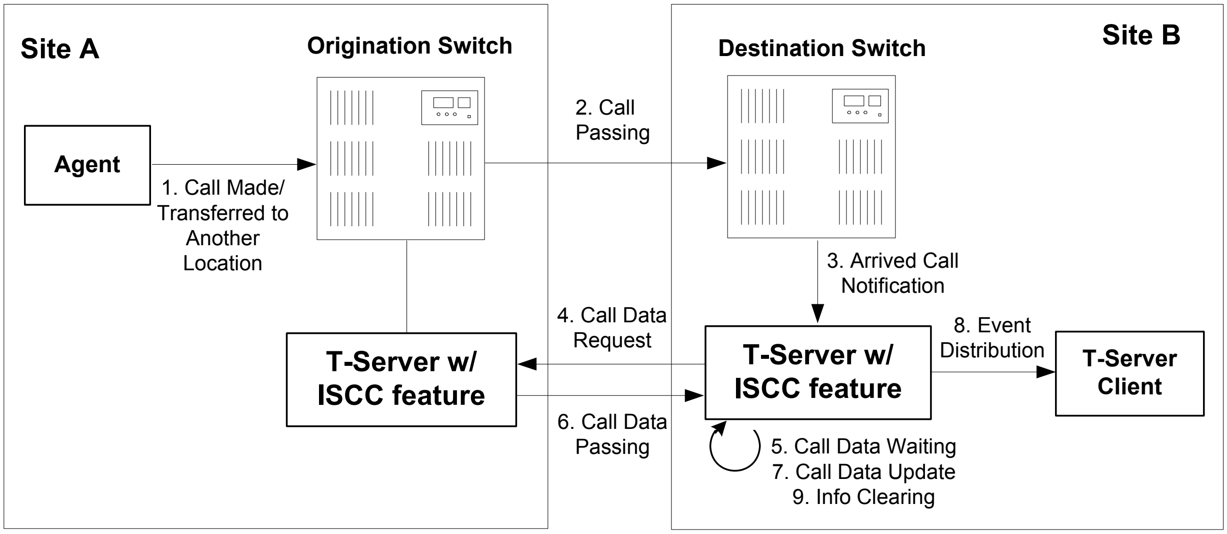


Figure 35: Steps in the ISCC/COF Process

Step 1

An agent makes or transfers a call manually to another location or a call is overflowed from Site A (origination location) to Site B (destination location).

Step 2

Switch A (the origination switch) passes the call to Switch B (the destination switch).

Step 3

The call reaches the destination switch, which notifies the destination T-Server about the arrived call.

Step 4

The destination T-Server verifies with remote locations whether the call was overflowed from any of them.

To determine which calls to check as possibly overflowed, T-Server relies on the `Switch` object configuration:

- If no COF DNs (that is, DNs of the `Access Resources` type with the `Resource Type` set to `cof-in` or `cof-not-in`) are configured for the destination switch, the ISCC/COF feature of the destination T-Server checks all arriving calls.

- If a number of COF DNs are configured for the destination switch, one of three scenarios occurs:
    - If the COF DNs with the cof-in setting for the Resource Type property are configured, the ISCC/COF checks for overflow only those calls that arrive to those cof-in DNs that are Enabled.
    - If no DNs with the cof-in setting for the Resource Type property are configured, but some DNs have the cof-not-in setting for the Resource Type property, the ISCC/COF checks for overflow only those calls that arrive to those cof-not-in DNs that are Disabled.
    - If no DNs with the cof-in setting for the Resource Type property are configured, some DNs have the cof-not-in setting for the Resource Type property, and some other DNs do not have any setting for the Resource Type property, the ISCC/COF checks for overflow only those calls that arrive to the DNs without any setting for the Resource Type property.
- In all other cases, no calls are checked for overflow.

To determine which location the call arrived from, T-Server checks the call type and checks whether the call has the NetworkCallID, ANI, or OtherDN attribute:

- If the call is not an inbound call, the request for call data is sent to all remote locations *except* those whose Switch Access Code has the ISCC Call Overflow Parameters property set to inbound-only=true.
- If the call of any type has the NetworkCallID attribute, the destination T-Server sends a request for call data to the remote locations of the same switch type as the destination location if their Switch Access Codes have the ISCC Call Overflow Parameters property set to match-callid.
- If the call of any type has the ANI or OtherDN attribute, the request for call data is sent to remote locations whose Switch Access Code has the ISCC Call Overflow Parameters property set to match-ani.

Step 5

The destination T-Server waits (suspending events related to that call) for the call data from the remote T-Server for the time interval specified in the cof-ci-req-tout configuration option. Within this interval, T-Server holds any events related to the call. In addition, the cof-ci-defer-delete option on the origination T-Server establishes the time interval only after which that T-Server deletes the call information. And the cof-ci-wait-all, if set to true, forces the origination T-Server to wait for responses related to possible call overflow situations before updating call data.

Step 6

The T-Server at the location from which the call was transferred or overflowed sends call data to the requesting T-Server.

Step 7

If a positive response to the call-data request is received, T-Server updates ConnID, UserData, and CallHistory, distributes all suspended events related to that call and deletes all information regarding the transaction (Step 9).

Step 8

If the timeout set by cof-ci-req-tout expires, T-Server distributes all suspended events, and starts the timeout specified by the cof-rci-tout option. If a positive response is received within the timeout set by cof-rci-tout, T-Server updates the ConnID, UserData, and CallHistory and notifies client applications by distributing EventPartyChanged.

Step 9

T-Server deletes all information regarding the transaction when one of these results occurs:

- The first positive response to the call-data request is received.
- Negative responses from all queried locations are received.
- The timeout specified by the cof-rci-tout option expires.

Number Translation Feature

The Number Translation feature of T-Server extends the ISCC/COF and direct-ani transaction type functions to provide more flexibility for handling calls distributed across multiple sites. T-Server translates the input string (ANI string) into a number defined by the translation rules. This processing is called number translation. T-Servers participating in handling calls at multiple sites exchange the translated numbers in order to match the call instances.

The translation process involves two algorithms, one for rule selection and the other for the actual translation. Through the first algorithm, T-Server selects a rule that will be used for number translation. Through the second algorithm, T-Server translates the number according to the selected rule definition. See “Number Translation Rules” on page 292 for more information on configuring rules for your environment.

Number translation occurs as follows:

1. The switch reports a number, typically via AttributeANI.
2. T-Server evaluates all configured inbound rules to determine which one is the best fit for the received number. The best fit is determined by comparing the length of, and the specific digits in, the input number with the inbound pattern of each configured rule. See “Rule Examples” on page 297 for specific examples.
3. T-Server translates the number according to the selected rule.

To enable T-Server to translate numbers, you must perform specific configuration tasks that are associated with translation. See "Configuring Number Translation" on page 298.

Number Translation Rules

T-Server uses the number translation rules that you define in the T-Server configuration object in two ways:

- **Rule selection—**To determine which rule should be used for number translation
- **Number translation—**To transform the number according to the selected rule

Using ABNF for Rules

The number translation rules must conform to the following syntax, represented using Augmented Backus-Naur Form (ABNF) notation. For more information about ABNF, see RFC 2234, "Augmented BNF for Syntax Specifications: ABNF."

Note: The notations are explained starting at the highest level, with the name of a component notation and a basic definition of each component that comprises it. Some components require more detailed definitions, which are included later in this section.

Common Syntax Notations

Syntax notations common to many of these rules include:

- `*`—Indicates that 0 to an infinite number of the item following this symbol are acceptable.
- `1*`—Indicates that one repetition is required. For T-Server, only one instance is acceptable.
- `/`—Indicates that any of the items mentioned, or a combination of those items, is acceptable.

Component Notations

Component notations include:

- `dialing-plan = *dialing-plan-rule`

  where:

  - `dialing-plan-rule` represents the name of the rule. Each rule must have a unique name. There are no other naming restrictions, and you do not need to model your names according to the examples in this chapter.

The rules are represented as separate options in the configuration. Also, fields from a rule are represented as parameters in a single option string.

- `rule = [name] in-pattern [out-pattern]`

  where:

  - `[name]` is the name for the rule option, for example, `rule-01`. In ABNF notation, the brackets `[]` indicate that 0 or 1 instance of the component is required. However, for T-Server, a name is required.
  - `in-pattern` is the part of the rule to which T-Server looks when attempting to match the input number.
  - `[out-pattern]` is the part of the rule that instructs T-Server on how to translate the input number into the required format. The brackets indicate that either 0 or 1 instance is required. You must create an `out-pattern` for number translation rules.

- `name = *( ALPHA / DIGIT / "-")`

  where:

  - `ALPHA` indicates that letters can be used in the name for the rule option.
  - `DIGIT` indicates that numbers can be used in the name for the rule option.
  - `"-"` indicates that a dash (-) can also be used in the option name, for example, `rule-01`.

- `in-pattern = 1*(digit-part / abstract-group)`

  where:

  - `digit-part` represents numbers. T-Server uses this when selecting the most appropriate rule from the entire dialing plan.
  - `abstract-group` represents one or more letters with each letter representing one or more numbers. T-Server uses this when transforming a dial string.

  For example, `[1-9]` is the `digit-part` (representing a range of numbers) and `ABBB` is the `abstract-group` for `in-pattern=[1-9]ABBB`.

- `out-pattern = 1*(symbol-part / group-identifier) *param-part`

  where:

  - `symbol-part` represents digits, symbols, or a combination. Symbols are rarely used. They are not used in the United States.
  - `group-identifier` are letters that represent groups of numbers. A letter in the `out-pattern` represents one or more digits, based on the number of times the letter is used in the `in-pattern`.
  - `*param-part` represents an additional parameter, such as `phone-context`. Reminder: an asterisk means that 0 to an infinite number of these are acceptable.

  For example, in `rule-04; in-pattern=1AAABBBCCC;out-pattern=91ABC, 91` is the `symbol-part`; `A, B,` and `C` are `group-identifiers` in the `out-pattern`, each representing three digits, since there are three instances of each in the `in-pattern`.

Note: Prefix an `out-pattern` value with a plus sign (+) for the inbound rule when the output must be in a global form (E.164 format).

- `digit-part = digits / range  / sequence`

  where:

  - `digits` are numbers 0 through 9.
  - `range` is a series of digits, for example, 1-3.
  - `sequence` is a set of digits.

- `symbol-part = digits / symbols`

  where:

  - `digits` are numbers 0 through 9.
  - `symbols` include such characters as `+, -,` and so on.

- `range = "[" digits "-"  digits "]" group-identifier`

  where:

  - `"[" digits "-"  digits "]"` represents the numeric range, for example, `[1-2]`.
  - `group-identifier` represents the group to which the number range is applied.

    For example, `[1-2]` applies to group identifier `A` for `in-pattern=[1-2]ABBB.` When T-Server evaluates the rule to determine if it matches the number, it examines whether the first digit of the number, identified as `group-identifier A`, is `1` or `2`.

- `sequence = "[" 1*(digits [","] ) "]" group-identifier`

  where:

  - `"[" 1*(digits [","] ) "]"` represents a sequence of digits, separated by commas, and bracketed. T-Server requires that each digit set have the same number of digits. For example, in `[415,650]` the sets have three digits.
  - `group-identifier` represents the group to which the number sequence is applied.

    For example, in `in-pattern=1[415,650]A*B, [415,650]` applies to `group-identifier A`. When T-Server evaluates the rule to determine if it matches the number, it examines whether the three digits (`group-identifier A`) following the 1 in the number are `415` or `650`.

- `abstract-group = fixed-length-group / flexible-length-group / entity`

  where:

  - `fixed-length-group` specifies a group composed of a specific number of digits and determined by how many times the group identifier is included in the `in-pattern`. For example, for `in-pattern=1AAABBBCCCC`, there are three digits in group `A` and `B` but four in group `C`.

    When you create an `out-pattern`, you include the group identifier only once because the `in-pattern` tells T-Server how many digits belong in that group. For example, `rule-04` (see page 297) is `in-pattern=1AAABBBCCCC; out-pattern=91ABC`.
  - `flexible-length-group` specifies a group composed of 0 or more digits in the group represented by the `group-identifier`. For example, in `in-pattern=1[415,650]A*B`, `*B` represents the flexible length group containing the remaining digits in the number.
  - `entity` represents digits defined for a specific purpose, for example, country code.

  The component `abstract-group` is used only for the `in-pattern`.

- `fixed-length-group = 1*group-identifier`

  See the earlier explanation under `abstract-group`.

- `flexible-length-group = "*"  group-identifier`

  See the earlier explanation under `abstract-group`.

- `entity = "#" entity-identifier  group-identifier`

  where:

  - `"#"` indicates the start of a Country Code `entity-identifier`.
  - `entity-identifier` must be the letter `C` which represents Country Code when preceded by a pound symbol (`#`). Any other letter following the # causes an error.
  - `group-identifier` represents the Country Code group when preceded by `#C`.

  The `entity` component is a special group that assumes some kind of predefined processing, such as the Country Code detection.

- `param-part = ";" param-name "=" param-value`

  where:

  - `";"` is a required separator element.
  - `param-name` is the name of the parameter.
  - `"="` is the next required element.
  - `param-value` represents the value for `param-name`.

- `param-name = "ext" / "phone-context" / "dn"`

  where:

  - `"ext"` refers to extension.
  - `"phone-context"` represents the value of the `phone-context` option configured on the switch.
  - `"dn"` represents the directory number.

- `param-value = 1*ANYSYMBOL`

  where:

  - `ANYSYMBOL` represents any number, letter, or symbol with no restrictions.
- `group-identifier = ALPHA`
- `entity-identifier = ALPHA`
- `digits = 1*DIGIT`
- `symbols = 1*("-" / "+" / ")" / "(" / ".")`

Recommendations for Rule Configuration

The configuration of rules for inbound numbers usually depends on the settings in the corresponding PBX. These settings often define the form in which the PBX notifies its client applications about the number from which an inbound call is coming.

As a general guideline, configure rules that define how to process calls from:

- Internal numbers.
- External numbers within the same local dialing area.
- External numbers within the same country.
- International numbers.

Rules for inbound numbers, typically for North American locations, might look like this:

1. Two rules to transform internal numbers (extensions):

   ```
   name=rule-01;in-pattern=[1-9]ABBB;out-pattern=AB
   ```

   ```
   name=rule-02;in-pattern=[1-9]ABBBB;out-pattern=AB
   ```

2. A rule to transform local area code numbers (in `333-1234` format in this example):

   ```
   name=rule-03;in-pattern=[1-9]ABBBBBB;out-pattern=+1222AB
   ```

3. A rule to transform U.S. numbers (in `+1(222)333-4444` format):

   ```
   name=rule-04;in-pattern=1AAAAAAAAAA;out-pattern=+1A
   ```

4. A rule to transform U.S. numbers without the `+1` prefix (in `(222)333-4444` format):

   ```
   name=rule-05;in-pattern=[2-9]ABBBBBBBBB;out-pattern=+1AB
   ```

5. A rule to transform U.S. numbers with an outside prefix (in `9 +1(222)333-4444` format):

   ```
   name=rule-06;in-pattern=91AAAAAAAAAA;out-pattern=+1A
   ```

6. A rule to transform international numbers with an IDD (international dialing digits) prefix (in `011 +44(111)222-3333` format):

   ```
   name=rule-07;in-pattern=011*A;out-pattern=+A
   ```

7. A rule to transform international numbers without an IDD prefix (in `+44(111)222-3333` format):

   `name=rule-08;in-pattern=[2-9]A*B;out-pattern=+AB`

Rule Examples

This section provides examples of six rules that are configured as options in the Genesys Configuration Database. It also provides examples of how T-Server applies rules to various input numbers.

Rules

rule-01 `in-pattern=[1-8]ABBB;out-pattern=AB`

rule-02 `in-pattern=AAAA;out-pattern=A`

rule-03 `in-pattern=1[415,650]A*B;out-pattern=B`

rule-04 `in-pattern=1AAABBBCCCC;out-pattern=91ABC`

rule-05 `in-pattern=*A913BBBB;out-pattern=80407913B`

rule-06 `in-pattern=011#CA*B;out-pattern=9011AB`

Examples

Here are examples of how T-Server applies configured above rules to various input numbers.

Example 1 T-Server receives input number `2326`.

As a result of the rule selection process, T-Server determines that the matching rule is `rule-01`:

`name=rule-01;in-pattern=[1-8]ABBB;out-pattern=AB`

The matching count for this rule is `1`, because Group `A` matches the digit `2`.

As a result of the parsing process, T-Server detects two groups: Group `A = 2` and Group `B = 326`.

T-Server formats the output string as `2326`.

Example 2 T-Server receives input number `9122`.

As a result of the rule selection process, T-Server determines that the matching rule is `rule-02`:

`name=rule-02;in-pattern=AAAA;out-pattern=A`

The matching count for this rule is `0`; however, the overall length of the input number matches that of the `in-pattern` configuration.

As a result of the parsing process, T-Server detects one group: Group `A = 9122`.

T-Server formats the output string as `9122`.

Example 3 T-Server receives input number `16503222332`.

As a result of the rule selection process, T-Server determines that the matching rule is `rule-03`:

```
name=rule-03;in-pattern=1[415,650]A*B;out-pattern=B
```

The matching count for this rule is `4`, because the first digit matches and all three digits in Group `A` match.

As a result of the parsing process, T-Server detects two groups: Group `A = 650` and Group `B = 3222332`.

T-Server formats the output string as `3222332`.

Example 4 T-Server receives input number `19253227676`.

As a result of the rule selection process, T-Server determines that the matching rule is `rule-04`:

```
name=rule-04;in-pattern=1AAABBBCCCC;out-pattern=91ABC
```

The matching count for this rule is `1`, because the first digit matches.

As a result of parsing process, T-Server detects three groups: Group `A = 925`, Group `B = 322`, and Group `C = 7676`.

T-Server formats the output string as `919253227676`.

Example 5 T-Server receives input number `4089137676`.

As a result of rule selection process, T-Server determines that the matching rule is `rule-05`:

```
name=rule-05;in-pattern=*A913BBBB;out-pattern=80407913B
```

The matching count for this rule is `3`, because three digits match.

