



Framework 8.0

T-Server for CSTA Connector Deployment Guide Wiki Redirect

ALERT: This document is available as a PDF only to support searches from the Technical Support Knowledge Base. Click here ([T-Server for CSTA Connector Deployment Guide](#)) to be redirected to the content in its original format.



Part

1

T-Server Deployment

Part One of this *T-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 One is divided into the following chapters:

- Chapter 1, “T-Server Fundamentals,” on [page 3](#), describes T-Server, its place in the Framework 8 architecture, T-Server redundancy, and multi-site issues. It stops short of providing configuration and installation information.
- Chapter 2, “T-Server and CSTA Connector General Deployment,” on [page 13](#), presents configuration and installation procedures for all T-Servers.
- Chapter 3, “High-Availability Deployment,” on [page 27](#), addresses high availability (HA).
- Chapter 4, “Multi-Site Support,” on [page 39](#), details the variations available for T-Server implementations across geographical locations.
- Chapter 5, “Starting and Stopping T-Server and CSTA Connector,” on [page 97](#), describes how, and in what order, to start up T-Server among other Framework components. It also provides possible stopping commands.

New for All T-Servers in 8.1

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 8.1 release of T-Server:

- T-Server no longer connects to applications that have disabled status in the configuration environment.

- The default value of the background-processing configuration option has been changed to true.
- T-Server now supports the Unresponsive Process Detection feature. The following configuration options enable this feature:

- heartbeat-period
- hangup-restart

For more information, refer to the *Framework 8.0 Management Layer User's Guide*.

- T-Server now supports IPv6. For more information, refer to the *Framework 8.1 Deployment Guide*.
- T-Server now supports vSphere 4 Hypervisor.
- T-Server now supports Acresto FLEXNet Publisher v11.9 license manager.



Chapter

1

T-Server Fundamentals

This chapter provides general information about T-Server features and functionality and about its configuration and installation. Generally, 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, page 3](#)
- [Advanced Disconnect Detection Protocol, page 9](#)
- [Redundant T-Servers, page 10](#)
- [Multi-Site Support, page 10](#)
- [Agent Reservation, page 10](#)
- [Client Connections, page 11](#)
- [Next Steps, page 12](#)

Learning About T-Server

The *Framework 8.1 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 1 illustrates the position Framework holds in a Genesys solution.

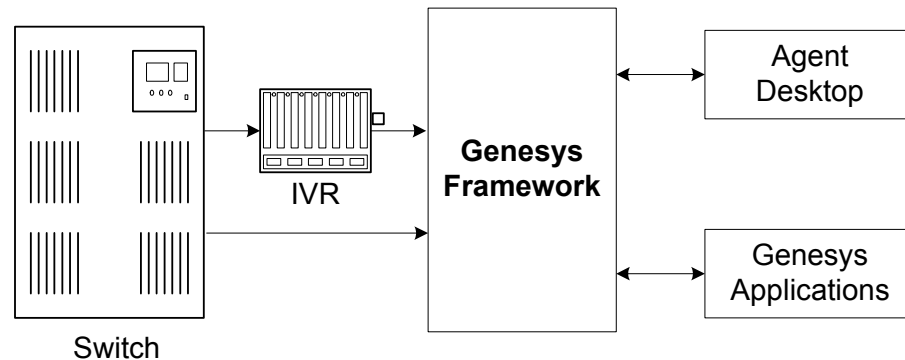


Figure 1: Framework in a Genesys Solution

Moving a bit deeper, Figure 2 presents the various layers of the Framework architecture.

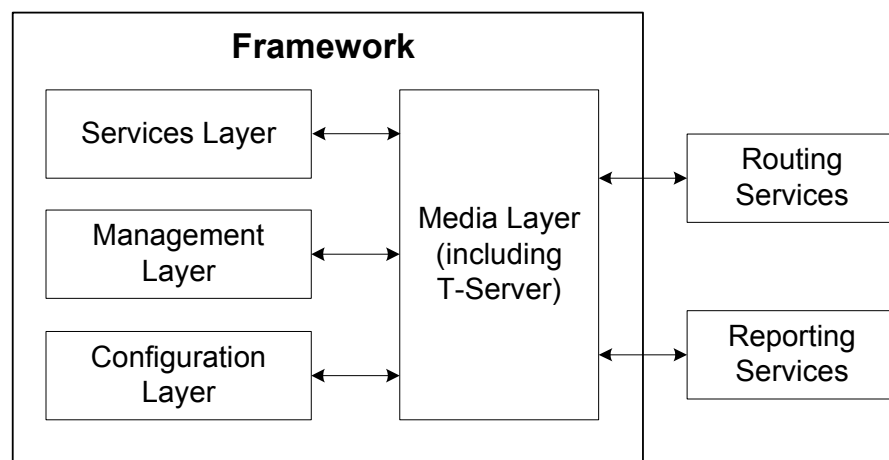


Figure 2: The Media Layer in the Framework Architecture

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 3 presents the generalized architecture of the Media Layer.

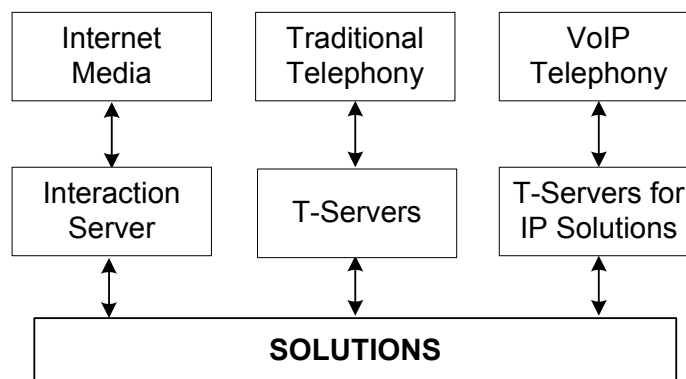


Figure 3: Media Layer Architecture

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, Interaction 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 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 8.x .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 3 on [page 5](#) (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 your T-Server 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 4](#), 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.

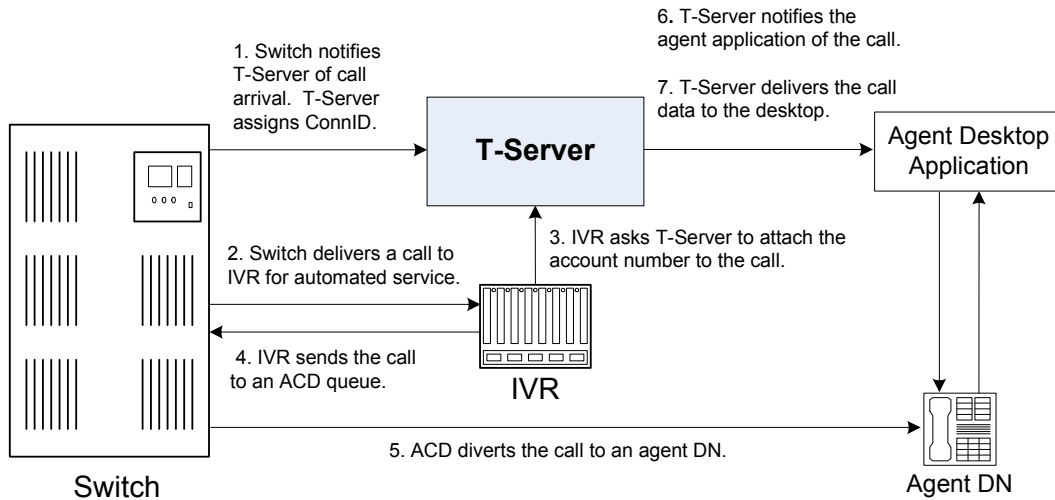


Figure 4: Functional T-Server Steps

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 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 after the polling signal, while the

response travels from one T-Server to another. If you do not 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.

Instructions for configuring T-Server redundancy are available in Chapter 3, “High-Availability Configuration and Installation.”

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 4, “Multi-Site Support,” on [page 39](#).

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 41](#)), 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 48](#) 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 8.x .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.

Starting with version 8.1, T-Server supports Agent Reservation failure optimization, to ensure that only agent reservation requests of the highest priority are collected. T-Server responds immediately with the `EventError` message to existing or new reservation requests of a lower priority while collecting the agent reservation requests of the highest priority only. This functionality is controlled with the `collect-lower-priority-requests` configuration option.

Client Connections

The number of connections T-Server can accept from its clients depend on the operating system that T-Server runs. [Table 1](#) illustrates the number of client connections that T-Server support.

Table 1: Number of T-Server's Client Connections

Operating System	Number of Connections
AIX 32-bit mode (versions 5.3)	32767
AIX 64-bit mode (versions 5.3, 6.1, 7.1)	32767
HP-UX 32-bit mode (versions 11.11)	2048
HP-UX 64-bit mode (versions 11.11, 11i v2, 11i v3)	2048
HP-UX Itanium (version 11i v3)	2048
Linux 32-bit mode (versions RHEL 4.0, RHEL 5.0)	32768
Linux 64-bit mode (version RHEL 5.0)	32768
Solaris 32-bit mode (version 9)	4096

Table 1: Number of T-Server's Client Connections (Continued)

Operating System	Number of Connections
Solaris 64-bit mode (versions 9, 10)	65536
Windows Server 2003, 2008	4096

Next Steps

Now that you have gained a general understanding of the roles and features available with T-Servers, you are ready to learn how T-Servers are installed and configured. That information is presented in the next few chapters of this *Deployment Guide*. So unless you are already familiar with T-Server deployment and operation procedures, continue with Chapter 2, “T-Server and CSTA Connector General Deployment,” on [page 13](#).



Chapter

2

T-Server and CSTA Connector General Deployment

This chapter contains general information for the deployment, configuration, and installation of your T-Server. You may have to complete additional configuration and installation steps specific to your T-Server and switch.

This chapter contains these sections:

- [Prerequisites, page 13](#)
- [Deployment Sequence, page 18](#)
- [Deployment of T-Server, page 18](#)
- [Next Steps, page 25](#)

Note: You *must* read the *Framework 8.1 Deployment Guide* before proceeding with this T-Server guide. That book contains information about the Genesys software you must deploy before deploying T-Server.

Prerequisites

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

Software Requirements

Framework Components

You can only configure T-Server after you have deployed the Configuration Layer of Genesys Framework. This layer contains DB Server, Configuration Server, and Configuration Manager. If you intend to monitor or control T-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 T-Server.

Refer to the *Framework 8.1 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 Reference Manual* for the list of operating systems and database systems supported in Genesys releases 6.x, 7.x, and 8.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.

Security

Starting with release 7.5, T-Server supports the Genesys Transport Layer Security (TLS) and can be configured for secure data exchange with the other Genesys components that support this functionality.

The Genesys TLS is not supported on all operating systems that T-Server itself supports. For information about the supported operating systems, see the *Genesys 8.x Security Deployment Guide*.

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 8.1 Deployment Guide* for recommendations on server locations.

Supported Platforms

Refer to the *Genesys Supported Media Interfaces Reference Manual* for the list of supported switch and PBX 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 T-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.

T-Server requires seat-related DN technical licenses to operate even in its most basic configuration. Without appropriate licenses, you cannot install and start T-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 Licensing Guide* available on the Genesys Documentation Library DVD.

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

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

Licensing Basic Implementations

A stand-alone T-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

T-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 T-Server licenses. Neither T-Server in a redundant pair configured for hot standby starts if this license is unavailable. Moreover, the primary and backup T-Servers must use the same licenses to control the same pool of DNs. If your T-Servers are configured with the hot standby redundancy type, order licenses for CTI HA support.

Licensing Multi-Site Implementations

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

Note: You do not need licenses for multi-site support if some T-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 T-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 T-Server.

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

About Configuration Options

Configuring T-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 T-Server configuration options on the **Options** tab of your T-Server Application object in Configuration Manager. The instructions for configuring and installing T-Server that you see here are only the most rudimentary parts of the process. Pay particular attention to the configuration options specific to your own T-Server.

Configuration options common to all T-Servers, independent of switch type, are described in the “T-Server Common Configuration Options” chapter. *T-Server-specific* configuration options are described in a separate chapter. T-Server also supports unified Genesys log options, as described in the “Common Configuration Options” chapter.

Options that configure values for the TSCP software in your T-Server are common to all T-Servers. Options based on the custom features of your switch apply to your T-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.

Deployment Sequence

This is the recommended sequence to follow when deploying T-Server and CSTA Connector.

Task Summary: Deployment Sequence

Objective	Related Procedures and Actions
1. Deploy Configuration Layer objects and ensure Configuration Manager is running.	See the <i>Framework 8.1 Deployment Guide</i> for details.
2. Deploy Network objects (such as Host objects).	See the <i>Framework 8.1 Deployment Guide</i> for details.
3. Deploy the Management Layer.	See the <i>Framework 8.1 Deployment Guide</i> for details.
4. Test your configuration and installation.	See Chapter 5, “Starting and Stopping T-Server and CSTA Connector,” on page 97 .

Note: 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 T-Server will run.

Deployment of T-Server

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

Configuration of Telephony Objects

This section describes how to manually configure T-Server telephony objects if you are using Configuration Manager. For information about configuring T-Server telephony objects using Genesys Administrator, refer to the *Framework 8.1 Genesys Administrator Help*.

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 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.

Note: The following script needs to be run in pre 8.1 ML environments in order for the BroadSoft BroadWorks Switching Office object to be available in Configuration Server (introduced as an object in 8.1 Configuration Server):

```
INSERT INTO cfg_locale VALUES(1, 8, 1, 81, 0, 'BroadSoft  
BroadWorks', 'Enum_CfgSwitchType')
```

Switches

1. Configure a `Switch` object for each switch on your telephony network. Assign each `Switch` object to the appropriate `T-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 4, “Multi-Site Support,” on [page 39](#), 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 T-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, T-Server does not register such DNs.

1. To configure Telephony objects within each switch, consult the switch documentation.
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.

Warning! When setting the Register flag for a DN, make sure you select the value according to your T-Server. The Register flag values are as follows:

- `False`—T-Server processes this DN locally, and never registers it on the switch.
 - `True`—T-Server always registers this DN on the switch during T-Server startup or CTI link reconnect.
 - `On Demand`—T-Server registers this DN on the switch only if a T-Server client requests that it be registered.
-

Multi-Site Operations

See the section, “Configuring Multi-Site Support” on [page 82](#), for information on setting up DNS for multi-site operations.

Configuration of T-Server

Use the *Framework 8.1 Deployment Guide* to prepare accurate configuration information. You may also want to consult *Configuration Manager Help* and/or *Genesys Administrator Help*, which contains detailed information about configuring objects.

Recommendations

Genesys recommends using an Application Template when you are configuring your T-Server application. The Application Template for your particular T-Server contains the most important configuration options set to the values recommended for the majority of environments. When modifying configuration options for your T-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 T-Server

Start of procedure

1. Follow the standard procedure for configuring all Application objects to begin configuring your T-Server Application object. Refer to the *Framework 8.1 Deployment Guide* for instructions.
2. In a Multi-Tenant environment, specify the Tenant to which this T-Server belongs on the General tab of the Properties dialog box.
3. On the Connections tab:
 - Add all Genesys applications to which T-Server must connect.

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

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

5. In a multi-site environment, you must complete additional T-Server configuration steps to support multi-site operations; see Chapter 4, “Multi-Site Support,” on [page 39](#).

End of procedure

Next Steps

- See “Installation of T-Server” on [page 22](#).

Procedure: Configuring multiple ports

Purpose: To configure multiple ports in T-Server for its client connections.

Start of procedure

1. Open the T-Server Application Properties dialog box.
2. Click the Server Info tab.
3. In the Ports section, click Add Port.
4. In the Port Properties dialog box, on the Port Info tab:
 - a. In the Port ID text box, enter the port ID.
 - b. In the Communication Port text box, enter the number of the new port.
 - c. In the Connection Protocol box, select the connection protocol, if necessary.
 - d. Select the Listening Mode option.

Note: For more information on configuring secure connections between Framework components, see *Genesys 8.x Security Deployment Guide*.

- e. Click OK.
5. Click OK to save the new configuration.

End of procedure

Installation of T-Server

The following directories on the Genesys 8.1 Media product DVD contain T-Server installation packages:

- `media_layer/<switch>/<platform>` for UNIX installations, where `<switch>` is your switch name and `<platform>` is your operating system.

- `media_layer\<switch>\windows` for Windows installations, where `<switch>` is your switch name.

Procedure: Installing T-Server on UNIX

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 T-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 T-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 T-Server application you configured in “Configuring T-Server” on [page 21](#) from the list of applications.
7. Specify the destination directory into which T-Server is to be installed, with the full path to it.
8. If the target installation directory has files in it, do one of the following:
 - Type 1 to back up all the files in the directory (recommended).
 - Type 2 to overwrite only the files in this installation package. Use this option only if the installation being upgraded operates properly.
 - Type 3 to erase all files in this directory before continuing with the installation.

