Framework Management Layer User's Guide

Management Framework Current

7/12/2023

Table of Contents

Framework Management Layer User's Guide

Published: September 26, 2017

9.x **Management Framework** is part of 9.x, which can include component releases from both 9.1.x, 9.0.x, and 8.5.x code streams. See Management Framework to check which component releases are part of 9.x.

Use this guide to introduce you to the concepts, terminology, and procedures relevant to this layer of the Genesys Framework.

The Management Layer

Functions

Management Layer Components

Management Layer Architecture

Troubleshooting

Management Layer Functionality

Solution Control and Management

Logging

Alarms

Fault Processing

SNMP Support

Monitoring and Control Functions

Using the Management Layer, including how to:

Monitor Solutions, Applications, and Hosts

Manage 3rd-Party Applications

Manage HA Configuration Servers and Configuration Server Proxies

Starting and Stopping Applications and Solutions

How to:

Start and Stop Applications and Solutions

Use Startup Files

Centralized Logs

Setting up a centralized log system, including how to:

- Configure Logging
- Manage and View Logs
- Trace Interactions and Set Up an Audit Trail
- Understand Log Formats and

Alarms

- How to Configure Alarm Conditions and Reactions
- E-Mail Alarm Reactions
- Predefined Alarm Conditions

SNMP Support

- SNMP Interface
- SNMP Managed Objects
- SNMP Traps
- Genesys MIB File Changes

Introduction to Management Layer

The Management Layer of Genesys Framework 8.5 provides the following functions:

- Solution and application control and monitoring—You control and monitor all solutions and applications from a single point. The Management Layer displays the real-time status of every configured Solution object, and you can activate and deactivate solutions and single applications from this layer. This control and monitoring also includes user-defined solutions.
- Centralized logging—Applications log maintenance events in the unified log format and the events are recorded in one central location. That format enables easy selection of required log records and centralized log storage for convenient access and solution-level troubleshooting. With centralized logging, you can also track individual interactions, audit activities in your contact center, and store alarm history.
- Alarm signaling—Flexible alarm signaling triggers alarms based on application maintenance events, thresholds for system performance variables, or Simple Network Management Protocol (SNMP) variables. Solution Control Server communicates alarms to Genesys Administrator and can write alarms to system logs. You can configure the system to convert alarms into SNMP traps and send them as emails to a specified email address. (The latter automatically enables paging notifications.) The Management Layer automatically associates alarms with the solutions they affect and stores alarms as active conditions in the system until either they are removed by another maintenance event or you clear them. Alarms are visible only if you have access to the application that generated the alarms.
- Application failure management—Fault-management functions consist of detection, isolation, and correction of application failures. For non-redundant configurations, the Management Layer automatically restarts applications that fail. For redundant configurations, this layer supports a switchover to the standby applications and also automatically restarts failed applications.
- Built-in SNMP functionality—Extended SNMP support enables both alarm processing and SNMP data exchange with an SNMP-compliant network management system (NMS). This means you can integrate a third-party NMS with a Genesys system to serve as an end-user interface for system control and monitoring and for alarm signaling.

> **Important**
>
> The SNMP functionality of the Management Layer is controlled by the Genesys licensing system. Refer to the *Genesys Licensing Guide* for information about ordering licenses that activate this functionality.

- Individual host monitoring—host parameters are monitored, including records of CPU and memory usage and information about currently running processes and services.
- Support for geographically distributed environments—a special mode in Genesys Solution Control Server simplifies the operation of a geographically distributed installation that uses a single Configuration Database.

> **Important**
>
> The Management Layer support for geographically distributed environments is controlled by the Genesys licensing system. Refer to the *Genesys Licensing Guide* for information about ordering licenses that activate

> this functionality.

See Architecture for information about the Management Layer components and Management Layer Functionality for more detailed information about Management Layer functions.

Refer to the *Framework Deployment Guide* for configuration and installation instructions.

Refer to the *Management Framework Migration Guide* for information about component compatibility, and for instructions about migrating between different releases of Framework components.

Components

This topic describes the components that make up the Management Layer of Management Framework.

Local Control Agent

Local Control Agent (LCA) is a daemon component that monitors, starts, and stops Genesys server applications as well as third-party server applications that you have configured in the Genesys configuration environment. In addition, LCA detects failures of Genesys servers and communicates their roles in redundancy context.

The LCA Installation Package (IP) also contains the Remote Deployment Agent (not shown in the diagram above) that deploys Genesys IPs as directed by Genesys Administrator Extension. Starting in release 8.5.1, the Remote Deployment Agent is no longer installed by default when installing LCA. Refer to the *Framework Deployment Guide* for more information about installing the Remote Deployment Agent.

Message Server

Message Server provides centralized processing and storage of all maintenance events that Genesys server applications generate. Events are stored as log records in the Centralized Log Database (also referred to as *Log Database*) where they are available for further centralized processing. Message Server also checks for log events configured to trigger alarms. If it detects a match, it sends the alarm to Solution Control Server (SCS) for immediate processing.

For usability reasons, Genesys does not recommend that you configure multiple Message Servers for one Log Database, with each Message Server assigned to handle logs for different applications. To view logs processed by a particular Message Server and therefore generated by a given application, you still have to filter the logs based on the application that generated them.

> **Important**
>
> Some solution components may also exchange messages via Message Server. You can find more details on this Message Server function in solution-specific documentation.

Solution Control Server

Solution Control Server (SCS) is the processing center of the Management Layer. It uses Local Control Agents to start solution components in the proper order, monitor their status, and provide a restart or

switchover in case of application failure. SCS also processes user-specified alarms.

Log Database

The Log Database stores all log records, including interaction-related records, alarm history records, and audit records.

SNMP Master Agent

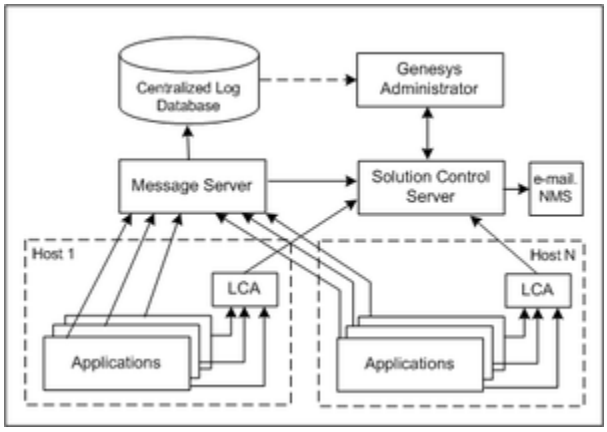
SNMP Master Agent (an optional component also not shown in the diagram above) is an interface between the Management Layer and an SNMP-compliant network management system (NMS). It distributes:

- SNMP traps, which are converted from alarms, from Solution Control Server to NMS.
- SNMP commands, which a user enters from an NMS interface, back to SCS for further processing.

For more information, refer to SNMP Support.

Architecture

The following figure shows interactions between Management Layer components and an application. To enable the Management Layer's solution-control and fault-management capabilities, you must install Local Control Agent on each host on which a monitored application (whether a Genesys server or a third-party application) is running. To enable the Management Layer's centralized-logging and alarm-signaling capabilities, you must configure each Genesys server application with a connection to Message Server.



Management Layer Architecture

Management Layer Functionality

This section describes the Management Layer capabilities that help you optimally manage the Genesys software serving your contact center, as follows:

- Solution Control and Monitoring Functions
- Logging Functions
- Alarm-Signaling Functions
- Fault Management Functions
- SNMP Support Functions

You can also manage Management Layer components such as Message Server, Solution Control Server, and SNMP Master Agent, via the Management Layer, as discussed in this section.

Solution Control and Management Functions

Use the Management Layer's control and monitoring functions to:

- Start single applications or entire solutions through a single control operation from Genesys Administrator.
- Shut down single applications or entire solutions in the same manner.
- Start all or a set of configured solutions.
- View the current runtime status of applications and entire solutions via Genesys Administrator.
- View all processes currently running on any host.
- View CPU and memory usage data for any host.
- View CPU usage data for each thread of a given process of an application.
- Manually switch operations from a primary server to its backup.
- Monitor the health of the NTP service, and change the signature of the NTP service or daemon.

For efficient solution maintenance, you should also use the monitoring functions for both solutions and applications. However, under normal circumstances, Genesys recommends that you always perform control functions over entire solutions as opposed to single applications. This ensures the correct logical order of application startup and shutdown and eliminates unnecessary error log events. In redundant configurations, the solution-level control operations also take into account the configured backup servers and start them up or shut them down automatically along with the primary servers.

Regardless of what level of control—applications or entire solutions—you choose to implement, the same architecture provides the control and monitoring functions. It consists of:

- Solution Control Server (SCS).
- As many instances of Local Control Agent (LCA) as there are computers with Genesys servers and/or with Management Layer–controlled third-party servers.
- Genesys Administrator, through which the control and monitoring capabilities are available directly to a user.

> **Important**
>
> A single SCS should be assigned to handle no more than 250 hosts and their applications. In this environment, SCS can handle the simultaneous failure of up to 50 hosts, while reflecting the actual runtime statuses of the hosts and affected applications in 15 seconds or less. If an SCS has to handle more than 250 hosts, the time to respond to host status changes might increase, especially if more than 50 of its hosts experience failures simultaneously. If you need SCS to handle more than 250

hosts, Genesys recommends that you configure multiple Solution Control Servers in distributed mode to limit the load on each SCS. Refer to the *Framework Deployment Guide* for information about setting up a distributed SCS environment.

Manual Switchover

The Management Layer provides an additional control function, a *manual switchover* of an application's operations to its backup application. Use this function, for example, for test purposes, during application upgrades, or during some hardware maintenance procedures. You can perform a manual switchover for any pair of redundant applications on which both primary and backup are running. During the switchover, the Management Layer changes the mode of the selected backup application to primary and the mode of the primary application to backup. The switchover mechanism is described in detail in Fault Management Functions.

You *cannot* manually switch over applications of these types:

- Configuration Server
- Database Access Point
- Solution Control Server

NTP Service Monitoring

Starting in release 8.1, the Management Layer can monitor the health of the NTP services on supported platforms.

LCA collects information about the state of the daemons at startup, and updates it periodically thereafter, when checking the state of third-party applications. Log events `00-08008` and `00-08009` report that the NTP service is available and is not available, respectively. Refer to the *Framework Combined Log Events Help* file for a description of these log events.

You can also configure the signature of an NTP service/daemon. Use the **signature** option in the **[ntp-service-control]** section of the Host object for which you want to configure the signature. Refer to the *Framework Configuration Options Reference Manual* for detailed information about this option.

Permissions

To use Genesys Administrator to monitor solutions and applications, the user must have Read permission for the corresponding objects in the Configuration Database. To perform any control operations on solutions and applications, the user must have `Execute` permission for the corresponding objects. To receive alarm reactions related to applications and hosts, the user must have Read permission for the corresponding objects in the Configuration Database. For more information about security settings, see the *Framework Deployment Guide*.

Remember that the Management Layer is a multi-client environment that makes solution-control functions available to an unlimited number of users simultaneously. The proper and responsible use of these functions is crucial for normal solution operations. Consider using the security capabilities of the Configuration Layer to limit access to these control functions to the trained personnel directly responsible for the contact center environment. Furthermore, schedule all control operations to occur during off-peak hours, preferably when the contact center is not processing any interactions, to ensure the availability of the customer interaction functions.

Controlling Third-Party Applications

You can apply the Management Layer control and monitoring functions to third-party applications that meet the necessary prerequisites.

These applications include, but are not limited to:

- SQL servers.
- CRM services.
- ERP services.

> ### Warning
> On Windows platforms, the Management Layer attempts to start an application without analyzing whether the application can run on an unattended computer (for instance, on a Windows computer with no user currently logged in) or whether the application can operate without a console window. Because the LCA that starts applications is always installed as a Windows Service, all processes start without a console window.

Logging Functions

The Management Layer collects Genesys application logs of defined levels and stores them in a centralized database.

Log Levels

Genesys applications can report log events at five levels of detail: *Alarm*, *Standard*, *Interaction*, *Trace*, and *Debug*. Only the first four are intended for on-site analysis by a user. Log events of the Alarm, Standard, Interaction, and Trace levels feature the same unified log record format and can be stored in the Centralized Log Database.

Logging During Normal Operation

For complete specifications of log events reported at the Alarm, Standard, Interaction, and Trace levels, see *Framework Combined Log Events Help*.

Alarm Log Level

The Alarm-level logs contain only log records of the Alarm level. SCS generates Alarm log events on behalf of other applications when receiving from them log events configured as Detection Events in Alarm Conditions. Using this level, Solution Control Server reports the occurrence and removal of all alarms to the Centralized Log Database.

This level contains the Management Layer translations of log events of other levels into Alarms.

Standard Log Level

Permanently enable *only* the Standard level of logging during solution operation in regular production mode. It contains high-level events that report both major problems and normal operations of in-service solutions.

An event is reported at the Standard level if it satisfies one of these criteria:

- Indicates that an attempt to perform any external operation has failed
- Indicates that the latest attempt to perform an external operation that previously failed has succeeded
- Indicates detection of a condition that has a negative impact on operations, actual or projected
- Indicates that a previously detected condition, which had a negative impact on operations, no longer exists
- Indicates a security violation of any kind
- Indicates a high-level data exchange that cannot be recognized or does not follow the expected logical sequence
- Indicates inability to process an external request

- Indicates successful completion of a logical step in an initialization process
- Indicates a transition of an application from one operational mode to another
- Indicates that the value of a parameter associated with a configurable threshold has exceeded that threshold
- Indicates that the value of a parameter associated with a configurable threshold that earlier exceeded the threshold has returned to its normal range

Interaction Log Level

The Interaction-level logs report the details of an interaction processed by solution components that handle interactions. The log contains information about the processing steps for each interaction by each solution component.

An event is reported at the Interaction level if it:

- Is a recognizable high-level data exchange with another application about an interaction.
- Indicates a change in real-time state of an interaction handled by the application (unless such a change is visible from the high-level data exchange).

The specific criteria depend on a particular component and its role in interaction processing.

Use the Interaction-level log records to analyze and troubleshoot new interaction-processing scenarios, especially when you introduce new solutions or enable new functions within existing solutions. Note that Interaction-level records contain the interaction attributes, such as Interaction ID, that you can use to search for log events related to the same interaction but generated by different applications.

> **Warning**
> Using the Interaction level generates a higher number of loggingevents on the network and that may adversely affect the performance of the DBMS, Message Servers, and interaction-processing components.

Trace Log Level

The Trace-level logs report the details of communications between the various solution components. The log contains information about the processing steps for each interaction by each solution component.

An event is reported at the Trace level if it satisfies one of these criteria:

- It is a recognizable high-level data exchange with another application.
- It is a recognizable high-level data exchange with an external system.
- It indicates a change in real-time state of user-level objects that the application handles (unless such a change can be seen from the high-level data exchange).

Use the Trace-level log records to analyze and troubleshoot new interaction-processing scenarios, especially when you introduce new solutions or enable new functions within existing solutions.

> **Warning**
> Using the Trace level generates a higher number of logging events on the network and that may adversely affect performance of the DBMS, Message Servers, and interaction-processing components.

Logging During Irregular Operation

Standard-level, Interaction-level, and Trace-level log events do not contain all the details you or someone else may need to analyze and troubleshoot solutions malfunctions. That is why Genesys Customer Care might ask you to provide relevant Debug-level logs when you request their assistance.

Because the Debug-level log events do not have a unified format, are not documented, and can only be stored in a text file, they are only useful to Genesys Customer Care. Logging at this level is likely to adversely affect application performance. Enable this log level only when a Genesys representative requests it. Keep in mind that running Genesys servers with the Debug level of logging is highly resource-intensive and, as such, is not recommended when you are in production mode.

Before you set a logging level more detailed than the Standard level, carefully consider whether a situation (such as the initial deployment or first signs of technical problems) really calls for it. Then, test for how more-detailed logging affects the network loads in a lab or controlled environment.

Note that changing the log level of a running application does not interrupt solution operations.

In addition to asking for Debug-level log records when you report a problem to them, Genesys Customer Care might also request that you reproduce the problem because:

- During regular operations, many contact-center systems, such as DBMSs, IVRs, and switches, do not employ logging at the level of detail required to diagnose serious technical issues.
- Reasons other than application failure can contribute to interaction-handling errors. For example, a call can be misrouted (delivered to a wrong DN) despite the fact that applications are functioning properly.