As a result of the parsing process, T-Server detects two groups: Group `A = 408` and Group `B = 7676`.

T-Server formats the output string as `804079137676`.

Example 6 T-Server receives input number `011441112223333`.

As a result of the rule selection process, T-Server determines that the matching rule is `rule-06`:

```
name=rule-06;in-pattern=011#CA*B;out-pattern=9011AB
```

The matching count for this rule is `3`, because three digits match.

As a result of the parsing process, T-Server detects two groups: Group `A = 44` and Group `B = 1112223333`.

T-Server formats the output string as `9011441112223333`.

Procedure:
Configuring Number Translation

Purpose: To configure the Number Translation feature in T-Server to provide more flexibility for handling calls distributed across multiple sites.

Overview

- The Number Translation feature becomes active when the ISCC/COF feature and/or the direct-ani transaction type are used.
- This configuration procedure must be completed within the T-Server Application object corresponding to your T-Server.

Start of procedure

1. Open the T-Server Application's Properties dialog box.
2. Click the Options tab.
3. Create a new section called extrouter or open an existing section with this name.
4. Create a new option called inbound-translator-<n>. This option points to another section that describes the translation rules for inbound numbers.
5. In this section, create one configuration option for each rule. Specify the rule name as the option name. The values of these options are the rules for the number translation.

   For the option description and its valid values, see Chapter 12, "T-Server Common Configuration Options," on page 343.
6. When you are finished, click Apply.
7. Click OK to save your changes and exit the Properties dialog box.

End of procedure

Network Attended Transfer/ Conference Feature

The Network Attended Transfer/Conference (NAT/C) feature is designed to enable agents working in multi-site contact centers to consult with each other before making call transfers or conferences, regardless of whether both agents work at the same or different sites. It also enables the agent who requests a consultation to maintain his or her conversation with the customer while the system is looking for an available agent and setting up the consultation call.

The NAT/C feature does not rely on the call transfer capabilities of the local switch.

There are two modes in which the network attended transfer/conference can be performed: *direct* and *URS-controlled*. Figure 36 shows the sequence of steps that occur in *URS-controlled* mode, when Agent A, who is handling a customer call, requests a consultation with another agent, and URS (Universal Routing Server) selects Agent B, who is working at another site. The *direct* mode is similar to the *URS-controlled* mode, with the difference that URS is not involved in the process (Step 2 and Step 3 are omitted).

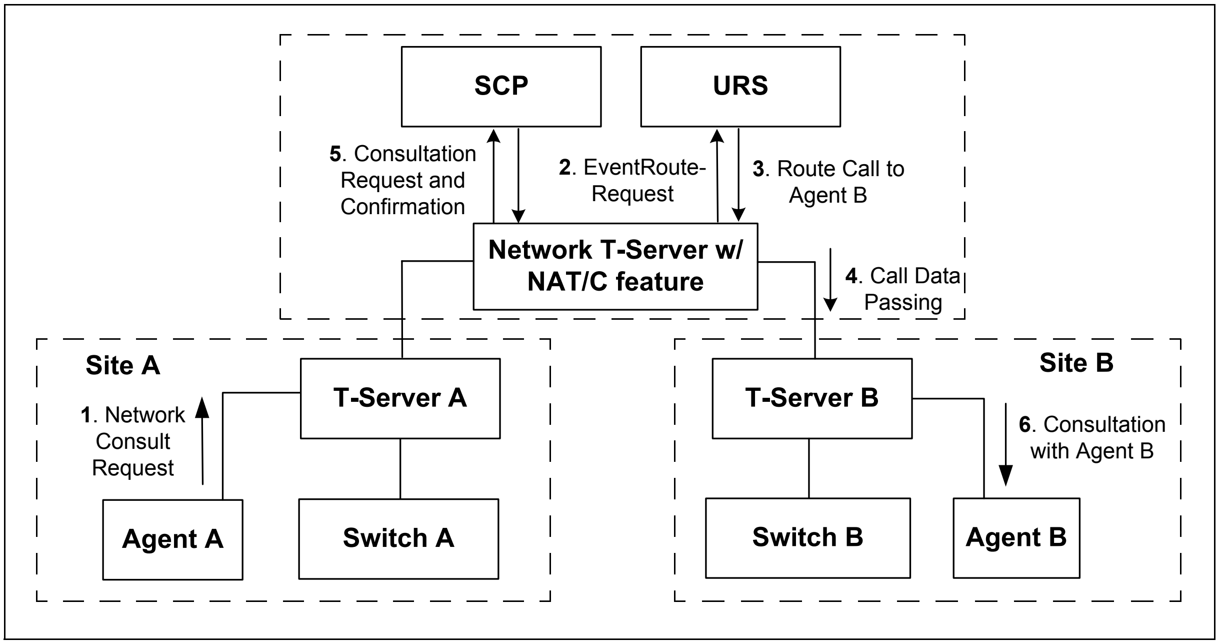


Figure 36: Steps in the NAT/C Process in URS-Controlled Mode

Step 1

Agent A makes a request for a consultation with another agent. A `TNetworkConsult` request is relayed to the Network T-Server. Depending on the parameter settings of the `TNetworkConsult` request, the NAT/C feature will operate in either *direct* or *URS-controlled* mode. For more information, see the *Voice Platform SDK 7.6 .NET* (or *Java) API Reference.*

Step 2

(*URS-controlled* mode only.) The Network T-Server sends `EventRouteRequest` to URS.

Step 3

(*URS-controlled* mode only.) URS locates an available agent at Site B and instructs the Network T-Server to route the call to Agent B. The Network T-Server confirms the initiation of the network transfer by sending `EventNetworkCallStatus` to T-Server A, which then relays it to Agent A.

Step 4

The Network T-Server proceeds to obtain the access number from T-Server B, and passes the call data to T-Server B. (See “ISCC Call Data Transfer Service” on page 269 for details.)

Step 5

The Network T-Server instructs the Service Control Point (SCP) to initiate a new voice path with Agent B. Once the connection is confirmed, the Network T-Server distributes `EventNetworkCallStatus` to both T-Server A and T-Server B, which then relay it to Agent A and Agent B respectively, to indicate that the consultation call is being established.

The Network T-Server also distributes `EventRouteUsed` to URS to confirm successful routing of the call to the selected agent.

Step 6

At this point, the customer is on hold, and Agent A is consulting with Agent B. Agent A can do one of the following:

- End the consultation and retrieve the original customer call
- Alternate between Agent B and the customer
- Set up a conference call with Agent B and the customer
- Transfer the customer call to Agent B

Note: All T-Servers support NAT/C requests with `AttributeHomeLocation` provided that this attribute identifies a network location that is capable of processing such requests. Refer to the *Network T-Server Deployment Guides* to determine whether a specific Network T-Server can process these requests.

Event Propagation Feature

The Event Propagation feature complements the ISCC and ISCC/COF features by distributing updated user data and party-related events to remote T-Servers. This feature is used when a call is being made, transferred, or conferenced to another location, and when, as a result, one or more instances of the call reside at one location while other call instances reside at another location. In this scenario, when a client at one location makes changes to user data, updated user data is passed *(propagated)* to T-Servers at other locations.

The Event Propagation feature consists of User Data update propagation and Party Events propagation.

User Data Propagation

User data propagation takes place when a client at one location makes changes to user data associated with a call that was made, transferred, conferenced, or routed to other locations. The remote clients involved with the call are notified about the changes with `EventAttachedDataChanged`.

When T-Server receives a local update to user data (that is, when a client of this T-Server has changed the call's user data), T-Server determines if parties at remote locations are involved with the call and, if so, sends (propagates) the updated user data to the T-Servers at remote locations.

When T-Server receives a remote update to user data (that is, when a client of a remote T-Server has changed the call's user data and the remote T-Server has used the Event Propagation feature to send the updated user data), T-Server:

1. Updates the user data of the corresponding local call.
2. Determines if parties at other remote locations are involved with the call and, if so, propagates the updated user data to T-Servers at other remote locations.

The locations to which user data is propagated are selected based on a call distribution topology. That is, the updated user data is passed directly to the location to which a call was sent and to the location from which the call was received, excluding the location from which the update was received.

For example, consider a call made from location A to location B, and then conferenced from location B to location C. The three instances of the call reside at different locations: the first instance is at location A, the second instance is at location B, and the third instance is at location C. The Event Propagation feature is employed in the following scenarios:

- When T-Server at location A receives a local update to user data, it notifies T-Server at location B (to which it sent the call) about changes to the call's user data. Thus, T-Server at location B receives a remote update to user data and, in turn, notifies T-Server at location C (to which it sent the call) about these changes.

  Although T-Server at location C receives a remote update to user data, it does not pass the notification to any other T-Servers, because it did not send the call to any other locations. As mentioned earlier, T-Servers at locations B and C update the user data of the corresponding local calls and notify their clients about the changes with `EventAttachedDataChanged`.

- When T-Server at location B receives a local update to user data, it notifies T-Server at location C (to which it sent the call) and T-Server at location A (from which it received the call) about changes to the call's user data. Thus, T-Servers at locations C and A receive a remote update to user data.

  Because T-Server at location C did not send the call to any other locations, and T-Server at location A originated the call, neither of these T-Servers passes the notification to any other T-Servers. T-Servers at locations C and A update the user data of the corresponding local calls and notify their clients about the changes with `EventAttachedDataChanged`.

- When T-Server at location C receives a local update to user data, it notifies T-Server at location B (from which it received the call) about changes to the call's user data. Thus, T-Server at location B receives a remote update to user data and, in turn, notifies T-Server at location A (from which it received the call) about these changes.

> Although T-Server at location A receives a remote update to user data, it does not pass the notification to any other T-Servers, because it originated the call. T-Servers at locations B and A update the user data of the corresponding local calls and notify their clients about the changes with `EventAttachedDataChanged`.

When a call is distributed between location A and location C using location B, and is then deleted on location B, propagation between locations A and C still occurs through the transit node at location B.

Party Events Propagation

Party events propagation takes place when a transfer or a conference is completed for a call that was made to or from one or more remote locations, or when a conference party is removed from the conference.

In these cases, the Event Propagation feature distributes party events, such as `EventPartyChanged`, `EventPartyAdded`, and `EventPartyDeleted`, to remote locations involved with the call, according to appropriate call model scenarios.

For example, consider a call made from DN 1 to DN 2 on location A. A `TInitiateConference` request is then issued for DN 2 to transfer the call to external DN 3 on location B. That transfer is made by means of ISCC routing. When this conference is completed on location A, the Event Propagation feature sends `EventPartyChanged` to location B and distributes this event to involved client applications that are connected to location B and registered for DN 3. After that, if a party of the conference is removed from the conference (for example, a party on DN 2), the Event Propagation feature sends `EventPartyDeleted` to location B and distributes this event to client applications registered for DN 3.

Warnings!

- The `OtherDN` and `ThirdPartyDN` attributes might not be present in the events distributed via the Event Propagation feature.
- The Event Propagation feature will not work properly with installations that use switch partitioning.

If a call involved in the propagation has no local parties but has two or more remote parties, the party events propagation is processed in the same manner as the propagation of user data updates.

For a complete event flow in such scenarios, refer to the *Genesys 7 Events and Models Reference Manual.*

Basic and Advanced Configuration

The basic Event Propagation feature configuration includes the setting of specific configuration options at the T-Server `Application` level. The advanced feature configuration allows you to customize the feature at the `Switch` level.

When determining whether to notify other T-Servers of changes to user data, or to distribute party events, T-Server checks:

1. Call topology (what location a call came from and to what location the call was then transferred or conferenced).
2. Outbound parameters of the Switch this T-Server relates to (whether propagation parameters are configured for the access codes this switch uses to reach the switch at the location a call came from and the switch at the location to which the call was then transferred or conferenced).

Warning! The direction of user-data or party-events propagation does not necessarily match the direction of call distribution. Therefore, the access code used to deliver the call can differ from the access code used for the purpose of Event Propagation.

If one of the T-Servers along the call distribution path has the Event Propagation feature disabled, that T-Server does not distribute events to remote locations.

Procedure: Activating Event Propagation: basic configuration

Purpose: To activate the Event Propagation feature for User Data updates and call-party–associated events (Party Events) distribution.

Start of procedure

1. Open the T-Server Application's Properties dialog box.
2. Click the Options tab.
3. Open the extrouter section.
4. Set the event-propagation option to the list value.

   This setting enables User Data propagation. If you need to enable Party Events propagation, perform Step 5.
5. Set the use-data-from option to the current value.

   This setting enables Party Events propagation.

   For the option description and its valid values, see Chapter 12, "T-Server Common Configuration Options," on page 343.
6. When you are finished, click Apply.
7. Click OK to save your changes and exit the Properties dialog box.

End of procedure

Next Steps

- For advanced feature configuration, do the following procedure:
  Modifying Event Propagation: advanced configuration, page 305

Procedure: Modifying Event Propagation: advanced configuration

Purpose: To modify access codes for advanced Event Propagation configuration.

Prerequisites

- Activating Event Propagation: basic configuration, page 304

Overview

You can set Event Propagation parameters using:

- The `Default Access Code` properties of the `Switch` that receives an ISCC-routed call (the destination switch).
- The `Access Code` properties of the `Switch` that passes an ISCC-routed call (the origination switch).

If you do not set up Event Propagation parameters for a given `Access Code,` T-Server uses corresponding settings configured for the `Default Access Code` of the destination switch.

The procedures for modifying `Default Access Codes` and `Access Codes` are very similar to each other.

Start of procedure

1. Among configured `Switches,` select the `Switch` that the configured T-Server relates to.
2. Open the `Switch's Properties` dialog box and click either the `Default Access Codes` tab or the `Access Codes` tab.
3. Select a configured `Default Access Code` or configured `Access Code` and click `Edit.`

   Note: If no `Default Access Code` is configured, see page 309 for instructions. If no `Access Codes` are configured, see page 310 for instructions.

4. In the `Switch Access Code Properties` dialog box that opens, specify a value for the `ISCC Protocol Parameters` field as follows:

- To enable distribution of both user data associated with the call and call-party–associated events[1], type:

  `propagate=yes`

  which is the default value.
- To enable distribution of user data associated with the call and disable distribution of call-party–associated events, type:

  `propagate=udata`
- To disable distribution of user data associated with the call and enable distribution of call-party–associated events, type:

  `propagate=party`
- To disable distribution of both user data associated with the call and call-party–associated events, type:

  `propagate=no`

5. Click `OK` to save configuration updates and close the `Switch Access Code Properties` dialog box.
6. Click `Apply` and `OK` to save configuration updates and close the `Switch Properties` dialog box.

End of procedure

ISCC Transaction Monitoring Feature

This feature allows T-Server clients to monitor ISCC transactions that occur during the call data transfer between T-Servers in a multi-site environment.

In order to be able to monitor ISCC messaging, a T-Server client must subscribe to the ISCC Transaction Monitoring. Once a subscription request is confirmed, a client will receive updates about all multi-site operations of this T-Server.

The `TTransactionMonitoring` request is used to instruct T-Server to start, stop, or modify a client's subscription to Transaction Monitoring feature notifications by setting the `TSubscriptionOperationType` parameter to `SubscriptionStart, SubscriptionStop,` or `SubscriptionModify` respectively. The transaction status is reported in `EventTransactionStatus` messages to the subscribed clients.

To determine whether the Transaction Monitoring feature is supported by a specific T-Server, a T-Server client may query T-Server's capabilities. See *Genesys 7 Events and Models Reference Manual* and *Voice Platform SDK 7.6 .NET (*or *Java) API Reference* for more information about support of this feature.

[1] The following are call-party–associated events: `EventPartyChanged`, `EventPartyDeleted`, and `EventPartyAdded`.

Configuring Multi-Site Support

Prior to configuring T-Server to support multi-site operation, you must read the “Licensing Requirements” on page 35, as well as previous sections of this chapter on multi-site deployment. In particular, Table 32 on page 283 shows which transaction types are supported by a specific T-Server, while Table 33 on page 288 shows whether your T-Server supports the NetworkCallID attribute for the ISCC/COF feature. Use this information as you follow the instructions in this chapter.

Note: Before attempting to configure a multi-site environment, Genesys recommends that you plan the changes you want to make to your existing contact centers. You should then gather the configuration information you will need (such as the names of each T-Server application, port assignments, switch names, and so on), and use Configuration Manager to create and partially configure each T-Server object. Review multi-site option values in the “Multi-Site Support Section” on page 352 and determine what these values need to be, based on your network topology.

For T-Server to support multi-site operation, you must create and configure three types of objects in the Configuration Layer:

1. Applications
2. Switches, including Access Codes
3. DNs

You must configure these objects for origination and destination locations. Multi-site support features activate automatically at T-Server startup. See “DNs” on page 314 for details.

Applications

Ensure that T-Server Application objects, and their corresponding Host objects, exist and are configured for origination and destination locations.

Once you’ve done that, use Configuration Manager to add this configuration to a T-Server Application.

Procedure:
Configuring T-Server Applications

Purpose: To configure T-Server Application objects for multi-site operation support.

Start of procedure

1. Open the T-Server Application's Properties dialog box.
2. Click the Connections tab and click Add to add a connection to the appropriate T-Server. The Connection Info Properties dialog box displays.
3. Use the Browse button to search for the T-Server you want to connect to, and fill in the following values:
   - Port ID
   - Connection Protocol
   - Local Timeout
   - Remote Timeout
   - Trace Mode
4. Click the Options tab. Create a new section called extrouter or open an existing section with this name.

   Note: If you do not create the extrouter section, T-Server works according to the default values of the corresponding configuration options.

5. Open the extrouter section. Configure the options used for multi-site support.

   Note: For a list of options and valid values, see "Multi-Site Support Section" on page 352, in the "T-Server Common Configuration Options" chapter in Part Two of this document.

6. When you are finished, click Apply.
7. Repeat this procedure for all T-Servers for origination and destination locations that are used for multi-site operations.

End of procedure

Next Steps

- See "Switches and Access Codes."

Switches and Access Codes

Ensure that Switching Office and Switch objects are configured for both origination and destination locations.

You configure Access Codes to a destination switch in the origination Switch's Properties dialog box. The only exception is the Default Access Code, which is configured at the destination Switch's Properties dialog box.