The list of file names will appear on the screen as the files are copied to the destination directory.
9. If asked which version of the product to install, the 32-bit or the 64-bit, choose the one appropriate to your environment.
10. If asked about the license information that T-Server is to use: specify either the full path to, and the name of, the license file, or the license server parameters.

11. As soon as the installation process is finished, a message appears announcing that installation was successful. The process places T-Server in the directory with the name specified during the installation.

End of procedure

Next Steps

- To verify manual installation, go to “Verifying the installation of T-Server” on [page 25](#).
- To test your configuration and installation, go to Chapter 5, “Starting and Stopping T-Server and CSTA Connector,” on [page 97](#), and try it out.
- To configure and install redundant T-Servers, see Chapter 3, “High-Availability Deployment,” on [page 27](#).
- To install T-Servers for a multi-site environment, proceed to Chapter 4, “Multi-Site Support,” on [page 39](#).

Procedure: Installing T-Server on Windows

Start of procedure

1. In the directory to which the T-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 T-Server.
3. When prompted, select the T-Server Application you configured in “Configuring T-Server” on [page 21](#) from the list of applications.
4. Specify the license information that T-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 T-Server is to be installed.
6. Click `Install` to begin the installation.
7. Click `Finish` to complete the installation.

By default, T-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 installation of T-Server” on [page 25](#).

- To test your configuration and installation, go to Chapter 5, “Starting and Stopping T-Server and CSTA Connector,” on [page 97](#), and try it out.
- To configure and install redundant T-Servers, see Chapter 3, “High-Availability Deployment,” on [page 27](#).
- To install T-Servers for a multi-site environment, proceed to Chapter 4, “Multi-Site Support,” on [page 39](#).

Procedure:

Verifying the installation of T-Server

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

Prerequisites

- [Procedure: Installing T-Server on UNIX, on page 23](#)
- [Procedure: Installing T-Server on Windows, on page 24](#)

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 configured and installed T-Server using Configuration Manager. If you want to test your configuration and installation, go to Chapter 5, “Starting and Stopping T-Server and CSTA Connector,” on [page 97](#), and try it out. Otherwise, if you want to configure and install redundant T-Servers, see Chapter 3, “High-Availability Deployment,” on [page 27](#). If you want to install T-Servers for a multi-site environment, proceed to Chapter 4, “Multi-Site Support,” on [page 39](#).



Chapter

3

High-Availability Deployment

This chapter describes the general steps for setting up a high-availability (HA) environment for your T-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.

Every switch/T-Server combination offers different high-availability options. 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. Some T-Servers support a switch's ability to provide two CTI links to two T-Servers or even one CTI link to two T-Servers.

This chapter describes the redundant architecture and how to configure T-Server so that it operates with either type. Information in this chapter is divided into the following sections:

- [Warm Standby Redundancy Type, page 27](#)
- [Hot Standby Redundancy Type, page 29](#)
- [Prerequisites, page 31](#)
- [Warm Standby Deployment, page 32](#)
- [Hot Standby Deployment, page 34](#)
- [Next Steps, page 38](#)

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.

Warm Standby Redundancy Architecture

Figure 5 illustrates the warm standby architecture. The standby server recognizes its role as a backup and does not process client requests until the Management Layer changes its role to primary. When a connection is broken between the primary server and the Local Control Agent (LCA, not shown in the diagram) running on the same host, a failure of the primary process is reported, and the switchover occurs; or, if the host on which the T-Server is running fails, the switchover also occurs. (See the *Framework 8.1 Deployment Guide* for information on LCA.) As a result:

1. The Management Layer instructs the standby process to change its role from backup to primary.
2. A client application reconnects to the new primary.
3. The new primary (former backup) starts processing all new requests for service.

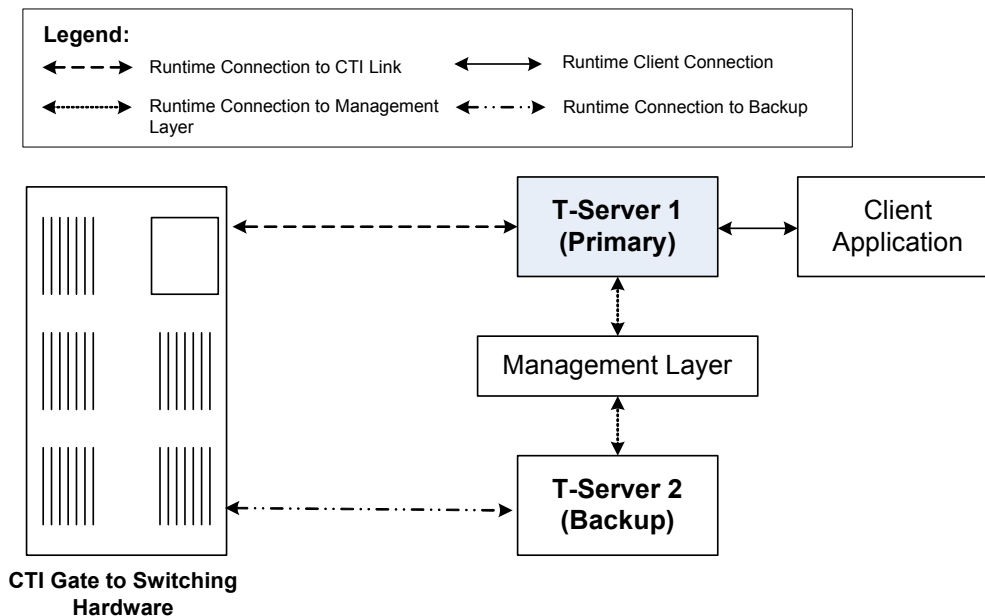


Figure 5: Warm Standby Redundancy Architecture

Although normal operations are restored as soon as the backup process takes over, the fault management effort continues. That effort consists of repeated

attempts to restart the process that failed. Once successfully restarted, the process is assigned the backup role.

Note: You can find full details on the role of the Management Layer in redundant configurations in the *Framework 8.1 Deployment Guide*.

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. Data synchronization and existing client connections to the backup guarantee higher availability of a component. (See Figure 6 on [page 30](#).)

Starting with release 7.1, the hot standby redundancy type is implemented in T-Servers for most types of switches. However, for some switches, you must compensate for the lack of link redundancy by using an additional Genesys component called *HA Proxy*.

Hot Standby Redundancy Architecture

[Figure 6](#) illustrates the switch-independent side of a hot standby implementation. Here, T-Servers start simultaneously and connect to the switch. At T-Server startup, the Management Layer assigns the role of the primary server to T-Server 1, and the role of backup to T-Server 2. T-Server clients register with both T-Servers, but only the primary T-Server handles client requests other than the registration requests. The internal T-Server information, such as a DN status, ConnID, UserData, and Call Type, is synchronized between the primary and backup T-Servers. Therefore, the backup T-Server has the same information as the primary T-Server.

If T-Server 1 fails, the Management Layer makes T-Server 2 the new primary server, and it starts processing client requests. The Management Layer attempts to restart T-Server 1, and if it is successful, it makes T-Server 1 the new backup server.

The details of hot standby redundancy implementation between T-Servers and their switches vary depending on switch support for multiple CTI links.

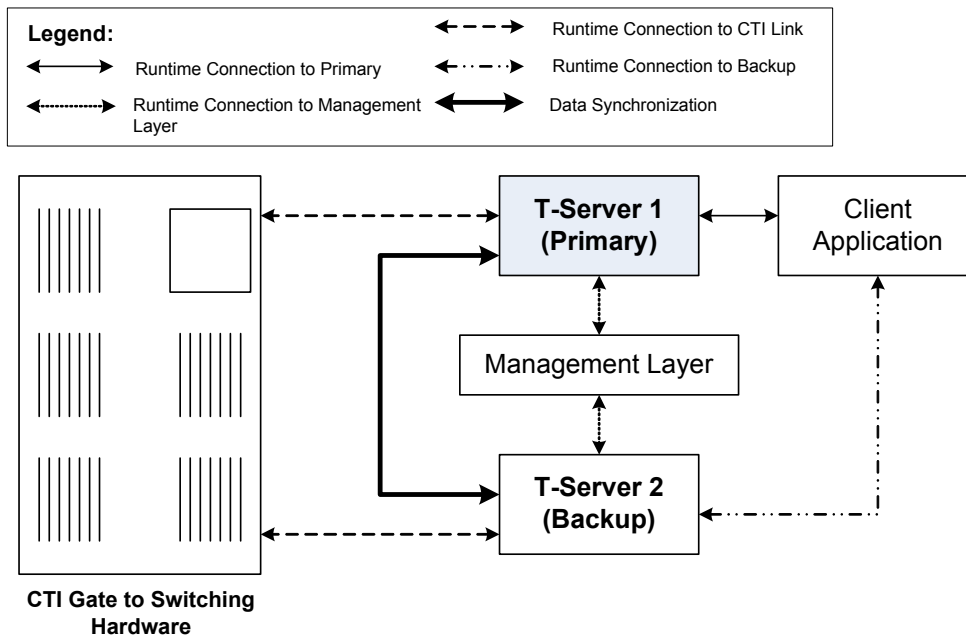


Figure 6: Hot Standby Redundancy Architecture

Benefits of Hot Standby Redundancy

The hot standby redundancy type provides the following benefits over the warm standby type:

- Using hot standby ensures the processing of interactions in progress if a failure occurs. After the primary T-Server (T-Server 1) fails, T-Server 2 handles all new interactions and takes over the processing of interactions that are currently in progress.
- T-Servers perform one-way (from primary to backup) synchronization of call-associated data, including, but not limited to:
 - Connection IDs.
 - Attached user data.
 - Inter Server Call Control (ISCC; formerly called External Routing) call references to another site in a multi-site environment (to support the ISCC/COF feature).

Note: Refer to “ISCC Call Data Transfer Service” on [page 41](#) for ISCC feature descriptions.

- When mirrored links are not available, HA Proxy helps T-Server synchronize the current states of agents, calls, parties, and devices between the primary and backup T-Servers.

However, keep the following hot standby limitations in mind:

- Client requests sent during the failure and switchover may be lost.
- Routing requests sent by the switch during the failure and switchover may be lost.
- T-Server does not synchronize interactions that begin before it starts, including incomplete ISCC-related transactions.
- 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 T-Servers.

Requirements

You must install the Management Layer if you are installing redundant T-Server applications. In particular, install Local Control Agent (LCA) on each computer that runs T-Server.

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

Synchronization Between Redundant T-Servers

When T-Servers operate in a high-availability environment, the backup T-Server must be ready to take on the primary role when required. For this purpose, both T-Servers must be running and must have the same information. When you configure redundant T-Servers to operate with the hot standby type, the primary T-Server uses the connection to the backup to deliver synchronization updates. Genesys recommends that you enable the Advanced Disconnect Detection Protocol (ADDP), described in Chapter 1, for this connection. Do so using the configuration options in the “Backup-Synchronization Section” section. Refer to the “T-Server Common Configuration Options” chapter for option descriptions.

Configuration Warnings

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

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

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

Warm Standby Deployment

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

General Order of Deployment

The general guidelines for T-Server warm standby configuration are:

1. Configure two T-Server Application objects as described in “Configuration of T-Server” on [page 21](#).
2. Make sure the Switch object is configured for the switch these T-Servers should serve, as described in “Configuration of T-Server” on [page 21](#).
3. Modify the configuration of the primary and backup T-Servers as instructed in the following sections.

After completing the configuration steps, ensure that both T-Servers are installed (see [page 34](#)).

Modification of T-Servers for Warm Standby

Modify the configuration of both the primary and backup T-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 number of ports, their Port IDs, and the Listening Mode settings of the primary and backup servers must match respectively.

Procedure:

Modifying the primary T-Server configuration for warm standby

Start of procedure

1. Stop both the primary and backup T-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 T-Server that you want to configure as a primary server.
4. Click the Switches tab.
5. Ensure that it specifies the Switch that this T-Server Application should serve. If necessary, select the correct Switch using the Browse button.
6. Click Apply to save the configuration changes.
7. Click the Server Info tab.
8. Specify the T-Server Application you want to use as the backup server. Use the Browse button next to the Backup Server field to locate the backup T-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

- [Procedure: Modifying the backup T-Server configuration for warm standby](#), on page 34

Procedure: Modifying the backup T-Server configuration for warm standby

Start of procedure

1. Make sure the two T-Servers are *not* running.
2. Open the Configuration Manager main window.
3. Open the Properties dialog box of the Application object for the T-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 T-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

Warm Standby Installation of Redundant T-Servers

The installation of a redundant T-Server is the same as that for the stand-alone T-Server. If you have not installed the primary and backup T-Servers yet, follow the instructions in “Installation of T-Server” on [page 22](#) for both installations.

Hot Standby Deployment

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

General Order of Deployment

The general guidelines for T-Server hot standby configuration are:

1. Configure two T-Server Applications objects as described in “Configuring T-Server” on [page 21](#).

2. Make sure the `Switch` object is configured for the switch these T-Servers should serve, as described in “Configuration of Telephony Objects” on [page 18](#).
3. Modify the configuration of the primary and backup T-Servers as instructed in the following sections.

After completing the configuration steps, ensure that both T-Servers are installed (see [page 37](#)).

Modification of T-Servers for Hot Standby

Modify the configuration of both the primary and backup T-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 number of ports, their Port IDs, and the Listening Mode settings of the primary and backup servers must match respectively.

Procedure:

Modifying the primary T-Server configuration for hot standby

Start of procedure

1. Stop both primary and backup T-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 T-Server that you want to configure as a primary server.
4. Click the `Switches` tab.
5. Ensure that it specifies the `Switch` that this T-Server `Application` should serve. If necessary, select the correct `Switch` 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`.

Note: For information on adding multiple ports, see “Configuring multiple ports” on [page 22](#).

- 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 T-Server will connect to the *default* port of the primary T-Server.

9. Specify the T-Server Application you want to use as the backup server. Use the `Browse` button next to the Backup Server field to locate the backup T-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 T-Servers, click the `Options` tab. Open or create the backup-sync section and configure corresponding options.
16. Click `Apply` and `OK` to save the configuration changes.

End of procedure

Next Steps

- [Procedure: Modifying the backup T-Server configuration for hot standby, on page 36](#)

Procedure: Modifying the backup T-Server configuration for hot standby

Start of procedure

1. Make sure the two T-Servers are *not* running.
2. Open the Configuration Manager main window.

3. Open the Properties dialog box of the Application object for the T-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 T-Server Application.
6. Click the Server Info tab.
7. In the Ports section, select the port to which the primary server will connect for HA data synchronization and click Edit Port.

Note: For information on adding multiple ports, see “Configuring multiple ports” on [page 22](#).

- 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 primary T-Server will connect to the *default* port of the backup T-Server.

8. Click Apply to save the configuration changes.
9. Click the Start Info tab.
10. Select Auto-Restart.
11. Click the Options tab.
12. Modify the values for all necessary configuration options. Genesys recommends that you set all configuration options for the backup T-Server to the same values as for the primary T-Server; the only exceptions are the log options and the server-id option.
13. Click Apply and OK to save the configuration changes.

End of procedure

Hot Standby Installation of Redundant T-Servers

The installation of a redundant T-Server is the same as that for the stand-alone T-Server. If you have not installed the primary and backup T-Servers yet, follow instructions in “Installation of T-Server” on [page 22](#) for both installations.

Next Steps

At this point, you have learned how to configure and install redundant T-Servers. Go to Chapter 5, “Starting and Stopping T-Server and CSTA Connector,” on [page 97](#), to test your configuration and installation, or continue with Chapter 4, “Multi-Site Support,” on [page 39](#), for more possibilities.



Chapter

4

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:

- [Multi-Site Fundamentals, page 40](#)
- [ISCC Call Data Transfer Service, page 41](#)
- [ISCC/Call Overflow Feature, page 61](#)
- [Number Translation Feature, page 65](#)
- [Network Attended Transfer/Conference Feature, page 73](#)
- [Event Propagation Feature, page 75](#)
- [ISCC Transaction Monitoring Feature, page 82](#)
- [Configuring Multi-Site Support, page 82](#)
- [Next Steps, page 96](#)

Note: Each switch/T-Server combination offers different multi-site options. For details describing your specific switch/T-Server environment, refer to the “T-Server Common Configuration Options” chapter.

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 2 on [page 57](#) and Table 3 on [page 62](#).

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, CallType, and CallHistory). 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 48](#) and “Transfer Connect Service Feature” on [page 60](#).
 - 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 61](#)).
 - Number Translation feature (see [page 65](#)).
 - Network Attended Transfer/Conference (NAT/C) feature (see [page 73](#)).

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 76](#))
 - Party Events propagation (see [page 77](#))

Note: ISCC automatically detects topology loops and prevents continuous updates.

Note: In distributed networks, Genesys recommends using call flows that prevent call topology loops and multiple reappearances of the same call instance. This approach ensures that all T-Servers involved with the call report the same ConnID, and also optimizes telephony trunk allocation by preventing 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 connection identifier of the call (attribute ConnID).
- Updates to user data attached to the call at the previous site (attribute UserData).
- The call type of the call (attribute CallType)—In multi-site environments the CallType of the call may be different for each of its different legs. For example, one T-Server may report a call as an Outbound or Consult call, but on the receiving end this call may be reported as Inbound.
- The call history (attribute CallHistory)—Information about transferring/routing of the call through a multi-site contact center network.