Reliability Logging

Starting in release 8.5, Genesys software supports a level of reliability logging, most notably to provide information for estimating product availability. This information is available from some log events, to which appropriate information has been added or made available. At the Management Framework level, three common log events (`00-05090`, `00-05091`, and `00-05064`) have been enhanced with the addition of application name, type, and DBID as extended attributes. Refer to *Framework Combined Log Events Help* for full descriptions of these log events.

Centralized Logging

The centralized logging function provides a number of advantages over the more traditional logging to a text file:

- Keeping log records of all applications in one place and presenting them in the unified log record format provides for a comprehensive view of the solutions' operations history.
- Using a relational DBMS such as the central log storage enables quick access to the required records and allows for advanced record selections, which you can base on a variety of search criteria.
- Viewing, via Genesys Administrator, the logs stored in a Centralized Log Database gives you an integrated view of the solutions' maintenance history and complements the solution-control and alarming capabilities.
- Deleting the obsolete logs or logs of a particular solution, host, or application makes the Log Database management more convenient.

Given these advantages, Genesys recommends using the centralized logging as the primary method for storing Standard-level log events of all applications. When enabling the log output for Interaction and Trace logs, store log events of both levels in the Centralized Log Database in addition to log events of the Standard level. Genesys does not recommend the simultaneous use of both local and centralized logging options except for some special, temporary purpose.

The centralized logging system consists of:

- One or more Message Servers that collect log events from applications.
- One or more Log Databases.
- Genesys Administrator.

Provided that the Standard level of log output is routinely used under normal production conditions, always limit the centralized logging system to one Message Server and one Log Database for all but large and geographically distributed interaction management networks.

If any part of the centralized logging system becomes unavailable, the log outputs of the affected applications are temporarily redirected to local binary files. Upon restoration of normal functioning, the applications automatically resume logging to the Centralized Log Database. The log records accumulated in the local binary files are automatically transferred to the Log Database.

If the connection between the Message Server and the Log Database is unavailable, messages are stored in a queue. When the connection is restored, the messages in the queue are written to the Log Database. See the "Message Server" section of the *Framework Configuration Options Reference Manual* for more information.

The format of records kept in the Log Database is specified in Log Formats.

Viewing Log Database Entries

Although you can use any general-purpose DBMS client to make advanced selections from the Log Database, Genesys Administrator's log-viewing capabilities may actually meet your needs just as well. In either interface, you can view an entire log. They also provide a number of predefined selections from the Log Database, which are based on the most typical maintenance-selection criteria:

- Records generated by components of a selected solution.
- Records generated by applications located on a selected host.
- Records of a specified output level.
- Records containing a specified combination of symbols in text.
- Records generated within a specified time interval.
- Records containing specified values of certain extended attributes.

You can use these predefined selection criteria in any combination.

To delete obsolete log records, you can use Genesys Administrator.

Logging and Application Performance

Follow these recommendations to increase an application's performance while enabling the application's logging:

- Always enable buffering of the log output when sending logs to a log file. Refer to the "Common Log Options" chapter in the *Framework Configuration Options Reference Manual*.
- Store log files on the local disk of the computer running the application rather than storing them using network file systems. Such systems may not perform very well and the added network traffic for this storage can affect application performance.
- Configure only log events of the Standard level to be sent to the Log Database. For log events of other levels, consider using the memory output as the safest output in terms of application performance.
- 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.

Logging Resilience

Starting in release 8.5, Genesys logging incorporates additional functionality specifically aimed at maintaining the integrity and usefulness of the logging system, including the Centralized Log, without causing input/output or run-time logging issues from affecting normal operations or causing the application to terminate unexpectedly.

Without this feature, logs are written to output by the application's main thread. Any delay or bottleneck in the log output system takes up processing time and space for non-log operations using that thread.

This problem is greatly resolved, if not eliminated, by the logging resilience feature. The feature is configured at the application level, and consists of two elements:

- An internal log queue, for which a dedicated thread processes log messages for output.
- A throttling mechanism, which only when a performance problem is detected, changes the verbose level of the log system (as specified by the **verbose** option in the `log` section). This element cannot be activated unless the dedicated process thread is active.

When logging resilience is activated (as specified by **enable-thread=true** in the `log` section), all log messages generated by the application are moved to an internal queue, from which they are written to output using the dedicated thread. This negates output problems backing up into regular processing.

Once the internal queue is established, the throttling mechanism can start (if configured to do so). This throttling mechanism works by monitoring the size of the internal log queue, and changing the value of the verbose option to increase or decrease the number of logs being generated, as follows:

- When the queue size approaches a configured threshold (specified by a non-zero value of the **throttle-threshold** option in the `log` section), the value of the **verbose** option is reduced by one level to reduce the number of log messages generated.
- When the queue size decreases to 50% or less of the threshold, the value of the **verbose** option is increased by one level to increase the number of log messages generated.

The verbose level is maintained at this level until both of the following conditions occur:

- A configured period of time (called the throttle period, and specified by the **throttle-period** option in the `log` section) expires. This timer is reset each time that throttling (up or down) occurs.
- One of:
  - The queue size increases and approaches the threshold, at which point the verbose level is throttled again. This can only occur until **verbose=none**, at which point there are no logs being processed for output.
  - The queue size drops to 50% or less of the threshold, at which point the verbose level is increased by one level. This can only occur until the value of verbose is back to its originally configured value.

Throttling is an optional part of the Logging Resilience feature, it can be disabled or stopped without interfering with the internal log queue.

For detailed descriptions of the three options used for and by this log resiliency functionality, refer to the “Common Configuration Options” chapter of the *Framework Configuration Options Reference Manual*.

Alarms

The Management Layer uses the Centralized Log Database to store detailed and structured information about Alarm activation and clearance. (See Alarm-Signaling Functions for more information about how alarms are generated.) Solution Control Server generates alarm-related information as log events of the Alarm level for each Alarm activation and clearance event. Solution Control Server attaches a set of extended attributes to each Alarm log event; in particular, the ID attribute uniquely identifies each Alarm.

For complete specifications of Alarm log events that SCS and Message Server report, and for information about extended attributes for each log event, see *Framework Configuration Options Reference Manual*.

Audit Trail

The Management Layer also uses the Centralized Log Database to store Audit-Trail records (from here on referred to as Audit records) that Framework components (in particular, Configuration Server and SCS) generate for configuration changes and control actions performed over processes, solutions, and alarms. Starting in release 8.1, the Audit records also include the name of the client application and details about the host on which the client application resides. This is to ensure compliance to Payment Card Industry Data Security Standard (PCI-DSS) 10.3.

Framework components generate Audit records as log events and, if available, attach extended attributes to Audit log events.

For information about setting up an audit trail and viewing the Audit logs, see How to Set Up and Use an Audit Trail.

For complete specifications of Audit log events that Framework components report and for information about extended attributes for each log event, see *Framework Configuration Options Reference Manual*.

History of Configuration Changes

Starting in release 8.5, Configuration Server keeps a history of configuration changes, including a record of new and previous values. For more information, refer to the *Framework Deployment Guide*.

Interaction Tracing

You can configure Framework components to send Interaction-level log events to the Centralized Log Database. You can later retrieve from the database all records related to a certain interaction, enabling you to trace its progress in the contact center.

> **Important**
> Storing Interaction-level log events in the Log Database might affect application performance, so Genesys does not recommend it in production environments.

Framework components might attach a set of extended attributes to each Interaction log event. In particular, each such event contains a unique identifier of the contact center interaction in the IID extended attribute.

> **Important**
> The set of extended attributes for Interaction-level log events may vary depending on a particular interaction's properties and the component that generates the log event.

For complete specifications of Interaction-level log events that Framework components report, and for information about extended attributes for each log event, see Framework Combined Log Events Help, available on the Genesys Documentation wiki.

Use Genesys Administrator to display all Interaction-level records from the Centralized Log Database. Use predefined selection criteria to search for all records with a particular Interaction ID.

For information about viewing Audit records with Genesys Administrator, see Framework Genesys Administrator Help.

Alarm-Signaling Functions

Maintenance events that the user may want to become aware of and react to immediately are communicated as Standard-level log events that Genesys applications generate. Each log event is assigned a unique number, which identifies the situation being reported. Thus, the alarm-signaling function of the Management Layer is based on the capability to detect the log events that have been pre-configured to trigger alarms and to send them to an alarm-processing center. In addition, the Management Layer monitors certain system and SNMP variables, which you can also use for alarm signaling.

This topic describes how the Management Layer implements an alarm system, and what is required to incorporate it into your configuration. For specific instructions, refer to Alarms.

Alarm Detection

The Management Layer detects alarms by matching the following against the alarm conditions you have configured:

- Log events coming from all applications
- The thresholds of the system performance variables (such as CPU or memory usage) and of local or remote SNMP variables. (SNMP threshold monitoring is available only when you have enabled SNMP functionality.)

SCS provides the same alarm-reaction processing for both Alarm-detection mechanisms.

Using Log Events for Alarm Detection

To use a log event to trigger an alarm, you must configure an object of the Alarm Condition type and associate it with a log event ID in the Configuration Layer. When you configure an Alarm Condition, you do the following:

- Specify the log event that should trigger this alarm during runtime.
- Assign an alarm category.
- Define the source of the alarm.
- Set conditions for automatic alarm clearance.

The source of an alarm can be a specific application, all applications of a particular type, or all applications of the interaction management network. In each case, the resulting alarm message contains the application name. So, you can know the exact source of the alarm.

> **Tip**
> You can also use log events of the Standard, Interaction, or Trace levels to trigger an alarm message.

Each application that can generate an alarm must be configured with a connection to Message Server to which it is able to send log events of the appropriate level. Otherwise, the Management Layer is unable to detect the log events. Once configured, an alarm condition automatically triggers an alarm in response to an occurrence of the log event on which the alarm condition is based. If the same log event occurs subsequently while the alarm is active, the clearance timeout is reset.

As previously noted, the alarm detection takes place in Message Server, so you must connect the potential sources of alarms to Message Server for alarm signaling to operate. If you are planning to use the recommended centralized logging function, your applications should already be connected to Message Server. Otherwise, you need to set up Message Servers and configure your applications to connect to them specifically for alarm-detection purposes.

> **Tip**
> If you are using Genesys Administrator 8.1.3, you can use the by using the Alarm Condition Wizard in Genesys Administrator. For more information, see the *Framework Genesys Administrator Help*.

The Configuration Layer also provides a number of preconfigured alarm conditions based on the events that cause service degradation in any environment. Before configuring your own alarm conditions, see if they may have been predefined in the Configuration Layer. For more information about predefined alarm conditions, see Predefined Alarm Conditions.

Using System Parameters and SNMP Thresholds for Alarm Detection

The alarm-detection mechanism for thresholds for system performance variables or SNMP variables is similar in many respects to the log-event-based mechanism. In particular, you must configure certain alarm conditions in the Configuration Database that indicate the values for the Management Layer to monitor.

When you are using thresholds for system performance variables, the Management Layer detects an alarm by periodically comparing the current value of the specified performance variable with the specified limits. If a change in the variable's value exceeds the specified limit, the Management Layer triggers an alarm.

When you are using SNMP variables as thresholds, the Management Layer detects an alarm by periodically comparing the current value of the specified SNMP variable, as identified by OID, with the specified limits. Currently, the following two SNMP variables are supported for this purpose:

- **gsClientExistNum** in table **gsInfoTable**
- **tsCallsExistNum** in table **tsInfoTable**

If a change in the variable's value exceeds the maximum limit of the specified minimum to maximum value range, an alarm is triggered. When the variable's value falls below the minimum limit of the specified range, the active alarm is cleared.

The Rising Threshold, which triggers an alarm when crossed *only if the value is rising*, must be a higher number than the Falling Threshold, which clears the alarm when crossed *only if the value is falling*. For example, if the Rising Threshold is 300 then the Falling Threshold must be less than 300.

This mechanism provides alarm signaling with both local SNMP variables—that is, variables from the Genesys MIB file, implemented locally in SCS—and with remote SNMP variables—that is, variables provided by third-party SNMP agents.

To monitor a variable of either type, use Genesys Administrator to create:

- An Alarm Detection Script
- A new Alarm Condition based on the Alarm Detection Script

For more information, see *Framework Genesys Administrator Help*.

Default Alarm Processing

Once it detects an alarm, Message Server sends it to Solution Control Server for processing. SCS processes the detected alarm in this way:

1. Stores the alarm in the system as active until it is removed manually, expires based on the configurable timeout, or is cleared by another log event, which you can optionally define in the Alarm Condition object as an automatic removal condition.
2. Generates log messages about every alarm detection and its removal.
3. Passes the alarm information and a list of all the running solutions that the alarm may affect to Genesys Administrator to display them for the user. SCS only passes alarm information about objects (such as applications or hosts) that the user currently logged in to Genesys Administrator has permissions to view. If necessary, the user can then take the appropriate action.

For alarm processing to take place, you must connect SCS to the Message Servers that detect the alarms.

Whenever you start Genesys Administrator, it automatically displays all active alarms currently registered in SCS as long as you have permissions to view the objects associated with the active alarms.

For more information, see *Genesys Administrator Help*.

Customized Alarm Processing

In addition to relying on the default alarm-processing actions, you can configure other actions (called Alarm Reactions) that the Management Layer is to take when it detects a specified alarm, such as:

- Shutdown a specified application.

- Start up a specified application.
- Restart the application that reported the alarm.
- Start up a specified solution.
- Send an email message with detailed information about the alarm to specified Internet addresses.
- Switchover operations from the application that reported the alarm to its backup application, for applications running in primary, backup, or either mode.
- Send an SNMP trap with detailed information about the alarm to a general-purpose network management system.
- Execute an operating system command.
- Change the value of a configuration option of a specified application, including the application that reported the alarm. (If the proposed change to an option is for a section or option that does not exist, the system creates both.)

Most of these reactions do not require any special arrangements. However, the switchover reaction type requires that the application in question have a backup application configured and running. The application restart and switchover mechanisms are described in detail in Fault Management Functions.

If you wish to use SNMP trap capabilities, you must install an SNMP Master Agent and configure your Solution Control Server to connect to it. You can use Genesys SNMP Master Agent or a third-party SNMP Master Agent you already have within you network management system—as long as it is compliant with the AgentX protocol. For instructions on these procedures and for detailed specification of the SNMP trap to which the Management Layer converts the alarms, see SNMP Support.

Though the Management Layer itself does not provide paging notifications, you can arrange these through the supported email or SNMP interfaces using your email server or network management system, respectively.

An alarm reaction is configured in the Configuration Database as a Script object of the Alarm Reaction type. For runtime execution of a particular alarm, you must associate the alarm reaction with the corresponding Alarm Condition object. You can configure any combination of supported reactions with respect to any alarm condition. The easiest way to do this is by using the Alarm Condition Wizard in Genesys Administrator.

You can configure alarms in the Management Layer that execute alarm reactions not only at alarm activation, but also at alarm clearance. To achieve this, add the alarm reaction Scripts that should be executed when the alarm is cleared to the Clearance Scripts list of the corresponding Alarm Condition object. You can also use the Alarm Condition Wizard in Genesys Administrator to accomplish this.

You can also use the mlcmd utility to clear all active alarms raised by an application or on the basis of a specified Alarm Condition.

Fault Management Functions

Faults—accidental and unplanned events causing a system to fail—present the biggest challenge to solution availability. The functions that detect, isolate, and correct various types of faults are partly incorporated into every Genesys component and partly implemented in the Management Layer of the Genesys Framework. The role of the Management Layer in application failure management is described in detail in this section.

Application Failures

A complete application failure may be a result of either an internal defect (for example, an infinite loop) or an external event (for example, a power failure). It may manifest as either a process non-response or termination. Typically, if a solution component stops working, the solution is no longer available to process customer interactions.

Since the application that fails cannot perform any functions, you must employ an external mechanism for both detection and correction of faults of this type. The Management Layer serves as such a mechanism. To detect an application failure, the Management Layer uses a simple monitoring component called Local Control Agent (LCA), which continuously maintains a connection with the application, confirming both its existence and ability to communicate. To make sure an application failure is never confused with a connection failure, the LCA that monitors a specific application always resides on the computer on which the application itself is running.