You can configure two types of switch `Access Codes` in the `Switch's Properties` dialog box:

- A `Default Access Code` (for inbound calls)—Specifies the access code that other switches can use to access this switch when they originate a multi-site transaction.
- An `Access Code` (for outbound calls)—Specifies the access code that this switch can use when it originates a multi-site transaction to access another switch.

When the origination T-Server processes a multi-site transaction, it looks for an access code to the destination switch. First, T-Server checks the `Access Code` of the origination `Switch`:

- If an access code to the destination switch is configured with the target type `Target ISCC` and with any transaction type except `Forbidden`, T-Server uses this access code to dial the destination switch.
- If the access code to the destination switch is not configured on the `Access Code` tab of the origination switch, the origination T-Server checks the `Default Access Code` tab of the destination switch. If an access code is configured there with the target type `Target ISCC` and with any transaction type except `Forbidden`, T-Server uses this access code to dial the destination switch.
- If no access code with the required properties is found, T-Server rejects the transaction.

Note: When migrating from previous releases of T-Servers to 7.6, or when using T-Servers of different releases (including 7.6) in the same environment, see “Compatibility Notes” on page 313.

Procedure: Configuring Default Access Codes

Purpose: To configure the `Default Access Codes` (one per `Switch` object) to be used by other switches to access this switch when they originate a multi-site transaction.

Prerequisites

- Ensure that `Switching Office` and `Switch` objects are configured for both origination and destination locations.

Start of procedure

1. Among configured Switches, select the Switch that the configured T-Server relates to.
2. Open the Switch Properties dialog box and click the Default Access Codes tab.
3. Click Add to open the Access Code Properties dialog box.
4. In the Code field, specify the access code used by remote switches to reach a DN at this switch. An access code is used as a prefix to the remote switch numbers.

   Note: If no prefix is needed to dial to the configured switch, you can leave the Code field blank.

5. In the Target Type field, select Target ISCC.
6. In the Route Type field, select a value corresponding to the transaction type you want to use (given that it is supported for your switch type).
7. When you are finished, click Apply.

End of procedure

Next Steps

- See “Configuring Access Codes.”

Procedure: Configuring Access Codes

Purpose: To configure the Access Codes (one or more per Switch object) that this switch can use when it originates a multi-site transaction to access another switch.

Prerequisites

- Ensure that Switching Office and Switch objects are configured for both origination and destination locations.

Start of procedure

1. Among configured Switches, select the Switch that the configured T-Server relates to.
2. Open the Switch Properties dialog box and click the Access Codes tab.
3. Click Add to open the Access Code Properties dialog box.

4. In the Switch field, specify the switch that this switch can reach using this access code. Use the Browse button to locate the remote switch.
5. In the Code field, specify the access code used to reach a DN at the remote switch from this switch. An access code is used as a prefix to the remote switch numbers.

Note: If no prefix is needed to dial from one switch to another, you can leave the Code field blank.

6. In the Target Type field, select Target ISCC.

   When you select Target ISCC as your target type, the Properties dialog box changes its lower pane to the Sources pane. It is here that you enter the extended parameters for your access codes, by specifying the ISCC Protocol and ISCC Call Overflow Parameters.

   To set these parameters, locate the two drop-down boxes that appear below the Target Type field in the Sources pane of that Properties dialog box.

   a. In the ISCC Protocol Parameters drop-down box, enter the appropriate ISCC Protocol parameter, as a comma-separated list of one or more of the following items shown in Table 34:

Table 34: Target Type: ISCC Protocol Parameters

ISCC Protocol Parameters	Description
dnis-tail=<number-of-digits>	Where <number-of-digits> is the number of significant DNIS digits (last digits) used for call matching 0 (zero) matches all digits.
propagate=<yes, udata, party, no>	Default is yes. For more information, see “Modifying Event Propagation: advanced configuration” on page 305.
direct-network-callid=<>	For configuration information, see Part Two of this document. (Use Table 32 on page 283 to determine if your T-Server supports the direct-network-callid transaction type.)

   b. In the ISCC Call Overflow Parameters drop-down box, enter call overflow parameters, as a comma-separated list of one or more of the following items shown in Table 35:

Table 35: Target Type: ISCC Call Overflow Parameters

ISCC Call Overflow Parameters	Description
`match-callid`	Matches calls using network `CallID`.
`match-ani`	Matches calls using ANI.
`inbound-only=<boolean>`	Default is `true`. Setting `inbound-only` to `true` disables COF on consultation and outbound calls.

7. In the `Route Type` field, select a value corresponding to the transaction type you want to use (given that it is supported for your switch type). Table 36 contains cross-reference information on transaction types that the Configuration Layer and T-Server use.

Table 36: Route Type and ISCC Transaction Type Cross-Reference

Route Type Field Value	ISCC Transaction Type
Default	The first value from the list of values specified in the `cast-type` option for the T-Server at the destination site
Direct	`direct-callid`
Direct ANI	`direct-ani`
Direct Digits	`direct-digits`
Direct DNIS and ANI	Reserved
Direct Network Call ID	`direct-network-callid`
Direct No Token	`direct-notoken`
Direct UUI	`direct-uui`
DNIS Pooling	`dnis-pooling`
Forbidden	External routing to this destination is not allowed
ISCC defined protocol	Reserved
PullBack	`pullback`

Table 36: Route Type and ISCC Transaction Type Cross-Reference (Continued)

Route Type Field Value	ISCC Transaction Type
Re-Route	`reroute`
Route	`route`

8. When you are finished, click `Apply`.

End of procedure

Next Steps

- After configuring a switch for multi-site support, proceed with the configuration of DNs assigned to this switch.

Compatibility Notes

When migrating from previous releases of T-Servers to 7.6, or when using T-Servers of different releases (including 7.6) in the same environment, keep in mind the following compatibility issues:

- The `Target External Routing Point` value of the `Target Type` field is obsolete and provided only for backward compatibility with T-Servers of releases 5.1 and 6.0. When two access codes for the same switch are configured, one with the `Target ISCC` target type and the other with the `Target External Routing Point` target type, T-Servers of releases 7.x, 6.5, and 6.1:
  - Use the `Target ISCC` access code for transactions with T-Servers of releases 7.x, 6.5, and 6.1.
  - Use the `Target External Routing Point` access code for transactions with T-Servers of releases 5.1 and 6.0.

  When the only access code configured for a switch has the `Target External Routing Point` target type, T-Server uses this access code for all transactions.
- When the `Target External Routing Point` value of the `Target Type` field is configured, you must set the `Route Type` field to one of the following:
  - `Default` to enable the `route` transaction type
  - `Label` to enable the `direct-ani` transaction type
  - `Direct` to enable the `direct` transaction type

Note: The `direct` transaction type in releases 5.1 and 6.0 corresponds to the `direct-callid` transaction type in releases 6.1, 6.5, and 7.x.

  - `UseExtProtocol` to enable the `direct-uui` transaction type
  - `PostFeature` to enable the `reroute` transaction type

  These values are fully compatible with the transaction types supported in T-Server release 5.1.

- For successful multi-site operations between any two locations served by release 5.1 T-Servers, identical `Route Type` values must be set in the `Switch's Access Code Properties` dialog boxes for both the origination and destination switches.

DNs

Use the procedures from this section to configure access resources for various transaction types.

Procedure: Configuring access resources for the route transaction type

Purpose: To configure dedicated DNs required for the `route` transaction type.

Prerequisites

- Ensure that `Switching Office` and `Switch` objects are configured for both origination and destination locations.

Start of procedure

1. Under a configured `Switch`, select the `DNs` folder. From the main menu, select `File > New > DN` to create a new `DN` object.
2. On the `General` tab of the DN's `Properties` dialog box, specify the number of the configured DN as the value of the `Number` field. This value must correspond to the Routing Point number on the switch.
3. Select `External Routing Point` as the value of the `Type` field.
4. If a dialable number for that Routing Point is different from its DN name, specify the number in the `Association` field.
5. Click the `Access Numbers` tab. Click `Add` and specify these access number parameters:
   - Origination switch.
   - Access number that must be dialed to reach this DN from the origination switch.

   In determining an access number for the Routing Point, T-Server composes it of the values of the following properties (in the order listed):

   a. Access number (if specified).

b. Switch access code from the switch of the origination party to the switch to which the Routing Point belongs, concatenated with its Association (if the Association value is specified).

c. Switch access code from the switch of the origination party to the switch to which the Routing Point belongs, concatenated with the number for the DN.

d. Default access code of the switch to which the Routing Point belongs, concatenated with its Association (if the Association value is specified).

e. Default access code of the switch to which the Routing Point belongs, concatenated with the number for the DN.

Note: If option use-implicit-access-numbers is set to true, the access number composed of switch access code and DN can be used for external transfers of calls originating at switches for which an access number is not specified.

6. When you are finished, click Apply.

End of procedure

Procedure: Configuring access resources for the dnis-pool transaction type

Purpose: To configure dedicated DNs required for the dnis-pool transaction type.

Start of procedure

1. Under a configured Switch, select the DNs folder. From the main menu, select File > New > DN to create a new DN object.
2. On the General tab of the DN's Properties dialog box, specify the number of the configured DN as the value of the Number field. This value must be a dialable number on the switch.
3. Select Access Resource as the Type field and type dnis as the value of the Resource Type field on the Advanced tab.
4. Click the Access Numbers tab. Click Add and specify these Access Number parameters:
   - Origination switch.

- Access number that must be dialed to reach this DN from the origination switch.

An access number for the access resource is determined in the same manner as for the `route` access resource.

5. When you are finished, click `Apply`.

End of procedure

Procedure: Configuring access resources for direct-* transaction types

Overview

You can use any configured DN as an access resource for the `direct-*` transaction types. (The `*` symbol stands for any of the following: `callid`, `uui`, `notoken`, `ani`, or `digits`.)

You can select the `Use Override` check box on the `Advanced` tab to indicate whether the override value should be used instead of the number value to dial to the DN. You must specify this value if the DN has a different DN name and dialable number. In fact, this value is required for T-Servers for some switch types—for example, Aspect ACD, Nortel Communication Server 2000/2100, and Spectrum.

Procedure: Configuring access resources for ISCC/COF

Purpose: To configure dedicated DNs required for the ISCC/COF feature.

Start of procedure

Note: Use Table 33 on page 288 to determine if your T-Server supports the ISCC/COF feature.

1. Under a configured `Switch`, select the `DNs` folder. From the main menu, select `File` > `New` > `DN` to create a new `DN` object.

   Note: The number of the access resource must match the name of a DN configured on the switch (usually, an ACD Queue) so that T-Server can determine if the calls arriving to this DN are overflowed calls.

2. On the General tab of the DN Properties dialog box, specify the number of the configured DN as the value for the Number field.
3. Select Access Resource as the value for the Type field.
4. On the Advanced tab, type cof-in or cof-not-in as the value for the Resource Type field.

Note: Calls coming to DNs with the cof-not-in value for the Resource Type are never considered to be overflowed.

5. When you are finished, click Apply.

End of procedure

Procedure: Configuring access resources for non-unique ANI

Purpose: To configure dedicated DNs required for the non-unique-ani resource type.

The non-unique-ani resource type is used to block direct-ani and COF/ani from relaying on ANI when it matches configured/enabled resource digits. Using non-unique-ani, T-Server checks every ANI against a list of non-unique-ani resources.

Start of procedure

1. Under a configured Switch, select the DNs folder. From the main menu, select File > New > DN to create a new DN object.
2. On the General tab of the DN Properties dialog box, specify the ANI digits that need to be excluded from normal processing.
3. Select Access Resource as the value for the Type field.
4. On the Advanced tab, specify the Resource Type field as non-unique-ani.
5. When you are finished, click Apply.

End of procedure

Procedure: Modifying DNs for isolated switch partitioning

Purpose: To modify DNs that belong to a particular partition where switch partitioning is used.

This configuration instructs T-Server to select an External Routing Point that has the same partition as the requested destination DN.

Note: When a target DN is not configured or has no configured partition name, T-Server allocates a DN of the External Routing Point type that belongs to any partition.

Start of procedure

1. Under a Switch object, select the DNs folder.
2. Open the Properties dialog box of a particular DN.
3. Click the Annex tab.
4. Create a new section named TServer.
5. Within that section, create a new option named epn. Set the option value to the partition name to which the DN belongs.
6. Repeat Steps 1–5 for all DNs, including DNs of the External Routing Point type, that belong to the same switch partition.
7. When you are finished, click Apply.

End of procedure

Configuration Examples

This section provides two configuration examples and describes how the configuration settings affect T-Server's behavior.

Multiple Transaction Types

This example demonstrates the difference in how ISCC directs a call when you specify two different transaction types (route and direct-ani).

In this example, you configure an origination and a destination switch for as described in "Switches and Access Codes" on page 308.

1. Among configured Switches, select the origination Switch.
2. Open the Switch Properties dialog box and click the Default Access Codes tab.
3. Click Add to open the Access Code Properties dialog box.
4. Set the Access Code field to 9.
5. When you are finished, click Apply.
6. Among configured Switches, select the destination Switch.
7. Under the destination Switch, configure a DN as described in "Configuring access resources for the route transaction type" on page 314.

8. Set the DN Number field to 5001234567.
9. Click the Advanced tab of this DN's Properties dialog box.
10. Select the Use Override check box and enter 1234567 in the Use Override field.
11. When you are finished, click Apply or Save.
12. Use a T-Server client application to register for this new DN with the destination T-Server and, therefore, with the switch.
13. Request to route a call from any DN at the origination switch to the destination DN you have just configured:
    - If you are using the route ISCC transaction type, the client requests that T-Server deliver a call to a destination location using the DN number 5001234567. ISCC requests that the switch dial one of the external routing points at the destination location, using the value either of the Access Number field or of the Access Code field, which is 9, concatenated with the external routing point at the destination location. The call is routed to the DN number 5001234567.
    - If you are using the direct-ani ISCC transaction type, the client requests that T-Server deliver a call to a destination location using the DN number 1234567, which is the Use Override value. ISCC requests that the switch dial 91234567, which is a combination of the Switch Access Code value and the Use Override value. The destination T-Server is waiting for the call to directly arrive at DN number 5001234567.

Call Overflow Methods

This section demonstrates how to indicate which overflow methods a switch supports.

In this example, for T-Server to use ANI/OtherDN matching in call overflow and manual transfer scenarios, set the ISCC Call Overflow Parameters to:

```
match-ani, inbound-only=true
```

when configuring Switch Access Codes as described on page 310.

With this setting, the switch's location is queried for call data each time the destination T-Server receives an inbound call with the ANI or OtherDN attribute.

For T-Server to use NetworkCallID matching in call overflow and manual transfer scenarios, set the ISCC Call Overflow Parameters to (for example):

```
match-callid, inbound-only=false
```

when configuring Switch Access Codes as described on page 310.

With this setting, the switch's location is queried for call data each time the destination T-Server receives a call of any type (including inbound) with the NetworkCallID attribute.

Next Steps

Continue with Chapter 4, “Starting and Stopping SIP Server,” on page 69 to test your configuration and installation.

Chapter

11 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:

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 it is otherwise specified in this document or in the documentation for your application, you set common configuration options in Configuration Manager in the corresponding sections on the `Options` tab of the `Application` object.

Warning! Configuration section names, configuration option names, and predefined option values are case-sensitive. Type them in the Configuration Manager interface 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 328.

Note: For definitions of the Standard, Interaction, Trace, and Debug log levels, refer to the *Framework 7.6 Deployment Guide* or to *Framework 7.6 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 328). 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`–`100`.
`<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 7.6 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 328). 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 328.

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 the compatible-output-priority 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, 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 332.

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

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.

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.

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. Use this configuration when trying to reproduce an application's failure. The memory log file will contain a snapshot of the application's log at the moment of failure; this should help you and Genesys Technical Support identify the reason for the 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 following options 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` (see page 338).

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 *URS 7.6 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

This section must be called `log-filter`.

default-filter-type

Default Value: `copy`
Valid Values:

`copy`	The keys and values of the KVList pairs are copied to the log.
`hide`	The keys of the KVList pairs are copied to the log; the values are replaced with strings of asterisks.
`skip`	The KVList pairs are not copied to the log.

Changes Take Effect: Immediately

Specifies the default way of presenting KVList information (including `UserData`, `Extensions`, and `Reasons`) in the log. The selected option will be applied to the attributes of all KVList pairs except the ones that are explicitly defined in the `log-filter-data` section.

Example

```
[log-filter]
default-filter-type=copy
```

Here is an example of a log using the default log filter settings:

```
message RequestSetCallInfo
   AttributeConsultType        3
   AttributeOriginalConnID     008b012ece62c8be
   AttributeUpdateRevision     2752651
   AttributeUserData           [111] 00 27 01 00
         'DNIS'                '8410'
         'PASSWORD'            '111111111'
         'RECORD_ID'           '8313427'
   AttributeConnID             008b012ece62c922
```

Log-Filter-Data Section

This section must be called `log-filter-data`.

<key name>

Default Value: `copy`
Valid Values:

`copy`	The key and value of the given KVList pair are copied to the log.
`hide`	The key of the given KVList pair is copied to the log; the value is replaced with a string of asterisks.
`skip`	The KVList pair is not copied to the log.

Changes Take Effect: Immediately

Specifies the way of presenting the KVList pair defined by the key name in the log. Specification of this option supersedes the default way of KVList presentation as defined in the `log-filter` section for the given KVList pair.

Note: If the T-Server common configuration option `log-trace-flag` is set to `-udata`, it will disable writing of user data to the log regardless of settings of any options in the `log-filter-data` section.

Example

```
[log-filter-data]
PASSWORD=hide
```

Here is an example of the log with option `PASSWORD` set to `hide`:

```
message RequestSetCallInfo
   AttributeConsultType        3
   AttributeOriginalConnID     008b012ece62c8be
   AttributeUpdateRevision     2752651
   AttributeUserData           [111] 00 27 01 00
         'DNIS'                '8410'
         'PASSWORD'            '****'
         'RECORD_ID'           '8313427'
   AttributeConnID             008b012ece62c922
```

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.

Use this option only with T-Servers.