Note: Load-sharing IVR Servers and Network T-Servers cannot be designated as the destination location for ISCC, except when cast-type is set to dnis-pool. Consult the *Universal Routing Deployment Guide* for specific configuration details.

Figure 7 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*.

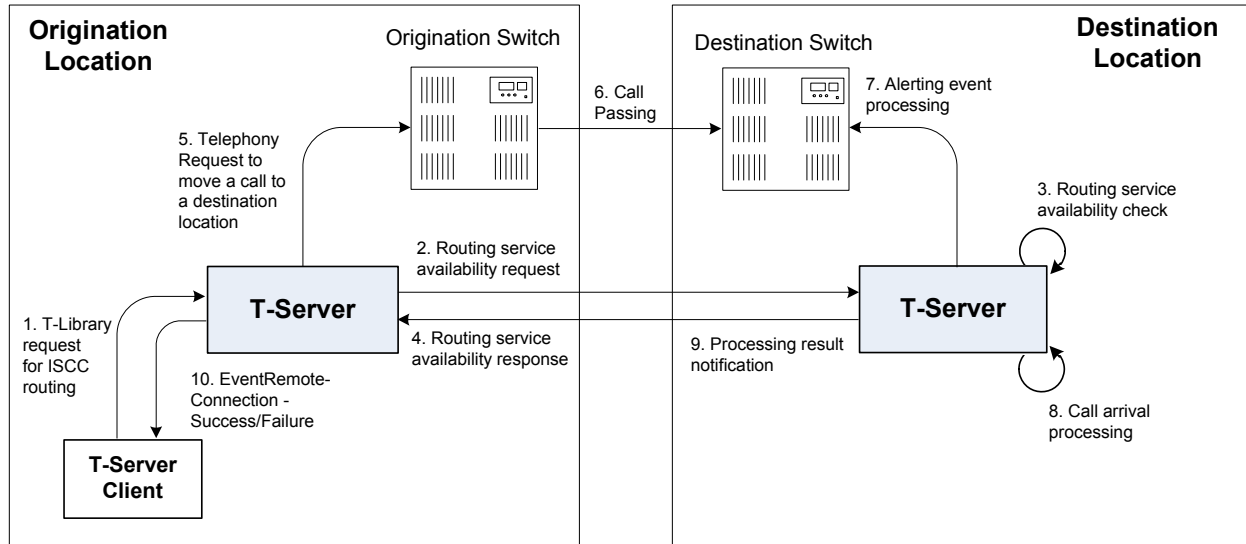


Figure 7: Steps in the ISCC Process

ISCC Call Flows

The following section identifies the steps (shown in Figure 7) 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 8.x .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-uu`, `direct-digits`, `reroute`, `dnis-pool`, `pullback`, or `route-uu`.
- 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: For more information on Access Codes and Default Access Code, see “Switches and Access Codes” on [page 84](#).

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`, `CallType`, 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 an Access Resource of type `dnis` 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. For option descriptions, refer to the “T-Server Common Configuration Options” chapter.

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`, `CallType`, 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`, `CallType`, 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.

Client-Controlled ISCC Call Flow

The following section identifies the steps that occur during a client-controlled ISCC transfer of a call.

Step 1

A client, such as Universal Routing Server (URS), that is connected to the T-Server at the origination location detects a call to be delivered to another destination location.

Step 2

The client chooses a destination location and the target DN for the call. Then, it sends the `TGetAccessNumber` request to the destination T-Server for routing service availability, indicating the target DN and other call context (`ConnID`, `UserData`, and `CallHistory` attributes).

Step 3

The destination T-Server receives the request for routing service availability. Depending on the ISCC transaction type, it stores the request information, including the call context. When appropriate, it allocates access resources for the coming call, such as External Routing Point.

If resources are unavailable, the request is queued at the destination T-Server until an appropriate ISCC resource is free or the client 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 `EventError` message to the client.

Step 4

The destination T-Server replies to the client with the `EventAnswerAccessNumber` message, which contains the allocated ISCC resource.

Step 5

The client requests that the origination T-Server delivers the call to the destination location using the allocated access resource.

Step 6

The origination T-Server receives and processes the client's request, and then sends a corresponding message to the switch.

Step 7

The call arrives at the destination switch and is reported to the destination T-Server via CTI. The call is matched by means of ISCC, based on the specified `cast-type` setting and allocated resource, and then the call is assigned a requested call context (such as `ConnID` or call data). Upon successful transaction completion, the destination T-Server notifies the client by sending `EventRemoteConnectionSuccess`.

The destination T-Server waits for the call no longer than the interval specified by the timeout that is 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 client by sending

`EventRemoteConnectionFailed`, deletes all data about the request, and, when appropriate, frees the resources previously allocated for the request.

The destination T-Server notifies the client whether the routing service succeeded or failed by sending either the `EventRemoteConnectionSuccess` or `EventRemoteConnectionFailure`, respectively.

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 49](#)).

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 49](#). Use Table 2 on [page 57](#) 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`.

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 49](#)
- `direct-notoken`, [page 51](#)
- `dnis-pool`, [page 52](#)
- `pullback`, [page 53](#)
- `reroute`, [page 54](#)
- `route` (aliased as `route-notoken`), the default transaction type, [page 55](#)

The following transaction types are unevenly supported for both the origination and destination T-Server roles:

- `direct-callid` (aliased as `direct`), [page 50](#)
- `direct-digits` (reserved for Genesys Engineering)
- `direct-network-callid`, [page 50](#)
- `direct-uui`, [page 51](#)
- `route-uui`, [page 56](#)

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 call 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 can use this network feature for call matching.

Warning! Depending on the switch platform, it may be 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 93](#) for details.)

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. It is applied only to the call that is in progress, and does not apply to functions that involve in the creation of a new call, such as `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.

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 to be 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. You must also ensure that the trunks involved do 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 only be reached from within the contact center (such as the second line of support, which customers cannot contact directly).

When using direct transaction types, Network T-Servers and load-sharing IVR Servers are not meant to act as 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`, `CallType`, 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 that employ a “behind the SCP” architecture, such as 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 only be used 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 by sending `EventRouteRequest` and attaching the call's user data.

Notes: The transaction type `reroute` can only be used 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 8](#).

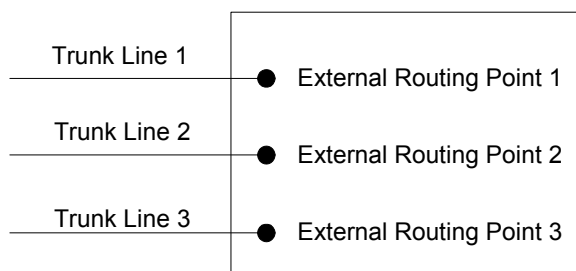


Figure 8: Point-to-Point Trunk Configuration

Note: Dedicated DNs of the External Routing Point type must be configured in a switch. See “Configuring Multi-Site Support” on [page 82](#).

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 9](#).

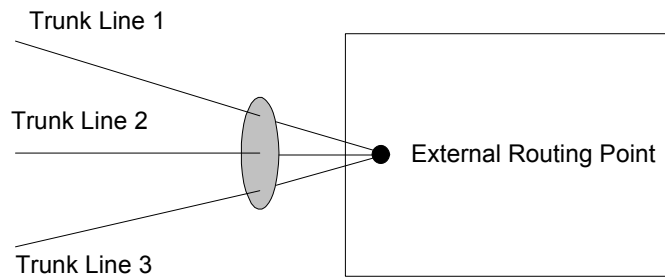


Figure 9: Multiple-to-Point Trunk Configuration

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 55](#)) and the UUI matching feature of the `direct-uui` transaction type ([page 51](#)). 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. You must also ensure that the trunks involved do not drop this data.

T-Server Transaction Type Support

[Table 2](#) 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 2: 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									
Aastra MXONE CSTA I	Yes			Yes ^a		Yes	Yes ^a				
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 ^c		Yes ^f	Yes ^f				
Avaya Communication Manager	Yes				Yes	Yes	Yes				
Avaya INDeX	Yes					Yes	Yes ^b				
Avaya TSAPI	Yes				Yes	Yes	Yes				
Cisco UCCE	Yes					Yes	Yes				
Cisco Unified Communications Manager	Yes			Yes		Yes	Yes				
CSTA Connector for BroadSoft BroadWorks	Yes					Yes	Yes				
DataVoice Dharma	Yes			Yes		Yes	Yes				
Digitro AXS/20	Yes			Yes		Yes					
EADS Intecom M6880	Yes			Yes		Yes	Yes				

Table 2: 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									
EADS Telecom M6500	Yes			Yes		Yes	Yes				
eOn eQueue	Yes			Yes		Yes					
Fujitsu F9600	Yes					Yes					
Huawei C&C08	Yes			Yes							
Huawei NGN	Yes					Yes	Yes				
Mitel MiTAI	Yes					Yes	Yes		Yes ^g		
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
Samsung IP-PCX IAP	Yes			Yes		Yes					
Siemens Hicom 300/HiPath 4000 CSTA I	Yes			Yes	Yes ^d	Yes	Yes				

Table 2: 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 3000	Yes			Yes		Yes					
Siemens HiPath 4000 CSTA III	Yes				Yes ^d	Yes	Yes				
Siemens HiPath DX	Yes				Yes ^h	Yes	Yes ⁱ				
SIP Server	Yes		Yes		Yes ^j	Yes					Yes
Spectrum	Yes	Yes		Yes		Yes ^f	Yes ^f				
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											

Table 2: T-Server Support of Transaction Types (Continued)

T-Server Type	Transaction Type										
	route		re-route	direct-callid	direct-uuui / route-uuui	direct-no-token	direct-ani	direct-digits	direct-network-callid	dnis-pool	pull-back
	one-to-one	multiple-to-one									
NGSN	Yes										Yes
Network SIP Server	Yes					Yes	Yes			Yes	
Sprint	Yes										
SR-3511											
Stentor											

- a. Not supported in the case of function `TRouteCall` 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 `UUUI` 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. Supported only for `TRouteCall` requests made from a Native Routing Point.
- h. Not supported if a `TMakeCall` request is made.
- i. Not supported if a `TInitiateTransfer` or `TInitiateConference` request is made from an outgoing call on a device.
- j. SIP Server supports the `direct-uuui` 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 silent 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 by using the `ASAI-send-DTMF-single` procedure.

ISCC/Call Overflow 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 without an explicitly defined destination location. 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 3 shows the T-Server types that provide the `NetworkCallID` of a call.

Table 3: T-Server Support of NetworkCallID for ISCC/COF Feature

T-Server Type	Supported NetworkCallID Attribute
Alcatel A4400/OXE ^a	Yes
Aspect ACD	Yes
Avaya Communication Manager ^{a,b}	Yes
Avaya TSAPI ^{a,b}	Yes
Cisco UCCE	Yes
Mitel MiTAI ^a	Yes
Nortel Communication Server 2000/2100 ^a	Yes
Nortel Communication Server 1000 with SCCS/MLS ^a	Yes
SIP Server ^a	Yes
Spectrum	Yes

a. Supported only if the `match-flexible` configuration parameter is used.

b. ISCC/COF is cross-compatible between T-Server for Avaya Communication Manager and T-Server for Avaya TSAPI.

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, CallType, and CallHistory are received for the matched call from the call's previous location.

Warning! Depending on the switch platform, it may be 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. For more information on feature configuration, see “Number Translation Feature” on [page 65](#).

ISCC/COF Call Flow

[Figure 10](#) 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.

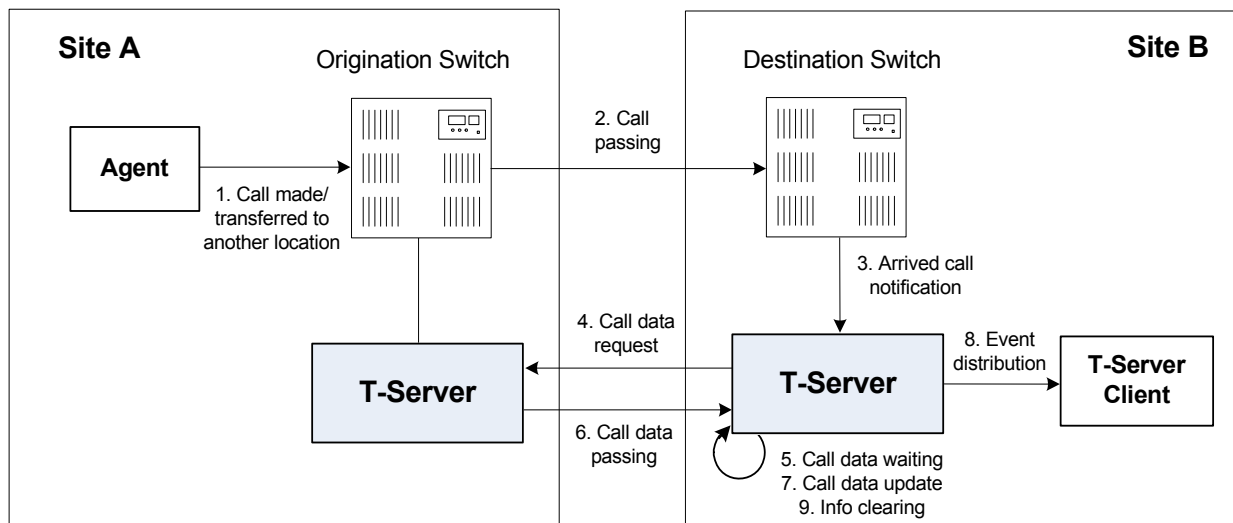


Figure 10: Steps in the ISCC/COF Process

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 overflowed at any of them.

To determine which calls to check as possibly having overflowed, T-Server relies on the Switch object and the presence of DNs on the Switch configured as the Access Resource type with the Resource Type set either to `cof-in` (COF-IN DNs) or to `cof-not-in` (COF-NOT-IN DNs):

T-Server skips an arriving call when one of following conditions is met:

- The call arrives at a DN configured as an Enabled COF-NOT-IN DN.
- COF-IN DNs are configured, but the call arrives at a DN other than one of the configured COF-IN DNs or to a COF-IN DN which is Disabled.

In all other cases, the call is 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, CallType, 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, CallType, 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 66](#) 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 71](#) 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 73](#).

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 following notation explanations begin with the highest level notation. Each explanation includes the name of a component notation and a basic definition of each component that it contains. 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 71](#)) 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=1AAAAAAAAA; 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]ABBBBBBBB; 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=91AAAAAAAAA; 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.
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 11 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).

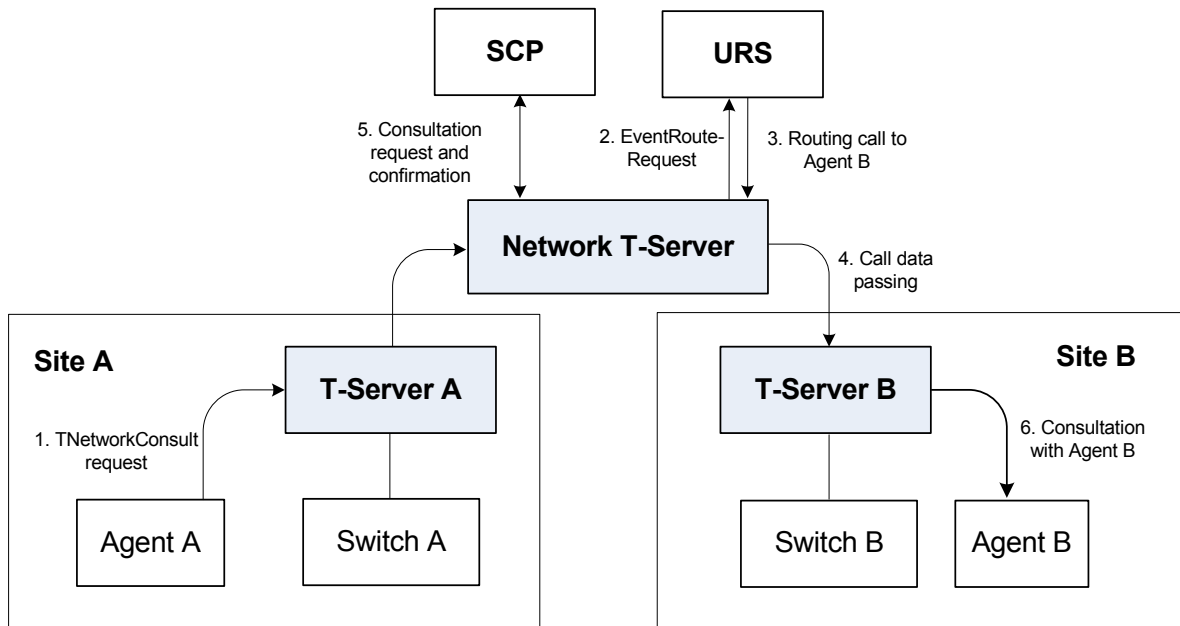


Figure 11: Steps in the NAT/C Process in URS-Controlled Mode

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 8.x .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 41](#) 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.