LCA is installed on a one-per-host basis and can connect to all Genesys applications located on the host. When a connection is broken, LCA generates a message to Solution Control Server, where an appropriate recovery action is chosen and executed according to the system configuration. SCS uses the Advanced Disconnect Detection Protocol (ADDP) to recognize a loss of connection with LCA. A loss of connection is interpreted as a failure of the host (that is, as failures of all Genesys components running on that host).

ADDP is, by default, enabled for the connection between SCS and LCA, with the ADDP timeout set to 9 seconds. With the default settings, SCS can detect and handle application failures in 20 seconds or less.

> **Important**
>
> However, if there is a particular risk of network delays, Genesys recommends setting ADDP timeouts to values equal to or greater than 10 seconds, rather than relying on default values to avoid false detection of disconnection. You can modify ADDP parameters for the connection between SCS and LCA in the Host object of the computer that runs LCA. For more information about these settings, refer to the *Framework Configuration Options Reference Manual*. For more information about ADDP, refer to the *Framework Deployment Guide*.

If you have not configured a backup application for the failed component, the correction procedure

normally consists of attempts to restart the failed process, if so configured. The same LCA component that detects application failures starts any Genesys application located on the host upon a command from SCS. If the application in question is a server, the clients automatically reconnect to this server once it is restarted and initialized.

> **Tip**
>
> Genesys recommends that you configure an automatic application restart procedure for all daemon applications.

> **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 you have configured an appropriate alarm condition and alarm reaction for them.

If a backup application is configured and started, the Management Layer automatically switches operations over to that application, given that you have a *high-availability* (HA) license (see the following notes). If the application is a server, the clients automatically connect to the backup server.

Notes:

- HA licenses are required only for those components that support automatic switchover.
- HA licenses are not required for manual switchover, either via the application-specific menu, or by stopping the primary server. All applications can be switched over manually.

The Management Layer currently supports warm standby between redundant components within the layer. It also supports switchovers between redundant client applications, regardless of the redundancy type specified by those applications. You must have an HA license to support either type of redundancy.

> **Warning**
>
> The Management Layer does not support Cold Standby redundancy type.

The Management Layer also provides more robust switchover capabilities, and, in particular, allows detection of situations when a running application is unable to provide service and treats this situation as an application failure. The Service Unavailable application status serves this purpose.

When an application reports that its status has changed to Service Unavailable, and a backup server for this application is configured and started, the Management Layer automatically switches operations over to the backup server. Respectively, when both primary and backup applications are running with the Service Unavailable status, the backup application may report that it can now

provide the service (that is, the backup application status changes to Started). In this case, the Management Layer automatically switches operations over to the backup application. As with a switchover resulting from an application failure, you must have an HA license to perform a switchover related to service unavailability.

> **Important**
>
> While some applications support the Service Unavailable status and report it under appropriate circumstances, others do not. (For instance, when T-Server loses its connection to the CTI Link, T-Server changes its status to Service Unavailable). The Management Layer operates based on the information supplied by an application and cannot automatically detect an application's inability to provide service. Refer to application-specific documentation to determine if the Service Unavailable status is supported on the application side.

Warm Standby Redundancy Type

Genesys uses the term `Warm Standby` to describe the redundancy type with which a backup server application remains initialized and ready to take over the operations of the primary server. Inability to process interactions that may have originated during the time it took to detect the failure is reduced to a minimum. Warm Standby redundancy type also eliminates the need to bring a standby server online, thereby increasing solution availability.

The standby server recognizes its role as a backup and does not process client requests until its role is changed to primary by the Management Layer. When a connection is broken between the primary server and the LCA running on the same host, a failure of the primary process is reported. As a result, the Management Layer instructs the standby process to change its role from standby to primary, and the former standby starts processing all new requests for service.

> **Important**
>
> To switch to primary mode, the backup Configuration Server must have an active connection to the Configuration Database during the failure of the primary Configuration Server.

While normal operations are restored as soon as the standby process takes over, the fault management effort continues. It consists of repeated attempts to restart the process that failed. Once successfully restarted, the process is assigned the standby role.

If Solution Control Server detects a loss of connection with the LCA of a host, SCS performs switchover for all applications located on the host, if backup applications are configured. There are two exceptions to this:

- A Configuration Server in backup mode ignores the switchover command if it detects another Configuration Server in primary mode. In other words, if the LCA residing on a host with a Configuration Server in primary mode goes down, the SCS requests that a Configuration Server in backup mode, on another host with an available LCA, switch over to primary mode. When the request is received, this Configuration Server checks whether the Configuration Server in primary mode is down, as indicated by

a lost connection between the two Configuration Servers. The Configuration Server in backup mode switches over to primary mode only if this connection is down. If the connection still exists, no switchover occurs.

- An SCS in backup mode does not try to switch itself over if it can still detect the SCS that is in primary mode. In other words, if an SCS in backup mode loses its connection to an LCA residing on a remote host with an SCS in primary mode—either because the LCA went down or a network timeout caused the SCS to drop its connection—the SCS in backup mode checks whether it is still connected to the remote SCS in primary mode. If that connection is also lost, the SCS switches over and runs in primary mode.

Hot Standby Redundancy Type

Genesys uses the term *Hot Standby* to describe the redundancy type with which a backup server application remains initialized, clients connect to both the primary and the 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. Configuration Layer and Management Layer components do not support Hot Standby between pairs of redundant components. They do support switchover between client applications configured with this type.

Hot Standby redundancy type with client connection support is only implemented in T-Servers for most types of switches and is not implemented in applications of other types. For a complete description of the hot standby redundancy type, refer to the deployment guide for your specific T-Server.

Hang-up Detection

Starting in release 8.0, LCA can use hang-up detection to detect unresponsive Genesys applications supporting this functionality. Users can then configure appropriate actions, including alarms if required.

To enable hang-up detection, use the configuration options **heartbeat-period** and **heartbeat-period-thread-class-<n>** to set the time interval in which a heartbeat message must be received before the application itself, or a thread of the application, is considered to be unresponsive for each application. A third option, **hangup-restart**, can be used to set the action that LCA takes when it deems the application to be non-responsive, either automatically restarting the application or just generating a notification of the situation.

For more information about these options, refer to the *Framework Configuration Options Reference Manual*.

> ### Warning
> Use this functionality with great care, and only with those applications for which support of this functionality has been announced. Failure to use it properly could result in unexpected behavior, from ignoring the options to an unexpected restart of the application.

To determine if your application supports hang-up detection, refer to application-specific documentation. Support by Management Framework components is indicated in the following table.

Component	Application Level	Thread Level	Hangup Thread	
Class	Name			
Configuration Server	Yes	Yes	1	`auth thread`
	Yes	Yes	1	`confserv_thread`
Solution Control Server	Yes	Yes	1	`mailer thread`
DB Server [a]	Yes	No	N/A	N/A
Message Server	Yes	Yes	1	`dbthread`
SNMP Master Agent	Yes	No	N/A	N/A

a. DB Server is no longer required by Management Framework starting in release 8.5.1, but can be used in place of the new internal database access system implemented in 8.5.x if desired. Refer to the *Framework Database Connectivity Reference Guide* for more information.

Important

- The option **hangup-restart** does not apply to Solution Control Server.
- The option **heartbeat-period-thread-class-<n>** does not apply to SNMP Master Agent.
- Solution Control Server must be running if the **hangup-restart** option is used.

Split Brain or No Brain State Detection

Split brain occurs when the state of both the primary and backup applications in an HA pair controlled by Solution Control Server (SCS) changes to primary, and no brain state can happen when one or both applications become unavailable or change state to backup. SCS generates the following log events to indicate the state of applications:

- 20621 - Indicates that an application is in "no brain" state.
- 20622 - Indicates that "split brain" condition has occurred for an application.
- 20623 - Indicates that an application is back to the normal state.

Important

These alarms might not work as expected if SCS is in the split brain state. Also, during switchover, the alarm is generated when the applications are either in primary/primary or backup/backup mode and is cleared automatically when the applications complete switchover.

You can create alarms using these log events to detect split brain or no brain state of applications. For more information on creating alarms using log events, see Using Log Events for Alarm Detection.

Detecting split-brain state of Solution Control Servers

Perform the following steps to detect if Solution Control Server (SCS) has entered the split-brain state:

1. Check if both primary and backup SCS are running.
2. Check if the following line is available in the logs of backup SCS:

   ```
   "SelfServer: Internal Change RUNMODE from 'BACKUP' to 'PRIMARY'".
   ```

   This log message indicates that the state of SCS has most likely entered the split brain state.

> **Important**
>
> Starting from Solution Control Server 8.5.100.38, the Log Event "10329" is generated when primary SCS disconnects from the backup SCS, indicating a possible SCS split brain state. The Log Event "10330" is generated when SCS returns to the normal state.

Perform the following checks to recover SCS from the split-brain state:

- Check and resolve if there are network connectivity issues between primary and backup SCS machines.
- Check whether DNS resolves quickly from the host where the primary SCS runs. If there is a delay in DNS resolution, troubleshoot the issue.
- Check whether there is any addp timeout triggering SCS to connect to host or resolve an FQDN which takes more time. In these cases, check the reason for addp timeout and increase the timeout value accordingly.

Remote Site Failures with Distributed Solution Control Servers

Starting in release 8.0, any Solution Control Server in the distributed environment can also detect the failure of a remote site controlled by another Solution Control Server in the environment and generate an appropriate log message.

Solution Control Server considers a remote site to have failed if it stops receiving polling messages from the Solution Control Server (or primary and backup Solution Control Servers, if configured) controlling the remote site within a specified time period. The time period is specified by the configuration option **alive_timeout**, configured on each Solution Control Server. Refer to the *Framework Configuration Options Reference Manual* for a detailed description of this option. In this

case, the primary Solution Control Server generates log event 43-20600.

If the remote site recovers and the Solution Control Server (or primary and backup Solution Control Servers, if configured) controlling that site starts to send polling messages, the primary Solution Control Server generates log event 43-20601 to indicate that the remote site is back in service.

For additional information, refer to the following documents:

- *Framework Deployment Guide* for more information about distributed Solution Control Servers, and for detailed instructions for configuring them
- *Framework Configuration Options Reference Manual* for a detailed description of the option **alive_timeout**
- *Framework Combined Log Events Help* for a full description of Solution Control Server log events 43-20600 and 43-20601.

Site Failure (Disaster Recovery)

Starting in release 8.5, Genesys software provides some Disaster Recovery/Business Continuity functionality, which enables you to continue operations and prevent data loss if the main site fails because of a natural, man-made, or unintended situation.

When the main site fails, the active Log Database is automatically replicated into the dormant Log Database installed at the remote site. The dormant Log Message Server is started and connects to that now-active Log Database. Another dormant Message Server is started and provides communication between the SCS at the failed site and the SCS at the remote site.

Refer to the *Framework Deployment Guide* for more information about this functionality.

SNMP Support Functions

The Management Layer provides you, as a network administrator, with a way to monitor and control Genesys applications when using an SNMP-compliant third-party network management systems (NMS) user interface. This functionality is available in two implementations:

- Built-in support using the Genesys SNMP Master Agent component.
- Net-SNMP, an open-source tool that is configured using the Genesys SNMP Master Agent Application object. Available starting in release 8.5.1, this implementation provides the same SNMP functionality as the built-in support of Genesys SNMP Master Agent.

Both implementations provide support for an SNMP-compliant NMS, which means that Solution Control Server not only converts Genesys alarms into SNMP traps, but also processes various NMS commands and generates SNMP traps based on changes in the current status of an individual application.

The following requirements apply to the components that integrate Genesys 7 or later with an SNMP-compliant third-party NMS:

- Solution Control Server must contain or be connected to SNMP functionality.
- An SNMP Master Agent application must be compliant with the AgentX protocol. Use either Genesys SNMP Master Agent (available on the Management Framework 8.5.0 or earlier product CD) or one provided by a third-party.
- The license file must contain licenses that enable the SNMP functionality of the Management Layer. Refer to the *Genesys Licensing Guide* for information about how to order licenses and set up the licensing system.

Depending on the type of NMS you are using, you may also need to modify it to ensure a successful integration. For example:

- Make sure that your NMS knows the communication port number of the SNMP Master Agent.
- If you use several SNMP Master Agent applications, make sure their communication port numbers are unique and are known to the NMS.

In addition, check documentation for your NMS to find out if:

- You must load a copy of the Genesys MIB file into NMS so that your NMS can monitor and control Genesys applications.
- You must modify your NMS as needed so that it can display and process SNMP traps that an SNMP Master Agent generates on behalf of Genesys applications, including SCS.

Monitoring and Control Functions

This chapter provides additional information and hands-on tips for using the following Management Layer monitoring and control functions, including information about involved applications and reference documents:

- Monitoring solutions, applications, and hosts
- Viewing system performance
- Managing 3rd-party applications
- Avoiding an unnecessary switchover
- Handling stuck calls
- Managing environments with redundant Configuration Servers, also referred to as high availablity (HA) configurations
- Managing environments that include one or more Configuration Server Proxies, also referred to as distributed Configuration Server configurations.
- Using the mlcmd Command-Line utility to assist you in monitoring and controlling your configuration.

How to Monitor Solutions, Applications, and Hosts

The monitoring function of the Management Layer allows a user to view the current runtime status of configured hosts, daemon applications, and entire solutions. The monitoring function requires the installation of:

- Solution Control Server (SCS)
- An instance of Local Control Agent (LCA) for each Genesys host computer
- Genesys Administrator

> **Important**
>
> For SCS to monitor an application, you must specify the name of the executable file of the application in the properties of the Application object.

Genesys Administrator

Starting in Management Framework release 8.0, you can monitor solutions, applications, and hosts through Genesys Administrator, a centralized Web-based user interface. The Dashboard, located on the **Monitoring** tab under **Environment**, displays the count and status of all applications and hosts that currently are configured. You can view the status of individual Solution, Application, and Host objects by selecting the appropriate category under **Provisioning > Environment**. Refer to *Genesys Administrator Help* for more information.

mlcmd Utility

Also starting in Management Framework release 8.0, you can also use the **mlcmd** command-line utility to view the status of hosts, application, and solutions. For detailed information about this utility, and instructions for using it, see How to Use the mlcmd Command-line Utility.

How to View System Performance

The monitoring function of the Management Layer allows a user to view the performance characteristics of configured hosts. System performance viewing requires the installation of the same components as for host monitoring. You can monitor a host computer that runs any Genesys-supported operating system as long as Local Control Agent is running on that computer.

Starting in release 8.5.1, you can also monitor how the growth of your configuration environment affects system performance by monitoring the number of configuration objects and identifying those objects with large configuration option sizes. This is done with database scripts that are run against the Configuration Database. For more information about these scripts, refer to "Monitoring Performance of Configuration Environment" in the *Framework Deployment Guide*.

How to Manage Third-Party Applications

In Genesys terms, a third-party application is an application not instrumented with Genesys libraries. This chapter describes which Management Layer functions you can use with third-party applications and how the Management Layer processes the related commands. It also lists the software prerequisites for and describes how to configure these applications.

Prerequisites

In Genesys terms, a third-party application is an application not instrumented with Genesys libraries. The Management Layer can monitor, start, and stop a third-party application as long as that application:

- Supports startup from a command line.
- Starts if the computer it runs on is unattended (for instance, on a Windows computer with no user logged in); however, this is not mandatory.
- Works without a console window on Windows; however, this is not mandatory.
- Is registered in the Configuration Database as an Application of the Third Party Server type.
- Runs on an operating system that Genesys supports.

> **Important**
>
> You cannot perform the centralized logging and alarm-signaling functions (including switchover) over a third-party application because they require built-in support on the application side.

Required Components

If you have configured third-party applications in the Genesys Configuration Database, Management Layer can control, monitor, start, and stop them. Even if you do not use the Management Layer to start a particular application, the application's runtime status is displayed. This functionality is also supported for:

- Third-party applications installed as Windows Services
- Third-party applications started with a script.

Managing third-party applications requires the installation of:

- Solution Control Server
- An instance of Local Control Agent (LCA) for each host computer running third-party applications
- Genesys Administrator

The monitoring views and control commands are available through Genesys Administrator, just as they are for managing Genesys applications. *Framework Genesys Administrator Help* provides detailed instructions for viewing the applications, and starting and stopping them.