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.

Changes from 7.5 to 7.6

Table 37 provides all the changes to common configuration options between release 7.5 and the latest 7.6 release.

Table 37: Common Log Option Changes from 7.5 to 7.6

Option Name	Option Values	Type of Change	Details
Log Section			
Use the following options only when requested by Genesys Technical Support.			
x-conn-debug-open	0, 1	New	See the description on page 333.
x-conn-debug-select	0, 1	New	See the description on page 334.
x-conn-debug-timers	0, 1	New	See the description on page 334.
x-conn-debug-write	0, 1	New	See the description on page 334.
x-conn-debug-security	0, 1	New	See the description on page 334.
x-conn-debug-api	0, 1	New	See the description on page 335.
x-conn-debug-dns	0, 1	New	See the description on page 335.
x-conn-debug-all	0, 1	New	See the description on page 335.
Extended Log Section (New Section)			
level-reassign-<eventID>	alarm, standard, interaction, trace, debug, none	New	See the description on page 336.
level-reassign-disable	true, false	New	See the description on page 338.
Common Section (New Section)			
Use the following options only when requested by Genesys Technical Support.			

Table 37: Common Log Option Changes from 7.5 to 7.6 (Continued)

Option Name	Option Values	Type of Change	Details
enable-async-dns	off, on	New	Use only with T-Servers. See the description on page 340.
rebind-delay	10–600	New	See the description on page 340.

Chapter

12 T-Server Common Configuration Options

This chapter describes the configuration options that are generally common to all T-Server types except where noted. It contains the following sections:

T-Server also supports common log options described in Chapter 11, "Common Configuration Options," on page 321.

Setting Configuration Options

Unless it is specified otherwise, you set configuration options in Configuration Manager in the corresponding sections on the `Options` tab for the T-Server `Application` object.

Mandatory Options

Except as noted for certain environments, the configuration of common options is not required for basic T-Server operation.

T-Server Section

The T-Server section contains the configuration options that are used to support the core features common to all T-Servers.

TServer This section must be called `TServer`.

ani-distribution

Default Value: `inbound-calls-only`
Valid Values: `inbound-calls-only, all-calls, suppressed`
Changes Take Effect: Immediately

Controls the distribution of the ANI information in `TEvent` messages. When this option is set to `all-calls,` the `ANI` attribute will be reported for all calls for which it is available. When this option is set to `suppressed,` the `ANI` attribute will not be reported for any calls. When this option is set to `inbound-calls-only,` the `ANI` attribute will be reported for inbound calls only.

background-processing

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

When set to `true`, T-Server processes all client requests in the background, giving higher priority to the rest of the messages. This ensures that it processes these messages without any significant delay.

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

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

When set to `false`, T-Server processes multiple requests from one T-Server client before proceeding to the requests from another T-Server client, and so on.

Note: Use of this option can negatively impact T-Server processing speed.

background-timeout

Default Value: `60 msec`
Valid Values: See "Timeout Value Format" on page 365.
Changes Take Effect: Immediately

Specifies the time interval that T-Server waits before processing client requests in background mode. You must set the `background-processing` option to `true` in order for this option to take effect.

check-tenant-profile

Default Value: `false`
Valid Values: `true`, `false`
Changes Take Effect: For the next connected client

When set to `true`, T-Server checks whether a client provides the correct name and password of a tenant. If it does, T-Server allows that client to register DNs that are included in the switch configuration in the Configuration Database, but it does not allow the client to register DNs that are *not* included in the switch configuration.

Note: To make T-Server compatible with 3.x and 5.x clients, set the `check-tenant-profile` option to `false`.

compatibility-port

Default Value: `0`
Valid Values: `0` or any valid TCP/IP port
Changes Take Effect: After T-Server has reconnected to the link

Specifies the TCP/IP port that 3.x clients use to establish connections with T-Server. Connections to this port are accepted only if T-Server has a connection with the switch. If set to `0` (zero), this port is not used.

Note: Starting with release 7.5, 3.x clients are no longer supported. You can use this option for backward compatibility with the previous T-Server releases.

consult-user-data

Default Value: `separate`
Valid Values:

`separate`	Stores user data for original and consultation calls in separate structures. The data attached to the original call is available for review or changes only to the parties of that call. The data attached to the consultation call is available only to the parties of the consultation call.
`inherited`	Copies user data from an original call to a consultation call when the consultation call is created; thereafter, stores user data separately for the original and the consultation call. Changes to the original call's user data are not available to the parties of the consultation call, and vice versa.
`joint`	Stores user data for an original call and a consultation call in one structure. The user data structure is associated with the original call, but the parties of both the original and consultation calls can see and make changes to the common user data.

Changes Take Effect: For the next consultation call created

Specifies the method for handling user data in a consultation call.

Note: A T-Server client can also specify the `consult-user-data` mode in the `Extensions` attribute `ConsultUserData` key for a conference or transfer request. If it is specified, the method of handling user data is based on the value of the `ConsultUserData` key-value pair of the request and takes precedence over the T-Server `consult-user-data` option. If it is not specified in the client request, the value specified in the `consult-user-data` option applies.

customer-id

Default Value: No default value. (A value must be specified for a multi-tenant environment.)
Valid Values: Any character string
Changes Take Effect: Immediately

Identifies the T-Server customer. You must set this option to the name of the tenant that is using this T-Server. You must specify a value for this option if you are working in a multi-tenant environment.

Note: Do not configure the `customer-id option` for single-tenant environments.

log-trace-flags

Default Value: `+iscc, +cfg$dn, -cfgserv, +passwd, +udata, -devlink, -sw, -req, -callops, -conn, -client`
Valid Values (in any combination):

`+/-iscc`	Turns on/off the writing of information about Inter Server Call Control (ISCC) transactions.
`+/-cfg$dn`	Turns on/off the writing of information about DN configuration.
`+/-cfgserv`	Turns on/off the writing of messages from Configuration Server.
`+/-passwd`	Turns on/off the writing of information about passwords.
`+/-udata`	Turns on/off the writing of attached data.
`+/-devlink`	Turns on/off the writing of information about the link used to send CTI messages to the switch (for multilink environments).
`+/-sw`	Reserved by Genesys Engineering.
`+/-req`	Reserved by Genesys Engineering.
`+/-callops`	Reserved by Genesys Engineering.
`+/-conn`	Reserved by Genesys Engineering.
`+/-client`	Turns on/off the writing of additional information about the client’s connection.

Changes Take Effect: Immediately

Specifies—using a space-, comma- or semicolon-separated list—the types of information that are written to the log files.

management-port

Default Value: `0`
Valid Values: `0` or any valid TCP/IP port
Changes Take Effect: After T-Server is restarted

Specifies the TCP/IP port that management agents use to communicate with T-Server. If set to `0` (zero), this port is not used.

merged-user-data

Default Value: `main-only`
Valid Values:

`main-only`	T-Server attaches user data from the remaining call only.
`merged-only`	T-Server attaches user data from the merging call.
`merged-over-main`	T-Server attaches user data from the remaining and the merging call. In the event of equal keys, T-Server uses data from the merging call.
`main-over-merged`	T-Server attaches data from the remaining and the merging call. In the event of equal keys, T-Server uses data from the remaining call.

Changes Take Effect: Immediately

Specifies the data that is attached to the resulting call after a call transfer, conference, or merge completion.

Note: The option setting does not affect the resulting data for merging calls if the `consult-user-data` option is set to `joint`. (See “consult-user-data” on page 346.)

server-id

Default Value: An integer equal to the `ApplicationDBID` as reported by Configuration Server
Valid Values: Any integer from `0–16383`
Changes Take Effect: Immediately

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

Warning! Genesys does not recommend using multiple instances of the Configuration Database.

user-data-limit

Default Value: `16000`
Valid Values: `0–65535`
Changes Take Effect: Immediately

Specifies the maximum size (in bytes) of user data in a packed format.

Note: When T-Server works in mixed 7.x/6.x environment, the value of this option must not exceed the default value of `16000` bytes; otherwise, 6.x T-Server clients might fail.

License Section

The License section contains the configuration options that are used to configure T-Server licenses. They set the upper limit of the seat-related DN licenses (`tserver_sdn`) that T-Server tries to check out from a license file. See “License Checkout” on page 350.

license This section must be called `license`.

Notes: T-Server also supports the `license-file` option described in the *Genesys 7 Licensing Guide.*

The `License` section is not applicable to Network T-Server for DTAG.

If you use two or more T-Servers, and they share licenses, you must configure the following options in the `license` section of the T-Servers.

num-of-licenses

Default Value: `0` or `max` (all available licenses)
Valid Values: `0` or string `max`
Changes Take Effect: Immediately

Specifies how many DN licenses T-Server checks out. T-Server treats a value of `0` (zero) the same as it treats `max`—that is, it checks out all available licenses.

The sum of all `num-of-licenses` values for all concurrently deployed T-Servers must not exceed the number of seat-related DN licenses (`tserver_sdn`) in the corresponding license file. The primary and backup T-Servers share the same licenses, and therefore they need to be counted only once. T-Server checks out the number of licenses indicated by the value for this option, regardless of the number actually in use.

num-sdn-licenses

Default Value: `0` or `max` (All DN licenses are seat-related)
Valid Values: String `max` (equal to the value of `num-of-licenses`), or any integer from `0`–`9999`
Changes Take Effect: Immediately

Specifies how many seat-related licenses T-Server checks out. A value of `0` (zero) means that T-Server does not grant control of seat-related DNs to any client, and it does not look for seat-related DN licenses at all.

The sum of all `num-sdn-licenses` values for all concurrently deployed T-Servers must not exceed the number of seat-related DN licenses (`tserver_sdn`) in the corresponding license file. The primary and backup T-Servers share the same licenses, and therefore they need to be counted only once. T-Server checks out the number of licenses indicated by the value for this option, regardless of the number actually in use.

Notes: For Network T-Servers, Genesys recommends setting this option to `0`.

Be sure to configure in the Configuration Database all the DNs that agents use (Extensions and ACD Positions) and that T-Server should control.

License Checkout

Table 38 shows how to determine the number of seat-related DN licenses that T-Server attempts to check out. See the examples on page 351.

Table 38: License Checkout Rules

Options Settings[a]		License Checkout[b]
num-of-licenses	**num-sdn-licenses**	**Seat-related DN licenses**
max (or 0)	max	all available
max (or 0)	x	x
max (or 0)	0	0
x	max	x
x	y	min (y, x)
x	0	0

[a] In this table, the following conventions are used: `x` and `y` - are positive integers; `max` is the maximum number of licenses that T-Server can check out; `min (y, x)` is the lesser of the two values defined by `y` and `x`, respectively.

[b] The License Checkout column shows the number of licenses that T-Server attempts to check out. The actual number of licenses will depend on the licenses' availability at the time of checkout, and it is limited to 9999.

Examples

This section presents examples of option settings in the `license` section.

Example 1

If...		Then...
Options Settings	**License File Settings**	**License Checkout**
num-of-licenses = max	tserver_sdn = 500	500 seat-related DNs
num-sdn-licenses = max		

Example 2

If...		Then...
Options Settings	**License File Settings**	**License Checkout**
num-of-licenses = 1000	tserver_sdn = 500	500 seat-related DNs
num-sdn-licenses = max		

Example 3

If...		Then...
Options Settings	**License File Settings**	**License Checkout**
num-of-licenses = 1000	tserver_sdn = 600	400 seat-related DNs
num-sdn-licenses = 400		

Example 4

If...		Then...
Options Settings	**License File Settings**	**License Checkout**
num-of-licenses = max	tserver_sdn = 5000	1000 seat-related DNs
num-sdn-licenses = 1000		

Agent-Reservation Section

The Agent-Reservation section contains the configuration options that are used to customize the T-Server Agent Reservation feature. See “Agent Reservation” on page 265 section for details on this feature.

agent-reservation This section must be called agent-reservation.

Note: The Agent Reservation functionality is currently a software-only feature that is used to coordinate multiple client applications. This feature does not apply to multiple direct or ACD-distributed calls.

reject-subsequent-request

Default Value: true
Valid Values:

true	T-Server rejects subsequent requests.
false	A subsequent request prolongs the current reservation made by the same client application for the same agent.

Changes Take Effect: Immediately

Specifies whether T-Server rejects subsequent requests from the same client application, for an agent reservation for the same Agent object that is currently reserved.

Note: Genesys does not recommend setting this option to false in a multi-site environment in which remote locations use the Agent-Reservation feature.

request-collection-time

Default Value: 100 msec
Valid Values: See “Timeout Value Format” on page 365.
Changes Take Effect: Immediately

Specifies the interval that agent reservation requests are collected before a reservation is granted. During this interval, agent reservation requests are delayed, in order to balance successful reservations between client applications (for example, Universal Routing Servers).

reservation-time

Default Value: 10000 msec
Valid Values: See “Timeout Value Format” on page 365.
Changes Take Effect: Immediately

Specifies the default interval that an AgentDN is reserved to receive a routed call from a remote T-Server. During this interval, the agent cannot be reserved again.

Multi-Site Support Section

The Multi-Site Support section contains the configuration options that are used to support multi-site environments with the Inter Server Call Control (ISCC) feature. The configuration options in this section are grouped with related options that support the same functionality (such as those for Transfer Connect Service or the ISCC/Call Overflow feature).

extrouter This section must be called `extrouter`.

For a description of the ways in which T-Server supports multi-site configurations and for an explanation of the configuration possibilities for a multi-site operation, see the "Multi-Site Support" chapter.

Note: In a multi-site environment, you must configure the `timeout`, `cast-type`, and `default-dn` options with the same value for both the primary and backup T-Servers. If you do not do this, the value specified for the backup T-Server overrides the value specified for the primary T-Server.

match-call-once

Default Value: `true`
Valid Values:

`true`	ISCC does not process (match) an inbound call that has already been processed (matched).
`false`	ISCC processes (attempts to match) a call as many times as it arrives at an ISCC resource or multi-site-transfer target.

Changes Take Effect: Immediately

Specifies how many times ISCC processes an inbound call when it arrives at an ISCC resource. When set to `false`, ISCC processes (attempts to match) the call even if it has already been processed.

Note: Genesys does not recommend changing the default value of the `match-call-once` option to `false` unless you have specific reasons. Setting this option to `false` may lead to excessive or inconsistent call data updates.

reconnect-tout

Default Value: `5 sec`
Valid Values: See "Timeout Value Format" on page 365.
Changes Take Effect: At the next reconnection attempt

Specifies the time interval after which a remote T-Server attempts to connect to this T-Server after an unsuccessful attempt or a lost connection. The number of attempts is unlimited. At startup, T-Server immediately attempts the first connection, without this timeout.

report-connid-changes

Default Value: `false`
Valid Values:

`true`	`EventPartyChanged` is generated.
`false`	`EventPartyChanged` is not generated.

Changes Take Effect: Immediately

Specifies whether the destination T-Server generates `EventPartyChanged` for the incoming call when the resulting `ConnID` attribute is different from the `ConnID` attribute of an instance of the same call at the origination location.

use-data-from

Default Value: `active`
Valid Values:

`active`	The values of `UserData` and `ConnID` attributes are taken from the consultation call.
`original`	The values of `UserData` and `ConnID` attributes are taken from the original call.
`active-data-original-call`	The value of the `UserData` attribute is taken from the consultation call and the value of `ConnID` attribute is taken from the original call.
`current`	If the value of `current` is specified, the following occurs: • Before the transfer or conference is completed, the `UserData` and `ConnID` attributes are taken from the consultation call. • After the transfer or conference is completed, `EventPartyChanged` is generated, and the `UserData` and `ConnID` are taken from the original call.

Changes Take Effect: Immediately

Specifies the call from which the values for the `UserData` and `ConnID` attributes are taken for a consultation call that is routed or transferred to a remote location.

Note: For compatibility with the previous T-Server releases, you can use the values `consult`, `main`, and `consult-user-data` for this option. These are aliases for `active`, `original`, and `current`, respectively.

ISCC Transaction Options

cast-type

Default Value: `route, route-uui, reroute, direct-callid, direct-uui, direct-network-callid, direct-notoken, direct-digits, direct-ani, dnis-pool, pullback`

Valid Values: `route, route-uui, reroute, direct-callid, direct-uui, direct-network-callid, direct-notoken, direct-digits, direct-ani, dnis-pool, pullback`

Changes Take Effect: For the next request for the remote service

Specifies—using a space-, comma- or semicolon-separated list—the routing types that can be performed for this T-Server.

The valid values provide for a range of mechanisms that the ISCC feature can support with various T-Servers, in order to pass call data along with calls between locations.

Because switches of different types provide calls with different sets of information parameters, some values might not work with your T-Server. See Table 32 on page 283 for information about supported transaction types by a specific T-Server. The "Multi-Site Support" chapter also provides detailed descriptions of all transaction types.

Notes: For compatibility with the previous T-Server releases, you can use the `direct` value for this option. This is an alias for `direct-callid`.

An alias, `route-notoken`, has been added to the `route` value.

default-dn

Default Value: No default value
Valid Values: Any DN
Changes Take Effect: For the next request for the remote service

Specifies the DN to which a call is routed when a Destination DN (`AttributeOtherDN`) is not specified in the client's request for routing. If neither this option nor the client's request contains the destination DN, the client receives `EventError`.

Note: This option is used only for requests with route types `route, route-uui, direct-callid, direct-network-callid, direct-uui, direct-notoken, direct-digits`, and `direct-ani`.

direct-digits-key

Default Value: `CDT_Track_Num`
Valid Values: Any valid key name of a key-value pair from the `UserData` attribute
Changes Take Effect: For the next request for the remote service

Specifies the name of a key from the `UserData` attribute that contains a string of digits that are used as matching criteria for remote service requests with the `direct-digits` routing type.

Note: For compatibility with the previous T-Server releases, this configuration option has an alias value of `cdt-udata-key`.

dn-for-unexpected-calls

Default Value: No default value
Valid Values: Any DN
Changes Take Effect: Immediately

Specifies a default DN for unexpected calls arriving on an External Routing Point.

network-request-timeout

Default Value: `20 sec`
Valid Values: See "Timeout Value Format" on page 365.
Changes Take Effect: For the next network request

For a premise T-Server, this option specifies the time interval that the premise T-Server waits for a response, after relaying a `TNetwork<...>` request to the Network T-Server. For a Network T-Server, this option specifies the time interval that the Network T-Server waits for a response from an SCP (Service Control Point), after initiating the processing of the request by the SCP.