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 Events and Models Reference Manual*.

Switch Partitioning

A multi-site environment with switch partitioning or intelligent trunks can be defined as a configuration of multiple virtual switches (or `Switch` objects) that

are defined in Configuration Manager under a single Switching Office object representing a physical switch. Each Switch object has its own instance of a T-Server application. All T-Server applications connect to the switch via the same or different CTI link or a gateway. (See [Figure 12.](#))

When the Event Propagation feature is active, updated user data and party-related events—`EventPartyChanged`, `EventPartyDeleted`, and `EventPartyAdded`—are propagated to T-Servers that are involved in call transactions, such as transfer or conference. However, with switch partitioning, the call instances may reside at one partition or at different partitions.

Site A

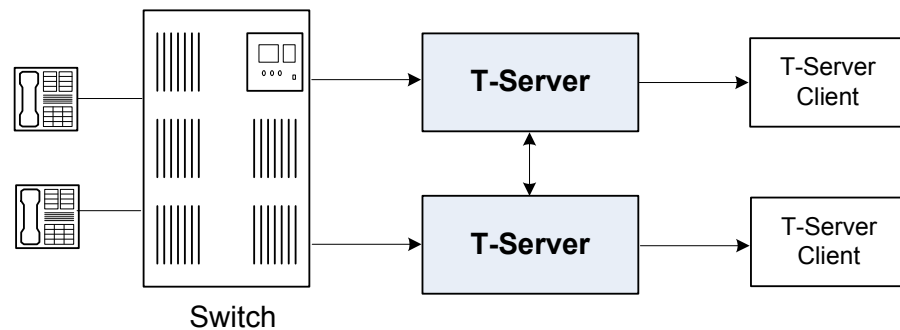


Figure 12: Switch Partitioning Architecture

Starting with version 8.0, in addition to `ConnIDs` and `UserData`, T-Server can synchronize the `CallType` attribute. Each T-Server is required to register all DNs it monitors. In a multi-partitioned environment, when configured, calls between partitions are reported as internal (`CallTypeInternal`). In a non-partitioned environment, such calls are reported as inbound (`CallTypeInbound`) and/or outbound (`CallTypeOutbound`), depending on the direction of a call. In order for T-Servers to report calls between specified partitions as internal, registered DNs of these partitions must be assigned to a Switch (T-Server), Switching Office, or Tenant, using the `dn-scope` configuration option. If DNs that are involved in calls are not in the T-Server scope, those DNs will be reported as inbound or outbound.

In addition, T-Server supports `LocalCallType` and `PropagatedCallType` attributes, which depend on the `propagated-call-type` configuration option setting for reporting.

To control race conditions that may occur in the switch-partitioned environment, use the `epp-tout` configuration option.

Notes: Because of possible delays in TCP/IP connections, a sequence of events sent for the same call by two or more T-Servers to clients may appear in an unexpected order. For example, in a simple call scenario with two partitions, `EventRinging` and `EventEstablished` messages may both arrive before `EventDialing`.

Genesys switch partitioning does not apply to hardware partitioning functionality that is supported on some switches.

Table 4 shows the T-Server types that support switch partitioning.

Table 4: T-Server Support for Switch Partitioning

T-Server Type	Supported
Alcatel A4400/OXE	Yes
Avaya Communication Manager	Yes
Avaya TSAPI	Yes
Cisco Unified Communications Manager	Yes
SIP Server	Yes

Event Propagation Configuration

The basic Event Propagation feature configuration includes a setting of specific configuration options at a T-Server Application level. The advanced feature configuration allows you to customize the feature at a 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 the “T-Server Common Configuration Options” chapter.
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:
[Procedure: Modifying Event Propagation: advanced configuration](#), on page 80

Procedure:**Modifying Event Propagation: advanced configuration**

Purpose: To modify access codes for advanced Event Propagation configuration.

Prerequisites

- [Procedure: Activating Event Propagation: basic configuration](#), on [page 80](#)

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 85](#) for instructions. If no Access Codes are configured, see [page 86](#) 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 events¹, 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:

-
1. The following are call-party-associated events: EventPartyChanged, EventPartyDeleted, and EventPartyAdded.

- 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. For more information about support of this feature, see *Genesys Events and Models Reference Manual* and *Voice Platform SDK 8.x .NET (or Java) API Reference*.

Configuring Multi-Site Support

Prior to configuring T-Server to support multi-site operation, you must read the "Licensing Requirements" on [page 15](#), as well as previous sections of this chapter on multi-site deployment. In particular, Table 2 on [page 57](#) shows which transaction types are supported by a specific T-Server, while Table 3 on [page 62](#) 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 name of each T-Server application, port assignments, and switch names), and use Configuration Manager to create and partially configure each T-Server object. Review option values in the *extrouter* section 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. DNSs

You must configure these objects for origination and destination locations. Multi-site support features activate automatically at T-Server startup. See “DNSs” on [page 90](#) 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 uses the default values of the corresponding configuration options.

5. Open the `extrouter` section. Configure the options used for multi-site support.
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 8.1, or when using T-Servers of different releases (including 8.1) in the same environment, see “Compatibility Notes” on [page 89](#).

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 5](#):

Table 5: Target Type: ISCC Protocol Parameters

ISCC Protocol Parameters	Description
dnis-tail=<number-of-digits>	Where <code>number-of-digits</code> is the number of significant DNIS digits (last digits) used for call matching. <code>0</code> (zero) matches all digits.
propagate=<yes, udata, party, no>	Default is <code>yes</code> . For more information, see “Modifying Event Propagation: advanced configuration” on page 80 .
direct-network-callid=<>	Use Table 3 on page 62 to determine if your T-Server supports the <code>direct-network-callid</code> 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 6](#):

Table 6: Target Type: ISCC Call Overflow Parameters

ISCC Call Overflow Parameters	Description
match-callid	Matches calls using network <code>CallID</code> .
match-ani	Matches calls using ANI. Note: When using <code>match-ani</code> , the <code>match-flexible</code> parameter must be set to <code>false</code> .

Table 6: Target Type: ISCC Call Overflow Parameters (Continued)

ISCC Call Overflow Parameters	Description
match-flexible	Supports flexible call matching based on the following values: Default Value: true Valid Values: true, false, and [matching-context-type], where [matching-context-type] is the switch-specific value, which must be the same as the value of the default-network-call-id-matching configuration option of the corresponding T-Server.
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 7](#) contains cross-reference information on transaction types that the Configuration Layer and T-Server use.

Table 7: 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 UII	direct-uuu
DNIS Pooling	dnis-pooling

Table 7: Route Type and ISCC Transaction Type Cross-Reference (Continued)

Route Type Field Value	ISCC Transaction Type
Forbidden	External routing to this destination is not allowed
ISCC defined protocol	Reserved
PullBack	pullback
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 8.1, or when using T-Servers of different releases (including 8.1) 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 8.x, 7.x, 6.5, and 6.1:
 - Use the Target ISCC access code for transactions with T-Servers of releases 8.x, 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 and later.

- UseExtProtocol to enable the direct-uu i 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):

- Access number (if specified).
- 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).
- 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.
- Default access code of the switch to which the Routing Point belongs, concatenated with its `Association` (if the `Association` value is specified).
- 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.

- 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

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

Start of procedure

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—such as Aspect ACD, Nortel Communication Server 2000/2100, and Spectrum.

End of procedure

Procedure: Configuring access resources for ISCC/COF

Purpose: To configure dedicated DNs required for the ISCC/COF feature.

Start of procedure

Note: Use Table 3 on [page 62](#) 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.
2. On the **General** tab of the DN Properties dialog box, enter the name of the configured DN in the **Number** field.

Note: The name of a DN of type **Access Resource** must match the name of a DN in your configuration environment (typically, a DN of type **Routing Point** or **ACD Queue**), so T-Server can determine whether the calls arriving at this DN are overflowed calls.

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 84](#).

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 90](#).
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 86](#).

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 86](#).

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 5, “Starting and Stopping T-Server and CSTA Connector,” on [page 97](#) to test your configuration and installation.



Chapter

5

Starting and Stopping T-Server and CSTA Connector

This chapter describes methods for stopping and starting T-Server and CSTA Connector. It includes these sections:

- [Command-Line Parameters, page 97](#)
- [Starting and Stopping with the Management Layer, page 99](#)
- [Starting with Startup Files, page 100](#)
- [Starting Manually, page 101](#)
- [Verifying Successful Startup, page 103](#)
- [Stopping Manually, page 103](#)
- [Starting and Stopping with Windows Services Manager, page 104](#)

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	<p>The license address. Use for the server applications that check out technical licenses. Can be either of the following:</p> <ul style="list-style-type: none"> • 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. <p>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.</p>
-V	<p>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.</p>
-nco X/Y	<p>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.</p>
-lmspath	<p>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.</p> <p>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.</p>
- transport-port <port number>	<p><port number> is the port number that a client will use for its TCP/IP connection to Configuration Server. See the Client-Side Port Definition section in the <i>Genesys 8.x Security Deployment Guide</i> for more information.</p>
- transport-address <IP address>	<p><IP address> is the IP address that a client will use for its TCP/IP connection to Configuration Server. See the Client-Side Port Definition section in the <i>Genesys 8.x Security Deployment Guide</i> for more information.</p>

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 T-Server to start with the Management Layer

Start of procedure

1. Open the T-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 97](#).
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 T-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 8.1 Deployment Guide*.) *Framework 8.0 Solution Control Interface Help* provides complete instructions on starting and stopping applications.

You can also use the Management Layer to start a T-Server that has failed. To enable T-Server's autorestart 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”](#) to identify which applications should be running for a particular application to start.

Procedure:

Starting T-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 T-Server on Windows with a startup file

Start of procedure

To start T-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 97](#).

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"`

You must specify the rest of the command-line parameters as for any other application.

The following sections provide general instructions for starting T-Server and CSTA Connector manually. Along with these instructions, refer to [Table 8](#), which lists executable file names for Windows and UNIX operating systems.

Table 8: T-Server and CSTA Connector Executable Names

T-Server Type	Executable File Name	
	UNIX	Windows
CSTA Connector for BroadSoft BroadWorks	ConnectorBW_server	ConnectorBW_server.exe
T-Server for CSTA Connector	tfcc_server	tfcc_server.exe

T-Server and CSTA Connector

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

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

The command-line parameters common to Framework server components are described on [page 97](#).

Procedure:

Starting T-Server on UNIX manually

Start of procedure

1. Go to the directory where T-Server is installed and type the following command-line:

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

2. Replace <switch>_server with the correct T-Server executable name, which depends on the type of the switch used.

See Table 8 on [page 102](#) for executable names.

End of procedure

Procedure: Starting T-Server on Windows manually

Start of procedure

1. Start T-Server from either the Start menu or the MS-DOS window. If using the MS-DOS window, go to the directory where T-Server is installed and type the following command-line:

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

2. Replace <switch>_server.exe with the correct T-Server executable name, which depends on the type of the switch used.
See Table 8 on [page 102](#) for executable names.

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 T-Server, check whether Solution Control Interface displays Started or Service Unavailable status for the corresponding application. Refer to the “Troubleshooting” section of the *Framework 8.0 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 T-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:

- T-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, T-Server, CSTA Connector, and Stat Server.

Procedure: Stopping T-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 T-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 97](#) and

<code>-service</code>	The name of the Application running as a Windows Service; typically, it matches the Application name specified in the <code>-app</code> 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.

Genesys T-Server for CSTA Connector

Supported Features and Options

T-Server for CSTA Connector Supported Features Agent Substitution for Monitored Agents



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Agent Substitution for Monitored Agents

Agent Substitution for Monitored Agents

Genesys reporting requires that the Genesys should use the agent's device for call events, reporting the agent id in a separate field. When the agent is monitored the switch will typically report call events using the agent id as the event device and T-Server needs to replace this with the agent device in TLibrary reporting.

The T-Server checks the device id of call events sent to it to see whether it matches the id of a currently logged in agent which has monitoring enabled. If there is a match the T-Server uses the agent's associated device for the ThisDN field in the reported Genesys TLibrary event(s) and the agent id for the agentID field.

T-Server for CSTA Connector Supported Features Business-Call Handling



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Business-Call Handling

Business-Call Handling

This section describes how T-Server handles different types of calls.

T-Server Call Classification

T-Server automatically assigns every call to one of four categories:

- business
- work-related
- private
- unknown

Based on this assignment, T-Server applies the appropriate business-call handling after the call is released.

Business Call Type Configuration

T-Server uses the following options to determine the business call type of a call. The options are given in order of precedence, highest to lowest:

- T-Server supports a request extension to define the business call type upon call initiation or answer.
- T-Server uses the originator agent state to determine if the call is work related.
- The Configuration Manager Annex tab configuration option, `bsns-call-type`, specifies the business call type for calls that pass through or arrive at a distribution device. If the call passes through a distribution device and there is no DN-level option present, the call will be classified as a business call as long as this is enabled by the option `bsns-call-dev-types`. Automatic classification of calls as business calls on distribution DNs can be disabled by the application option `bsns-call-dev-types`. If the corresponding flag of this option is disabled, calls passing distribution DNs of that type will not change their respective business classification. The request extension and DN-level option will still be effected. Distribution devices include the following device types:
 - Routing Point
 - ACD Queue
 - Routing Queue
 - External Routing Point
- T-Server supports options that are configured on the Application level to define whether specific call types (inbound, outbound, internal or unknown) are to be classified as business calls.

The extension `BusinessCallType` can be attached to `TMakeCall`, `TInitiateTransfer`, `TMuteTransfer`, `TInitiateConference` and `TMakePredictiveCall` to specify the business call type for the new call. This extension will take precedence over all other Configuration Manager business call type configuration options. The extension can also be attached to `TAnswerCall` to specify the business call type for the answering party and call.

The DN-level option `bsns-call-type` specifies the business call type for calls that pass through or arrive at a distribution device (Routing Point, Route Queue or ACD Queue). If the call passes through a distribution device and there is no Annex tab option present, the call will be classified as a business call. If the call passes through more than one distribution device, then the usual rules for assigning a business call type are followed. Once set, the business call type cannot be overridden unless it is changed to be a business call.

When the configuration options `inbound-bsns-calls`, `internal-bsns-calls` and `outbound-bsns-calls` are set at the Application level, they control whether the call type of the associated calls are to be classified as business calls. T-Server will not classify the business type of the call using these options until the destination is known. Also these options are not be used to set the originating party's business type as business until after the `Dialing` event has been reported. (This is to ensure that Genesys reporting is consistent regardless of the switch reported order of events).

The private request, `TSetBusinessCall` allows T-Server clients to set the business call type of an existing call to be "business". T-Server responds to a successful request by distributing the private event `EventBusinessCallSet`.

Business Calls

The `bsns-call-dev-types` option determines whether a call on a distribution device is promoted to a business call type. Use the following configuration options to define what additional calls to or from an agent are classified as business calls:

- `inbound-bsns-calls`
- `outbound-bsns-calls`
- `inherit-bsns-type`
- `internal-bsns-calls`
- `unknown-bsns-calls`

Work-Related Calls

T-Server categorizes as a work-related call any non-business call that an agent makes while in ACW. T-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, T-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, T-Server cancels the legal-guard timer and reapplies it at the end of the work-related call.

Private Calls

T-Server categorizes as a private call any call that does not fall into the business or work-related categories. T-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.

Unknown Calls

Any call that does not fall into any of the above categories is classified as an unknown call.

T-Server for CSTA Connector Supported Features Call Release Tracking



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Call Release Tracking

Call Release Tracking

T-Server now provides information about which party initiated the release of a call. This can be valuable for different applications to provide historical and real-time call reporting.

The following T-Library SDK call models can now be reported in this way:

- Normal call release.
- Abnormal call release.
- Call release from a conference.
- Rejection of an alerting call.
- Release for a failed or blocked call to a busy destination.

DN-Based Reporting

In DN-based reporting, information about the call release initiator will be reported in the `AttributeExtension` using extension key `ReleasingParty` in `EventReleased` and `EventAbandoned` events, when those events are distributed.

One of the following values will be reported in the `ReleasingParty` key:

- 1 `Local`—The call is released because the `ThisDN` value in the `EventReleased` was requesting the release.
- 2 `Remote`—The call is released because the other party (which is remote to `ThisDN`) in the `EventReleased` or `EventAbandoned` events was requesting release operation.
- 3 `Unknown`—The call is released, but T-Server cannot determine the release initiator.

Call-Based Reporting

Independently of DN-based reporting, T-server provides the call release initiator in `AttributeCtrlParty` for `EventCallPartyDeleted` and `EventCallDeleted` events. For scenarios where T-Server cannot provide the release initiator, `AttributeCtrlParty` will not appear in event reporting.

T-Server will provide `AttributeCtrlParty` reporting (for the party that initiated the call release) either:

- When the call is released using a GCTI request and T-Server is aware of the result of the requested operation, or;
- The PBX CTI protocol provides reliable information about the identity of party that released.