Configuring Third-Party Applications

You must create an Application object for each third-party application that you want to use, and configure it appropriately. **[+] Show steps**

Purpose

To create and configure an Application object for a third-party application using Genesys Administrator.

Prerequisites

- Management Layer components are installed and running.
- The third-party application is installed.
- You are logged in to Genesys Administrator.

Start of procedure

1. Register the third-party application in the Configuration Database as an Application object of the Third Party Server type.
2. In the **Server Info** section of the Application object's **Configuration** tab, specify the following:
   - **Working Directory**— The full path to the directory from which the application starts.
   - **Command Line**— The command line used for starting the application; usually, it is the name of the executable file.
3. (Optional) If you want to start and stop the third-party application using the Management Layer, do the following:
   a. In the **Server Info** section of the Application object's **Configuration** tab, in the **Command Line Arguments** field, specify any additional parameters used to start the application.
   b. If the application is started with a batch file or script, specify the name of

the command used for launching that file or script. In the Annex list of the Application object's **Options** tab, create a section named **[start_stop]** and an option named **start_command**.

c. As a value for this option, specify the command that launches the batch file or script, including the full path to the executed file or script. (The **start_command** option may also contain the command to start the executable file.)

d. If the application is stopped with a batch file or script that performs the correct shutdown of the application, specify the name of the command used for launching that file or script. In the Annex list of the Application object's **Options** tab, create (or open) a section named **[start_stop]** and create an option named **stop_command**. For its value, specify the command that launches the batch file or script, including the full path to the executed file or script.

Monitoring Third-Party Applications

Monitoring functionality is provisioned by the ability of Local Control Agent to determine whether:

- A third-party application is started.
- A third-party application is stopped.

Determining Whether Applications Are Started

LCA uses the so-called command–line matching mechanism to determine if a third-party application is started. This means that LCA periodically retrieves a list of all currently running processes, and then compares command lines of all processes from that list with possible command lines of the third-party applications being checked.

LCA uses the following command–line matching rules for this comparison:

- The command line of a process is equal to the set of these elements:
  Working Directory + Command Line [+ Command Line Arguments]
  where **Working Directory**, **Command Line**, and **Command Line Arguments** are the properties of the Application object in the Configuration Database. If the **Command Line Arguments** property is empty, it is not used.
  The processes started with the full path specification are evaluated based on this rule.
- The command line of a process is equal to the set of these elements:
  Command Line [+ Command Line Arguments]
  where **Command Line** and **Command Line Arguments** are the properties of the Application object in the Configuration Database. If the **Command Line Arguments** property is empty, it is not used.
  The processes started without the full path specification are evaluated based on this rule.
- For Windows operating systems only, the command line of a process is equal to the set of these elements:
  + Working Directory + Command Line [+ Command Line Arguments]

where **Working Directory**, **Command Line**, and **Command Line Arguments** are the properties of the Application object in the Configuration Database. If the property **Command Line Arguments** is empty, it is not used.
The processes started with the full path specification are evaluated based on this rule when the path contains spaces.

If LCA finds a process whose command line matches that of a third-party application, LCA assumes that the application has started and then:

1. Stores the PID (process identifier) for that application.
2. Sets the application status to Started.
3. Sends a notification to SCS.

Determining Whether Applications Are Stopped

LCA uses the so-called PID-check mechanism to determine if a third-party application is stopped. This means that LCA tracks the PIDs (process identifiers) for all currently running processes. Using relevant operating system commands, LCA determines if a process with a particular PID is running. If not, LCA considers the corresponding third-party application stopped and:

1. Sets the application status to Stopped.
2. Sends a notification to SCS.

Starting Third-Party Applications

You can start a third-party application in the following ways:

- User command from Genesys Administrator
- Alarm Reaction
- Automatically at host start-up
- Auto-Restart

When the Management Layer receives a request to start a particular third-party application, SCS generates a command line and passes it to LCA, which executes the required operating system function.

SCS generates the command line based on the parameters you configured for a specific third-party Application object in the Configuration Database, which may include:

- **Working Directory**—as specified in the Application object's properties.
- **Command Line**—as specified in the Application object's properties.
- **Command Line Arguments**—as specified in the Application object's properties.
- Start Command—as specified by the **start_command** option of the Application object's Annex.

If you have not specified values for the first three listed parameters or have not created a

start_command option and provided a value for it, the Management Layer cannot start the application. For more information about these parameters, refer to Configuring Third-Party Applications.

Solution Control Server forms the command line as follows:

- If you have specified the **start_command** option, SCS uses its value to form the command line and ignores the other parameters.
- If you have not specified the start_command option, SCS uses the values of the **Command Line** and **Command Line Arguments** to form the command line, while LCA executes an appropriate operating system function in the directory specified as the **Working Directory** for this application.

LCA passes all required parameters to the operating system function (`CreateProcess` on Windows or execvp on UNIX) and calls the function, after which two scenarios can occur:

- The operating system function returns an error. In this case, LCA passes the error to SCS, which retains the Stopped status for the third-party application.
- The operating system function does not return an error. In this case, LCA determines the status of the third-party application and passes the status to SCS. (See Determining Application Status for a description of the methods LCA uses to determine the application status.)

Whichever scenario occurs, the startup process is then considered finished.

Starting Third-Party Applications Automatically

Management Layer supports the automatic start-up of third-party applications. You must be aware of, and correct if necessary, the configuration of the startup timeout for automatically started third-party applications.

The Management Layer must know the correct status of the third-party application at the exact moment when determining if the application is stopped and is to be started, that is, when the startup timeout for the third-party application expires.

Incorrect configuration of the automatic third-party application start-up configuration can cause multiple instances of the same application to be started.

Restarting Third-Party Applications Automatically

The Management Layer also supports the automatic restart of third-party applications following an unexpected termination. You must select **Auto-Restart** (in the Application's properties) when configuring the application. Note that the startup timeout is not used when restarting third-party applications.

Determining Application Status

The method LCA uses to determine the current status of a third-party application depends on the method SCS uses for forming the startup command line:

- If you have not configured the **start_command** option, SCS uses the values of the **Command Line** and **Command Line Arguments** to form the command line. In this case, LCA stores the PID returned by the operating system function and immediately passes the Started status to SCS.

- If you have configured the **start_command** option, SCS uses the value of this option to form the command line. In this case, LCA passes the Pending status to SCS and determines if the application has started successfully, as described in Determining Whether Applications Are Started.

Ensuring Command Line Correctness

If you want to monitor a third-party application, use the running process and its arguments as a model for the command line and command line arguments in Genesys Administrator. Follow these steps:

1. Use a system tool (for example, the UNIX tool ps) to display the running process and its arguments.
2. In the third-party Application object's properties, do the following:
   a. In the **Command Line** field, enter the exact value of the running process.
   b. In the **Command Line Arguments** field, enter the exact value of the running process arguments.

Stopping Third-Party Applications

You can stop a third-party application in the following ways:

- User command from Genesys Administrator
- Alarm Reaction

> **Important**
> The Shutdown Timeout does not apply to third-party applications.

The Management Layer can only stop a third-party application that has a status of Started; that is, LCA knows the PID for this application.

When the Management Layer receives a request to stop a particular third-party application, SCS passes it to LCA, which executes the required operating system function.

LCA processes the request as follows:

- If you have not configured the **stop_command** option and the application runs on UNIX, LCA sends the SIGINT signal to the process with the PID corresponding to the third-party application. Then, LCA sets the application status to Stopped and notifies SCS.
- If you have not configured the **stop_command** option and the application runs on Windows, LCA calls the TerminateProcess function for the process with the PID corresponding to the third-party application. Then, LCA sets the application status to Stopped and notifies SCS.
- If you have configured the **stop_command** option, LCA either executes the specified operating system command or launches the specified script or batch file. LCA sets the status of the third-party application to Pending and then determines the actual status of the application, which, when determined, it passes to SCS. (See Determining Whether Applications Are Stopped for a description of the methods LCA uses to determine application status.)

At this point, the process of stopping a third-party application is considered finished.

Example

To demonstrate how you can use the Management Layer to control a third-party application, such as License Manager, running on a Windows-based computer, do the following:

1. Install License Manager to directory **d:\flexlm**. Use the License Manager installation procedure for Windows described in the *Genesys Licensing Guide*.
2. Create two *.bat files, one (named **lmgrd_run.bat**) for starting License Manager and the other (named **lmgrd_stop.bat**) for shutting down the application. The file content should be as described in Start Script and Stop Script Content.
3. Save both files to the **d:\flexlm** directory.
4. Create an Application object of the Third Party Server type and name it **FLEXlm**. Refer to the configuration procedures in Configuring Third-Party Applications.
5. In the third-party Application object's **Startup Info** section, set the parameters as follows:
   - Specify `d:\flexlm` as the value for **Working Directory**.
   - Specify `lmgrd` as the value for **Command Line**.
   - Specify `-c d:\flexlm\license.dat` as the value for **Command Line Arguments**.

   > **Important**
   >
   > Make sure that the combined string
   >
   > *<Working directory> + <Command Line> + <space> + <Command Line Arguments>*
   > matches the command line in the **lmgrd_run.bat** file, which is
   >
   > `d:\flexlm\lmgrd -c d:\flexlm\license.dat`.

6. Start **lmgrd_run.bat** manually. After 20 or so seconds, check the Application object's status. Its status should be Started.
7. In the FLEXlm Application object's Annex:
   a. Create a section called `start_stop`.
   b. Create two options, **start_command** and **stop_command**, in this new section.
   c. Specify full paths to the appropriate *.bat files as the option values:
      start_command = d:\flexlm\lmgrd_run.bat
      stop_command = d:\flexlm\lmgrd_stop.bat
   d. Save configuration changes.
8. Try to stop and start the FLEXlm application using appropriate commands in Genesys Administrator.

Start Script and Stop Script Content

The content of the **lmgrd_run.bat** and **lmgrd_stop.bat** files depends on whether you run License Manager as a regular application or as a Windows Service.

For a regular application, the file content should be as follows:

- **lmgrd_run.bat** `@echo "Starting FLEXlm License Manager" d:\flexlm\lmgrd -c d:\flexlm\license.dat`
- **lmgrd_stop.bat** `@echo "Stopping FLEXlm License Manager" d:\flexlm\lmutil lmdown -q -c d:\flexlm\license.dat`

For a Windows Service, the file content should be as follows:

- **lmgrd_run.bat** `net start <FLEXlm Service Name>`
- **lmgrd_stop.bat** `net stop <FLEXlm Service Name>`

How to Avoid an Unnecessary Switchover

You can minimize the chance that a network problem causes a switchover between a functioning primary server and its backup. When disconnected from an LCA running on any host, SCS initiates a switchover for all applications running on the same host with the LCA. However, the disconnect can result from either long-term issues (such as the host being down or LCA terminating) or temporary issues (such as slowness of the network or a temporary network problem).

You can configure SCS to verify the connection status in a few seconds to confirm whether the connection issue is resolved. To do so, create the **disconnect-switchover-timeout** option in the `general` section on the **Options** tab of the SCS properties. Set the option value to any positive integer, which means the number of seconds, and depends on your typical network conditions. When SCS initiates a switchover process, it waits the interval you specified and checks the LCA connection:

- If the problem has been temporary, the connection is restored and the applications on the LCA host are in Running status. In this case, SCS does not perform a switchover.
- If the problem is serious, the LCA remains disconnected and the status of the applications on the LCA host is unknown. In this case, SCS proceeds with the switchover.

> **Important**
>
> The **disconnect-switchover-timeout** option setting has no effect on a manual switchover, a switchover resulting from an alarm reaction, or a switchover resulting from service unavailability at the primary server.

Refer to the *Framework Configuration Options Reference Manual* for detailed option descriptions.

How to Handle Stuck Calls

This topic describes the procedures related to detecting and ways of dealing with calls that appear to be stuck.

A stuck call occurs when information about the release of a call in the communication system fails to reach one or more of the components of a CTI solution. One possible cause is the temporary loss of communication between CTI applications and devices, such as switching and interactive voice response systems, in the communication infrastructure.

Having missed the call release information, CTI applications continue to treat such calls as active, which results in less efficient operation and inaccurate reporting.

Because T-Servers are directly involved in communications with the switching systems, they play a critical role in detecting and handling stuck calls.

Which Method To Use?

Stuck calls can be handled by any of three methods:

1. Using T-Server switch-specific functionality
2. Using the SNMP interface in the Management Layer
3. Using the gstuckcalls utility in the Management Layer

T-Server Switch-Specific Functionality

This method offers stuck calls detection and cleanup built-in to T-Server.

This is the basic form of using the stuck call feature in T-Server that provides for minimal customization and provisioning from the user's perspective.

Characteristics:

- A simple, timeout-based detection mechanism is used internally in T-Server.
- This method does not utilize Management Layer capabilities—no automatic reactions to be executed upon detection of stuck calls.
- You must set up and manage each T-Server manually and individually, using Genesys Administrator.
- Unwieldy for managing multiple T-Servers.

See Using T-Server.

SNMP Interface in the Management Layer

This method provides an SNMP interface, good for SNMP-based installations and in an SNMP-oriented application.

Characteristics:

- Relies on external SNMP-aware applications (such as SNMP Perl scripts) to monitor and detect stuck calls in T-Server.
- Stuck call detection logic is highly customizable; information such as filters and timestamp properties lies in the new SNMP tables.
- The script provides for a central point of management, and can be tailored to manage a single or multiple T-Server applications.
- Does not utilize the Management Layer capability to monitor and react to alarm events; the script must do it.
- Requires SNMP licensing.

See Using the SNMP Interface.

gstuckcalls Utility

This method has the advantages of the second method but does not require SNMP.

Characteristics:

- The stuck call detection logic is highly customizable.
- This method is integrated with the Management Layer. A stuck calls event can be configured and reacted upon when corresponding log messages are received by SCS.
- This method does not require an SNMP license.
- The scripts in this utility generate an XML file with a summary of all calls retrieved from T-Server, which makes it useful as a quick-look diagnostic tool.

The **gstuckcalls** utility uses the **logmsg** utility, which allows sending specified log messages to trigger and clear Management Layer alarms based on conditions external to the Management Layer.

See Using the gstuckcalls Utility.

Prerequisites

The Stuck Calls functionality requires that Perl be installed on the SCS host computer. The following table lists the names and minimum versions of Perl extension modules required. Users may need to install some or all of them, depending on their current Perl installation. **[+] Show table**

Perl Module	Recommended Version
HTML::Parser	3.25 and higher
SOAP::Lite	0.60 and higher
XML::DOM	1.43 and higher
XML::Parser	2.34 and higher
XML::SAX	0.12 and higher
XML::NamespaceSupport	1.08 and higher
XML::RegExp	0.03 and higher
HTTP::Cookies	1.39 and higher

These modules are available from the Comprehensive Perl Archive Network (CPAN) website (http://www.CPAN.org).

The Framework stuck calls functionality—including the Perl scripts **GStuckCallsDetect.pl** and **GStuckCallsClear.pl**, and the above modules—were tested using Perl version 5.6.1.

Using T-Server

To support stuck calls handling in T-Server, a set of configuration options has been introduced. These options control stuck call detection, notification, and automatic cleanup. For more information, refer to the "T-Server Common Configuration Options" chapter in the latest version of the Deployment Guide for your T-Server.

To support stuck calls handling in the Management Layer, a set of log messages have been added to the T-Server Common Part. Download the *Framework Combined Log Events Help* for more information.

To support stuck calls handling in client applications of T-Server, a new property has been added to the T-Server events that define the end of the call (EventReleased and EventAbandoned). See the *Genesys Events and Models Reference Manual* for more information.

Based on a specified timeout, T-Server waits for call information being updated. After the timeout is expired, T-Server considers a call as a stuck call and reports a standard log message.

Processing of timeouts and notifications is implemented in the T-Server Common Part, but the actual call cleanup involves interaction with the switch-dependent part for each T-Server.

Configuration Options Summary

Three new options must be configured in the section call-cleanup.

- **notify-idle-tout**: This option specifies the time interval that T-Server waits for a call being 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.
- **cleanup-idle-tout**: This option specifies the time interval that T-Server waits for a call being 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 call information from memory unconditionally. The option description in the Deployment Guide for each T-Server reflects the actual implementation for that particular T-Server.