When the allowed time expires, the T-Server cancels further processing of the request and generates `EventError`.

register-attempts

Default Value: `5`
Valid Values: Any positive integer
Changes Take Effect: For the next registration

Specifies the number of attempts that T-Server makes to register a dedicated External Routing Point.

register-tout

Default Value: `2 sec`
Valid Values: See "Timeout Value Format" on page 365.
Changes Take Effect: For the next registration

Specifies the time interval after which T-Server attempts to register a dedicated External Routing Point. Counting starts when the attempt to register a Routing Point fails.

request-tout

Default Value: `20 sec`
Valid Values: See "Timeout Value Format" on page 365.
Changes Take Effect: For the next request for remote service

Specifies the time interval that a T-Server at the origination location waits for a notification of routing service availability from the destination location. Counting starts when the T-Server sends a request for remote service to the destination site.

resource-allocation-mode

Default Value: `circular`
Valid Values:

`home`	T-Server takes an alphabetized (or numerically sequential) list of configured DNs and reserves the first available DN from the top of the list for each new request. For example, if the first DN is not available, the second DN is allocated for a new request. If the first DN is freed by the time the next request comes, the first DN is allocated for this next request.
`circular`	T-Server takes the same list of configured DNs, but reserves a subsequent DN for each subsequent request. For example, when the first request comes, T-Server allocates the first DN; when the second request comes, T-Server allocates the second DN; and so on. T-Server does not reuse the first DN until reaching the end of the DN list.

Changes Take Effect: Immediately

Specifies the manner in which T-Server allocates resources (that is, DNs of the `External Routing Point` type and Access Resources with `Resource Type dnis`) for multi-site transaction requests.

resource-load-maximum

Default Value: `0`
Valid Values: Any positive integer
Changes Take Effect: Immediately

Specifies the maximum number of ISCC routing transactions that can be concurrently processed at a single DN of the `External Routing Point` route type. After a number of outstanding transactions at a particular DN of the `External Routing Point` type reaches the specified number, T-Server considers the DN not available. Any subsequent request for this DN is queued until the number of outstanding transactions decreases. A value of `0` (zero) means that no limitation is set to the number of concurrent transactions at a single External Routing Point. In addition, the `0` value enables T-Server to perform load balancing of all incoming requests among all available External Routing Points, in order to minimize the load on each DN.

route-dn

Default Value: No default value
Valid Values: Any DN
Changes Take Effect: Immediately

Specifies the DN that serves as a Routing Point for the `route` transaction type in the multiple-to-one access mode.

timeout

Default Value: `60 sec`
Valid Values: See "Timeout Value Format" on page 365.
Changes Take Effect: For the next request for remote service

Specifies the time interval that the destination T-Server waits for a call routed from the origination location. Counting starts when this T-Server notifies the requesting T-Server about routing service availability. The timeout must be long enough to account for possible network delays in call arrival.

use-implicit-access-numbers

Default Value: `false`
Valid Values: `true`, `false`
Changes Take Effect: After T-Server is restarted

Determines whether an External Routing Point in which at least one access number is specified is eligible for use as a resource for calls coming from switches for which an access number is not specified in the External Routing Point. If this option is set to `false`, the External Routing Point is not eligible for use as a resource for calls coming from such switches. If this option is set to `true`, an implicit access number for the External Routing Point, composed of the switch access code and the DN number of the External Routing Point, will be used.

Note: If an External Routing Point does not have an access number specified, this option will not affect its use.

Transfer Connect Service Options

tcs-queue

Default Value: No default value
Valid Values: Any valid DN number
Changes Take Effect: For the next request for the remote service

Specifies the TCS DN number to which a call, processed by the TCS feature, is dialed after the originating external router obtains an access number. This option applies only if the `tcs-use` option is activated.

tcs-use

Default Value: `never`
Valid Values:

`never`	The TCS feature is not used.
`always`	The TCS feature is used for every call.
`app-defined`	In order to use the TCS feature for a multi-site call transfer request, a client application must add a key-value pair with a TC-type key and a nonempty string value to the `UserData` attribute of the request.

Changes Take Effect: Immediately

Specifies whether the Transfer Connect Service (TCS) feature is used.

Note: For compatibility with the previous T-Server releases, you can use the value `up-app-depended` for this option. This is an alias for `app-defined`.

ISCC/COF Options

cof-ci-defer-create

Default Value: `0`
Valid Values: See "Timeout Value Format" on page 365.
Changes Take Effect: Immediately

Specifies the time interval that T-Server waits for call data from the switch before generating a negative response for a call data request from a remote T-Server. If T-Server detects the matching call before this timeout expires, it sends the requested data. This option applies only if the `cof-feature` option is set to `true`.

cof-ci-defer-delete

Default Value: `0`
Valid Values: See "Timeout Value Format" on page 365.
Changes Take Effect: Immediately

Specifies the time interval that T-Server waits before deleting call data that might be overflowed. If set to `0`, deletion deferring is disabled. This option applies only if the `cof-feature` option is set to `true`.

cof-ci-req-tout

Default Value: `500 msec`
Valid Values: See "Timeout Value Format" on page 365.
Changes Take Effect: For the next COF operation

Specifies the time interval during which T-Server will wait for call data requested with respect to a call originated at another site. After T-Server sends the call data request to remote T-Servers, all events related to this call will be suspended until either the requested call data is received or the specified timeout expires. This option applies only if the `cof-feature` option is set to `true`.

cof-ci-wait-all

Default Value: `false`
Valid Values:

`true`	T-Server waits for responses from all T-Servers that might have the requested call data before updating the call data with the latest information.
`false`	T-Server updates the call data with the information received from the first positive response.

Changes Take Effect: Immediately

Specifies whether T-Server, after sending a request for matching call data, waits for responses from other T-Servers before updating the call data (such as `CallHistory`, `ConnID`, and `UserData`) for a potentially overflowed call. The waiting period is specified by the `cof-ci-req-tout` and `cof-rci-tout` options. This option applies only if the `cof-feature` option is set to `true`.

cof-feature

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

Enables or disables the Inter Server Call Control/Call Overflow (ISCC/COF) feature.

cof-rci-tout

Default Value: `10 sec`
Valid Values: See “Timeout Value Format” on page 365.
Changes Take Effect: For the next COF operation

Specifies the time interval that T-Server waits for call data from other T-Servers’ transactions. Counting starts when `cof-ci-req-tout` expires. This option applies only if the `cof-feature` option is set to `true`.

local-node-id

Default Value: `0`
Valid Values: `0` or any positive integer
Changes Take Effect: Immediately

This option, if enabled, checks all networked calls against the specified `NetworkNodeID` (the identity of the switch to which the call initially arrived). If the `NetworkNodeID` is the same as the value of this option, the request for call information is *not* sent. The default value of `0` disables the functionality of this option. To establish an appropriate `NetworkNodeID`, specify a value other than the default. This option applies only if the `cof-feature` option is set to `true`.

Note: This option applies only to T-Server for Nortel Communication Server 2000/2100 (formerly DMS-100).

default-network-call-id-matching

Default Value: No default value
Valid Values: `sip`
Changes Take Effect: Immediately

When this option is set to `sip`, SIP Server will use the content of the `X-ISCC-CofId` header for the ISCC/COF call matching.

To activate this feature, the `cof-feature` option in the `extrouter` section of the SIP Server `Application` object must also be set to `true`.

Note: This option applies only to SIP Server.

Event Propagation Option

event-propagation

Default Value: `list`
Valid Values:

`list`	Changes in user data and party events are propagated to remote locations through call distribution topology.
`off`	The feature is disabled. Changes in user data and party events are not propagated to remote locations.

Changes Take Effect: Immediately

Specifies whether the Event Propagation feature is enabled.

Number Translation Option

inbound-translator-<*n*>

Default Value: No default value.
Valid Value: Any valid name
Changes Take Effect: Immediately

Specifies the name of another configuration section as the value for the `inbound-translator` option. For example,

```
inbound-translator-1 = ani-translator
```

where `ani-translator` is the name of the configuration that describes the translation rules for inbound numbers.

Translation Rules Section

The section name is specified by the `inbound-translator-<n>` option. It contains options that define translation rules for inbound numbers.

You can choose any name for this section, provided that it matches the value of the section. Every option in this section corresponds to a rule and must conform to the format described below. You can configure as many rules as necessary to accommodate your business needs.

rule-<*n*>

Default Value: No default value
Valid Value: Any valid string in the following format:
`in-pattern=<input pattern value>;out-pattern=<output pattern value>`
Changes Take Effect: Immediately

Defines a rule to be applied to an inbound number. The two parts of the option value describe the input and output patterns in the rule. When configuring the pattern values, follow the syntax defined in "Using ABNF for Rules" on page 292. See "Configuring Number Translation" on page 298 for examples of these rules as well as detailed instructions for creating rules for your installation. For example, a value for this configuration option might look like this:

```
rule-01 = in-pattern=0111#CABBB*ccD;out-pattern=ABD
```

Backup-Synchronization Section

The Backup-Synchronization section contains the configuration options that are used to support a high-availability (`hot standby` redundancy type) configuration.

backup-sync This section must be called `backup-sync`.

Note: These options apply only to T-Servers that support the `hot standby` redundancy type.

addp-remote-timeout

Default Value: `0`
Valid Values: Any integer from `0–3600`
Changes Take Effect: Immediately

Specifies the time interval that the redundant T-Server waits for a response from this T-Server after sending a polling signal. The default value of `0` (zero) disables the functionality of this option. To establish an appropriate timeout, specify a value other than the default. This option applies only if the `protocol` option is set to `addp`.

addp-timeout

Default Value: 0
Valid Values: Any integer from 0–3600
Changes Take Effect: Immediately

Specifies the time interval that this T-Server waits for a response from another T-Server after sending a polling signal. The default value of 0 (zero) disables the functionality of this option. To establish an appropriate timeout, specify a value other than the default. This option applies only if the protocol option is set to addp.

addp-trace

Default Value: off
Valid Values:

off, false, no	No trace (default).
local, on, true, yes	Trace on this T-Server side only.
remote	Trace on the redundant T-Server side only.
full, both	Full trace (on both sides).

Changes Take Effect: Immediately

Specifies whether the option is active, and to what level the trace is performed. This option applies only if the protocol option is set to addp.

protocol

Default Value: default
Valid Values:

default	The feature is not active.
addp	Activates the Advanced Disconnect Detection Protocol.

Changes Take Effect: When the next connection is established

Specifies the name of the method used to detect connection failures. If you specify the addp value, you must also specify a value for the addp-timeout, addp-remote-timeout, and addp-trace options.

sync-reconnect-tout

Default Value: 20 sec
Valid Values: See “Timeout Value Format” on page 365.
Changes Take Effect: Immediately

Specifies the time interval after which the backup T-Server attempts to reconnect to the primary server (for a synchronized link).

Call-Cleanup Section

The Call-Cleanup section contains the configuration options that are used to control detection and cleanup of stuck calls in T-Server. For more information on stuck call handling, refer to the "Stuck Call Management" chapter in the *Framework 7.6 Management Layer User's Guide*.

call-cleanup This section must be called `call-cleanup`.

cleanup-idle-tout

Default Value: `0`
Valid Values: See "Timeout Value Format" on page 365.
Changes Take Effect: Immediately

Specifies the time interval that T-Server waits for a call to be updated from its last update. After this time elapses, if no new events about the call are received, T-Server clears this call as a stuck call, either by querying the switch (if a CTI link provides such capabilities) or by deleting the call information from memory unconditionally. The default value of `0` disables the stuck calls cleanup.

notify-idle-tout

Default Value: `0`
Valid Values: See "Timeout Value Format" on page 365.
Changes Take Effect: Immediately

Specifies the time interval that T-Server waits for a call to be updated from its last update. After this time elapses, if no new events about the call are received, T-Server reports this call as a stuck call. The default value of `0` disables the stuck calls notification.

periodic-check-tout

Default Value: `10 min`
Valid Values: See "Timeout Value Format" on page 365.
Changes Take Effect: Immediately

Specifies the time interval for periodic checks for stuck calls. These checks affect both notification and cleanup functionality, and are made by checking the T-Server's own call information with call information available in the switch. For performance reasons, T-Server does not verify whether the `notify-idle-tout` or `cleanup-idle-tout` option has expired before performing this checking.

Note: Setting this option to a value of less than a few seconds can affect T-Server performance.

Examples

This section presents examples of option settings in the `call-cleanup` section.

Example 1

```
cleanup-idle-tout = 0
notify-idle-tout = 0
```

```
periodic-check-tout = 10
```

With these settings, T-Server will not perform any checks for stuck calls.

Example 2

```
cleanup-idle-tout = 0
notify-idle-tout = 5 min
periodic-check-tout = 10 min
```

With these settings, T-Server performs checks every 10 minutes and sends notifications about all calls that have been idle for at least 5 minutes.

Example 3

```
cleanup-idle-tout = 20 min
notify-idle-tout = 5 min
periodic-check-tout = 10 min
```

With these settings, T-Server performs checks every 10 minutes, sends notifications about all calls that have been idle for at least 5 minutes, and attempts to clean up all calls that have been idle for more than 20 minutes.

Security Section

The `Security` section contains the configuration options that are used to configure secure data exchange between T-Servers and other Genesys components. Refer to the *Genesys 7.6 Security Deployment Guide* for complete information on the security configuration.

Timeout Value Format

This section of the document describes the values to use for those T-Server common options that set various timeouts. The current format allows you to use fractional values and various time units for timeout settings.

For timeout-related options, you can specify any value that represents a time interval, provided that it is specified in one of the following formats:

`[[[`*`hours`*`:]`*`minutes`*`:]`*`seconds`*`][`*`milliseconds`*`]`

or

`[`*`hours`*` hr][`*`minutes`*` min][`*`seconds`*` sec][`*`milliseconds`*` msec]`

Where a time unit name in italic (such as *hours*) is to be replaced by an integer value for this time unit.

Integer values with no measuring units are still supported, for compatibility with previous releases of T-Server. When you do not specify any measuring units, the units of the default value apply. For example, if the default value equals `60 sec`, specifying the value of `30` sets the option to 30 seconds.

Example 1

The following settings result in a value of 1 second, 250 milliseconds:

```
sync-reconnect-tout = 1.25
sync-reconnect-tout = 1 sec 250 msec
```

Example 2

The following settings result in a value of 1 minute, 30 seconds:

```
timeout = 1:30
timeout = 1 min 30 sec
```

Changes from Release 7.5 to 7.6

Table 39 lists the configuration options that:

- Are new or changed in the 7.6 release of T-Server
- Have been added or changed since the most recent 7.5 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 39: Option Changes from Release 7.5 to 7.6

Option Name	Option Values	Type of Change	Details
TServer Section			
ani-distribution	inbound-calls-only, all-calls, suppressed	New	See the option description on page 344.
compatibility-port	0 or any valid TCP/IP port	Obsolete	See the option description on page 345.
extrouter Section			
use-data-from	active, original, current, active-data-original-call	New value	New option value, `active-data-original-call`. See the option description on page 354.
default-network-call-id-matching	sip	New	See the option description on page 361.
backup-sync Section			
network-provided-address	true, false	Obsolete	

Part

3

Part Three: DMX Reference Information

Part Three of this *SIP Server Deployment Guide* contains reference information specific to the Distributed Media eXchange (DMX) application. The information is divided among these chapters:

- Chapter 13, "DMX Deployment," on page 369, presents configuration and installation procedures for DMX.
- Chapter 14, "DMX Reference," on page 393, presents reference information for DMX.

New in DMX for 7.6

There are no new features available in the initial 7.6 release of DMX.

Chapter

13 DMX Deployment

This chapter describes the deployment of Distributed Media eXchange (DMX). It contains the following sections:

Overview

Distributed Media eXchange (DMX) negotiates codecs and converts protocols between SIP and endpoints using H.323 (H.225/245). It also enables the exchange of audio and video media types across many networks, such as PSTN, PBX, LAN, and WAN.

Note: Microsoft NetMeeting and Windows Messenger are not supported with DMX beginning with release 7.2.

DMX General Network Modes

DMX can perform in Traditional and Load Balancing modes.

Traditional Mode

In this mode, DMX is deployed with or without SIP Server (or any third-party SIP Service). Several combinations are possible, and in all combinations DMX works as a proxy between H.323 and SIP networks. DMX can send requests to different destinations depending on the scenario used:

Scenario One

In the first scenario (Figure 37), DMX is deployed between the gateway/PSTN and SIP Server and functions as a proxy between SIP Server and the H.323 gateway.

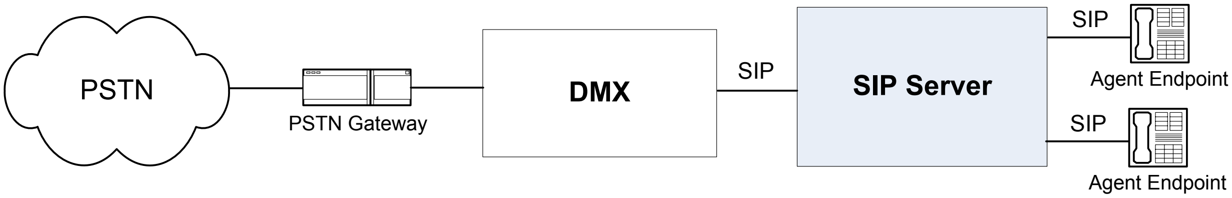


Figure 37: Scenario One: DMX Traditional Mode

For each call DMX would have to find the proper destination to send messages:

- If the destination is SIP Server, DMX obtains its IP address using the link from the `Connections` tab from Configuration Manager. If there are several servers listed, DMX gives preference to the first SIP Server.
- If the destination is H.323, DMX resolves the IP address of the destination gateway from the `TO` header within the incoming `INVITE` message from SIP Server. For example, if the `TO` header of the `INVITE` message is defined as 5650@192.168.83.6, then the H.323 `SETUP` message would be sent to IP address 192.168.83.6.
- If SIP Server modifies the Request-URI so that IP address in the Request-URI differs from `TO` header, DMX processes the IP address from the Request URI header. For example, if the `TO` header is defined as 5650@192.168.83.6, but the Request-URI is defined as 5650@192.168.83.76, then H.323 `SETUP` message would be sent to IP address 192.168.83.76.