T-Server for CSTA Connector Supported Features Call Type Prediction



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Call Type Prediction

Call Type Prediction

T-Servers use CTI-provided information to assign a call type to a call. On occasions when the CTI information is either insufficient or arrives too late for T-Server to assign a definite call type, T-Server can now use a call type prediction procedure to assign a call type on a "best possible guess" basis.

Call Type Prediction is only effective if the common `dn-scope` option is not configured. If this option is configured, the call type assignment follows the rules enforced by the option.

The following table shows how T-Server assigns call types in different scenarios.

Call Type Prediction

Call Direction/OtherDN	External	Internal	Unknown
Incoming	CallTypeInbound	CallTypeInternal	CallTypeUnknown
Outgoing	CallTypeOutbound	CallTypeInternal	CallTypeUnknown

The feature is enabled/disabled by the configuration option `call-type-by-dn`.

You can configure this feature in T-Server using the following options:

- `call-type-by-dn`
- `call-type-rules`
- `rule-<n>`, where `n=1-N`

T-Server for CSTA Connector Supported Features Emulated Agents



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Emulated Agents

Emulated Agents

T-Server will emulate the following functionality for agents in situations where the PBX does not support it:

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

- Login and logout
- Agent set ready
- Agent set not ready (using various work modes)
- Automatic after call work (ACW)
- After call work in idle
- Automatic legal-guard time to provide a minimum break between business related calls

Depending on the PBX capability T-Server can emulate none, some or all of these features.

Emulated Agent Login/Logout

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

- `emulate-login`
- `emulated-login-state`
- `agent-strict-id`
- `agent-emu-login-on-call`

Agent Logout on Client Unregistering from DN

In some scenarios (such as a power failure/disconnection or when a desktop stops responding), agents may still receive calls but be unable to handle them. To prevent this problem, T-Server can be configured to automatically logout the agent in such circumstances.

When a client desktop or application disconnects from T-Server while an agent is still logged in, T-Server receives a notification that the application is unregistering from the agent's DN. Also, T-Server is able to uniquely identify the client application which sends a T-Library request, including `TAgentLogin` and `TRegisterAddress`.

T-Server can associate the client application (the one that sends the initial `TAgentLogin` request) with the agent and automatically log that agent out when the client application unregisters the agent DN while the agent is still logged in. (The initial `TAgentLogin` request is the one which first logs the agent in).

This feature is enabled/disabled by the following configuration options:

- `agent-logout-on-unreg`
- `agent-logout-reassoc`


HA Considerations

If T-Server is running in HA mode, a client connecting to one T-Server will be connected to both with the same session ID. Therefore the client's session ID must be used as part of the association data to ensure consistency across the primary and backup T-Servers. The primary T-Server will send an HA synchronization message to the backup when there is a change in client associations.

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.

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

 **Note:** The *Genesys Events and Models Reference Manual* and the *Voice Platform SDK 8 .NET (or Java) API Reference* define which agent state/agent mode transitions are permissible.

Emulated After-Call Work (ACW)

T-Server can apply emulated wrap-up (after-call work or ACW) for agents after a business call is released, unless the agent is still involved in another business call (see [Business Calls](#)).

Timed and Untimed ACW

T-Server applies emulated ACW for an agent after any business call is released from an established state. T-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

T-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.


T-Server reports ACW using `EventAgentNotReady` with `workmode` (`AgentAfterCallWork`) and indicates the amount of ACW it will apply using the extension `WrapUpTime`.

T-Server sends the `EventNotReady (ACW)` before the `EventReleased` at the end of the business call.

Emulated ACW Period

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

The 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 backwards compatibility, the default value of `untimed-wrap-up-value` is 1000.

 **Note:** Changing the value of untimed ACW should be done with care, because 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, or disable ACW altogether. If raised it may change untimed or disabled ACW to timed ACW. The use of the new 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`.

See the following related options for more details:

- `wrap-up-time`
- `untimed-wrap-up-value`
- `wrap-up-threshold`


Pending ACW

An agent can request emulated ACW, or override the period of (emulated) ACW to be applied to themselves, while on an established call. T-Server will apply the emulated ACW when the call is released. The agent sends `TAgentSetReady` with `workmode = 3` to request pending ACW while on an established call. The extension `WrapUpTime` indicates the amount of ACW that T-Server will apply, using the following parameters and rules:

- Extension missing - untimed ACW
- Value = 0 - ACW is disabled
- Value greater than 0 - period of timed ACW in seconds
- Value = `untimed` - untimed ACW
- Invalid value - request is rejected

If the request is successful, T-Server sends `EventAgentReady` with `workmode = 3` (ACW). T-Server will also indicate that the agent is in a pending ACW state by adding the extension `ReasonCode` with the new value `PendingACW`. It will also indicate the period of ACW to be applied using the extension `WrapUpTime`.

An agent may alter the period of pending ACW by sending a new `TAgentSetReady` with `workmode = 3`, using a different value for the extension `WrapUpTime`. If the request is successful, T-Server sends another `EventAgentReady` event, indicating the new value in the extension `WrapUpTime`.

 **Note:** To enable this feature the agent desktop the extension `WrapUpTime` must be enabled on the agent desktop.

Emulated ACW In Idle

An agent can activate wrap-up time on request when idle, by issuing a `TAgentSetNotReady` with `workmode = 3` (`AgentAfterCallWork`) to request emulated ACW while idle.

You can configure this feature in T-Server using the following options:

- `timed-acw-in-idle`
- `acw-in-idle-force-ready`

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 `TAgentSetNotReady` with `workmode = 3` (`AgentAfterCallWork`). T-Server determines the period of the extended ACW from the extension `WrapUpTime`, as follows:

- Value = 0 - No change to ACW period, but T-Server reports how much ACW time remains.
- Value greater than 0 - T-Server adds the given number of seconds to the timed ACW period. Untimed ACW remains unaffected.
- Value = `untimed` - T-Server applies untimed ACW.

T-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, T-Server places the agent back into 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, T-Server applies legal guard if any is configured. No legal guard is applied if the emulated ACW was untimed.

 **Note:** To enable this feature the agent desktop the extension `WrapUpTime` must be enabled on the agent desktop.

Emulated Legal-Guard Time

T-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 to guarantee 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.

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

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

The period of legal guard is determined by the configuration option `legal-guard-time`.

HA Synchronization

On startup and link re-establishment, the Hot Standby backup T-Server requests the primary T-Server to send details of all agents. The primary T-Server replies with all the information required for switchover, including all emulated and switch-based data.

From this point on, the primary T-Server also sends a similar synchronization message whenever an emulated agent's state changes.

This means that a higher level of synchronization between the two T-Servers is maintained at all times.

T-Server for CSTA Connector Supported Features Error Messages



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Error Messages

Error Messages

The following tables present the complete set of error messages T-Server distributes with `EventError`.

T-Server-Defined Errors

Code	Description
50	Unknown error
51	Unsupported operation for the switch
52	Internal error
53	Invalid attribute
54	Switch not connected
98	Cannot complete transfer
119	Invalid password
177	Target DN invalid
714	Invalid Call_ID

ISCC (Inter Server Call Control) Errors

Code	Description
1000	Invalid or missing server location name
1001	Remote server disconnected
1002	Remote server has not processed request
1003	Wrong protocol version
1004	Remote link disconnected
1005	External routing feature not initiated
1006	No free CDNs
1007	No access number
1008	TCS feature is not initiated
1009	Invalid route type
1010	Invalid request
1011	No primary server was found on location
1012	Location is invalid or missing
1013	Timeout performing requested transaction
1014	No configured access resources are found
1015	No registered access resources are found

1016	Client is not authorized
1017	Invalid transaction type
1018	Invalid or missing transaction data
1019	Invalid location query request
1020	Invalid origin location

Switch-Specific Errors

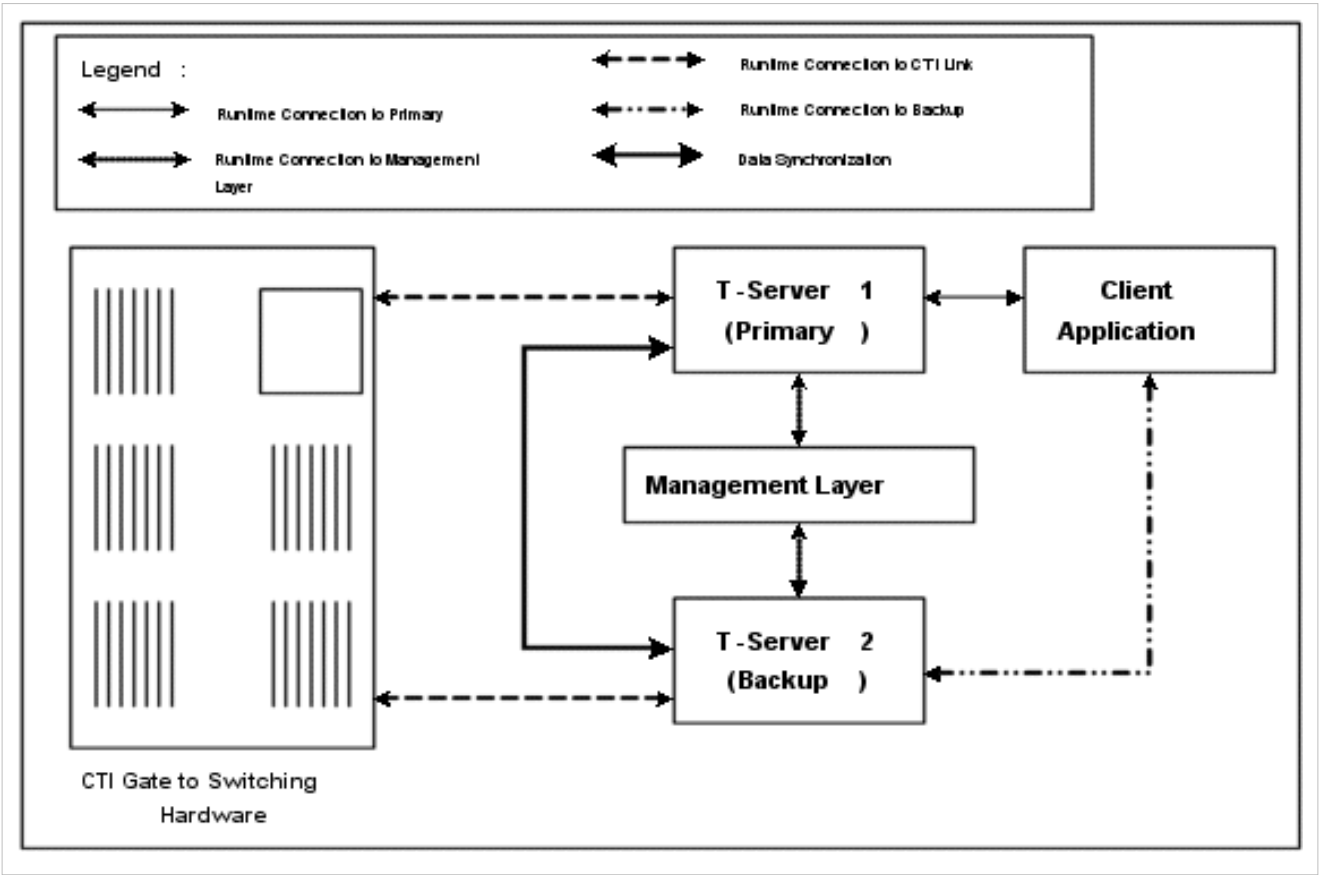
Code	Description
109	Link down or bad link specified
231	DN is busy
232	No answer at DN
233	Call rejected
1102	Badly structured APDU
1110	Duplicate invocation (packet missed)
1131	Unexpected error response
1132	Unrecognized error
1141	Request incompatible with object
1143	Object not known
1147	Request caused privilege violation on device
1153	Invalid call destination
1154	Invalid feature requested
1156	Invalid cross-reference identifier
1161	Invalid object state
1162	Invalid Connection ID
1173	Resource is out of service
1181	Object monitor limit exceeded
1190	Unspecified security error

T-Server for CSTA Connector Supported Features Hot-Standby HA Synchronization

Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Hot-Standby HA Synchronization

Hot-Standby HA Synchronization

This section describes how T-Server supports hot-standby HA synchronization. The following Figure shows the process of successful detection of T-Server synchronization. The primary T-Server is assumed to have successfully completed switch synchronization.



Primary T-Server Still in Start-up Phase

If the primary T-Server is still in the process of switch synchronization when it receives a `aBackup Ready` message from the backup T-Server, the primary T-Server sends the `Full Sync Done` message immediately. This allows the backup T-Server to send `EventLinkConnected` and become available. The Management Layer then sets the backup T-Server as the new primary, and vice versa. Once the old primary T-Server finishes switch synchronization, it initiates T-Server synchronization with the new primary T-Server as shown in the table above.

Primary T-Server's Link with the Switch is Down

If the primary T-Server has lost communication with the switch when it receives a `Backup Ready` message from the other T-Server, it sends the `Full Sync Done` message immediately. It can be assumed to have lost synchronization with the switch itself and there is no guarantee that it will recover communication with the link, which the backup T-Server currently has.

Backup T-Server Fails During Synchronization

If the backup T-Server fails while waiting for synchronization, the primary T-Server stops the synchronization process.

Primary T-Server Fails During Synchronization

If the primary T-Server fails while waiting for synchronization, then the backup T-Server sends an `EventLinkConnected` message immediately.

Call Synchronization Between T-Servers

An integral part of T-Server synchronization is the synchronization of the Connection IDs of the calls between T-Servers. It is the Connection IDs of calls created by the backup T-Server during the switch synchronization phase that differ from those in the primary T-Server—those created afterwards are synchronized by the normal HA mechanism. When the primary T-Server receives the `Backup Ready` message from the backup T-Server, it tags all current calls. Once all tagged calls have been released, the primary T-Server can be certain that the Connection IDs for all current calls have been synchronized with the backup T-Server because they were created after the backup T-Server completed its startup phase. If no further T-Server synchronization is required, the primary T-Server sends the `Full Sync Done` message to the backup T-Server.

Configuration Option

The configuration option `ha-sync-dly-lnk-conn` enables control of this feature.

T-Server for CSTA Connector Supported Features Keep-Alive Feature Handling



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Keep-Alive Feature Handling

Keep-Alive Feature Handling

T-Server may not always receive timely notification when the CTI link stops functioning. In order for T-Server to detect link failure and initialize alarm and recovery procedures, T-Server usually needs to actively check the link's integrity. This is referred to as Keep-Alive or "KPL" functionality.

Keep-alive functionality involves sending a KPL request which elicits either a positive or negative response from the CTI link. The responses are counted in four cumulative counters. If the relevant counter reaches the maximum configured limit, T-Server either:

- Decrements the relevant warning/failure KPL tolerance counter
- Attempts to reconnect to the link
- Sends a warning message to Message Server

Three configuration options are available in the Link-Control section of T-Server:

- `kpl-interval` sets the interval timer for KPL requests.
- `kpl-tolerance` sets the threshold at which T-Server either attempts to reconnect to the link or issues a warning message.

T-Server for CSTA Connector Supported Features Link Bandwidth Monitoring



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Link Bandwidth Monitoring

Link Bandwidth Monitoring

T-Server provides bandwidth monitoring on a CTI link and can notify the Genesys Management Layer when the Configuration Layer limits are exceeded.

When configured high or low thresholds are reached, T-Server sends alarm messages `LINK_ALARM_HIGH LMS` or `LINK_ALARM_LOW LMS`, as appropriate.

High and Low Watermarks

Configuration option `link-alarm-high`, specified as a percentage of the `max-bandwidth` value, defines an upper threshold bandwidth value which when breached raises a `LINK_ALARM_HIGH LMS` message.

Configuration option `link-alarm-low`, specified as a percentage of the `max-bandwidth` value, defines a lower threshold bandwidth value which when breached raises a `LINK_ALARM_LOW LMS` message.

Alarm Set Algorithm

T-Server measures requests sent to the CTI link and whenever there is a 99.7% probability that a high or low watermark threshold has been crossed, an appropriate LMS message is generated.

If `link-alarm-high` is set to 0 (zero), no high alarm will be generated.



Note: A high or low watermark LMS message will only be generated when there is at least a 99.7% probability that the requisite threshold has been crossed. Therefore, if the `link-alarm-low` is set to 0 (zero), it cannot be crossed and no low alarm can be generated. Since a subsequent high alarm LMS message will only be generated after a low watermark message, no further alarms can be raised.