- **periodic-check-tout**: This option specifies the time interval for periodic checks for stuck calls (affects both notification and cleanup functionality) by checking the T-Server's own call information with call information available in the switch. For performance reasons, T-Server does not verify whether the **notify-idle-tout** or **cleanup-idle-tout** option has expired before performing this checking.

T-Server Common Log Events

Three T-Server common log events support stuck calls management: `01-20020`, `01-20021`, and `01-20022`. Refer to the latest *Framework Combined Log Events Help* for detailed specifications of these log events.

EventReleased on special DN

The value of the **TReliability** parameter indicates that the update was forced by external request:

- `TReliabilityExternal = 3`
- `TReliabilityExternal - cleared by an external SNMP request`

Using the SNMP Interface

The following tables in the T-Server-Specific SNMP Objects support management of stuck calls using the SNMP Interface in the Management Layer. These tables allow you to retrieve only those call instances that were defined by the filters in the tsCallFilterTable table thus reducing network traffic and increasing application performance.

- tsCallFilterTable provides the interface for setting call filter criteria for the tsCallInfoTable table. It also provides the interface for clearing calls by the call's Connection ID.
- tsCallInfoTable stores the latest snapshot of active calls from a given T-Server, and contains information about active calls filtered by conditions set in the tsCallFilterTable.

Using the gstuckcalls Utility

The **gstuckcalls** utility contains two stuck calls management scripts, **GStuckCallsDetect.pl** and **GStuckCallsClear.pl**, which support the detection and automatic clearing of stuck calls. Both scripts use the **gstuckcallsscript.cfg** configuration file, also part of the utility.

> **Important**
>
> When using the **gstuckcalls** utility, the password is saved as plain text in the configuration file and this file is usually not encrypted. Therefore, Genesys recommends to carefully guard the configuration file containing the passwords.

How to Install the Utility

If you installed Solution Control Server 8.5.100.41 and earlier, the **gstuckcalls** utility containing the stuck calls scripts and configuration file is already installed, and is located in the same folder in which Solution Control Server was installed.

Staring with SCS 8.5.100.42 release, the **gstuckcalls** utility is not installed by default with Solution Control Server. You can install it separately by following the instructions in the section Installing the Solution Control Server Utilities of the Framework Deployment Guide. After the utilities are installed, the **gstuckcalls** utility is stored in the location that you specified during the installation.

Starting in 8.1.2, you can install the Solution Control Server utilities without installing Solution Control Server itself.

How to Run the Utility

To start the stuck calls utility:

- In Windows, enter **gstuckcalls.exe** on the command line.
- On UNIX, enter **gstuckcalls** on the command line.

The utility will automatically run the scripts.

GStuckCallsDetect.pl Script

The **GStuckCallsDetect.pl** script performs these functions:

1. Retrieves all the information about all T-Servers from the configuration.
2. Queries each T-Server for stuck calls according to the specified filter using the **gstuckcalls** utility.
3. If stuck calls are found, sends log message `0095000` on behalf of the T-Server.
4. If stuck calls are not found, sends log message `095010` on behalf of the T-Server.

If you require alarming for stuck calls, schedule this script for periodic execution by using tools provided by your operating system, such as Scheduled Tasks for Windows and Cron for UNIX.

GStuckCallsClear.pl Script

The **GStuckCallsClear.pl** script clears stuck calls in the specified T-Server. If you need to clear stuck calls automatically, use this script as an alarm reaction for the active alarm Stuck Calls Detected. This script performs the following:

1. Connects to the specified T-Server.
2. Uses the **gstuckcalls** utility to clear all stuck calls according to the specified filter.

gstuckcallsscript.cfg Configuration File

The **GStuckCallsDetect.pl** and **GStuckCallsClear.pl** scripts use this configuration file. It has the following format:

```
[cfgserver]
host=<host>
port=<port>
username = <username>
password = <password>
[msgserver]
host=<host>
port=<port>
[filter]
createdbefore=<seconds>
createdafter=<seconds>
updatedbefore=<seconds>
updatedafter=<seconds>
```

Stuck Calls Alarm Log Messages

The following alarm log messages have been added to support detection and automatic clearance of stuck calls:

- **09500:** Stuck calls detected
- **09501:** Stuck calls not detected

Configuring the Alarm Condition

To enable automatic stuck calls detection, configure the corresponding Alarm Condition with the following settings:

For Detect Event:

- Log Event ID set to `9500`
- Selection Mode set to `Select By Application Type`
- Type set to `T-Server`

For Cancel Event:

- Log Event ID set to `9501`

See Using Log Events for Alarm Detection for more information.

The active alarm Stuck Calls Detected is communicated when log message `9500` is received. This happens when the **GStuckCallsDetect.pl** script detects stuck calls in a T-Server. Scheduling the **GStuckCallsDetect.pl** script for periodic execution (for instance, once per day) ensures automatic stuck calls detection and alarming.

To clear stuck calls automatically, follow these steps:

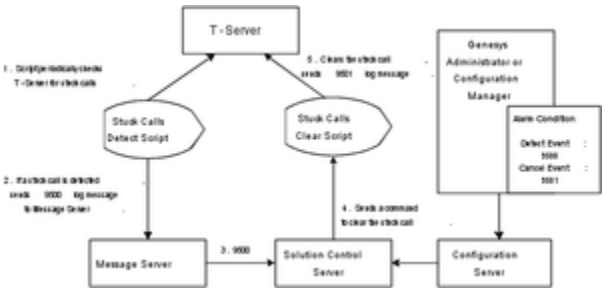
1. Configure the alarm reaction type Execute OS Command for the Alarm Condition Stuck Calls Detected.

2. Configure this alarm reaction to execute the **GStuckCallsClear.pl** script.

The script clears stuck calls at the corresponding T-Server and sends log message 9501 to the Management Layer, which then clears the active alarm Stuck Calls Detected.

Stuck Calls Scripts Flow Chart

The following figure provides a flow chart to help you better understand the scripts.



Stuck Calls Script Flowchart

How to Manage Environments with HA Configuration Servers

The Management Layer is designed to operate with one running Configuration Server (one primary-backup pair of Configuration Servers). That is, you must connect the Management Layer components and all the applications that it controls to the same Configuration Server.

If you need to use the Management Layer capabilities in an environment with two running Configuration Servers (two primary-backup pairs of Configuration Servers), you must use an independent Management Layer installation for each running Configuration Server (each primary-backup pair). To make two Management Layer installations independent when a host computer serves two configurations, install two Local Control Agent applications on such a computer, one LCA communicating to one Management Layer installation and the other LCA communicating to the other, and make the LCA port numbers unique.

How to Manage Environments with Configuration Server Proxies

The Management Layer fully supports the distributed configuration environment available when using Configuration Server Proxy.

The term *Configuration Server Proxy* is used to identify a Configuration Server instance running in so-called *Proxy* mode. Refer to the *Framework Deployment Guide* for more information.

This support means that the Management Layer:

1. Displays the current real-time status of Configuration Server Proxy and performs its startup, shutdown, or automatic switchover to the backup Application just as for any other Genesys Application.

   > **Important**
   > You cannot manually switch over Configuration Server Proxy Applications.

2. Correctly starts Applications that are clients of Configuration Server Proxy.

When receiving a request to start an Application, the Management Layer determines whether the Application is configured as a client of Configuration Server or of Configuration Server Proxy. For this purpose, the Management Layer checks the list of connections configured for an Application.

The Application is considered a client of Configuration Server Proxy when both of these conditions are met:

- The Application's configuration contains a connection to an Application of the Configuration Server type.
- In its turn, the Application of Configuration Server type contains in its configuration a connection to another Application of the Configuration Server type.

The Application is considered a client of Configuration Server when either of these conditions is met:

- The Application's configuration contains no connection to an Application of the Configuration Server type.
- The Application's configuration contains a connection to an Application of the Configuration Server type, but this latter Application's configuration does not contain a connection to an Application of the Configuration Server type.

> **Important**
> Genesys recommends that you configure connections to Configuration Server for

Applications that are clients of Configuration Server in an environment with Configuration Server Proxy.

If the Management Layer finds the Application to be a client of Configuration Server, the Management Layer uses the Configuration Server parameters to start the Application. For more information, see Processing the Start Command for Applications.

If the Management Layer finds the Application to be a client of Configuration Server Proxy, the Management Layer also checks the configuration to determine whether a backup Application is configured for this Configuration Server Proxy:

- If no backup Application is configured, the stand-alone Configuration Server Proxy is considered to be running in Primary mode.
- If a backup Application is configured, the Management Layer identifies which Configuration Server Proxy of the primary-backup redundancy pair is currently running in Primary mode.

Then, the Management Layer starts the Application that is a client of Configuration Server Proxy. SCS generates a command line and passes it to LCA, which executes the command. The command line contains:

- The Application’s working directory as specified in the Application object’s properties.
- The host name of Configuration Server Proxy currently running in Primary mode.
- The port number of Configuration Server Proxy currently running in Primary mode.
- The Application name as specified in the Application object’s properties.

Important

Make sure that Configuration Server Proxy is running during its client startup.

How to Use the mlcmd Utility

You can use the **mlcmd** command-line utility to :

- Query the status of hosts, applications, or solutions.
- Start, stop, and gracefully stop applications and solutions.
- Send a custom command to an application.

Installing the Utility

If you installed Solution Control Server 8.5.100.41 and earlier, the **mlcmd** utility is already installed, and is located in the same folder in which Solution Control Server was installed.

Staring with SCS 8.5.100.42 release, the **mlcmd** utility is not installed by default with Solution Control Server. You can install it separately by following the instructions in the section "Solution Control Server Utilities' of the *Framework Deployment Guide*. After the utilities are installed, the **mlcmd** utility is stored in the location you specified during the installation.

Starting in 8.1.2, you can install the Solution Control Server utilities without installing Solution Control Server itself.

Using the Utility

All **mlcmd** command parameters are made in a single command. The general syntax is as follows:

```
mlcmd <mandatory parameters> <optional parameters> <command parameter>
```

You must provide all the mandatory parameters and one operation parameter. The parameters are listed in the following table. Starting in release 8.1, you must authenticate yourself with **mlcmd** by logging in to the utility. If authentication is successful, you can use the utility as part of operations. The parameters are listed in the following table.

> **Important**
> The **mlcmd** utility does not support issuing the `switchapp` command to Configuration Server Proxy.

[+] Show table

Parameter	Description
COMMON PARAMETERS	
-help	Prints the version of the utility and its usage.
-cshost *hostname*	Mandatory. Configuration Server host name.
-csport *portnumber*	Mandatory. Configuration Server port number.
-csuser *username*	Mandatory. Username of user.
-cspassword *password*	Mandatory. The password of the user.
-csappname *application_name*	Mandatory. Application name of the interface used to access Management Framework (that is, Genesys Administrator or Genesys Administrator Extension).
-scshost *SCS_host_name*[a]	Optional. Solution Control Server host name.
-scsport *SCS_port_name*[a]	Optional. Solution Control Server port number.
-timeout *seconds*	Optional. Specifies the amount of time (in seconds) that the utility will wait for SCS to perform the requested action. If SCS does not respond within the specified time, the utility will return an error. Minimum value permitted = 10 seconds. Default value = 60 seconds. If the specified value is less than 10, the command will use the default value.
-secure	Optional. Specifies that a secure connection should be used by clients when connecting to SCS.
-cert *certificate*	Optional. Use only if -secure is used. On Windows, specifies the security certificate thumbprint; on UNIX, specifies the path to the host's security certificate.
-key *key*	Use only if **-secure** is used. On UNIX, specifies the path to the file with the private key; not used on Windows.
-ca-cert *ca-cert*	Use only if **-secure** is used. On UNIX, specifies the path to the file with the CA certificate; not used on Windows.
COMMAND PARAMETERS	
WARNING! Specify only one of the following command parameters and any associated sub-parameters in a single invocation of the utility command.	
-getallalarms	Requests the object type, DBID, status, DBID of the Alarm Condition, time it was triggered, time it expires, and message text of all active alarms.
-getallappstatus	Requests the DBID, status, and runmode of all applications.
-getappstatus *Application Name* | *DBID* [-usedbid]	Requests the status of the application specified by *Application Name* (the default), or the *DBID* if usedbid is specified.
-getappstatus-runmode *Application Name* | *DBID* [-	Requests the status and runmode of the application

Parameter	Description
usedbid]	specified by **Application Name** (the default), or the **DBID** if usedbid is specified.
-gethoststatus *Host Name* | *DBID* [-usedbid]	Requests the status of the host specified by *Host Name* (the default), or the *DBID* if usedbid is specified.
-getsolstatus *Solution Name* |*DBID* [-usedbid]	Requests the status of the solution specified by *Solution Name* (the default), or the *DBID* if usedbid is specified.
-clear-app-alarms *Application Name* | *DBID* [-usedbid]	Clears all active alarms raised by the Application specified by the *Application Name* (the default), or the *DBID* if usedbid is specified.
-clear-cond-alarms (-dbid *Alarm Condition DBID*) | (-name *Alarm Condition Name*)	Clears all active alarms raised on behalf of the Alarm Condition with a DBID specified by *Alarm Condition DBID* or with the name specified by *Alarm Condition Name*.
-startapp *Application Name* | *DBID* [-usedbid]	Starts the application with the specified *Application Name* (the default), or the *DBID* if usedbid is specified.
-start-initapp *Application Name* | *DBID* [-usedbid]	Starts the application with the specified *Application Name* (the default), or the *DBID* if usedbid is specified, without waiting for the status to change to Initializing/Running.
-stopapp *Application Name* | *DBID* [-usedbid]	Stops the application with the specified *Application Name* (the default), or the *DBID* if usedbid is specified.
-switchapp *Application Name* | *DBID* [-usedbid]	Switches over the application with the specified *Application Name* (the default), or the *DBID* if usedbid is specified.
-stopapp-graceful *Application Name* | *DBID* [-usedbid]	Gracefully stops the application with the specified *Application Name* (the default), or the *DBID* if usedbid is specified.
-startsol *Solution Name* | *DBID* [-usedbid]	Starts the solution with the specified *Solution Name* (the default), or the *DBID* if usedbid is specified.
-stopsol *Solution Name* | *DBID* [-usedbid]	Stops the solution with the specified *Solution Name* (the default), or the *DBID* if usedbid is specified.
-stopsol-graceful *Solution Name* | *DBID* [-usedbid]	Gracefully stops the solution with the specified *Solution Name* (the default), or the *DBID* if usedbid is specified.
-get-app-performance {*Application Name* | *DBID* [-usedbid]} [-result *xml file name*]	Requests information about a given process of an application with the specified *Application Name* (the default), or the *DBID* if usedbid is specified. The information, including CPU Usage for each thread, is stored in an XML file named *application name*_Performance_*time stamp*.xml, if *xml file name* is not specified, or <xml file name>.xml if it is.
CUSTOM COMMAND PARAMETERS	
Name | *DBID* [-usedbid]	Sends custom commands to the application with

Parameter	Description
	the specified *Application Name* (the default), or the *DBID* if usedbid is specified. Must be used with one or both of the **-custom-subcommand** and **-custom-data parameters**.
-custom-subcommand *subcommand* where *subcommand* is a number -custom-data *path* where *path* is the path to a data file	Use only with the **-custom-command** option. Both of these options specify the content of a custom data packet as follows: • If the **-custom-subcommand** option is used, the *subcommand* value is converted to a 4 B value (network byte order) and added to the custom data packet. • If the **-custom-data** option is used, the contents of the file given by *path* is added to the custom data packet. The custom data packet is then sent to the application with a custom command.
-custom-response *path*	Use only with the **-custom-command** option. Specifies the path to the file in which the response to the custom command is to be stored. If this parameter is not used, the response is discarded.
-custom-print-response	Use only with the **-custom-command** option. Specifies that the first 4 B of the custom response must be converted from network byte order into a decimal format and sent to the standard output.

[a] If **-scshost** and **-scsport** are not specified, and if authentication is successful, **mlcmd** will retrieve the SCS connection parameters from Configuration Server. If there is more than one SCS, the utility will select the first SCS it finds, and retrieve the host and port information from the application object for that SCS.

Utility Output

For all parameters, the utility returns a numeric code when it has finished, regardless of whether execution was successful. This code can then be used in downstream processing as necessary. See Return Codes for a full list of return codes.