Scenario Two

In the second scenario (Figure 38), DMX is deployed as a proxy between SIP Server and H.323 caller endpoints.

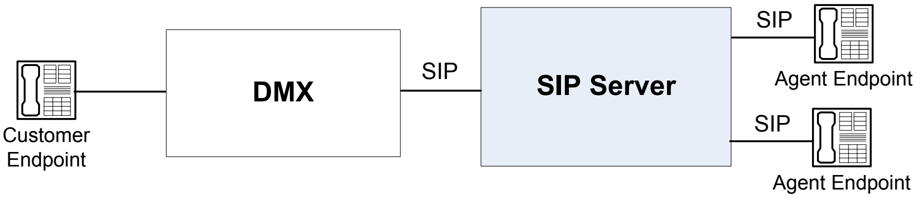


Figure 38: Scenario Two: DMX Traditional Mode

For each call, DMX needs to find the proper destination to send messages:

- For inbound calls from H.323 agents, DMX obtains its IP address using the link from the `Connections` tab from Configuration Manager. If there are several servers listed, DMX gives preference to the first SIP Server.
- For inbound calls from the SIP Server, DMX resolves the IP address from the number/alias established by the URI if the destination URI of the incoming `INVITE` message does not have a direct IP address. To do this, the H.323 client should have been previously registered by RAS. For example, if a H.323 client has registered with DMX as number 5650, with an IP address of 192.168.83.6, the destination URI of the incoming `INVITE` message is defined as 5650 and the H.323 `SETUP` message is sent to IP 192.168.83.6.
- For inbound calls from a SIP Server or SIP Client that includes an `INVITE` message with a direct IP address of the H.323 client, the IP addresses could be retrieved using these methods:
  - If the IP address from the `TO` header is similar to the DMX IP address, DMX resolves the IP address from the number/alias established by the URI if the destination URI of the incoming `INVITE` message does not have a direct IP address. To do this, the H.323 client should have been previously registered by RAS. For example, if an H.323 client has registered with DMX as number 5650, with an IP address of 192.168.83.6, the destination URI of the incoming `INVITE` message is defined as 5650 and the H.323 `SETUP` message is sent to IP address 192.168.83.6.
  - If the IP address from the `TO` header is not equal to the DMX IP address, DMX uses the IP address from the URI path within the incoming `INVITE` message. For example, if the destination URI of the `INVITE` message is defined as 5650@192.168.83.76 and the DMX was started at host identified by IP address 192.168.83.7, the H.323 `SETUP` message would be sent to IP 192.168.83.76.
  - If the `TO` header field specifies the desired “logical” recipient of the request, SIP Server can affect the Request-URI of the message. Therefore the IP in the Request-URI field can differ from the IP in `TO` header. In this scenario, the Request-URI field has higher priority and DMX processes it only. For example, if the `TO` header is defined as 5650@192.168.83.6, but the Request-URI is defined as 5650@192.168.83.76, then an H.323 `SETUP` message would be sent to IP address 192.168.83.76.

Scenario Three

In the third scenario (Figure 39), DMX is deployed only as a proxy between the H.323 agents and the SIP agents.

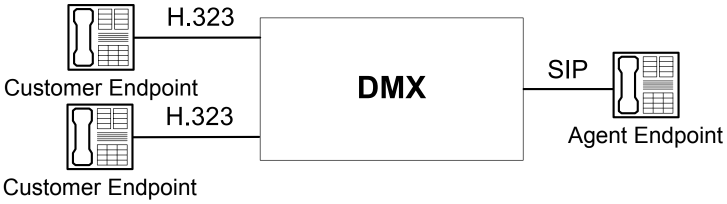


Figure 39: Scenario Three: DMX Traditional Mode

The H.323 agent must register with DMX before any calls are made. DMX would have to find the proper destination to send messages for each call:

- For inbound calls from H.323 agents, DMX obtains its IP address by using the link from the `Connections` tab from Configuration Manager. If there are several servers listed, DMX gives preference to the first SIP Server.
- For inbound calls from SIP agents, DMX resolves the IP address from the number/alias established by the URI if the destination URI of the incoming `INVITE` message does not have a direct IP address. To do this, the H.323 client should have been previously registered by RAS. For example, if an H.323 client has registered with DMX as number 5650, with an IP address of 192.168.83.6, the destination URI of the incoming `INVITE` message is defined as 5650 and the H.323 `SETUP` message is sent to IP address 192.168.83.6.
- For inbound calls from a SIP Server or SIP Client that includes an `INVITE` message with a direct IP address of the H.323 client, the IP addresses could be retrieved using these methods:
    - If the IP address from the `TO` header is similar to the DMX IP address, DMX resolves the IP address from the number/alias established by the URI if the destination URI of the incoming `INVITE` message does not have a direct IP address. To do this, the H.323 client should have been previously registered by RAS. For example, if a H.323 client has registered with DMX as number 5650, with an IP address of 192.168.83.6, the destination URI of the incoming `INVITE` message is defined as 5650 and the H.323 `SETUP` message is sent to IP address 192.168.83.6.
    - If the IP address from the `TO` header is not equal to the DMX IP address, DMX uses the IP address from the URI path within the incoming `INVITE` message. For example, if the destination URI of the `INVITE` message is defined as 5650@192.168.83.76 and the DMX was started at a host identified by IP address 192.168.83.7, the H.323 `SETUP` message would be sent to IP address 192.168.83.76.

Load Balancing Mode

In addition to Traditional mode, DMX can work as a gatekeeper with a load balancing mechanism. In this scenario, DMX can distribute H.323 calls between DMX clients which reduces the call load among several DMXs. All DMX clients must be registered with DMX gatekeeper.

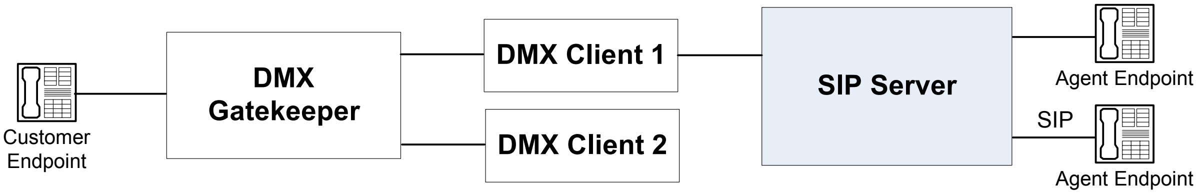


Figure 40: DMX Load Balancing Mode

If the H.323 agent sends a RAS Admission Request to the DMX gatekeeper, DMX transfers the call to the DMX client with its IP address in the RAS Admission Confirm response. If the H.323 agent sends a `SETUP` message to the DMX gatekeeper, DMX processes the call. Therefore, DMX can work either as a DMX gatekeeper or as a DMX client simultaneously, depending on the message received: if DMX receives a `SETUP` message, it processes it and performs in Traditional mode; otherwise, DMX works as a gatekeeper.

DMX gatekeeper uses a comma-separated list in the `services` option of the `Gatekeeper` section as the determination for call distribution among client DMXs. Ordinarily, calls are distributed in round-robin fashion among all DMXs registered with a gatekeeper, but if a client DMX has a service number specified, all calls matching that number are sent only to that client. The option `access-registrations` in the properties of the DMX gatekeeper should be set to `true`.

DMX client determines gatekeeper registration from the `primary-gk-host` and `primary-gk-port` options in the `Gatekeeper` section of the `Options` tab in Configuration Manager.

DMX Configuration

You need to ensure your system has the following components installed and configured in order to properly operate DMX:

- Genesys Framework components including DB Server, Configuration Server, and Configuration Manager. Genesys Framework includes SIP Server, which is the most important part of the Media Layer.
    - This layer enables Genesys solutions to communicate with various media, including traditional telephony systems, voice over IP (VOIP), e-mail, and the Web.

- This layer also provides the mechanism for distributing interaction-related business data, also referred to as *attached data,* within and across solutions.

You can expand the basic setup by using multiple DMXs. Remember that by default DMX uses the preassigned TCP/UDP ports (5060 for SIP and 1720 for H.323 communications), and so it cannot be run on the same host as any other SIP/H.323 application (such as a VoIP endpoint, or a second DMX). However, if the DMX configuration uses only one VoIP protocol (either SIP or H.323), you can change the port used by DMX by using the `h225-port` option, or the `sip-port` option. The same port should also be specified in the DMX option `callport`.

Templates

DMX 7.6 requires that you use the appropriate template for your SIP Server release. DMX is supplied with four templates matching different modes of DMX operation as shown in Table 40.

Table 40: DMX 7.6 Templates

Template Name	Description
dmx_client_76.apd	Requires the configuration of a DMX gatekeeper. This DMX client will receive and process the calls distributed by DMX gatekeeper.
dmx_dualmode_76.apd	Functions as both a gatekeeper and a client to a gatekeeper, allowing it to share the work of processing calls.
dmx_gk_76.apd	Acts to regulate other DMX modules. Deploying DMX in a gatekeeper topology will result in an enhanced availability of the solution, as well as the ability to use the component within a load balancing and/or multi-tenant application. By choosing the gatekeeper or client modes of operation, you must configure at least 2 DMX modules. One acts as a gatekeeper and one as a client to handle calls. The gatekeeper DMX monitors client DMXs and is able to reroute calls if a client fails. Because the gatekeeper distributes calls in round-robin fashion, the gatekeeper/client configuration enables load balancing among multiple DMXs.
dmx_standalone_mode_76.apd	Operates in the stand-alone mode with no gatekeeper required. You have the option of deploying the DMX server as a stand-alone H.323/SIP call control device. The stand-alone configuration results in the simplest deployment where only one DMX is required.

Note: Since DMX gatekeeper mode is implemented only in H.323, gatekeeper and client templates can be used only in an H.323 environment. Stand-alone and dual-mode templates can be used in SIP, H.323, and mixed SIP-H.323 environments.

Wizard Configuration

To use the Wizards to install DMX, install the Wizard from the CD, run the Genesys Wizard Manager, and follow the instructions.

Manual Configuration

This section describes setting up a simple configuration for DMX, in a single-tenant environment. For each configuration field, the tables in this section provide the value used in this example configuration, plus a description of what should go in the field so that users can tailor the example to their own environment. If a field is not mentioned, you should leave it unchanged.

Procedure:
Configuring DMX manually

Start of procedure

1. Import application templates:
   a. Open Configuration Manager and select the `Environment\Application Templates` folder.
   b. Select `File > Import Application Template` and browse to the Templates directory.
   c. Import the template for DMX.
2. Create a DMX `Application` object:
   a. Select the `Applications` folder, and then select `File > New > Application`.
   b. Select from one of the templates and enter field values in the resulting `Properties` dialog box, as shown in Table 41.

Table 41: DMX Properties

Tab	Field	Description
General	Name	Any descriptive name
Server Info	Host	The host on which DMX runs
	Communication Port	Not used since release 7.2; the value 0 is a placeholder to satisfy Configuration Manager minimum information conditions
Start Info	Working Directory	Pathname to the directory or folder holding the DMX application
	Command Line	Name of the DMX executable file
	Command-line Arguments	Host name and port number of Configuration Server, name of DMX application
	Timeout (Startup, Shutdown)	The time interval, in seconds, during which the application is expected to start/shut down
Connections		Any SIP Server configured in Configuration Manager. If there is more than one SIP Server application specified, DMX will choose the first one.

3. When you are finished, click `Apply` to save configuration changes.

End of procedure

DMX Configuration Options

DMX configuration options are grouped into `call`, `link`, `x-config`, and `gatekeeper` sections, as follows.

Call Section

This section must be named `call`. The name is case-sensitive.

call-address

Default Value: `$HOST`
Valid Values: `$HOST`, any IP address
Changes Take Effect: Immediately

Specifies the IP address of the DMX.

call-port

Default Value: None (protocol dependent)
Valid Values: `1-65535`
Changes Take Effect: Immediately

This option represents a non-standard value of the TCP/IP port to call DMX.

call-protocol

Default Value: `smcp`
Valid Value: `smcp`
Changes Take Effect: On restart

Warning! Do not alter this option from its Default Value.

Specifies the type of the application containing this call section.

Link Section

This section must be called `link`. The name is case-sensitive.

protocol

Default Value: `tcp`
Valid Value: `tcp`
Changes Take Effect: Immediately

Warning! Do not alter this option from its Default Value.

Describes low-level protocol.

verbose

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

Warning! Do not alter this option from its Default Value.

Sets verbosity level for this link.

X-Config Section

DMX connects directly to Configuration Server.

Every `x-config` section must include an option called `x-type,` with values `dmx` or `sm` specifying to which application it applies. The `x-config` section can also include an option `x-name` specifying the application name; if `x-name` is absent, the application name is taken from the object (application) name in the configuration system.

The following list includes the equivalent command-line parameter for each option.

This section must be called `x-config`. The name is case-sensitive.

amd-mode

Default Value: `false`
Valid Values: `true, false, 1, 0`
Changes Take Effect: Immediately

Enables or disables Answering Machine Detection in outgoing H.323 calls according to the AudioCodes specification.

Note: This option should be set to `false` if DMX is used with a non-AudioCodes gateway or if Call Progress Detection is not used.

audio-codec

Default Value: `1,3`

Valid Values: Comma-separated list of any of the following:

`1`	G.711 mu-Law
`2`	G.711 A-Law
`3`	G.723
`4`	G.729.A
`8`	MS-GSM and GSM Full Rate

Command-line Equivalent: `-audio_codec`
Changes Take Effect: Immediately

Specifies the audio codec(s) to be used by this DMX. G.711 operates at a higher bit rate than G.723 and G.729.A, providing better quality but consuming more network resources. MS-GSM/GSM FR is intermediate, providing moderate quality and low network resource consumption. For G.711, mu-Law is used in North America and Japan and A-Law is used elsewhere, including international routes.

bit-rate

Default Value: `6217`
Valid Value: `1-19200`
Command-line Equivalent: `-bit_rate`
Changes Take Effect: Immediately

Specifies the maximum bit rate for video codec(s). Applies only if the `video-codec` option (page 385) has a value other than `0` (zero).

cif-mpi

Default Value: `0`
Valid Value: `0-32`
Command-line Equivalent: `-cif_mpi`
Changes Take Effect: Immediately

Specifies the minimum picture interval, in units of 1/29.97, for the encoding and decoding of Common Intermediate Format (CIF) pictures for both H.261 and H.263 codecs. A value of `0` (zero) indicates no capability for CIF pictures. Applies only if the `video-codec` option (page 385) has a value other than `0` (zero). The maximum value is `4` when using the H.261 codec. The maximum value is `32` for the H.263 codec.

debug-level

Default Value: `0`
Valid Values: `0-3`
Command-line Equivalent: `-debug_level`
Changes Take Effect: Immediately

Applies only if the Log option `verbose` is set to `all`. It sets the amount of detail in log events of the Debug level. A higher number produces more detail.

delay-own-tcs

Default Value: `2000`
Valid Values: `0-2000`
Command-line Equivalent: None
Changes Take Effect: Immediately

Resolves the DMX and Cisco CallManager interoperability issue. The value is specified in milliseconds.

dtmf-payload-type

Default Value: `101`
Valid Values: `96—127`
Command-line Equivalent: None
Changes Take Effect: Immediately

Specifies the RTP Payload for DTMF digits carried in the RTP packets according to RFC2833 standards.

Note that in some cases, this value should be identical in both the gateway and in DMX. For example, when working with a Cisco Gateway, the gateway value and the DMX value for this option should be the same.

dtmf-sip

Default Value: `rtp-nte`
Valid Values: `no`, `rtp-nte`
Command-line Equivalent: `-dtmf_sip`
Changes Take Effect: Immediately

Defines the way DTMF signals are processed in the SIP protocol. If the value is `no`, no DTMF signal is processed. The `rtp-nte` value allows DTMF relay using NTE RTP packets. DTMF tones are encoded in the NTE format and transported in the same RTP channel as the voice.

dtmf-source

Default Value: `auto`
Valid Values:

`auto`	Chooses source based on Terminal Capabilities Set of the remote endpoint.
`h245ui`	H.245 user input signals will take place.
`h245sig`	H.245 signal will be applied.
`No`	No DTMF signals will be processed.
`rtp-nte`	Allows you to encode DTMF signals into the RTP packets and transport them on the same channel as voice.

Command-line Equivalent: `-dtmf`
Changes Take Effect: Immediately

Specifies the source for DTMF generation. Use the following formats:

`dtmf-source=value1,value2` or

`dtmf-source-value1`

Where:

- `value1` or `value2` can be `rtp-nte.`
- `h245ui, h245sign` cannot be assigned simultaneously.
- The `no` value has higher priority and, therefore, can silently discard other values.

h225-port

Default Value: `1720`
Valid Value: Any positive integer (any valid port)
Command-line Equivalent: `-h225_port`
Changes Take Effect: Upon restart

Specifies the listening port for the H.225 protocol. Use this option to override the default port setting for DMX if you want to run DMX on the same machine as another H.323 application.

h264-level

Default Value: `71`
Valid Values: See Table 42
Changes Take Effect: Immediately

This is a mandatory setting. It is supported only when using the H.264 protocol with a vPoint client. This option defines parameters which are used during the H.264 codec negotiation procedure. Use this option to improve video quality by setting the `level` or the degree of the video's capabilities (for example, its pixel resolution, speed of decoding, number of Macroblocks per second, and so on). The video quality also depends on the endpoint's encode/decode stream capability.