Configuration Options

The following configuration options are used to set bandwidth monitoring on a CTI link:

- `link-alarm-high`
- `link-alarm-low`
- `use-link-bandwidth`

LMS Messages

High alarm

STANDARD Link bandwidth: %d1 requests per second exceeds alarm threshold %d2 requests per second on CTI link ID %d3

Attributes:

%d1 represents estimated requests rate

%d2 represents `link-alarm-high * max_bandwidth / 100`

%d3 represents the CTI Link ID

Low alarm

STANDARD Link bandwidth: %d1 requests per second dropped below alarm threshold %d2 requests per second on CTI link ID %d3

Attributes:

%d1 represents estimated requests rate

%d2 represents $\text{link-alarm-low} * \text{max_bandwidth} / 100$

%d3 represents the CTI Link ID

Setting the link-alarm-low option to a value of 0 (zero) will not create a link alarm low LMS message. The link bandwidth must drop below the set low alarm level in order to create the low watermark message. For a high watermark, the bandwidth recorded must exceed the set high alarm watermark to create the high watermark LMS message. The consequence of setting the low alarm watermark to 0 (zero) is that T-Server will only generate one high watermark LMS message since a low watermark LMS message is never created. Therefore, T-Server will remain in high watermark alarm state indefinitely and never generate a subsequent LMS high watermark message. If the link-alarm-low option is set to a value higher than the value of the link-alarm-high option, then the two values are swapped. However, the values are not swapped if either value is set to 0 (zero).

See the LinkLoad extension attribute for the link bandwidth feature.

HA Considerations

If the primary T-Server is at the high watermark prior to a switchover, its state is not transferred to the backup T-Server.

T-Server for CSTA Connector Supported Features No-Answer Supervision



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features No-Answer Supervision

No-Answer Supervision

This section describes T-Server's No-Answer Supervision feature.

Agent No-Answer Supervision

This feature provides the following functionality:

- If an agent does not answer a call within a specified timeout, T-Server can divert the call to a sequence of overflow destinations. Alternatively, you can configure T-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 T-Server to either log out the agent or set the agent to NotReady to prevent further calls from arriving.

Configuration Options

T-Server provides three configuration options for defining the behavior of the Agent No-Answer Supervision feature:

- `agent-no-answer-action`
- `agent-no-answer-overflow`
- `agent-no-answer-timeout`

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, T-Server can divert the call to a sequence of overflow destinations. Alternatively, you can configure T-Server to return calls automatically to the last distribution device.

T-Server provides two configuration options for defining the behavior of No-Answer Supervision with devices of type `Extension`:

- `extn-no-answer-overflow`
- `extn-no-answer-timeout`

Recall Scenarios

The configuration option `recall-no-answer-timeout` allows you to configure No-Answer Supervision for recall scenarios.

Configuration Options for Device-Specific Overrides

T-Server provides three configuration options with which you can configure device-specific overrides for individual devices. You set the values for these options on the `Annex` tab of the `TServer` section of the individual device in the Framework Configuration Layer. These are the options:

- `nas-private`
- `no-answer-overflow`
- `no-answer-timeout`
- `recall-no-answer-timeout`

Extension Attributes for Overrides for Individual Calls

For all of the No-Answer Supervision options, you can specify the corresponding `Extension` attribute in `TRequestRouteCall`, to override the configured value for individual calls. This method allows the no-answer behavior to be determined in a routing strategy. These are the three `Extension` attributes:

- `NO_ANSWER_ACTION`
- `NO_ANSWER_OVERFLOW`
- `NO_ANSWER_TIMEOUT`

Private Calls

You can also apply No-Answer Supervision to private calls, using the configuration option `nas-private`.

Reporting

The configuration option `nas-indication` allows you to configure reporting of extensions related to No-Answer Supervision for reporting scenarios.

T-Server for CSTA Connector Supported Features Private Services and Events



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Private Services and Events

Private Events

The following describes private services and events.

Service	Attribute	Value	Description
Hot-desking established	Event Number	1	Successful association between guest and host device has been established. T-Server will send corresponding EventPrivateInfo when de-association occurs.
	ThisDN	guest extension	
	OtherDN	host extension	Host associated with the guest extension
	AttributePrivateMsgID	1	CSTA Genesys specific private event
	Connector Function	privateEvent / guestAssociatedEvent	

Service	Attribute	Value	Description
Hot-desking cancelled	Event Number	2	Association between guest and host device has been terminated.
	ThisDN	guest extension	
	OtherDN	host extension	Host (if known) which was associated with the guest extension.
	AttributePrivateMsgID	2	CSTA Genesys specific private event.
	Connector Function	privateEvent / guestAssociatedEvent	

Service	Attribute	Value	Description
Set Account Code	Event Number	501	Account code set on the device.
	ThisDN	This device	
	ConnID	Connection ID of the call	
	PrivateID	0	
	AttributeExtensions	Should have a key, name defined by the <code>acccode-name</code> option.	

T-Server for CSTA Connector Supported Features Request Handling Enhancements



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Request Handling Enhancements

Request Handling Enhancements

T-Server introduces two major new enhancements to queue handling: request conflict resolution and a new device queue.

Requests submitted by different clients are treated no differently to requests submitted by the same client. For this reason, having multiple clients controlling the same device can result in unexpected behavior.



Note: While this configuration is supported, it should be recognised that there is no special handling for multiple clients.

Use the following T-Server configuration options to configure this feature:

- `device-rq-gap`
- `rq-conflict-check`

T-Server for CSTA Connector Supported Features Smart OtherDN Handling



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Smart OtherDN Handling

Smart OtherDN Handling

For T-Server clients that provide the `Agent ID` value as the `OtherDN` in requests to T-Server, T-Server can convert this `OtherDN` value using its knowledge of the association between the `Agent ID` and the DN to ensure the correct execution of the request by the switch. For switches expecting an `Agent ID` in the place of a DN for a particular operation, T-Server can convert the `OtherDN` value supplied by client to the `Agent ID` that the switch expects.

The following configuration options are provided to modify this feature's behaviour:

- `convert-otherdn`
- `dn-for-undesired-calls`

Extension key `ConvertOtherDN` is also provided to enable this feature to be applied on a call-by-call basis.

Supported Requests

The following table shows the requests that assume the use of the `OtherDN` value as a switch directory number, and can therefore support `SmartOtherDN` Handling.

Requests That Support Smart OtherDN Handling

TRequest	Meaning of OtherDN Attribute	AgentID-to-DN Conversion	Reserved DN Conversion
TMakeCall	Call destination	Yes	Yes
TMakePredictiveCall ^[1]	Call destination	Yes	Yes
TRedirectCall	New destination for call	Yes	Yes
TInitiateTransfer	Call destination	Yes	Yes
TMuteTransfer	Call destination	Yes	Yes
TSingleStepTransfer	New destination	No	No
TInitiateConference	Call destination	Yes	Yes
TSingleStepConference	New destination for call	Yes	Yes
TDeleteFromConference	Conference member to be deleted	Yes	Yes
TListenDisconnect	Request target	Yes	Yes
TListenReconnect	Request target	Yes	Yes
TCallSetForward	Request target	Yes	Yes
TGetAccessNumber ^[2]	DN for which Access Number is requested	No	No
TSetCallAttributes ^[3]	Not specified	No	No
TReserveAgentAndGetAccessNumber ^[3]	DN for which Access Number is requested	No	No
TMonitorNextCall	Agent DN to be monitored	Yes	Not applicable
TCancelMonitoring	Agent DN that was monitored	Yes	Not applicable
TRouteCall ^[3]	New destination for call		
* RouteTypeUnknown		Yes	Yes
* RouteTypeDefault		Yes	Yes
* RouteTypeOverwriteDNIS		Yes	Yes
* RouteTypeAgentID		Yes	Yes

[1] TMakePredictiveCall assumes that the directory number should be outside the switch; however, this request could also support Smart OtherDN Handling.

[2] T-Server cannot intercept these requests.

[3] Only the listed route types are applicable for OtherDN conversion.

T-Server for CSTA Connector Supported Features T-Library Functionality



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features T-Library Functionality

T-Library Functionality

The following table presents T-Library functionality supported in T-Server for Connector. The table entries use these notations:

- N—Not supported
- Y—Supported
- I—Supported, but reserved for Genesys Engineering
- E—Event only supported

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 Events and Models Reference Manual and the Voice Platform SDK 8 .NET (or Java) API Reference*.

This table reflects only that switch functionality used by Genesys software and might not include the complete set of events offered by the switch.

Certain requests in the table are reserved for Genesys Engineering and are listed here merely for completeness of information.

Notes describing specific functionalities appear at the end of the table.

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 ^[1]		EventRegistered	Y
TUnregisterAddress		EventUnregistered	Y
Call-Handling Requests			
TMakeCall ^[2]	Regular	EventDialing	Y
	DirectAgent		N

	SupervisorAssist		N
	Priority		N
	DirectPriority		N
TAnswerCall ^[3]		EventEstablished	Y
TReleaseCall		EventReleased	Y
TClearCall		EventReleased	Y
THoldCall		EventHeld	Y
TRetrieveCall		EventRetrieved	Y
TRedirectCall		EventReleased	Y
TMakePredictiveCall		EventDialing*, EventQueued	Y

Transfer/Conference Requests

TInitiateTransfer		EventHeld, EventDialing*	Y
TCompleteTransfer		EventReleased*, EventReleased	Y
TInitiateConference		EventHeld, EventDialing*	Y
TCompleteConference ^[4]		EventReleased*, EventRetrieved, EventPartyAdded	Y
TDeleteFromConference		EventPartyDeleted*, EventReleased	Y
TReconnectCall		EventReleased, EventRetrieved*	Y
TAlternateCall		EventHeld*, EventRetrieved	Y
TMergeCalls	ForTransfer	EventReleased*, EventPartyChanged	N
	ForConference	EventReleased*, EventRetrieved, EventPartyChanged, EventPartyAdded	N
TMuteTransfer		EventHeld, EventDialing*, EventReleased, EventReleased	Y
TSingleStepTransfer		EventReleased*, EventPartyChanged	Y
TSingleStepConference		EventRinging*, EventEstablished	N

Call-Routing Requests

TRouteCall	Unknown	EventRouteUsed	Y
	Default		Y
	Label		N
	OverwriteDNIS		Y
	DDD		Y
	IDDD		Y
	Direct		N
	Reject		Y
	Announcement		N
	PostFeature		N
	DirectAgent		N
	Priority		N
	DirectPriority		N
	AgentID		N

	CallDisconnect		Y
Call-Treatment Request			
TApplyTreatment	Unknown	(EventTreatmentApplied + EventTreatmentEnd)/EventTreatmentNotApplied	N
	IVR		N
	Music		Y
	RingBack		Y
	Silence		Y
	Busy		Y
	CollectDigits		Y
	PlayAnnouncement		Y
	PlayAnnouncementAndDigits		Y
	VerifyDigits		N
	RecordUserAnnouncement		N
	DeleteUserAnnouncement		N
	CancelCall		Y
	PlayApplication		N
	SetDefaultRoute		N
	TextToSpeech		N
	TextToSpeechAndDigits		N
	FastBusy		N
	RAN		N
TGiveMusicTreatment		EventTreatmentApplied	Y
TGiveRingBackTreatment		EventTreatmentApplied	Y
TGiveSilenceTreatment		EventTreatmentApplied	Y
DTMF (Dual-Tone Multifrequency) Requests			
TCollectDigits		EventDigitsCollected	Y
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	WorkModeUnknown	EventAgentLogin	Y
	ManualIn ^[5]		Y
	AutoIn ^[6]		Y

	AfterCallWork ^[7]		Y
	AuxWork		Y
	WalkAway		Y
	ReturnBack		Y
	NoCallDisconnect		Y
TAgentLogout		EventAgentLogout	Y
TAgentSetIdleReason		EventAgentIdleReasonSet	N
TAgentSetReady ^[8]		EventAgentReady	Y
TAgentSetNotReadyh	WorkModeUnknown	EventAgentNotReady	Y
	ManualIn		Y
	AutoIn		Y
	AfterCallWork		Y
	AuxWork		Y
	WalkAway		Y
	ReturnBack		Y
	NoCallDisconnect		Y
TMonitorNextCall	OneCall	EventMonitoringNextCall	Y
	AllCalls		Y
TCancelMonitoring		EventMonitoringCanceled	Y
TCallSetForward	None	EventForwardSet	Y
	Unconditional		Y
	OnBusy		Y
	OnNoAnswer		Y
	OnBusyAndNoAnswer		N
	SendAllCalls		N
TCallCancelForward	None	EventForwardCancel	Y
	Unconditional		Y
	OnBusy		Y
	OnNoAnswer		Y
	OnBusyAndNoAnswer		N
	SendAllCalls		N
TSetMuteOff		EventMuteOff	N
TSetMuteOn		EventMuteOn	N
TListenDisconnect		EventListenDisconnected	N
TListenReconnect		EventListenReconnected	N
TSetDNDOOn		EventDNDOOn	Y
TSetDNDOOff		EventDNDOOff	Y
TSetMessageWaitingOn		EventMessageWaitingOn	N
TSetMessageWaitingOff		EventMessageWaitingOff	N

Query Requests

TQuerySwitch	DateTime	EventSwitchInfo	N
	ClassifierStat		N
TQueryCall	PartiesQuery	EventPartyInfo	Y
	StatusQuery		N
TQueryAddress	AddressStatus	EventAddressInfo	Y
	MessageWaitingStatus		N
	AssociationStatus		N
	CallForwardingStatus		Y
	AgentStatus		Y
	NumberOfAgentsInQueue ^[9]		Y
	NumberOfAvailableAgentsInQueue ^[10]		Y
	NumberOfCallsInQueue ^[11]		Y
	AddressType		Y
	CallsQuery		Y
	SendAllCallsStatus		N
	QueueLoginAudit		Y
	NumberOfIdleTrunks		N
	NumberOfTrunksInUse		N
	DatabaseValue		N
TQueryLocation	DNSStatus		Y
	QueueStatus		Y
	AllLocations	EventLocationInfo	I
	LocationData		I
	MonitorLocation		I
	CancelMonitorLocation		I
	MonitorAllLocations		I
	CancelMonitorAllLocations		I
TQueryServer	LocationMonitorCanceled		I
	AllLocationsMonitorCanceled		I
	EventServerInfo		Y

User-Data Requests

TAttachUserData	EventAttachedDataChanged	Y
TUpdateUserData	EventAttachedDataChanged	Y
TDeleteUserData	EventAttachedDataChanged	Y
TDeleteAllUserData	EventAttachedDataChanged	Y

ISCC (Inter Server Call Control) Requests

TGetAccessNumber	EventAnswerAccessNumber	Y
TCancelReqGetAccessNumber	EventReqGetAccessNumberCanceled	Y

Special Requests

TReserveAgent	EventAgentReserved	I
TSendEvent	EventACK	I
TSendEventEx	EventACK	I
TSetCallAttributes	EventCallInfoChanged	Y
TSendUserEvent	EventACK	Y
TPrivateService	EventPrivateInfo	Y

Network Requests

TNetworkConsult	EventNetworkCallStatus	N
TNetworkAlternate	EventNetworkCallStatus	N
TNetworkTransfer	EventNetworkCallStatus	N
TNetworkMerge	EventNetworkCallStatus	N
TNetworkReconnect	EventNetworkCallStatus	N
TNetworkSingleStepTransfer	EventNetworkCallStatus	N
TNetworkPrivateService	EventNetworkPrivateInfo	N

- [1] Every configured device is monitored as soon as the connection with the switch is established. Extensions are monitored using MonitorDeviceCalls (telephony events) and MonitorACDFeatures (agent states) functionality. Routing Points are monitored using MonitorDevice and ACD Queues using MonitorQueue.
- [2] Functions on digital phones without human intervention.
- [3] This function is not available for analog phones (extension type 2).
- [4] Only three-party conferences are supported.
- [5] If a queue is configured with the parameter WK (Work mode), the agent state is NotReady after login.
- [6] If a queue is configured with the parameter WK (Work mode) the agent state is Ready after login. However, you can set the agent state to NotReady.
- [7] After an ACD call, an agent is automatically put into the AfterCallWork state.
- [8] Functions only if the queue is configured with the WK (Work mode) parameter.
- [9] Only on Agent Group, not queue.
- [10] Only on Agent Group, not queue. Ready/Not Ready only.
- [11] Only on queue, not Agent Group.

T-Server for CSTA Connector Supported Features User Data Keys



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features User Data Keys

User Data Keys

T-Server supports the use of the user data keys in the following table:

User Data Keys

Extension		Used In	Description
Key	Type		
ACCOUNT_CODE	string integer	TAttachUserData TUpdateUserData EventUserDataChanged	Alternative to key ACCOUNT_CODE_<N> . The name of this extension is defined by the configuration option <code>accode-name</code> .
LegalGuardTime	integer	Call-related requests	Specifies the amount of emulated Legal Guard time allocated to the agent at the end of a business call.
WrapUpTime	integer	Call-related requests	Specifies the amount of emulated wrap-up time allocated to all agents at the end of the business call. This value is effective for the duration of this agent's login session.

T-Server for CSTA Connector Supported Features Using the Extensions Attribute



Home > T-Server for CSTA Connector> T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Supported Features > T-Server for CSTA Connector Supported Features Using the Extensions Attribute

Using the Extensions Attribute

T-Server supports the use of the `Extensions` attribute as documented in the *Genesys Events and Models Reference Manual* and the *Voice Platform SDK 8 .NET (or Java) API Reference*.