If any errors occur when processing this utility, a log message is generated and sent to **stderr**. Output is never sent to **stdout** unless the **-help** or **-custom-print-response** parameters are specified.

The parameter **get-app-performance** enables you to output CPU Usage data in an XML file. See XML Data Output for Information Query.

Return Codes

A zero-value (0) or a positive two-digit return code indicates that processing was completed successfully. If the command included one of the parameters used to retrieve the status of a host, application, or solution, the return code indicates the status. A negative return code (or on UNIX, a positive value in the range of 247 to 255) indicates that processing did not complete successfully. Success and failure return codes are given in the following tables.

[+] Show tables

Success Codes Returned by mlcmd Utility

Parameter Used	Code	Description
-gethoststatus	0	Host status is UNKNOWN
	1	Host status is DISCONNECTED
	2	Host status is RUNNING
	3	Host status is UNAVAILABLE
	4	Host status is UNREACHABLE
-getappstatus	0	Application status is UNKNOWN
	1	Application status is STOPPED
	2	Application status is STOP_TRANSITION
	3	Application status is STOP_PENDING
	4	Application status is START_TRANSITION
	5	Application status is START_PENDING
	6	Application status is RUNNING
	7	Application status is INITIALIZING
	8	Application status is SERVICE_UNAVAILABLE
	9	Application status is SUSPENDING
	10	Application status is SUSPENDED
-getsolstatus	0	Solution status is UNKNOWN
	1	Solution status is STOPPED
	2	Solution status is STOP_PENDING
	3	Solution status is START_PENDING
	4	Solution status is STOP_TRANSITION
	5	Solution status is START_TRANSITION
	6	Solution status is RUN_PENDING

	7	Solution status is RUNNING
	8	Solution status is STARTED
-getappstatus-runmode	0	Execution completed successfully
	The code returned by -getappstatus-runmode is a combination of the application status and the runmode. See the following table "Codes Returned Using -getappstatus-runmode Parameter" to separate the two elements.	
All others not mentioned previously	0	Execution completed successfully

Codes Returned Using -getappstatus-runmode Parameter

	0	1	2	3	4	5	6	7	8	9	10
RunMode	**Application Status (see -get_appstatus in the table above)**										
0 (PRIMARY)	0[a]	1	2	3	4	5	6	7	8	9	10
1 (BACKUP)	32	33	34	35	36	37	38	39	40	41	42
2 (EXIT)	64	65	66	67	68	69	70	71	72	73	74

[a] Execution completed successfully.

Error Codes Returned by mlcmd Utility

All but UNIX	UNIX	
Code		**Description**
-1	255	Cannot connect to Configuration Server and/or SCS.
-2	254	Unexpected disconnection from Configuration Server and/or SCS.
-3	253	Timeout expired.
-4	252	No specified object found.
-5	251	Management Layer cannot execute the operation.
-6	250	Incorrect parameters specified.
-7	249	Internal utility error.
-8	248	Command execution failed.
-9	247	User does not have correct permissions to execute the command.

XML Data Output for Information Query

When you use the **get-app-performance** command, the results are stored in an XML file named as

follows:

- Default: **<*application name*>_Performance_<timestamp>.xml**
- User specifies name in command: **<*User-specified name*>.xml**

The format of the record in the XML file is as follows:

[+] Show format

```
<AppName>                          <!—The Application name-->
  <getprocessinfo>
    <process>
      <pid>94236</pid>
      <execname>MessageServer.exe</execname>
      <vmsize>4243456</vmsize>
      <pctcpu>0.00</pctcpu>
      <priority>normal</priority>
      <cmdline>cmdline</cmdline>
      <threads>
      <!-- This section can be absent if application is on host with old version of LCA -->
        <thread>
          <tid>1st thread ID</tid>
          <pctcpu>1st thread CPU usage</pctcpu>
        </thread>
        <thread>
          <tid>2nd thread ID</tid>
          <pctcpu>2nd thread CPU usage</pctcpu>
        </thread>
        ...
        <thread>
          <tid>Nth thread ID</tid>
          <pctcpu>Nth thread CPU usage</pctcpu>
        </thread>
      </threads>
    </process>
  <getprocessinfo>
  <ProcSizeInBytes/>
</AppName>
```

Starting and Stopping Applications and Solutions

This section gives hands-on tips on how to start and stop applications and solutions using the Management Layer, including information about the applications involved and reference documents:

- How to start and stop Applications and Solutions manually and automatically
- Using startup files

How to Start and Stop Applications and Solutions

With the control function of the Management Layer, you can start and stop individual applications with a single control function through Genesys Administrator. You can also use the control function to start and stop solutions, with one exception: you must use Genesys Administrator to start and stop a solution of type Default Solution Type or Framework.

Starting with release 8.0, you can also stop applications and solutions gracefully, also called *Graceful Stop* or graceful shutdown. See Graceful Stop for more information.

The control function requires the installation of:

- Solution Control Server (SCS)
- An instance of Local Control Agent (LCA) for each Genesys host computer
- Genesys Administrator

> **Important**
> Genesys recommends that you start and stop entire solutions as opposed to starting and stopping single applications.

Starting and Stopping Applications and Solutions Manually

To manually start and stop applications and solutions, use the start and stop commands in Genesys Administrator, locate as follows:

- **Start**, **Stop**, and **Graceful Stop** buttons on the toolbar
- **Start**, **Stop**, and **Graceful Stop** commands in the **Tasks** panel

Genesys Administrator Help provides detailed instructions on how to start and stop solutions and applications.

Starting in Management Framework release 8.0, you can also use the **mlcmd** command-line utility to start and stop applications and solutions. For detailed information about this utility, and detailed instructions for using it, see How to Use the mlcmd Utility.

Starting Applications Automatically

Solution Control Server starts applications without a user command (that is, automatically) when:

- SCS starts
- The computer running the applications is restarted

If you do not modify an Application object as described in this section, SCS automatically starts the application only at the application's host restart, given that the application has been running prior to the host restart.

To enable this, you must configure the **autostart** option for each application, as described in the following procedure. **[+] Show steps**

Prerequisites

- The Configuration Layer has been installed and configured, and is running.
- Either the Application object that you want to configure already exists, or you are performing this procedure while you are configuring it.
- You are logged in to Genesys Administrator.

Start of Procedure

To enable SCS to start an application automatically every time SCS establishes a connection with LCA and the latter does not report the Started status for the application:

1. In Genesys Administrator, go to **Provisioning > Environment > Applications**, and double-click the application to open its **Configuration** tab.
2. Click the **Options** tab, and select **Advanced View (Annex)** from the **View** drop-down list.
3. If there is a section called `sml`, select it.
4. Click **New**.
5. In the **New Option** dialog box, specify the following:
   a. In the **Section** field, type `sml`, unless it is already displayed.
   b. In the **Name** field, type `autostart`.
   c. In the **Value** field, type `true`.
6. Click **OK**.
7. Click **Save and Close**.

To disable an automatic application startup in scenarios in which SCS starts or in which an application was not running prior to its computer restart, either delete the **autostart** option or set its value to `false`.

What Happens During Start and Stop?

Processing the Start Command for Applications

When the Management Layer receives a request to start a particular application, SCS generates a command line and passes it to LCA, which executes the command. The command line contains:

- The working directory of the application as specified in the Application object's Properties.
- The host name and port number of Configuration Server currently running in Primary mode.

> **Important**
>
> If you are using Configuration Server Proxy, SCS substitutes the host and port parameters of Configuration Server Proxy, where appropriate. For more information, see How to Manage Environments with Configuration Server Proxies

- The application name as specified in the Application object's Properties.

Be sure that the working directory, executable (or startup) file name, application name, and startup timeout parameters are specified correctly in the Application object's Properties; otherwise, the Management Layer will be unable to start the application.

Unless an application is explicitly configured with a connection to Configuration Server Proxy, SCS starts the application against the Configuration Server to which SCS is currently connected. The port for connection to Configuration Server which SCS provides to the application is determined as follows:

- If the application is configured with a connection to Configuration Server, SCS starts the application using the same PortID as that configured for the connection between the application and Configuration Server.
- If the application is not configured with a connection to Configuration Server, SCS starts the application using **PortID = default**.

Processing the Start Command for Solutions

When the Management Layer receives a request to start a particular solution, SCS via LCA starts all applications included in the solution, in the order specified in the Solution object. The solution is considered Started when all mandatory applications that belong to it or their backup applications start successfully. When a mandatory solution component does not start, SCS determines if the solution configuration contains a backup server configured for this application. Then, one of the following happens:

- If the solution configuration does not contain a backup server, SCS interrupts the solution startup

procedure and the solution status remains Stopped.

- If the solution configuration contains a backup server, SCS attempts to start the backup application:
  - If successful, the solution startup procedure continues through completion. Then, the Management Layer applies the restart mechanism to applications that could not start.
  - If unsuccessful, SCS interrupts the solution startup procedure and the solution status remains Stopped.

Note that after starting a mandatory application, SCS attempts to start a backup server configured for this application. This only happens when the backup server is included in the same solution. If the backup server application does not start while the primary server application is running, SCS proceeds by starting the next mandatory component.

At SCS Startup

When SCS starts, it:

1. Establishes connections with all LCAs in the system and receives current statuses of all configured applications.
2. Checks the applications' configurations in the Configuration Database to determine whether you have enabled the **autostart** option.
3. For applications that have Stopped status and have the **autostart** option enabled, SCS:
   a. Waits for the interval specified in the **Startup Timeout** property of a particular Application object.
   b. Checks whether the application's status changes after the timeout has expired. If not, SCS starts the application as described in Processing the Start Command for Applications.

At Computer Restart

When a computer restarts, and on which Management Layer–controlled applications are installed, SCS:

1. Establishes a connection with the LCA running on that computer.
2. For applications that were running (that is, had Started or an equivalent status) prior to the computer restart, SCS:
   a. Waits for the interval specified in the **Startup Timeout** property of a particular Application object.
   b. Checks whether the application's status changes after the timeout has expired. If not, SCS starts the application as described in Processing the Start Command for Applications.
3. Identifies applications that were not running (that is, had Stopped status) prior to the computer restart.
4. Checks the applications' configurations in the Configuration Database to determine whether you have enabled the **autostart** option (using the procedure in Starting Applications Automatically).
5. For applications that have Stopped status and have the `autostart` option enabled, SCS:
   a. Waits for the interval specified in the **Startup Timeout** property of a particular Application object.
   b. Checks whether the application's status has changed after the timeout expires. If not, SCS starts the application as described in Processing the Start Command for Applications.

As a result, both the applications that were running prior to a computer restart and the applications that were not running but whose configuration contained the **autostart** option set to `true` are started automatically after you restart a computer.

Starting Third-Party Applications Automatically

Management Layer supports the automatic start-up and restart of third-party applications as described above. However, control and monitoring of third-party applications is performed in a different manner than that for Genesys applications. For details about how third-party applications are controlled, refer to How to Manage Third-Party Applications.

Graceful Stop

When you stop an application or a solution, it shuts down, ceasing all processing immediately. This can have a detrimental effect on the rest of the system. Starting with release 8.0, you can stop an application or a solution gracefully in a manner known as a *graceful stop*, or *graceful shutdown*.

> ### Important
> You must use Genesys Administrator to stop solutions of type Framework or Default Solution Type.

Applications that are being stopped gracefully refuse any new requests, but continue to process their current requests. Applications are stopped only after they have finished processing all of their requests.

The graceful stop command can be issued to any application. Applications that support this functionality process the command as described in the preceding paragraph. Applications that do not support the graceful stop functionality are stopped ungracefully.

For each application, you can specify a timeout with the **suspending-wait-timeout** configuration option in the Application object's Annex. If the status of the application does not change to Suspending after the graceful stop command but before the timeout expires, the application is considered not to support graceful shutdown, and is stopped ungracefully.

> ### Important
> The Shutdown Timeout does not apply to third-party applications. See Stopping Third-Party Applications

For a solution to stop gracefully, the graceful stop command is issued to each of its composite applications. How each composite application handles the command depends on whether the application supports the graceful-stop functionality.

Important

Because a number of solutions can share the same applications, some solution components may continue to have a status of Started after you stop the solution.

For more information about graceful stop, refer to *Genesys Administrator Help*. For details about the **suspending-wait-timeout** configuration option, refer to the *Framework Configuration Options Reference Manual*.

How to Use Startup Files

Some Genesys applications require special scripts to start and stop the application. Refer to the deployment procedures of your specific product to determine if separate start and stop files must be configured.

For Genesys applications that require both start and stop scripts to function properly, you must configure these scripts instead of single batch files. For Application objects that have the commands **start_stop.start_command** and **start_stop.stop_command** specified in their **Annex**, the specified values are used to start and stop the application on the target host. If these parameters are not specified, by default, system termination signals would be sent to the running process when the stop is issued. But if the **stop_command** is specified, an external script will be invoked instead for those applications that require special handling for termination.

> **Important**
>
> For Genesys applications for which the command line, command line argument, and **start_command** are all specified, the value in **start_command** would take precedence.

If your application requires startup files, do the following:

1. Modify the application to use a start up file. **[+] Show steps**

   ### Prerequisites

   - Configuration Layer components are installed and running.
   - An Application object exists for the application that is use a startup file.
   - You are logged in to Genesys Administrator.

   ### Steps

   a. In Genesys Administrator, open the **Configuration** tab of the Application object.

   b. In the **Server Info** section, modify the following properties:

      - `Command Line` property—Instead of the application executable file or startup file name, specify the command prompt name (**cmd.exe**) with the full path to it. For example:
        `D:\Windows\system32\cmd.exe`
      - **Command Line Arguments** property—Instead of the Configuration Server parameters and application name, specify the startup file name (**startServer.bat**) with the full path to it, preceded with the `/c` command line parameter. For example:
        `/c D:\GCTI\MessageServer\startServer.bat`

   c. Click Save to save the configuration changes.

2. Modify the startup file for each application. **[+] Show steps**

Prerequisites

- The applications must be configured to use startup files, as described in the previous step.

Steps

a. Using the text-editor of your choice, open one of the following files that are located in the applications' directories:
   - On UNIX—**run.sh**
   - On Windows—**startServer.bat**

b. If necessary, modify the existing line, or insert a new line specifying the application's executable file name with the full path to it, followed by this sequence of symbols:
   `%1 %2 %3 %4 %5 %6`
   For example, for a Message Server application installed in the GCTI directory on the D drive, the content of a startup file to use with the Management Layer looks like this:
   `D:\GCTI\MessageServer\MessageServer.exe %1 %2 %3 %4 %5 %6`

c. Save the file.

Centralized Logs

This section gives hands-on tips on how to use the Centralized Log system, including information about involved applications and reference documents:

- How to configure logs
- How to customize logs
- How to view logs
- How to manage log records
- How to trace interactions
- How to set up and use an audit trail
- Log formats
- Log specifications
- How to use the logmsg command-line utility

How to Configure Logging

The logging function requires the installation of:

- One or more Message Servers that collect log events from applications.
- One or more Log Databases, and one or more DB Servers, which connect Message Server with the DBMS in which you have set up the Log Database.

To configure logs, configure log options for a single application in Genesys Administrator on the **Options** tab of the respective Application object. For more information, refer to *Genesys Administrator Help*. The log options themselves are described in the "Common Configuration Options" section of the *Framework Configuration Options Reference Manual*.

Log levels are defined in Logging Functions. Changing the log level of a running application does not interrupt solution operations. The exception to this rule is Configuration Server, as follows:

- You must configure options for Configuration Server in its configuration files.
- You must restart Configuration Server for the new values to take effect.
- You cannot use the Log Wizard for Configuration Server.

For complete specifications of log events reported at the Alarm, Standard, Interaction, and Trace levels, see *Framework Combined Log Events Help*.

LCA Logging

Unlike with other server applications, you do not configure an Application object for LCA. However, you can change the default settings for common log options for LCA.

Starting in release 8.5, the LCA configuration file, called **lca.cfg**,is created automatically during installation of the IP, and stored in the same directory as the LCA executable file. Edit the file directly to specify new values for appropriate options. The configuration file contains only the `log` section.

Important

You can also specify a custom name for the configuration file. To start LCA with a custom name use the following format:

executable_name port -c configuration_file_name

On UNIX, for example:
`lca 7117 -c lca_custom.cfg`

Where `lca_custom.cfg` is the user defined configuration file.