Table 42: Valid Values for the h.264-level Option

Level Parameter Value	H.264 Level Number
15	1
22	1.1
29	1.2
36	1.3
43	2
50	2.1
64	3
71	3.1
78	3.2
85	4
92	4.1
99	4.2
106	5
113	5.1

h264-profile

Default Value: `64`
Valid Values: `16, 32, 64` (see Table 43)
Changes Take Effect: Immediately

This is a mandatory setting. It is supported only when using the H.264 protocol with a vPoint client. This option defines parameters which are used during the H.264 codec negotiation procedure. This option allows you to define the profile, which improves video quality. Each profile that you define represents a set of algorithmic features. Table 42 describes the Codes and Profiles used with this option, which are based on the H.264 standard. The video quality also depends on the endpoint's encode/decode stream capability.

Table 43: Profile Values and Descriptions

Valid Profile Value	Profile Description
64	Baseline Profile. The basic goal of H.264 was to provide a royalty-free baseline profile to encourage early application of the H.264 protocol. The baseline profile consists of most of the major features of the H.264 codec, with the exception of: B slices and weighted prediction; CABAC encoding; field coding; and SP & SI slices. Thus, the Baseline Profile is appropriate for many progressive scan applications such as video conferencing and video-over-IP, but not for interlaced television or multiple stream applications.
32	Main Profile. The Main Profile contains all of the features as in Baseline, except for flexible macro block ordering (FMO), arbitrary slice order (ASO) and redundant slices. However, it adds field coding, B slices and weighted prediction, and CABAC entropy coding. This profile is appropriate for efficient coding of interlaced television applications where bit or packet error is not excessive, and where low latency is not a requirement.
16	Extended Profile. This profile contains all features from the Baseline profile and Main Profile, except that CABAC is not supported. In addition, the Extended Profile adds SP and SI for stream switching, and up to 8 slice groups. This profile is appropriate for server-based streaming applications where bit-rate scalability and error rate are very important; for example, Mobile video services.

Note: The vPoint HD conferencing system supports the Baseline Profile.

packet-size

Default Value: `20`

Valid Values: Comma-separated list of any of: `1, 2, 3, 4, 20, 30` (see Table 44)

Command-line Equivalent: `-pstr`

Changes Take Effect: Immediately

Specifies maximum packet size, as follows:

Table 44: Values of Packet-Size Option

Codec	Default Value	Valid Values	Unit
G.711 mu-Law or A-Law	20	20, 30	milliseconds per packet
G.723, G.729.A	1	1, 2, 3, 4	frames per packet
MS-GSM	Not applicable		
GSM Full Rate	Not applicable. **Note:** This option does not apply to the MS-GSM and GSM Full Rate codecs.		

If an audio-codec option is specified for more than one codec, the values of packet-size correspond one-to-one, in the same order. For example, suppose an audio-codec has values 1, 2, 4 and its packet-size has values 20, 30, 2. That means that this DMX uses codecs G.711 mu-Law at 20 milliseconds per packet, G.711 A-Law at 30 milliseconds per packet, and G.729.A at 2 frames per packet. You do not have to list the audio-codec values in ascending order (so audio-codec = 4, 2, 1 and packet-size = 2, 30, 20 would have the same effect as the example just described), although it is probably easier to do so. If a packet-size value is invalid for the codec it corresponds to, it is ignored and that codec receives the default packet size. This option does not apply to the MS-GSM codec.

progress-inband

Default Value: false
Valid Values: true, false, 0, 1
Changes Take Effect: Immediately

When a caller makes a call from an IP network to a PSTN via DMX and Cisco gateway, an in-band tone from the Cisco gateway is received. The Cisco gateway should be configured with the following options for the Inband tones feature to work properly:

- Dial-peer POTS (for example, dial-peer voice 522 pots)
- Contain the following configuration options:
  - progress_ind alert enable 8
  - progress_ind progress enable 8

In some IOSes, these options may be present but hidden (that is, no help or autocomplete options are available, but explicitly-issued commands will work).

Note: All dial peers (POTS type) should be configured separately. That is, there is no global configuration command that will preconfigure all POTS dial peers. For additional information regarding Cisco GW configuration, please refer to http://www.cisco.com.

qcif-mpi

Default Value: `1`
Valid values: `0-32`
Command-line Equivalent: `-qcif_mpi`
Changes Take Effect: Immediately

Specifies the minimum picture interval in units of 1/29.97 for the encoding and/or decoding of Quarter Common Intermediate Format (QCIF) pictures for both H.261 and H.263 codecs. A value of `0` (zero) indicates no capability for QCIF pictures. This option applies only if the `video-codec` option has a value other than `0` (zero). The maximum value is `4` when using the H.261 codec. The maximum value is `32` for the H.263 codec.

silence-supression

Default Value: `NULL`
Valid Values: `true, false, 1, 0`
Command-line Equivalent: None
Changes Take Effect: Immediately

If this option is either (`a`) not present, or (`b`) set to anything other than a valid value (including `NULL`), the endpoints of a call can negotiate whether or not to apply silence suppression. If this option has one of the valid values, silence suppression is (`true` or `1`) or is not (`false` or `0`) applied, without regard to anything the endpoints may do.

Note: For audio-codec G.723 (value `3`), this option must be set to `false`.

sip-port

Default Value: `5060`
Valid Values: Any integer greater than `1024`
Command-line Equivalent: `-sip_port`

Changes Take Effect: After DMX is started

Defines the port where DMX waits for incoming connections from SIP endpoints.

sip-transport

Default Value: `udp`
Valid Values: `tcp` or `udp`
Command-line Equivalent: `-sip_transport`

Changes Take Effect: Immediately

Defines the transport protocol (TCP or UDP) used to initiate outgoing sip calls.

Note: Preferably, you should set this option identically for all DMXs in a multi-DMX configuration.

still-img-trans

Default Value: `false`
Valid Values: `true, false`
Command-line Equivalent: `-still_img_trans`
Changes Take Effect: Immediately

Indicates capability for still images as specified in Annex D of the H.261 codec. This option applies only if the option `video-codec` has a value other than `0` (zero).

tcp-port-range

Default Value: None
Valid Values: `x - y` where `x` is any valid port number greater than `1024` and `y` is any valid port number greater than `x`
Command-line Equivalent: `-tcp_port_range`
Changes Take Effect: Immediately

Defines a range of ports for H.245 communication with DMX (may be necessary when operating with a firewall).

trade-off-cap

Default Value: `true`
Valid Values: `true, false`
Command-line Equivalent: `-trade_off_cap`
Changes Take Effect: Immediately

Applies to both H.261 and H.263 codecs. If `true`, the encoder is able to vary its tradeoff between temporal and spatial resolution as commanded by the remote terminal. This option only applies if the `video-codec` has a value other than `0` (zero).

video-codec

Default Value: `0`
Valid Values:

`0`	No video
`1`	H.261
`2`	H.263
`3`	H.264

Command-line Equivalent: `-video_codec`
Changes Take Effect: Immediately

Specifies the preferred video codec.

x-type

Default Value: `dmx`
Valid Value: `dmx`
Command-line Equivalent: None
Changes Take Effect: Immediately

Specifies to which component the `x-config` options apply.

Gatekeeper Section

This section must be called `gatekeeper`. The name is case sensitive.

RAS Server Options

accept-registrations

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

Enables/disables RAS server (gatekeeper) functionality. In the `off/false` position, no communication sockets are opened, no RAS requests are accepted, and none of the other RAS server options have any effect.

accept-calls

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

Setting this option to `true/on` allows a gatekeeper DMX to also process calls; in other words, the server functions as one of its own clients. The advantage of this is that the gatekeeper can include itself in call distribution for load balancing.

If this option is set to `off/false,` a gatekeeper DMX will still process calls if (1) it has no client DMX registered, and (2) it is connected to a SIP Server.

discovery-port

Default Values: `0`
Valid Values: `0` (zero) or `1718` only
Changes Take Effect: Immediately

Defines the port on which the gatekeeper DMX listens for autodiscovery messages. A value of `0` (zero) disables the auto-discovery function. Autodiscovery handles the situation in which an endpoint wants to register as a RAS client but is not configured with any gatekeeper's address. The endpoint broadcasts a message to all gatekeepers on the network. Gatekeepers that have autodiscovery enabled then respond to the message, informing the endpoint of their addresses.

main-port

Default Value: `1719`
Valid Values: integer between `1024` and `32767`
Changes Take Effect: Immediately

Defines the port on which the gatekeeper DMX listens for RAS requests.

RAS Client Options

backup-gk-host

Default Value: None
Valid Values: Any host name or IP address
Changes Take Effect: Immediately, if `primary-gk-host` and `primary-gk-port` are specified

Specifies a backup gatekeeper host.

backup-gk-port

Default Value: none
Valid Values: Any integer greater than `1024`
Changes Take Effect: Immediately, if `primary-gk-host` and `primary-gk-port` are specified.

Specifies the port of the backup gatekeeper.

primary-gk-host

Default Value: None
Valid Values: Any host name or IP address
Changes Take Effect: Immediately, if `primary-gk-host` and `primary-gk-port` are specified.

Specifies the gatekeeper with which this DMX registers.

primary-gk-port

Default Value: `1719`
Valid Values: Any integer greater than `1024`
Changes Take Effect: Immediately, if `primary-gk-host` and `primary-gk-port` are specified.

Specifies the gatekeeper port to which this DMX connects.

register

Default Value: `true`
Valid Values: `on/off, true/false`
Changes Take Effect: Immediately, if `primary-gk-host` and `primary-gk-port` are specified.

This is the main option that enables/disables RAS client functionality. In the `off/false` position, the executable does not try to register and communicate with a gatekeeper, and none of the other RAS client options have any effect.

services

Default Value: None
Valid Values: Any comma-separated series of positive integers
Changes Take Effect: Immediately, if `primary-gk-host` and `primary-gk-port` are specified.

A comma-separated list of E.164 (phone) numbers, to be used as a destination pattern during call distribution among client DMXs. Ordinarily, calls are distributed in round-robin fashion among all DMXs registered with a gatekeeper. But if a client DMX has a Service Number specified, all calls matching that number are sent to it only. Consider for example a configuration with one gatekeeper DMX and clients DMX-A, DMX-B, and DMX-C. DMX-A has the `services` option set to 3, while DMX-B and DMX-C have no services specified. Calls whose dialed number begins with a number other than 3 are distributed among all three client DMXs in a round-robin fashion. Calls whose dialed number begins with 3 are sent to DMX-A only.

Server/Client Option

registration-timeout

Default Value: `60`
Valid Values: Any positive integer
Changes Take Effect: Immediately

Specifies an interval (in seconds) within which the client DMX must repeat a registration request to the gatekeeper DMX. If three of these timeout intervals pass without any registration request, the gatekeeper assumes that the client has failed. You must set this option for both the client and the gatekeeper.

Changes from 7.5 to 7.6

There are no option changes between release 7.5 and release 7.6.

Configuring for Use with the Management Layer

The Management Layer, part of Genesys Framework 7.6, centralizes many functions (such as starting and stopping, status monitoring, alarm processing, and processing of log records) for all components of the Genesys Solution. It is required for configuring redundant systems (as described in the next section). Detailed information about the Management Layer is available in the Framework 7.6 documentation. To operate DMX along with the Management

Layer, configure DMX applications as previously described with the exceptions described in Table 45.

Table 45: Configuring DMX for the Management Layer

Object	Tab	Field	Values to Enter
Host	General	LCA Port	Number of the port on which LCA (Local Control Agent) is running
DMX	Connections	Server: Add	Add the name of the Message Server Add the name of the T-Server
	Start Info	Working Directory	Path name to the directory or folder holding the DMX application
		Command Line	dmx.exe
		Command-line Arguments	-host <hostname> -port <portnumber> -app <appname>, where <hostname> and <portnumber> are the host and port of Configuration Server and <appname> is the name of the DMX application
		Timeout (Startup, Shutdown)	120 recommended
		Redundancy Type	Unspecified
		Auto-Restart	Put a check mark in the box if you want the Management Layer to automatically restart the application.

VOIP Gateways

For every gateway that you plan to make a part of your interaction management solution, you need to have the following information available:

- Type of gateway. It usually corresponds to the gateway vendor, brand name, and model number.

Note: A gateway must support either SIP RFC 3261 or H.323 (v1 or v2).

- Version of the gateway software.
- Type of interface to the PBX/PSTN; for example T1 (ISDN PRI), FXO.

Voice Terminal Endpoints

Voice terminal endpoints enable the agent to receive voice calls. Three types of voice terminals can be used. Table 46 shows the information you should have available for each type.

Table 46: Voice Terminals and Required Information

Voice Terminal Type	Required Information
Multimedia PC with sound card, headphones, microphone	IP address and host name of PC
Traditional analog or digital phone connected to a VoIP gateway (via a traditional PBX for digital phone)	• Extension numbers corresponding to PBX and/or gateway dialing plan • IP address of the gateway or RAS
IP endpoint (SIP and H.323 compliant)	IP address of endpoint
SIP Endpoints	Cisco ATA 186 (See note below) Cisco IP Phone CP-7960G PingTel Xten X-Lite Polycom IP500
H.323 Endpoints	Open Phone Cisco ATA 186 (See note below)
Gateways	AudioCodes IPM260 (PCI-based) Cisco 5300 Cisco 3725 RadVision VoIP Gateway LG-1100 LG-2100 Vega-50 Vega-1000 SIP IP Phone Polycom IP500

Installing DMX

Configuration Wizards facilitate component deployment. The DMX configuration and installation involves many steps, and Genesys strongly recommends that you set up DMX using the wizard rather than manually. The DMX Wizard guides you through a series of steps and options to customize your deployment of DMX.

Note: You should not attempt to run DMX on the same host as SIP Server or any other SIP clients.

Installing using the Wizard

To use the Wizard to install DMX, install the Wizard from the CD, run the Genesys Wizard Manager, and follow the instructions.

Installing Manually

Consider the following when installing DMX:

- Only one DMX can be active on a given physical server.
- No other SIP or H.323 applications/components can be located on the same machine as DMX.

Note: To override this default behavior, see the description of the DMX `h225-port` and `SIP-port` options.

On the product CD, the `media_layer` directory includes a subdirectory containing installation files for the DMX component. It contains subdirectories named `windows` and `solaris`, which contain the appropriate installation files for those operating systems.

UNIX

For each component's directory, open the `aix`, `linux`, or `solaris` subdirectory, then open and read the file `README_FIRST`. It contains complete instructions on expanding the compressed file, running the shell script `<COMPONENT>-INSTALL.SH`, and completing the installation.

Note: Do not include spaces in the destination directory name when using a UNIX operating system.

You are now ready to start DMX. See "Starting and Stopping DMX" on page 392.

Windows 2000/2003

Open the subdirectory `windows`, then double-click `Setup.exe`. Follow the InstallShield instructions.

InstallShield creates a batch file in the folder of the component it installs. After installation, open the batch file and ensure that it includes the parameter `-app`

<appname>, where <appname> is the name of the application object you created in Configuration Manager.

Note: DMX is installed as a Windows 2000/2003 Service by default.

You are now ready to start DMX. See "Starting and Stopping DMX" on page 392.

Starting and Stopping DMX

Be sure that DB Server, Configuration Server, and License Manager are running before starting DMX.

Procedure: Starting DMX using Solution Control Interface

Purpose: To start DMX using Solution Control Interface (SCI).

Start of procedure

1. Start SCI either by clicking `Sci.exe` or selecting `Solution Control Interface` from the `Start` menu.
2. Go to the `Applications` view.
3. Select the DMX application in the `List` pane.
4. Click `Start` or select `Start` from the shortcut menu.

   The application's status changes from `Stopped` to `Started`.

End of procedure

Procedure: Stopping DMX using Solution Control Interface

Purpose: To stop DMX using SCI.

Start of procedure

1. Select the DMX component in the `Applications` folder.
2. Select `Stop` from the `Action` menu.

   The application's status changes from `Started` to `Stopped`.

End of procedure

Chapter

14 DMX Reference

This chapter provides reference information for DMX. It includes:

- DMX Log Events, page 393
- VOIP and Video-Over-IP Support, page 399

DMX Log Events

DMX displays some log events from the common.lms file. However, many other log events have their text hard coded within DMX rather than coming from the common.lms file. Table 47 lists the hard-coded events. In the Message Text column, <number> and <text> stand for attributes that are filled in when the actual message is generated. The Attributes column lists the names of the attributes (if any) in the order that they appear in the message text. Thus, for example, ID#40009 could also be represented as:

- Option <section name>:<option name> was set to <option value> from the command line.