Additionally, the extensions described in the following table are also supported.


Use of the Extensions Attribute

Extension		Used In	Description
Key	Type		
ACCOUNT_CODE	string integer	TPrivateService EventPrivateInfo	The requested or reported account code. The key name of this extension is defined by the configuration option <code>acccode-name</code> .
ReasonCode	string integer	TAgentNotReady EventAgentNotReady	When an account code is used as a Walk-Away code, this extension specifies the requested or reported account code while the agent is in <code>NotReady</code> state. The default method of reporting Not Ready activation information.
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: <ul style="list-style-type: none"> <code>no-answer-timeout</code> <code>agent-no-answer-timeout</code> <code>extn-no-answer-timeout</code>
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: <ul style="list-style-type: none"> <code>no-answer-action</code> <code>agent-no-answer-action</code>
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: <ul style="list-style-type: none"> <code>no-answer-overflow</code> <code>agent-no-answer-overflow</code> <code>extn-no-answer-overflow</code>
SUPERVISED_ROUTE	string	TRouteCall	Overrides the value in configuration option <code>supervised-route-timeout</code> for individual calls.
ConvertOtherDN	string integer	See Smart OtherDN Handling.	A value of 0 disables all conversions for the call. A value of 1 forces the relevant conversion for the call.
EmulateLogin	string	TAgentLogin	With a value of <code>yes</code> , T-Server performs an emulated login. With a value of <code>no</code> , T-Server attempt a real login.
	string	EventAgentLogin EventAddressInfo EventRegistered	A value of <code>yes</code> indicates that the T-Server has performed an emulated login.
WrapUpTime	integer	TAgentLogin	Specifies the amount of emulated wrap-up time (in seconds) allocated to this agent at the end of a business call. This value is effective for the duration of this login's agent session. It can be overridden by the value in the <code>WrapUpTime</code> extension in <code>TAgentNotReady</code> .
	integer	TAgentNotReady	Specifies the amount of emulated wrap-up time (in seconds) allocated to this agent at the end of a business call. This value is effective only for the lifespan of this request.
BusinessCall	integer	Call-related events	Specifies the business type of a call. Valid values are: <code>0/private</code> "Private call", <code>1/business</code> "Business call", <code>2/work</code> "Work-related call".
BusinessCallType	string integer	TMakeCall TInitiateTransfer TMuteTransfer TInitiateConference TMakePredictiveCall TAnswerCall	Specifies the call business type to be used by TServer for the new call or the answering party. Valid values are: <code>0/private</code> "Private call", <code>1/business</code> "Business call", <code>2/work</code> "Work-related call"

AgentLogoutOnUnregister	string	TAgentLogin TRegisterAddress	Specifies whether T-Server performs an automatic logout of an agent whenever their client application unregisters its DN from T-Server. Valid values are: <code>true</code> T-Server will logout emulated and native agents on unregister, <code>false</code> T-Server will not logout emulated or native agents on unregister, <code>emu-only</code> T-Server will logout emulated agents only on unregister.
AssociateClientWithLogin	boolean	TAgentLogin TRegisterAddress	Specifies whether the client should be associated with the agent session.
	boolean	EventAgentLogin EventRegistered EventPrivateInfo	Specifies that the client has been associated with the agent session.
AgentEmuLoginOnCall	boolean	TAgentLogin TAgentLogout	Specifies whether T-Server allows an emulated agent login or logout from a device where there is a call in progress.
LegalGuardTime	integer	TAgentLogin	Specifies the amount of emulated legal guard time allocated to an agent at the end of a business call.
SyncEmuACW	integer	TAgentLogin	Specifies whether T-Server synchronizes emulated ACW and/or legal guard with the switch for native agents.
ReleasingParty	string	EventReleased EventAbandoned	Identifies which party was the initiator of the call release. Possible values are: 1 Local, 2 Remote, 3 Unknown
LinkLoad	string	EventRouteRequest	A value of <code>1High</code> indicates that T-Server is in a high watermark condition. The feature is disabled if the <code>use-link-bandwidth</code> option is set to 0 (zero). Possible values are: 0 OK, 1 High
Association	string	TRegisterAddress	Defines the association of a DN that is not configured in Configuration Manager. T-Server uses the value <code>none</code> (empty string) when the extension is not provided.
SwitchSpecificType	string integer	TRegisterAddress	Defines the switch-specific type of a DN that is not configured in Configuration Manager.
sdn-licenses-in-use	integer	EventServerInfo	Specifies how many SDN licenses are currently in use.
sdn-licenses-available	integer		Specifies how many SDN licenses are currently available.
PICKUP	string	TMakeCall	The value of the <code>PICKUP</code> extension can be anything. If the Extension is present in <code>TMakeCall</code> T-Server will pick up the call ringing on the device specified by <code>OtherDN</code> in the request.
ASSIST	string	TInitiateConferece TInitiateTransfer	The value of the <code>ASSIST</code> extension can be anything. An agent can add a Supervisor to a call in order to receive assistance by passing the <code>ASSIST</code> extension in either <code>TInitiateConference</code> or <code>TInitiateTransfer</code> and specifying the Supervisor's device as <code>OtherDN</code> in the request.
INTRUDE	string	TSingleStepConference	The value of <code>INTRUDE</code> extension can be anything. A device an intrude on a call on another device by passing the <code>INTRUDE</code> extension in <code>TSingleStepConference</code> and specifying in <code>OtherDN</code> the device with the call to be intruded on.
OrigSV-n, NumOfOrigSVs	string	EventPartyChanged, EventPartyAdded	Indicates the supervisors monitoring the main call in a transfer/conference scenario. The <code>NumOfOrigSVs</code> extension indicates the number of supervisors listening to the main call.
ConsultSV-n, NumOfConsultSVs	string	EventPartyChanged, EventPartyAdded	Indicates the supervisors monitoring the consultation call in a transfer/conference scenario. The <code>NumOfConsultSVs</code> extension indicates the number of supervisors listening to the consultation call.

ReleaseController	string, integer	TReleaseCall (CSTA Clear Connection)	Allows to override the settings of the configuration option <code>enable-controller-release</code> . 0—Corresponds to the option value <code>false</code> . A positive integer—Corresponds to the option value <code>true</code> .
-------------------	-----------------	--------------------------------------	--

Common Configuration Options common Section

 Home > Common Configuration Options > Common Configuration Options common Section

common Section

This section must be called `common`.

enable-async-dns

- Default Value: `off`
- Valid Values:

<code>off</code>	Disables asynchronous processing of DNS requests.
<code>on</code>	Enables asynchronous processing of DNS requests.

- Changes Take Effect: Immediately
- Enables the asynchronous processing of DNS requests such as, for example, host-name resolution.

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.

Common Configuration Options log Section



Home > Common Configuration Options > Common Configuration Options log Section

log Section

This section must be called `log`.

verbose

- Default Value: `all`
- Valid Values:

<code>all</code>	All log events (that is, log events of the Standard, Trace, Interaction, and Debug levels) are generated.
<code>debug</code>	The same as <code>all</code> .
<code>trace</code>	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.
<code>interaction</code>	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.
<code>standard</code>	Log events of the Standard level are generated, but log events of the Interaction, Trace, and Debug levels are not generated.
<code>none</code>	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.

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

buffering

- Default Value: `true`
- Valid Values:

<code>true</code>	Enables buffering.
<code>false</code>	Disables buffering.

- Changes Take Effect: Immediately
- Turns on/off operating system file buffering. The option is applicable only to the `stderr` and `stdout` output. Setting this option to `true` increases the output performance.

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

segment

- Default Value: `false`
- Valid Values:

<code>false</code>	No segmentation is allowed.
<code><number> KB</code> or <code><number></code>	Sets the maximum segment size, in kilobytes. The minimum segment size is 100 KB.
<code><number> MB</code>	Sets the maximum segment size, in megabytes.
<code><number> hr</code>	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:

<code>false</code>	No expiration; all generated segments are stored.
<code><number> file</code> or <code><number></code>	Sets the maximum number of log files to store. Specify a number from 1—1000.
<code><number> day</code>	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.

Warning: If an option's value is set incorrectly"ut of the range of valid values"it will be automatically reset to 10 .

keep-startup-file

- Default Value: `false`
- Valid Values:

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

Warning: 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

Warning: 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:

<code>time</code>	The time string is formatted according to the <code>HH:MM:SS.sss</code> (hours, minutes, seconds, and milliseconds) format.
<code>locale</code>	The time string is formatted according to the system's locale.
<code>ISO8601</code>	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:

<code>true</code>	Attaches extended attributes, if any exist, to a log event sent to log output.
<code>false</code>	Does not attach extended attributes to a log event sent to log output.

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

check-point

- Default Value: `1`
- Valid Values: `0—24`
- Changes Take Effect: Immediately
- Specifies, in hours, how often the application generates a `check point` log event, to divide the log into sections of equal time. By default, the application generates this log event every hour. Setting the option to `0` prevents the generation of `check-point` events.

memory

- Default Value: No default value
- Valid Values: `<string>` (memory file name)
- Changes Take Effect: Immediately
- Specifies the name of the file to which the application regularly prints a snapshot of the memory output, if it is configured to do this (see Log Output Options). 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.

Warning: 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:

<code><number> KB</code> or <code><number></code>	The size of the memory output, in kilobytes. The minimum value is 128 KB.
<code><number> MB</code>	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.

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:

<code>true</code>	The log of the level specified by Log Output Options is sent to the specified output.
<code>false</code>	The log of the level specified by Log Output Options and higher levels is sent to the specified output.

- Changes Take Effect: Immediately
- Specifies whether the application uses 6.x output logic. For example, you configure the following options in the `log` section for a 6.x application and for a 7.x application:
 - `[log]`
 - `verbose = all`
 - `debug = file1`
 - `standard = file2`
- The log file content of a 6.x application is as follows:
 - `file1` contains Debug messages only.
 - `file2` contains Standard messages only.
- The log file content of a 7.x application is as follows:
 - `file1` contains Debug, Trace, Interaction, and Standard messages.
 - `file2` contains Standard messages only.
- If you set `compatible-output-priority` to `true` in the 7.x application, its log file content will be the same as for the 6.x application.

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

Log Output Options

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

You can use:

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

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

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.

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

all

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

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

- Changes Take Effect: Immediately
- Specifies the outputs to which an application sends all log events. The log output types must be separated by a comma when more than one output is configured. For example:
- all = stdout, logfile

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

alarm

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

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

- **Changes Take Effect:** Immediately
- Specifies the outputs to which an application sends the log events of the `Alarm` level. The log output types must be separated by a comma when more than one output is configured. For example:
- `standard = stderr, network`

standard

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

<code>stdout</code>	Log events are sent to the Standard output (<code>stdout</code>).
<code>stderr</code>	Log events are sent to the Standard error output (<code>stderr</code>).
<code>network</code>	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
<code>memory</code>	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
<code>[filename]</code>	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):**

<code>stdout</code>	Log events are sent to the Standard output (<code>stdout</code>).
<code>stderr</code>	Log events are sent to the Standard error output (<code>stderr</code>).
<code>network</code>	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
<code>memory</code>	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
<code>[filename]</code>	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"or example:
- `debug = stderr, /usr/local/genesys/logfile`

Warning: 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.

Warning: 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.

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

Debug Log Options

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

x-conn-debug-open

- Default Value: 0
- Valid Values:

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

- Changes Take Effect: After restart
- Generates `Debug` log records about "open connection" operations of the application.

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

x-conn-debug-select

- Default Value: 0
- Valid Values:

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

- Changes Take Effect: After restart
- Generates `Debug` log records about "socket select" operations of the application.

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

x-conn-debug-timers

- Default Value: 0
- Valid Values:

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

- Changes Take Effect: After restart
- Generates `Debug` log records about the timer creation and deletion operations of the application.

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

x-conn-debug-write

- Default Value: 0
- Valid Values:

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

- Changes Take Effect: After restart
- Generates `Debug` log records about "write" operations of the application.

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

x-conn-debug-security

- Default Value: 0
- Valid Values:

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

- Changes Take Effect: After restart
- Generates `Debug` log records about security-related operations, such as Transport Layer Security and security certificates.

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

x-conn-debug-api

- Default Value: 0
- Valid Values:

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

- Changes Take Effect: After restart
- Generates `Debug` log records about connection library function calls.

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

x-conn-debug-dns

- Default Value: 0
- Valid Values:

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

- Changes Take Effect: After restart
- Generates `Debug` log records about DNS operations.

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

x-conn-debug-all

- Default Value: 0
- Valid Values:

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

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

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

Common Configuration Options log-extended Section



Home > Common Configuration Options > Common Configuration Options log-extended Section

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

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

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

Warning: Use caution when making these changes in a production environment. Warning: 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. Warning: Likewise, changing the log level to a lower priority may cause the log event to be not logged at all, or to be not logged to specific outputs, thereby losing important information. The same applies to any alarms associated with that log event.

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

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

Example

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

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

Before the log levels of the log are changed:

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

Extended log configuration section:

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

After the log levels are changed:

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

level-reassign-disable

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

Common Configuration Options sml Section



Home > Common Configuration Options > Common Configuration Options sml Section

sml Section

This section must be called `sml`.

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

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

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

heartbeat-period

- Default Value: `None`
- Valid Values:

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

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

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

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

heartbeat-period-thread-class-<n>

- Default Value: None
- Valid Values:

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

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

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

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

hangup-restart

- Default Value: `true`
- Valid Values: `true`, `false`
- Changes Take Effect: Immediately
- If set to `true` (the default), specifies that LCA is to restart the unresponsive application immediately, without any further interaction from Solution Control Server.

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

suspending-wait-timeout

- Default Value: 10
- Valid Values: 5-600
- Changes Take Effect: Immediately
- Specifies a timeout (in seconds) after the Stop Graceful command is issued to an application during which the status of the application should change to `Suspending` if the application supports graceful shutdown. If the status of the application does not change to `Suspending` before the timeout expires, it is assumed that the application does not support graceful shutdown, and it is stopped ungracefully.
- Use this option if you are unsure whether the Application supports graceful shutdown.

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

T-Server Common Configuration Options License Section



Home > T-Server Common Configuration Options > T-Server Common Configuration Options License Section

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.

Warning: T-Server also supports the `license-file` option described in the *Genesys Licensing Guide*. Warning: 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 DN 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.

Warning: For Network T-Servers, Genesys recommends setting this option to 0. Warning: Be sure to configure in the Configuration Database all the DN's that agents use (Extensions and ACD Positions) and that T-Server should control.

License Checkout

The following table shows how to determine the number of seat-related DN licenses that T-Server attempts to check out.

Options Settings ^[1]		License Checkout ^[2]
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

References

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 DN's
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 DN's
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		

References

T-Server Common Configuration Options

TServer Section

 Home > T-Server Common Configuration Options > T-Server Common Configuration Options TServer Section

TServer Section

The `TServer` section contains the configuration options that are used to support the core features common to all T-Servers.

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: `true`
- 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.

background-timeout

- Default Value: `60 msec`
- Valid Values: See Timeout Value Format.
- 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 only allows a client to register if the client provides the correct name and password of a T-Server Tenant. If the client provides the Tenant name concatenated with a slash (/) and the Tenant password for the Tenant to which T-Server belongs as the value of `AttributeApplicationPassword` in the `TRegisterClient` request, 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.

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.

Warning: 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.

Warning: Do not configure the `customer-id` option for single-tenant environments.

dn-scope

- Default Value: undefined
- Valid Values: undefined, switch, office, tenant
- Changes Take Effect: Immediately
- Related Feature: Switch Partitioning
- Specifies whether DNs associated with the `Switch`, `Switching Office`, or `Tenant` objects will be considered in the T-Server monitoring scope, enabling T-Server to report calls to or from those DNs as internal.

With a value of `tenant`, all DNs associated with the switches that are within the `Tenant` will be in the T-Server monitoring scope. With a value of `office`, all DNs associated with the switches that are within the `Switching Office` will be in the T-Server monitoring scope. With a value of `switch`, all DNs associated with the `Switch` will be in the T-Server monitoring scope.

- With a value of `undefined` (the default), pre-8.x T-Server behavior applies and the switch partitioning is not turned on.

Warning: Setting the option to a value of `office` or `tenant`, which requires T-Server to monitor a large set of configuration data, may negatively affect T-Server performance.

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 AttributePassword in TEvents.
+/-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 "sing a space-, comma- or semicolon-separated list"he 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.

Warning: 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`.)

server-id

- Default Value: An integer equal to the value `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.

Warning: 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.

Warning: When T-Server works in mixed 8.x/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.