The LCA configuration file has the following format:

```
[log]
```

```
<log option name>=<log option value>
<log option name>=<log option value>
```

For more information about common log options and the LCA configuration file, refer to the *Framework Configuration Options Reference Manual*.

Customizing Log Events

Log levels are defined in Logging Functions. Each log event has a default log level. You can customize log events for any application by changing the default log level of an event to a more appropriate level, or by disabling the event completely.

You can toggle the customizations on and off, without deleting them. This enables you to specify the customized log levels at any time, but only use them when appropriate. Note that this option enables or disables all of the customizations; it cannot be applied to specific ones.

Customizations are unique to the application in which they are created. For example, application A customizes a set of log events. application B does not customize any log events. If the feature is activated, the log events will have the customized properties of the application which generated them—if generated by application A, the log events will be customized as specified by A; if generated by application B, the log events will not be customized. If application B was to customize the same set of log events, but with different custom definitions, the log events generated by application B would be customized as specified by B.

> ## Warning
>
> Use caution when making these changes in a production environment.
>
> Depending on the log configuration, changing the log level to a higher priority may cause the log event to be logged more often or to a greater number of outputs. This could affect system performance.
>
> Likewise, changing the log level to a lower priority may cause the log event to be not logged at all, or not logged to specific outputs, thereby losing important information. The same applies to any alarms associated with that log event.

In addition to the precautionary message above, take note of the following:

- 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 therefore not subject to any log configuration.
- Changing the log level of a log using this feature 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 an 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. 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.

> **Important**
>
> Predefined log events of the Debug level are also in the unified log record format, and therefore can be assigned to another log level. However, any application can generate a raw text stream and call it a debug log event. Such non-unified log messages cannot be reassigned.

Refer to *Genesys Administrator Help* and the *Framework Configuration Options Reference Manual* for instructions and detailed examples of customized logs.

How to View Centralized Logs

With the Management Layer logging function, you can view the log records stored in Centralized Log Database, filter log records by their level, and search for records meeting the specified criteria. The log-viewing function requires the installation of:

- One or more Message Servers that collect log events from applications

> **Important**
>
> For usability reasons, Genesys does not recommend that you configure multiple Message Servers for one Log Database, with each Message Server assigned to handle logs for different applications. To view logs processed by a particular Message Server and therefore generated by a given application, you still have to filter the logs based on the application that generated them.

- One or more Log Databases
- Genesys Administrator

> **Important**
>
> Log filtering can be enabled and disabled for individual applications. Refer to the "Hide Selected Data in Logs" section in the *Genesys Security Deployment Guide* for more information.

For complete specifications of log events reported at the Alarm, Standard, Interaction, and Trace levels, see the *Framework Combined Log Events Help*.

Before you view Centralized Log records, make sure that:

- Management Layer components are installed and running.
- Centralized Logging is enabled.
- You are logged in to Genesys Administrator.

Then follow these steps:

1. In Genesys Administrator, do one of the following, as appropriate, to display a list of log records in the Details panel:
   - To view all log records stored in the Centralized Log Database, go to **Monitoring > Environment > Centralized Log**.
   - To view all log records for a specific Application object, go to **Provisioning > Environment > Applications** and select the Application.
   - To view all log records for all applications running on a specific Host object, go to **Provisioning >**

Environment > Hosts and select the Host.

2. To view an individual record or a subset of records, do one or both of the following:
   - Define one or more filtering criteria using the query builder fields above the list of log records.
   - Select a log record and, with one button, define a filter based on the value of certain fields in that record.

Refer to *Genesys Administrator Help* for more information about viewing contents of the Centralized Log Database.

How to Manage Log Records

You can manage records in the Log Database by:

- Using Genesys Administrator.
- Creating your own scripts for automated database purging.

> **Important**
> Log records also contain alarm history. Be careful not to delete current alarm history records when you remove log records from the Log Database.

The log-managing function requires the installation of the same components as for the log-viewing function.

Using the Log Database Maintenance Wizard

You can use the Management Layer logging function to manage log records stored in the Centralized Log Database. With the Log Database Maintenance Wizard available in Genesys Administrator, you can specify criteria—through a custom SQL statement or individual selections—for the search and removal of log records from the database. Available criteria categories include log type, log level, log generation time, log source, or extended attributes. Log source can be a particular application, applications that belong to a particular solution, or applications that run on a particular host computer.

Launch the Log Maintenance Wizard from **Environment > Centralized Log** under the **Monitoring** tab, and follow the instructions for each step. *Genesys Administrator Help* contains more information about the Wizard.

Automating the Purging Functionality

This section describes how to automate the removal of obsolete log records from the Log Database. Database purging involves the periodic, automated execution of appropriate SQL statements within your SQL server.

To enable automated purging:

1. Prepare SQL statements that remove log records. **[+] Show more information**

   As you create SQL statements that delete records from the Log Database tables, keep in mind that these SQL statements must contain one or more criteria for selecting the log records you want to remove. You can base the selection criteria on the values of

the log record fields, such as log record generation time, application name, host name, and so forth. For example, you might remove older log messages from the G_LOG_MESSAGES table and their corresponding attributes, if any, from the G_LOG_ATTRS table with the following SQL statements (in the order specified):

```
DELETE FROM G_LOG_ATTRS
WHERE LRID IN (SELECT G_LOG_MESSAGES.ID
  FROM G_LOG_MESSAGES
  WHERE (TIMEGENERATED > <start datetime>)
  AND (TIMEGENERATED < <end datetime>)
)
DELETE FROM G_LOG_MESSAGES
  WHERE (TIMEGENERATED > <start datetime>)
  AND (TIMEGENERATED < <end datetime>)
```

Refer to the structure description in Log Format for more information about Log Database tables and fields. Combine the selection criteria to achieve the level of purging that suits your environment.

Check the Log Database Maintenance Wizard in Genesys Administrator, for examples of the records-removal SQL statements that the Wizard prepares. The Wizard provides the graphical interface through which you specify various log-records selection criteria, and it displays the resulting SQL statements.

2. Schedule automated execution of the SQL statements. **[+] Show more information**

To enable automated purging of log records, schedule the periodic, automatic execution of the SQL statements you have prepared (for example, once a week). The simplest way to do this is to use either SQL server utilities or operating system services.

<tabber> Using SQL Server Utilities= If you decide to use SQL Server utilities, refer to your SQL Server documentation to determine whether that server provides tools for automatic execution of SQL statements. |-| Using OS Services= If you decide to use scheduling tools available in your operating system, you should:

1. Prepare a command (either an executable file or a batch/shell file) that executes your SQL statements.
2. Use an operating system tool that enables you to schedule the specified command for execution.

To prepare a command that executes your SQL statement(s), use either a batch file or shell script. A command like this usually calls an SQL Server–specific tool to execute command-line SQL statements and passes an SQL statement to this tool as a parameter. For example, you can use the following tools to execute command-line SQL statements:

- **isql.exe** (a Microsoft SQL tool)
- **sqlplus** (an Oracle tool)

To schedule a specified command for execution with the required frequency, consider using these tools:

- `cron` on UNIX platforms
- `at` on Windows platforms

How to Trace Interactions

You can trace interactions by using Interaction-level log records. You can view these logs using Genesys Administrator. The installation requirements for enabling the Interaction view are the same as for the Centralized Log.

Before tracing interactions, make sure that:

- Management Layer components are installed and running.
- Centralized Logging is enabled.
- You are logged in to Genesys Administrator.

Then do the following:

1. In Genesys Administrator, select **Monitoring > Environment > Centralized Log**, and select the **Interaction** tab.
2. To view a single Interaction log record, click the triangle to the left of the record.
3. To view all Interaction-level log records for a specific application or host, do one of the following:
   - Filter these records by entering the name of the application or host in the **Application** or **Host** field, respectively.
   - Select **Provisioning > Environment > Applications or Hosts > <name of object>**, and select the **Interaction** tab.

For more information, refer to *Genesys Administrator Help*.

How to Set Up and Use an Audit Trail

You can use Management Layer's centralized logging functionality to set up an audit trail using audit logs.

Setting Up the Audit Trail

Audit logs are log messages of type Standard or Trace that are marked as audit-related, and are logged in response to some action or event that requires an audit. To determine which logs are actually Audit logs, refer to *Framework Combined Log Events Help*. This help file identifies Audit logs for each component.

To set up the Audit trail, use the log configuration option **verbose**, and set the output to `network` to ensure that the logs will be stored in the Log Database, ready for viewing.

For more information about the options used in setting up an audit trail, refer to the *Framework Configuration Options Reference Manual*. For more information about log levels, see Log Levels.

Standard-Level Audit Logs

To set up an Audit trail using only Standard-level Audit logs, configure the following options:

- In the Application objects representing the components that have Audit logs and for which you want to set up an audit trail:

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

  and optional, but recommended:

  ```
  print-attributes=true
  ```

- In Message Server:

  ```
  [messages]
  db-storage=true
  ```

This will ensure that log events of Standard level will be stored in the Log Database, ready for viewing. Use the *Framework Combined Log Events Help* to identify which of the logs are Audit logs.

Standard- and Trace-Level Audit Logs

> **Warning**
> Trace-level logging generates a significantly greater number of logs than Standard-

> level logging, and may affect the performance of your system.

To set up an audit trail with Standard- and Trace-level Audit logs, configure the following options:

- In the Application objects representing the components that have Audit logs and for which you want to set up an audit trail:

  ```
  [log]
  verbose=trace
  trace=network
  ```

  and optional, but recommended:

  ```
  print-attributes=true
  ```

- In Message Server:

  ```
  [messages]
  db-storage=true
  ```

This will ensure that log events of Standard, Interaction, and Trace level will be stored in the Log Database, ready for viewing. Use the *Framework Combined Log Events Help* to identify which of the Standard- and Trace-level logs are Audit logs.

Viewing Audit Logs

You can view Audit logs using Genesys Administrator. Do the following:

1. In Genesys Administrator, select **Monitoring > Environment > Centralized Log**, and select the **Audit** tab.
2. To view a single Audit log record, click the small triangle to the left of the record.
3. To view all Audit log records for a specific application or host, do one of the following:
   - Filter these records by entering the name of the application or host in the **Application** or **Host** field, respectively. If the **Filter** panel is not visible, click the binoculars icon in the far right end of the Filter bar.
   - Select **Provisioning > Environment > Applications or Hosts> <name of object>**, and select the **Audit** tab.

For more information, refer to *Genesys Administrator Help*.

Log Format

A log record is a data record that stores information communicated in a single log event. Log records are stored in the Centralized Log Database in the following tables:

- G_LOG_MESSAGES
- G_LOG_ATTRS

G_LOG_MESSAGES

The G_LOG_MESSAGES table consists of one row for each log message. The structure of the table is described in the following table: **[+] Show table structure**

Structure of the G_LOG_MESSAGES Table

Field Name	Type	Description
ID	numeric	The unique identifier of the record stored in this table.
MESSAGE_ID	integer	The unique identifier of the event.
TIMEGENERATED	datetime	The time when the record was written to the database, in GMT format.
TIMEWRITTEN	datetime	The time when the record was written to the database, in GMT format.
PRIORITY	integer	The log level of the reported event. This field can have one of the following values: 2—Trace-level events. The Trace level of logging reports the details of communication between the various solution components and contains information about the processing steps for each interaction by each solution component. 3—Interaction-level events. The Interaction level of logging reports the details of an interaction process by solution components that handle interactions and contains information about the processing steps for each interaction by each solution component. 4—Standard-level events. The Standard level of logging contains high-level events that report both major problems and normal operations of in-service solutions. 5—Alarm-level events. The Alarm level of logging reports the events related to

		alarm detection, processing, and removal.
ORIGIN	integer	Reserved for future use.
CATEGORY	integer	Identifies the type of the log record and can have one of the following values: 0—for application-related log events 2—for audit-related log events
DATALEN	integer	Reserved for future use.
APPDBID	integer	Reserved for future use.
APPTYPE	integer	The type of application that reported the event. Refer to APPTYPE Field Values for a list of the valid values.
APPNAME	string	The name of the application, as specified in the Configuration Database, that reported the event.
HOSTNAME	string	The name of the host, as specified in the Configuration Database, on which the application that reported the event runs.
MESSAGETEXT	string	Text defining and describing the event.

APPTYPE Field Values

The following table provides the valid values for the **APPTYPE** field in the **G_LOG_MESSAGES** table. **[+] Show values**

Application Types for APPTYPE Field in the G_LOG_MESSAGES Table

Value	Application Type
00	Applications of all types when reporting log events common to all Genesys applications
01	T-Server
01	Programmable Gateway Framework
02	Stat Server
03	Billing Server
06	Voice Treatment Server
08	DB Server
09	Call Concentrator
10	CPD (Call Progress Detection) Server
11	List Manager

12	Outbound Contact Server
15	Universal Routing Server
21	Configuration Server
23	Third Party Server
26	DART Server (Obsolete)
28	Custom Server
29	External Router
31	Virtual Routing Point
32	Database (Obsolete)
33	Web Option
34	Detail Biller
35	Summary Biller
36	Network Overflow Manager
37	Backup Control Client
38	CC Analyzer Data Sourcer
40	IVR Interface Server
41	I-Server
42	Message Server
43	Solution Control Server
46	DB Server
48	WFM Data Aggregator
50	WFM Schedule Server
52	ETL Proxy
54	GVP-Voice Communication Server
55	VSS System
58	CC Analyzer Data Mart
59	Chat Server
60	Callback Server
61	Co-Browsing Server
62[a]	SMS Server
63	Contact Server
64	E-Mail Server
65	Media Link
66	Web Interaction Requests Server
67	Web Stat Server
68	Web Interaction Server
69	Web Option Route Point
74	Voice over IP Controller

77	HA Proxy
78	Voice over IP Stream Manager
79	Voice over IP DMX Server
80	Web API Server
81	Load Balancer
82	Application Cluster
83	Load Distribution Server
84	G-Proxy
85	Genesys Interface Server
86	GCN Delivery Server
88	IVR DirectTalk Server
89	GCN Thin Server
90	Classification Server
91	Training Server
92	Universal Callback Server
93	CPD Server Proxy
94	XLink Controller
95	K-Worker Portal
96	WFM Server
97	WFM Builder
98	WFM Reports
99	WFM Web
100	Knowledge Manager
101	IVR Driver
102	IVR Library
103	LCS Adapter
104	Desktop NET Server
105	Siebel7 ConfSynchComponent
106	Siebel7 CampSynchComponent
107	Generic Server
108	Generic Client
109	Call Director
110	SIP Communication Server
111	Interaction Server
112	Integration Server
113	WFM Daemon
114	GVP Policy Manager
115	GVP Cisco Queue Adapter

116	GVP Text To Speech Server
117	GVP ASR Log Manager
118	GVP Bandwidth Manager
119	GVP Events Collector
120	GVP Cache Server
121	GVP ASR Log Server
122	GVP ASR Package Loader
123	GVP IP Communication Server
124	GVP Resource Manager
125	GVP SIP Session Manager
126	GVP Media Gateway
127	GVP Soft Switch
128	GVP Core Service
129	GVP Voice Communication Server
130	GVP Unified Login Server
131	GVP Call Status Monitor
132	GVP Reporter
133	GVP H323 Session Manager
134	GVP ASR Log Manager Agent
135	GVP Genesys Queue Adapter
136	GVP IServer
137	GVP SCP Gateway
138	GVP SRP Server
139	GVP MRCP TTS Server
140	GVP CCS Server
141	GVP MRCP ASR Server
142	GVP Network Monitor
143	GVP OBN Manager
144	GVP Self Service Provisioning Server
145	GVP Media Control Platform
146	GVP Fetching Module
147	GVP Media Control Platform Legacy Interpreter
148	GVP Call Control Platform
149	GVP Resource Manager
150	GVP Redundancy Manager
151	GVP Media Server
152	GVP PSTN Connector
153	GVP Reporting Server

154	GVP SSG (formerly GVP ASG)
155	GVP CTI Connector
156	Resource Access Point
157	Workspace Desktop Edition (formerly Interaction Workspace)
158	Advisors
159	ESS Extensible Services
160	Customer View
161	Orchestration Server
162	Reserved
163	Capture Point
164	Rules ESP Server
165	Genesys Administrator
166	iWD Manager
167	iWD Runtime Node
168	Business Rules Execution Server
169	Business Rules Application Server
170	VP Policy Server
171	Social Messaging Server
172	CSTA Connector
173	VP MRCP Proxy
174	UCM Connector
175	OT ICS Server
176	OT ICS OMP Infrastructure
177	Advisors–Contact Center Advisor
178	Advisors–Frontline Advisor
179	Advisors–Advisors Platform
180	Advisors–Advisors Genesys Adapter
181	Advisors–Advisors Cisco Adapter
182	Federation Server
183	Federation Stat Provider
184	Genesys Administrator Server
185	Web Engagement Backend Server
186	Web Engagement Frontend Server
187	WebRTC Gateway
188	LRM Server
189	Recording Crypto Server
190	Genesys Knowledge Center Server
191	Genesys Knowledge Center CMS

[a] Prior to release 8.0, this value was used for the Application type IS Transport Server. In release 8.0 and later, this value is used for the Application type SMS Server.