Table 47: DMX Log Events

ID	Level	Message Text	Attributes
40004	STANDARD	Total number of legs: <number> (v. <string>)	Number of legs, DMX version
40009	STANDARD	Option <text>:<text> was set to <text> from the command line	Section, name, value
40011	STANDARD	(p:<number>) LCA port was set from the command line	Port
40012	STANDARD	DMX general options: <text>	String of options

Table 47: DMX Log Events (Continued)

ID	Level	Message Text	Attributes
40015	STANDARD	DMX audio settings: <text>	String of options
40016	STANDARD	DMX video settings: <text>	String of options
40018	STANDARD	Version: <text>	Version number
40019	STANDARD	Compilation Date: <text> <text>	Date, time
40021	STANDARD	Command line: <text>	Command-line string
40023	STANDARD	Cannot resolve IP address by host name <text> (e:)	Hostname
40039	STANDARD	(p:<number>) Port open for client H225s (H225)<text>	Port number
40040	STANDARD	(p:<number>) Port closed for client H225s (H225)<text>	Port number
40041	STANDARD	(p:<number>) Cannot open port for clients H225 (H225) (e:)	Port number
40044	STANDARD	(l:<number>) (c:<number>) Function <text> failed (e:<number>) (H225)	Leg ID, socket
40045	STANDARD	(c:<number>) Connection failed (e:) (H225)	Socket
40048	STANDARD	(l:<number>) (c:<number>) Function <text> failed (e:<number>) (H245)	Socket
40049	STANDARD	(c:<number>) Connection failed (e:) (H245)	Socket
40050	STANDARD	(l:<number>) Leg not found by ID (e:)	Leg ID
40052	STANDARD	Leg creation error (e:)	
40054	STANDARD	(l:<number>) (c:<number>) Received unsupported message (H225)	Socket
40055	STANDARD	Wrong protocol decriminator <hexadecimal number> (e:) (Q931)	Protocol discriminator
40057	STANDARD	Wrong CRV length <number>, message length <number> (e:) (Q931)	Length
40060	STANDARD	ASN1 decode error (e:<text>), decode <number>, PDU <number> (H225)	Error message, decode, PDU

Table 47: DMX Log Events (Continued)

ID	Level	Message Text	Attributes
40061	STANDARD	ASN1 encode error (e:<text>), encode <number> (H225)	Error message, decode
40064	STANDARD	ASN1 decode error (e:<text>), decode <number>, PDU <number> (H225)	Error message, decode, PDU
40065	STANDARD	ASN1 encode error (e:<text>), encode <number> (H225)	Error message, decode
40083	STANDARD	(CSI:) Can't connect to Configuration Server on <text>:<text> <text> (e:)	Host, port, error
40084	STANDARD	(CSI:) CfgLib error: <text> (e:)	Error description
40085	STANDARD	(CSI:) No Message Server for application <text> (e:)	Application name
40087	STANDARD	(CSR:) Error on reconnection <text> (e:)	Explanation
40088	STANDARD	(CSR:) DMX-server name <<text>> on reconnection is not as specified <<text>>(e:)	Old name, new name
40089	STANDARD	(CSR:) DMX-server type <number> is not as should be <number> (e:)	Old type, new type
40100	STANDARD	(CSU:) Message Server <text> on <text>:<text> added to DMX	Name, host, port
40102	STANDARD	(CSE:) Delta object=NULL (e:)	
40105	STANDARD	(CSU:) Message Server <text> removed	
40108	STANDARD	(CSS:) Primary Configuration Server on <text>:<text>	Host, port
40109	STANDARD	(CSS:) Backup Configuration Server on <text>:<text>	Host, port
40110	STANDARD	(CSS:) No backup Configuration Server	
40111	STANDARD	(CSS:) Message Server <text> Host's name was changed to <text>	Name, new host name
40112	STANDARD	(CSS:) Configuration Server Disconnected	
40113	STANDARD	(CSE:) Error: Can't find <text> with dbid <number> (e:)	Type of object, database ID

Table 47: DMX Log Events (Continued)

ID	Level	Message Text	Attributes
40114	STANDARD	(CSE:lib) List is not empty but head is null (e:)	
40115	STANDARD	(CSU:) Can’t reconnect because do not have host/port info (e:)	
40118	STANDARD	(RSO:) ‘services’ for client <text> was set to <text>	Client name, services
40128	STANDARD	(e:)(RSD:) Selected object not found	
40129	STANDARD	RAS Discovery port closed	
40132	STANDARD	Previous try was rejected, try in <number>	Seconds
40133	STANDARD	(RSS:) RAS ASN1 decode: Error decode = <number>,pdu=<number> (e:)	Decode, PDU
40134	STANDARD	(RSS:) Registration of <text> confirmed, aliases <text> created	New client, aliases
40135	STANDARD	(RSS:) Registration Rejected due to duplicated aliases <text>	Aliases
40137	STANDARD	(RSS:) Registration confirmed, Reregistration without aliases, aliases cleared	
40139	STANDARD	(RSC:) Impossible to prepare Registration message (e:)	
40140	STANDARD	(RSC:) Impossible to prepare UnRegistration message (e:)	
40141	STANDARD	(RSC:) Opened Listening Socket <number> for RAS client on Port: <number>	Socket, port
40142	STANDARD	(RSS:) Opened Listening Socket for RAS Discovery on Port: <number>	Port
40144	STANDARD	(RSS:) Opened Listening Socket for RAS Request and Status on Port: <number>	Port
40145	STANDARD	(RSS:) Listening Socket for RAS Request and Status Closed	
40146	STANDARD	(RSC:) Stop working as Client	
40147	STANDARD	(CSR:) Configuration Server reconnected on <text>:<text>	Host, port

Table 47: DMX Log Events (Continued)

ID	Level	Message Text	Attributes
40148	STANDARD	(CSR:)Error,kvlist empty (e:)	
40149	STANDARD	(CSR:) Try to reconnect as <text> to <text>:<text>	
40150	STANDARD	(l:<number>) LCSE Error: Timer T103 fired (e:) (H245)	Leg ID
40151	STANDARD	(l:<number>) LCSE Error: Improper state (e:<number>) (H245)	Leg ID, error code
40152	STANDARD	(l:<number>) CESE Error: Timer T101 fired (e:) (H245)	Leg ID
40153	STANDARD	(l:<number>) LCSE Error: Timer T106 fired (e:) (H245)	Leg ID
40154	STANDARD	(l:<number>) LCSE Error: Improper state (e:<number>) (H245)	Leg ID, error code
40155	STANDARD	(l: <number>) DMX attempts to call itself (e:)	Leg ID
40156	STANDARD	(CSI:) LcaPort <text> LCA port	Port number
40157	STANDARD	(CSI:) DMX application has empty set of options	
40158	STANDARD	(CSI:)DMX application <text> defined as third party application configured incorrectly:<text> (e:)	DMX name, description of problem
40159	STANDARD	(CSR:)Bad result of ConfGetConfServerDBID(..) error_type <number>, object-type <number>,object_property <number> (e:)	Error type, object type, object property
40160	STANDARD	(CSR:)Bad result of ConfGetConfServerBackupDBID(..) error_type <number>, object-type <number>,object_property <number> (e:)	Error type, object type, object property
40161	STANDARD	(CSU:)We have more then one Message Server attached, <text> and <text> (e:)	
40162	STANDARD	(CSU:) Empty set of connections of DMX, Message Server not searched	
40163	STANDARD	(CSU:) Message Server <text> changed	

Table 47: DMX Log Events (Continued)

ID	Level	Message Text	Attributes
40164	STANDARD	(CSU:)ConfGetErrorInfo can’t get details of error (e:)	
40165	STANDARD	(RSO:) ‘gatekeeper’ Option <text> was set to <text> Option name, value	
40166	STANDARD	(RSD:) Add new client <text>, services: <text>; current number of clients <number>	Name of client, services list, number of clients
40167	STANDARD	(RSS:) RAS ReRegistration Confirmed, aliases set was empty, added new one(s)	
40168	STANDARD	(RSS:)RAS Registration, Null call address, EndPoint ID unknown (e:)	
40169	STANDARD	(RSS:) RAS Registration, response from client <text> not received, unregistered by force	Name of client
40170	STANDARD	(RSS:) Agent <text> unregistered	Name of client
40175	STANDARD	(RSC:)<text>, File:<text>,Line:<number> (e:)	Description of situation, file, line
40175	STANDARD	(RSC:) Send Unregistration message	
40179	STANDARD	(ts:<text>) TS_EXCEPTION occurred (e:) (nco)	
40182	STANDARD	List of local ip addresses: <text> Addresses	
40183	STANDARD	(l:<number>) No more available ports. Increase tcpport-range value (e:)	Leg ID
40184	STANDARD	(l:<number>) No common audio codec found (e:)	Leg ID
40190	STANDARD	Option <text>:<text> is wrong. Using default value <text>	Option name, option value, default option value
40195	STANDARD	(p:<number>) Port closed for SIP clients	Port number
40197	STANDARD	(l:<number>) Sent SIP ‘<text’	LegID, SIP message
40201	STANDARD	(l:<number>) Received SIP ‘<text>’	LegID, SIP message
50001	TRACE	(l:<number>) Leg created, CRV = <hexadecimal number>	Socket on H225, leg ID

Table 47: DMX Log Events (Continued)

ID	Level	Message Text	Attributes
50002	TRACE	(l:<number>) Leg destroyed	Leg ID
50034	TRACE	(RSD:) No Registered Clients	
50038	TRACE	(RSD:) Total num of clients =<number>	Number of clients
50039	TRACE	(RSD:) Add service <text> for server itself	Service name
50040	TRACE	(RSD:) Stop server as client for itself	
50054	TRACE	(RSD:)<<text>>: Alias <text> was resolved <text>:<number>	
50056	TRACE	(RSD:) Admission request from <text> Confirmed: <text>	Sender, destination

VOIP and Video-Over-IP Support

The DMX specifications in Table 48 include information for working with VOIP and/or Video over IP.

Table 48: DMX Specifications: VOIP and Video-over-IP

Supported Features	Possible Values
VoIP protocols supported:	H.323 set (H.225.0, H.245)
	SIP rfc3261 (TCP and UDP)
	SIP SDP rfc2327
Codecs supported:	
voice:	G.711 (mLaw, ALaw), G.723.1, G.729.A, GSM, MS-GSM
video:	H.261, H.263, H.264; CIF and QCIF
Silence suppression:	Supported in G.711, G.729.A and GSM
DTMF type support:	H.245 user input, H.245 signal, rfc 2833 (RTP-NTE)

Table 48: DMX Specifications: VOIP and Video-over-IP (Continued)

Supported Features	Possible Values
Additional protocol support:	H.225.0 Facility call Forward (server)
	H.450.2 FACILITY Single-step Transfer
	SIP REFER (Single-step Transfer)
	H.225.0 RAS Gatekeeper (server/client)
Video still image support:	Supported in H.261
Firewall configuration support:	Supported in modes: standalone firewall; using DMX as proxy in DMZ

Appendix

Session Initiation Protocol (SIP) Overview

This chapter provides basic information about the SIP and H.323 protocols and contains the following sections:

SIP Overview

Session Initiation Protocol (SIP) was originally designed in 1996 as an Application Layer control (signalling) protocol for creating, modifying, and terminating media sessions. See Figure 41 on page 402 for more information about layers in the TCP/IP model.

IETF

The Internet Engineering Task Force (IETF) is a volunteer group that works with the World-wide Web Consortium (W3C) to develop and promote Internet standards such as Transmission Control Protocol/Internet Protocol (TCP/IP). They also created documents known as "Request for Comments" (RFC) that discuss new research, innovations, and methodologies applicable to Internet technologies. The IETF is organized into a large number of working groups, one of which is SIP. There are a large number of SIP-related RFCs.

TCP/IP Model

SIP can work over any component in the Transport layer such as TCP or UDP. SIP works with several other protocols and is only involved in the signalling portion of a session. SIP contains Session Description Protocol (SDP) information that describes the media content of the session, and is a packet in Realtime Transport Protocol (RTP). RTP is the carrier for the actual voice or video content itself. Normally, SIP clients use port 5060 over TCP or UDP protocols to connect to SIP Servers and other SIP endpoints.

Note: Hardware SIP endpoints share characteristics with traditional telephones, but use SIP and RTP to communicate.

Software SIP endpoints are also very common in many Instant Messaging applications.

Figure 41 contains information about layers in the TCP/IP model.

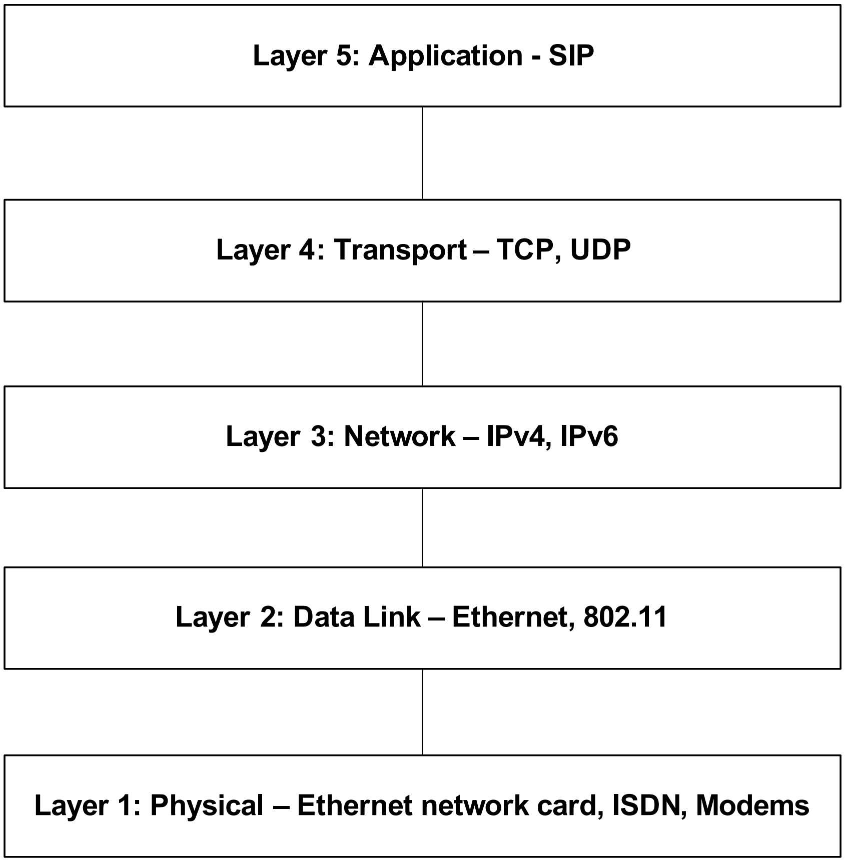

Layer 5: Application - SIP
Layer 4: Transport – TCP, UDP
Layer 3: Network – IPv4, IPv6
Layer 2: Data Link – Ethernet, 802.11
Layer 1: Physical – Ethernet network card, ISDN, Modems

Figure 41: Five-layer TCP/IP Model

SIP Functionality

SIP provides signalling and call setup protocols for IP communication that are similar to those found in the Public Switched Telephone Network (PSTN). SIP also enables features such as Proxy Servers and User Agents. These features are similar to normal telephone operations. Although SIP and PSTN networks are different, the agent does not notice a difference. SIP can also enable many of the advanced call processing features that are found in a complex system such as Signalling System 7 (SS7). However, SIP is a peer-to-peer protocol, and requires only a core network with hardware or software endpoints. With SIP, the advanced call processing features are implemented in the endpoints instead of being implemented in a SS7 network.

Proxy and Registrar Servers

SIP requires proxy and registrar network components to work. Although two SIP endpoints can communicate without a SIP Server (peer-to-peer), this is not very practical so traditional switch hardware or software switch functionality is needed to act as registrar and proxy.

A *registrar* is a server that accepts REGISTER requests from endpoints and allows agents to specify their current location for the proxy servers.

A proxy server performs for the following functionalities:

- Route media requests to an agent's location
- Authenticate and authorize agents for media services
- Implement call-routing strategies
- Provide media features to agents

SIP Messaging

SIP is similar to HTTP and provides request-response text-based messaging, so it is easy to understand because it shares many HTTP status codes.

SIP is able to process the following messages:

Request Methods

Table 49: SIP Request Methods

Request Method	Description
INVITE	Indicates a client is being invited to participate in a call session.
ACK	Confirms that the client has received a final response to an INVITE request.
BYE	Terminates a call and is sent by either the sender or the receiver.
CANCEL	Cancels any pending searches but does not terminate a call that has already been accepted.
OPTIONS	Queries the capabilities of servers.
REGISTER	Registers the address listed in the To header field with a SIP server.
INFO	Sends mid-session information that does not modify the session state.

Table 49: SIP Request Methods (Continued)

Request Method	Description
REFER	Ask recipient to issue SIP request, such as a call transfer.
MESSAGE	Transports instant messages using SIP.
UPDATE	Modifies the state of a session without changing the state of the dialog.

Provisional Responses

PRACK Provisional acknowledgement

Notifications

SUBSCRIBE Subscribes for an Event of Notification from the Notifier.

NOTIFY Notifies the subscriber of a new Event.

Publishing Event Changes

PUBLISH Publishes an event to the server.

Responses

1xx Informational Responses

100 Trying

180 Ringing

181 Call Is Being Forwarded

182 Queued

183 Session Progress

2xx Successful Responses

200 OK

202 accepted

3xx Redirection Responses

300 Multiple Choices

301 Moved Permanently

302 Moved Temporarily

305 Use Proxy

380 Alternative Service

4xx Client Failure Responses

400 Bad Request

401 Unauthorized

402 Payment Required (Reserved for future use)

403 Forbidden

404 Not Found: User not found

405 Method Not Allowed

406 Not Acceptable

407 Proxy Authentication Required

408 Request Timeout

410 Gone

413 Request Entity Too Large

414 Request-URI Too Long

415 Unsupported Media Type

416 Unsupported URI Scheme

420 Bad Extension

421 Extension Required

423 Interval Too Brief

479 Regretfully, we were not able to process the URI

480 Temporarily Unavailable

481 Call/Transaction Does Not Exist

482 Loop Detected

483 Too Many Hops

484 Address Incomplete

485 Ambiguous

486 Busy Here

487 Request Terminated

488 Not Acceptable Here

489 Bad Event

491 Request Pending

493 Undecipherable

494 Security Agreement Required

5xx Server Failure Responses

500 Server Internal Error

501 Not Implemented

502 Bad Gateway

503 Service Unavailable

504 Server Time-out

505 Version Not Supported

513 Message Too Large

6xx Global Failure Responses

600 Busy Everywhere

603 Decline

604 Does Not Exist Anywhere

606 Not Acceptable

H.323 Overview

H.323 is a part of a group of protocols that provides for communications using Integrated Services Digital Network (ISDN), Public Switched Telephone Network (PSTN) or Signaling System 7 (SS7). H.323 was developed to transport media over Local Area Networks (LANs) but it has since become a means for implementing Voice Over Internet Protocol (VOIP) networks.

Note: In Genesys, the DMX component provides H.323 protocol conversion for SIP Server.

ITU-T and Call Models

H.323 is a protocol that the ITU Telecommunication Standardization Sector (ITU-T) developed to provide media communication sessions for a packet network. H.323 was the first VoIP protocol to set standards for the basic call model and additional services that were needed to complement existing PSTN service expectations. The H.323 protocol is integrated easily into existing ISDN-based telephone switches. As a result, an IP switch is a H.323 Gatekeeper as well as a provider of additional calling services.

H.xxx Protocols

The H.323 protocol also uses other ITU-T protocols such as:

H.225: Describes call signalling, media, stream packets, media stream synchronization, and control messaging formats.

H.235: Describes forms of security.

H.239: Describes how to use two media streams when having a video conference call.

H.245: Describes the messages and procedures used for manipulating media channels, capability exchange, control and indications.

H.261, H.263, H.264: Describe video encoding formats.

H.450: Describes any additional calling services.

H.460: Describes firewall implementation.

Index