T-Server Common Configuration Options Translation Rules Section



Home > T-Server Common Configuration Options > T-Server Common Configuration Options Translation Rules Section

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. See Configuring Number Translation 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
```

T-Server Common Configuration Options

agent-reservation Section



Home > T-Server Common Configuration Options > T-Server Common Configuration Options agent-reservation Section

agent-reservation Section

The `agent-reservation` section contains the configuration options that are used to customize the T-Server Agent Reservation feature. See the Agent Reservation section for details on this feature.

Warning: 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.

collect-lower-priority-requests

- Default Value: `true`
- Valid Values: `true`, `false`
- Changes Take Effect: Immediately
- Specifies whether an agent reservation request is collected, depending on its priority during the time interval specified by the `request-collection-time` configuration option. When set to `false`, during the `request-collection-time` interval T-Server collects reservation requests of the highest priority only, rejecting newly submitted requests that have a lower priority or rejecting all previously submitted requests if a request with a higher priority arrives. When set to `true` (the default), agent reservation requests are collected as they were in pre-8.x releases.

reject-subsequent-request

- Default Value: `true`
- Valid Values:

<code>true</code>	T-Server rejects subsequent requests.
<code>false</code>	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.

Warning: 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.
- 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.
- Changes Take Effect: Immediately
- Specifies the default interval for which a an Agent DN is reserved. During this interval, the agent cannot be reserved again.

T-Server Common Configuration Options backup-sync Section



Home > T-Server Common Configuration Options > T-Server Common Configuration Options backup-sync Section

backup-sync Section

The backup-synchronization section contains the configuration options that are used to support a high-availability (`hot standby redundancy` type) configuration.

Warning: 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:

<code>off, false, no</code>	No trace (default).
<code>local, on, true, yes</code>	Trace on this T-Server side only.
<code>remote</code>	Trace on the redundant T-Server side only.
<code>full, both</code>	Full trace (on both sides).

- Changes Take Effect: Immediately
- Specifies whether `addp` messages are traced in a log file, to what level the trace is performed, and in which direction. This option applies only if the `protocol` option is set to `addp`.

protocol

- Default Value: `default`
- Valid Values:

<code>default</code>	The feature is not active.
<code>addp</code>	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.
 - 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).
-

T-Server Common Configuration Options

call-cleanup Section



Home > T-Server Common Configuration Options > T-Server Common Configuration Options call-cleanup Section

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 8.0 Management Layer User's Guide.

cleanup-idle-tout

- Default Value: 0
- Valid Values: See Timeout Value Format.
- 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.

Warning: If the call-cleanup functionality is enabled in T-Server for Avaya Communication Manager, the UCID (Universal Call ID) feature must be enabled on the switch as well. This allows the UCID to be generated and passed to T-Server.

notify-idle-tout

- Default Value: 0
- Valid Values: See Timeout Value Format.
- 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.
- 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 check.

Warning: 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.

T-Server Common Configuration Options extrouter Section



Home > T-Server Common Configuration Options > T-Server Common Configuration Options extrouter Section

extrouter Section

The `extrouter` 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 of the document are grouped with related options that support the same functionality, as follows:

- ISCC Transaction Options
- Transfer Connect Service Options
- ISCC/COF Options
- Event Propagation Options
- Number Translation Option
- GVP Integration Option

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.

Warning: 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:

<code>true</code>	ISCC does not process (match) an inbound call that has already been processed (matched).
<code>false</code>	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.

Warning: 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: 5sec
- Valid Values: See Timeout Value Format.
- 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: current
- Valid Values:

active	The values of <code>UserData</code> and <code>ConnID</code> attributes are taken from the consultation call.
original	The values of <code>UserData</code> and <code>ConnID</code> attributes are taken from the original call.
active-data-original-call	The value of the <code>UserData</code> attribute is taken from the consultation call and the value of <code>ConnID</code> attribute is taken from the original call.
current	<p>If the value of <code>current</code> is specified, the following occurs:</p> <ul style="list-style-type: none"> • Before the transfer or conference is completed, the <code>UserData</code> and <code>ConnID</code> attributes are taken from the consultation call. • After the transfer or conference is completed, <code>EventPartyChanged</code> is generated, and the <code>UserData</code> and <code>ConnID</code> 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.

Warning: 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-uid, reroute, direct-callid, direct-uid, direct-network-callid, direct-notoken, direct-digits, direct-ani, dn timer pool, pullback
Valid Values: route, route-uid, reroute, direct-callid, direct-uid, direct-network-callid, direct-notoken, direct-digits, direct-ani, dn timer 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 type provide calls with different sets of information parameters, some values might not work with your T-Server. The Multi-Site Support section also provides detailed descriptions of all transaction types.

Warning: For compatibility with the previous T-Server releases, you can use the `direct` value for this option. This is an alias for `direct-callid`. Warning: 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`.

Warning: This option is used only for requests with route types `route`, `route-uu`, `direct-callid`, `direct-network-callid`, `direct-uu`, `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.

Warning: 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.
- **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.
- 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.
- 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:

<code>home</code>	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.
<code>circular</code>	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 the `Resource Type` set to `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.
- 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.

Warning: 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 <code>UserData</code> attribute of the request.

- Changes Take Effect: Immediately
- Specifies whether the Transfer Connect Service (TCS) feature is used.

Warning: 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.
- 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.
- 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.
- 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.
- 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`.

Warning: This option applies only to T-Server for Nortel Communication Server 2000/2100.

default-network-call-id-matching

- Default Value: No default value
- Valid Values: See the "T-Server-Specific Configuration Options" chapter for an option description for your T-Server
- Changes Take Effect: Immediately
- When a value for this option is specified, T-Server uses the `NetworkCallID` attribute for the ISCC/COF call matching.
- To activate this feature, the `cof-feature` option must be set to `true`.

Warning: SIP Server and several T-Servers support the `NetworkCallID` attribute for the ISCC/COF call matching in a way that requires setting this option to a specific value. For information about the option value that is specific for your T-Server, see the "T-Server-Specific Configuration Options" chapter of your *T-Server Deployment Guide*.

Event Propagation Options

compound-dn-representation

- Default Value: `true`
- Valid Values: `true`, `false`
- Changes Take Effect: Immediately
- Specifies which format T-Server uses to represent a DN when reporting an `OtherDN` or `ThirdPartyDN` attribute in event propagation messages.
- When set to `true`, the `<switch>:DN` (compound) format is used. This option value supports backward compatibility for pre-8.x T-Server ISCC/EPP functionality and is provided for multi-site deployments where the same DNs are configured under several switches.

When set to `false`, the DN (non-compound) format is used. This option value ensures more transparent reporting of `OtherDN` or `ThirdPartyDN` attributes and is recommended for all single-site deployments, as well as for multi-site deployments that do not have the same DNs configured under several switches. This option applies only if the `event-propagation` option is set to `list`.

Warning: Local DNs are always represented in the non-compound (DN) form.

epp-tout

- Default Value: 0
- Valid Values: See Timeout Value Format.
- Changes Take Effect: Immediately
- Specifies the time interval during which T-Server attempts to resolve race conditions that may occur in deployments that use switch partitioning or intelligent trunks. This option applies only if the `event-propagation` option is set to `list`.

Warning: If the time interval is not long enough to account for possible network switching delays, T-Server may produce duplicated events, such as events that are propagated by the ISCC and generated locally.

event-propagation

- Default Value: `list`
- Valid Values:

<code>list</code>	Changes in user data and party events are propagated to remote locations through call distribution topology.
<code>off</code>	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.

propagated-call-type

- Default Value: `false`
- Valid Values: `true`, `false`
- Changes Take Effect: Immediately
- Related Feature: Switch Partitioning

Determines what T-Server reports as the value of the `CallType` attribute in events related to calls that have been synchronized with another site via ISCC, as follows:

- When set to `false`, T-Server reports in events related to calls that have been synchronized with another site via ISCC the same value for the `CallType` attribute as it did in pre-8.0 releases and adds the new `PropagatedCallType` attribute with the value of the `CallType` attribute at the origination site. This provides backward compatibility with existing T-Server clients.
- When set to `true`, T-Server reports in events related to calls that have been synchronized with another site via ISCC the same value for the `CallType` attribute as at the origination site, and adds the new `LocalCallType` attribute with the same value as `CallType` in pre-8.0 releases.

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.

GVP Integration Option**handle-vsp**

- Default Value: `no`
- Valid Values:

<code>requests</code>	ISCC will process and adjust requests related to this DN and containing a <code>Location</code> attribute before submitting them to the service provider.
<code>events</code>	ISCC will process and adjust events received from the service provider and containing a <code>Location</code> attribute before distributing them to T-Server clients.
<code>all</code>	ISCC will process and adjust both events and requests.
<code>no</code>	No ISCC processing of such requests and events takes place.

- Changes Take Effect: Immediately
- Specifies the way ISCC handles events from, and requests to, an external service provider registered for a DN using the `AddressType` attribute set to `VSP`.

T-Server for CSTA Connector Agent-Specific Override Options



Home > T-Server for CSTA Connector > T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Options > T-Server for CSTA Connector Agent-Specific Override Options

Agent-Specific Override Options

You can only set the configuration options described in this section in the `TServer` section of the `Annex` tab of the relevant configuration object in the Configuration Layer for individual agents in the Framework Configuration Layer, therefore they need to be listed separately. You cannot define them in the main `TServer` configuration section.

TServer Section

monitor

Default Value: `false`

Valid Values: `true`, `false`

Changes Take Place: Immediately

Specifies whether the switch should monitor the agent. The value overrides the global T-Server option `monitor-agents`.

no-answer-action

Default Value: `none`

Valid Values:

`none`—T-Server takes no action on agents when business calls are not answered.

`notready`—T-Server sets agents `NotReady` when business calls are not answered.

`logout`—T-Server automatically logs out agents when business calls are not answered.

Changes Take Effect: Immediately

Related Feature: No-Answer Supervision

The value of the option `no-answer-action` overrides any value of the option `agent-no-answer-action` set at the Application level.

This option is defined in a section called `TServer` on the `Annex` tab of any `Agent Login` object in the Configuration Layer. If an emulated or real PBX agent receives a T-Server business call and the agent fails to answer the call within the time defined in option `agent-no-answer-timeout`, the `no-answer-action` option determines the action T-Server performs on this agent.



Note: If a call is abandoned before either `agent-no-answer-timeout` or `no-answer-timeout` or `supervised-route-timeout` expires (depending on which timer is applicable), T-Server performs no action on this agent.

T-Server for CSTA Connector DN-Level Options



Home > T-Server for CSTA Connector > T-Server for CSTA Connector Deployment Guide > T-Server for CSTA Connector Options > T-Server for CSTA Connector DN-Level Options

DN-Level Options

You can only set the configuration options described in this section in the `TServer` section of the `Annex` tab of the relevant configuration object in the Configuration Layer. You cannot define them in the main `TServer` configuration section.

TServer Section

bsns-call-dev-types

Default Values: `+acdq +rp +rpq +xrp`

Valid Values: A set of space separated flags.

`+/-acdq`—Turns on/off the classification of the call type as business on an ACD Queue.

`+/-rp`—Turns on/off the classification of the call type as business on a Routing Point.

`+/-rpq`—Turns on/off the classification of the call type as business on a Routing Queue.

`+/-xrp`—Turns on/off the classification of the call type as business on an External Routing Point.

Changes Take Effect: Immediately

Related Feature: Business-Call Handling

Specifies which types of distribution devices will be exempt from default business-call handling. By default, T-Server classifies any call arriving at a distribution device (ACD Queue, Routing Point, Routing Queue, External Routing Point) as a business call. Using this option, you can disable automatic classification for calls to a particular type of distribution device. For example, if the value for this option is set to `-rp`, calls to Routing Point DN's will not be automatically classified as `business`, allowing the routing strategy to use the `BusinessCallType` Extension.

This option does not affect the application of the DN-level `bsns-call-type` option.

bsns-call-type

Default Value: none

Valid Values:

`business`—The call is classified as a business call.

`private`—The call is classified as a private call.

`ignore`—The distribution point has no effect on business call classification.

Changes Take Effect: Immediately

Related Feature: Business-Call Handling

Specifies the business call type for calls that pass through or arrive at the associated device.

This option takes precedence over the following options that are set at the Application level: `inbound-bsns-calls`, `inherit-bsns-type` and `outbound-bsns-calls`. This option may be over-ridden by the extension `BusinessCallType`.

dn-for-undesired-calls

Default Value: No default value

Valid Values: Any valid switch DN

Changes Take Effect: Immediately

Related Feature: Smart OtherDN Handling

Specifies the DN that T-Server uses as the request destination if the client provides a reserved DN in the request.



Note: You can set a value for this option in the appropriate DN `Annex` tab in the `TServer` section. When set there, this value overrides the default value for the DN.

max-outstanding

Default Value: 8

Valid Values: 1–1000

Changes Take Effect: Immediately

Specifies the maximum number of outstanding sent requests awaiting the response from the link.

override-switch-acw

Default Value: `false`

Valid Value: `true`, `false`

`true`—T-Server will override the switch ACW.

`false`—ACW will override emulated ACW.

Changes Take Effect: Immediately

Specifies whether T-Server emulated ACW will override the switch ACW for calls distributed via a Route Point.

The option can be set in Configuration Manager in the following places in order of precedence (highest to lowest):

1. In the `TServer` section in the `Annex` tab of DNs of type `Routing Point`.
2. In the `TServer` section of the application

prd-dist-call-ans-time

Default Value: 0

Valid Value: Any integer from 0–10


Changes Take Effect: Immediately

Specifies the interval (in seconds) during which an agent can answer a predictive call before T-Server abandons it. With a value of 0 (zero), T-Server does not automatically abandon the call, which then rings on the agent desktop until it is answered.

When an emulated predictive dial is made from an emulated Routing Point, and options `nas-indication` and `supervised-route-timeout` are set, the value in `prd-dist-call-ans-time` takes precedence. For predictive dialing to work, you must set values greater than 0 (zero) for both options.

This option can be defined in two places:

1. In the `T-Server Application` object. This defines the default value to be applied for the predictive calls initiated from all distribution devices
2. In the `TServer` section in the `Annex` tab of any ACD Queue or Routing Point that is to be used as the origination device for a predictive call. When set there, this value overrides the value of the T-Server option set at the Application level for all calls that originate from that ACD Queue or Routing Point.

 **Note:** When using T-Server 8.0 with Outbound Contact Server (OCS) 7.6 or lower, this option must be set to 0 (zero).

rq-gap

Default Value: 0

Valid Value: Any integer from 0–1000

Changes Take Effect: Immediately

Specifies the minimum interval (in milliseconds) between succeeding CTI requests sent over the link. You can adjust the value to meet CTI-link load and performance requirements.

You can set this option in the `TServer` section in the `Annex` tab of a device.

Article Sources and Contributors

T-Server for CSTA Connector Supported Features Agent Substitution for Monitored Agents *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4656> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features Business-Call Handling *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4657> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features Call Release Tracking *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4658> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features Call Type Prediction *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4659> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features Emulated Agents *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4660> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features Error Messages *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4661> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features Hot-Standby HA Synchronization *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4662> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features Keep-Alive Feature Handling *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4663> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features Link Bandwidth Monitoring *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4664> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features No-Answer Supervision *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4665> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features Private Services and Events *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4666> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features Request Handling Enhancements *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4667> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features Smart OtherDN Handling *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4668> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features T-Library Functionality *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4669> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features User Data Keys *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4670> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Supported Features Using the Extensions Attribute *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=5248> *Contributors:* Ronaldb, WikiSysop

Common Configuration Options common Section *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4640> *Contributors:* Ronaldb, WikiSysop

Common Configuration Options log Section *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4642> *Contributors:* Ronaldb, WikiSysop

Common Configuration Options log-extended Section *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4641> *Contributors:* Ronaldb, WikiSysop

Common Configuration Options sml Section *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4643> *Contributors:* Ronaldb, WikiSysop

T-Server Common Configuration Options License Section *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4644> *Contributors:* Ronaldb, WikiSysop

T-Server Common Configuration Options TServer Section *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4645> *Contributors:* Ronaldb, WikiSysop

T-Server Common Configuration Options Translation Rules Section *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4646> *Contributors:* Ronaldb, WikiSysop

T-Server Common Configuration Options agent-reservation Section *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4647> *Contributors:* Ronaldb, WikiSysop

T-Server Common Configuration Options backup-sync Section *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4648> *Contributors:* Ronaldb, WikiSysop

T-Server Common Configuration Options call-cleanup Section *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4649> *Contributors:* Ronaldb, WikiSysop

T-Server Common Configuration Options extrouter Section *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4650> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector Agent-Specific Override Options *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4651> *Contributors:* Ronaldb, WikiSysop

T-Server for CSTA Connector DN-Level Options *Source:* <http://developerzone.genesyslab.com/wiki/index.php?oldid=4653> *Contributors:* Ronaldb, WikiSysop

Image Sources, Licenses and Contributors

Image:Welcome.png *Source:* <http://developerzone.genesyslab.com/wiki/index.php?title=File:Welcome.png> *License:* unknown *Contributors:* WikiSysop

File:Information.png *Source:* <http://developerzone.genesyslab.com/wiki/index.php?title=File:Information.png> *License:* unknown *Contributors:* WikiSysop

Image:fr_dep-ts_connector_functions-3.gif *Source:* http://developerzone.genesyslab.com/wiki/index.php?title=File:Fr_dep-ts_connector_functions-3.gif *License:* unknown *Contributors:* Ronaldb, WikiSysop