G_LOG_ATTRS

Some log events also contain Extended Attributes that an application may attach for audit purposes. The G_LOG_ATTRS table contains one row for each extended attribute. The structure of the table is described in the following table:

[+] Show table structure

Structure of the G_LOG_ATTR Table

Field Name	Type	Description
ID	numeric	The unique identifier of the record stored in this table.
LRID	numeric	The unique identifier of the log record stored in the G_LOG_MESSAGES table to which this extended attribute belongs.
MESSAGE_ID	integer	The unique identifier of the event.
ATTR_NAME	string	The name of the extended attribute.
ATTR_VALUE	string	The value of the extended attribute in string format.

Log Specifications

The *Framework Combined Log Events Help* contains descriptions of all log events generated by Genesys components. This file is available for download from the Genesys Documentation website. It is regularly updated as new and updated Genesys components are released.

To obtain the latest version, download it from the Genesys Documentation website and extract the contents.

logmsg Command-Line Utility

The **logmsg** utility enables you to configure log messages to be sent to a specific Message Server on behalf of the specified application. For example, if a specific situation occurs, you can call **logmsg** to send a log message to a specific Message Server for information purposes or to perform an appropriate action. You can also use **logmsg** to generate and clear Management Layer alarms based on conditions external to the Management Layer.

Overview

The **logmsg** utility was developed to serve the need of customer scripts to raise alarms in the Management Layer. It is packaged with Solution Control Server, and uses existing Management Layer functionality to generate and clear Management Layer alarms based on conditions external to the Management Layer. When an external alarm condition is detected, such as by a script, **logmsg** is called to send a log message with the corresponding Log Event ID. SCS then triggers an active alarm specified by the Alarm Condition identified by the Detect Log Event ID.

logmsg is used by the stuck calls scripts (see Handling Stuck Calls). However, you can also use it independently of the stuck calls scripts, as described in this section.

Installing logmsg

If you installed Solution Control Server 8.5.100.41 and earlier, **logmsg** is already installed, and is located in the same folder in which Solution Control Server was installed.

Staring with SCS 8.5.100.42 release, the **logmsg** utility is not installed by default with Solution Control Server. You can install it separately by following the instructions in the section "Solution Control Server Utilities" in the *Framework Deployment Guide*. After the utilities are installed, **logmsg** is stored in the location you specified during the installation.

Starting in 8.1.2, you can install the Solution Control Server utilities without installing Solution Control Server itself.

Using the Utility

logmsg is located in the same folder in which Solution Control Server was installed.

All logmsg command parameters are made in a single command. The general syntax is as follows:

```
logmsg -h <host> -p <port> -lmid <ID> -aname <name> -atype <type> -adbid <dbid>
-hname <host>[-urgent <level>] -lmtext <text> [-lmattfile <filename>]
```

The parameters are described in the following table, and are mandatory unless stated otherwise in the table.

[+] Show parameters

logmsg Parameters

Parameter	Description
-h <host>	The name of the Host on which is running the Message Server to which you are sending the log message.
-p <port>	The Message Server port through which logmsg communicates with Message Server.
-lmid <ID>	The ID of the log event to be sent.
-aname <name>	The name of the application on behalf of which the log message is being sent.
-atype <type>	The type of the application on behalf of which the log message is being sent.
-adbid <dbid>	The DBID of the application on behalf of which the log message is being sent.
-hname <host>	The name of the host on which the application is running.
-urgent <level>	Optional. The urgency level of the log message, as follows: 0 - NONE 1 - DEBUG 2 - INFORMATION 3 - INTERACTION 4 - ERROR 5 - ALARM
-lmtext <text>	The text of the log message.
-lmattfile <filename>	Optional. The name of the file containing the log message attribute pairs, with each pair consisting of an <attribute_name> and an <attribute_value>.

Configuring Alarms for Events External to the Management Layer

You can also use **logmsg** to generate and clear Management Layer alarms based on conditions external to the Management Layer. For example, you may configure an alarm to be triggered in response to a stuck call.

When an external alarm condition is detected, such as by a script, call **logmsg** to send a log message with the corresponding Log Event ID. SCS then triggers an active alarm specified by the Alarm Condition identified by the Detect Log Event ID.

Before you use this utility for this purpose, you must configure one Alarm Condition object for each condition for which an alarm is to be triggered. Each condition must specify a Detect Log Event ID and a Cancel Log Event ID.

> **Important**
>
> Detect and Cancel Log Events created in this situation must have an event ID in the range of `9600–9800`; this range is reserved for custom alarms.

Alarms

This section gives hands-on tips on how to use alarms, including information about the applications involved and reference documents:

- How to configure Alarm Conditions and Alarm Reactions
- How to configure email Alarm Reactions
- Pre-defined Alarm Conditions included in your Genesys software

How to Configure Alarm Conditions and Alarm Reactions

Although you can configure Alarm Condition objects and Alarm Reaction scripts manually, using Genesys Administrator, the Management Layer provides an automated procedure.

You can configure new Alarm Conditions based on either source for their detection:

- Log Events—These Alarm Conditions trigger an alarm when an application or applications generate a specified log event.
- Alarm Detection Scripts—These Alarm Conditions trigger an alarm when a certain system variable changes in a specified manner.

For alarms based on log events, alarm detection takes place in Message Server. Therefore, if you configure log event–based Alarm Conditions, you must configure your applications to connect to Message Server.

Using Log Events for Alarm Detection

To configure alarm conditions or alarm reactions in Genesys Administrator, create the alarm condition or alarm reaction under **Provisioning > Environment > Alarm Conditions**, being sure to specify the appropriate log event on the **Configuration** tab. You can also use pre-configured Alarm Condition objects. Refer to *Genesys Administrator Help* for detailed instructions about creating and using these necessary objects.

You can also use the Alarm Condition Wizard to associate Alarm Reactions with the Alarm Condition you are configuring.

For complete specifications of log events reported at the Alarm, Interaction, Standard, and Trace levels, see *Framework Combined Log Events Help*.

Using Detection Scripts for Alarm Detection

Management Layer provides an additional alarm-detection mechanism, called Advanced Alarm Detection. Through this mechanism, you can configure Alarm Conditions:

- Based on the threshold for a system performance variable (CPU or memory usage).
- Based on the threshold for a local or remote SNMP variable (available only when you have enabled SNMP functionality).

Set alarms based on the Advanced Alarm Detection methods using Alarm Detection scripts, and then

associate them with corresponding Alarm Conditions. When an Alarm Condition object refers to an Alarm Detection script, the alarm detection for this Alarm Condition is performed as specified by the Alarm Detection script, regardless of whether any log event is specified as a Detection Event.

Using Alarm Detection and Reaction Scripts with Alarm Conditions

To configure an Alarm Detection Script, do the following:

[+] Show steps

1. Log in to GAX if required.
2. **Configuration** > **Environment** > **Detection/Reaction Scripts**.
3. Navigate to the folder in which you want to store the new script, and click **New**.
4. Enter the following information for the new script:
   a. Enter the **Name** of the script.
   b. Make sure that **State Enabled** is checked.
   c. From the **Script Type** drop-down list, select **Alarm Detection**.
   d. From the **Detection Types** drop-down list, select one of the following tabs and proceed accordingly. Note that you must have SNMP functionality in your environment to monitor SNMP variables.

Host System Variable Threshold

To monitor a performance variable for a given Host, provide the following information:

i. Enter the DNS or IP address of the **Host** to be monitored.
ii. From the **Select Monitored Resource** drop-down list, select the resource to be monitored, either **Host CPU Usage (%)** or **Host Memory Usage (MB)**.

Application System Variable Threshold

To monitor a performance variable for a given Application, provide the following information:

i. Select the **Application** to be monitored.
ii. From the **Select Monitored Resource** drop-down list, select the resource to be monitored, either **Process CPU Usage (%)** or **Process Memory Usage (MB)**.

Local SNMP Variable Threshold

> **Important**
> You must have SNMP functionality configured in your environment to monitor SNMP variables.

To monitor a local SNMP variable, enter the oid of the variable in **Specify Local SNMP Variable**.

Remote SNMP Variable Threshold

> **Important**
> You must have SNMP functionality configured in your environment to monitor SNMP variables.

To monitor a remote SNMP variable, provide the following information:

i. Select the **SNMP Agent Application** in which the variable is used.

ii. Enter the name of the variable to be monitored in **Specify Local SNMP Variable**.

e. In **Specify Sample Interval**, specify (in seconds) the frequency with which to take a sample of the required information.

f. In **Specify Falling Threshold**, specify the level at which the monitored attribute must fall below to clear the alarm. Your entry must be in the same units of measure as the units of measure specified for the resource or variable that you are monitoring.

g. In **Specify Rising Threshold**, specify the level at which the monitored attribute must rise above to trigger the alarm. Your entry must be in the same units of measure as the units of measure specified for the resource or variable that you are monitoring.

5. Click **Save.**

If there is a specific automated action that you want to happen as a result of the alarm, configure an Alarm Reaction script. For example, to generate an SNMP trap when this alarm is triggered, do the following:

[+] Show steps

1. Log in to GAX if required.
2. **Configuration** > **Environment** > **Detection/Reaction Scripts**.
3. Navigate to the folder in which you want to store the new script, and click **New**.
4. Enter the following information for the new script:
   a. Enter the **Name** of the script.
   b. Make sure that **State Enabled** is checked.
   c. From the **Script Type** drop-down list, select **Alarm Reaction**.
   d. From the **Alarm Reaction Types** drop-down list, select **Send an SNMP trap**. You must have SNMP

functionality in your environment to monitor SNMP variables.

e. Click **Save**.

To create the corresponding Alarm Condition object, follow these steps:

[+] Show steps

1. Log in to GAX if required.
2. **Configuration** > **Environment** > **Alarm Conditions**.
3. Navigate to the folder in which you want to store the new Alarm Condition, and click **New**.
4. On the **General** tab, enter the following information for the new script:
   a. Enter the **Name** of the Alarm Condition.
   b. (Optional) Provide a short **Description** of the Alarm Condition.
   c. From the **Category** drop-down list, select the severity of the Alarm Condition, either **Critical**, **Major**, or **Minor**.
   d. In the **Detect Script** field, specify the name of the Alarm Detection Script that you created above, using the provided browser control if necessary.
   e. In the **Cancel Timeout** field, specify the time interval (in seconds) after which the alarm will be cleared by SCS. If the host CPU is still not normalized, the alarm will be generated again.
   f. In the **Detect Log Event ID** and **Cancel Log Event ID** fields, enter any numeric dummy value. These two fields are not used if you are using an Alarm Detection Script, and while ignored by the Management Layer in this case, are still mandatory.
   g. From the **Detect Selection** drop-down list, select the mode for event selection that the Management Layer uses for Alarm Condition analysis. Refer to Alarm Conditions in GAX Help for a detailed description of this field.
   h. Make sure that **State Enabled** is checked.
5. If you have created an Alarm Reaction Script for this Alarm Condition, link the Reaction Script to this Alarm Condition, as follows:
   a. On the **Reactions Scripts** tab, click **Add**.
   b. Navigate to, and select, the Alarm Reaction Script that you created above.
6. If you have created an Alarm Clearance Script for this Alarm Condition, link the Clearance Script to this Alarm Condition. In this case, an Alarm Reaction of type "Send an SNMP trap" also acts as its own Alarm Clearance Script, since there is no waiting involved; that is, once the Alarm Reaction is complete, the Alarm can be cleared—the trap has been sent. So, you just need to specify the same script as the Clearance Script, as follows:
   a. On the **Clearance Scripts** tab, click **Add**.
   b. Navigate to, and select, the Alarm Reaction Script that you created above.
7. Click Save.

Important

If you select **Select By Any** in the **Detect Selection** field (step 4g) when creating the Alarm Condition object, be aware that SCS creates an alarm if a situation involving *any* application or host produces the Detect Event ID. Likewise, it clears the alarm if *any* application or host produces the Clearance Event ID. As a result, specific Alarm Conditions such as Host Inaccessible, Host Unavailable, and Host Unreachable (see Predefined Alarm Conditions) will trigger an alarm for the first Host that encounters one of the conditions, and stay triggered for all other hosts that encounter one of these conditions. The alarm is cleared only when the first of the affected hosts recovers.
To avoid this problem, and to more accurately monitor these conditions, Genesys suggests that you manually clear each alarm as it occurs on a host, so that next time it occurs, it will be generated by the affected host. Or, you can set **Cancel Timeout** (step 4e) to a minimal value so the alarm is automatically cleared shortly after detection.

Types of Alarm Reactions

You can configure alarm reactions of the following types:

- Start a specified application
- Stop a specified application
- Restart the application that generated the alarm
- Start a specified solution
- Send an email
- Send an SNMP trap
- Switch over to the backup application
- Execute OS command
- Change application option

The configuration procedure for most of the alarm reactions is self-explanatory in the Alarm Reaction Wizard. You must supply information for these configuration parameters:

- A unique name for the Alarm Reaction configuration object (for all types of alarm reactions).
- A name of the application or solution the alarm reaction is configured for (for alarm reactions of such types as Start a specified application, Stop a specified application, Start a specified solution, and Restart the application that generated the alarm).

You must provide additional information for alarm reactions of the following types:

- Switchover
- Send an E-Mail

- Send an SNMP Trap
- Execute OS command
- Change application option

Switchover

> **Warning**
> You must have a high-availability (HA) license to enable Solution Control Server to successfully process an alarm reaction of the Switchover type. The lack of the license prevents the switchover between primary and backup applications of any type.

When configuring an alarm reaction of the Switchover type, you can specify whether Solution Control Server should perform the switchover when an application, which generates an alarm, is running in a particular mode:

- Select `primary` if you want SCS to perform a switchover only if the application that has generated an alarm is currently operating in Primary mode.
- Select `backup` if you want SCS to perform a switchover only if the application that has generated an alarm is currently operating in Backup mode.
- Select `perform switchover always` if you want SCS to perform a switchover regardless of the operating mode of the application that generates the alarm.

You might use these options, for instance, when associating an alarm reaction of the Switchover type for T-Server with the CTI Link Disconnected log event. Selecting `primary` for the alarm reaction configuration may prevent an unwanted switchover if the T-Server that produced this log event currently operates in Backup mode.

Send an E-Mail

To configure an alarm reaction of type Send an E-Mail, specify the recipients of the email in the Alarm Reaction Wizard. Then, compose the subject and text of the email message, using reserved variables. See *Genesys Administrator Help* for detailed instructions on configuring the email script, including an example. See E-Mail Alarm Reactions for more information about the email interface itself.

Send an SNMP Trap

To configure an alarm reaction of the Send an SNMP Trap type, specify a Name for the Alarm Reaction configuration object. All necessary information is automatically provided by SCS, given that the SNMP Master Agent application is configured correctly.

See SNMP Interface for more information about enabling the SNMP alarm signaling.

Execute OS command

To configure an Alarm Reaction of the Execute OS Command type, specify the name of the operating

system command that is to be executed when an alarm is detected. If necessary, include the full path to the executed command.

> **Important**
>
> Although you can specify any valid command name, use alarm reactions of this type with caution. To avoid unauthorized actions, limit access to Solution Control Server and Genesys Administrator to the Administrators group.

SCS executes all alarm reactions. In the case of an alarm reaction of the Execute OS Command type, SCS executes the specified command on its own host computer. Therefore, a currently logged in user must have sufficient permissions to execute the specified operating system command.

SCS passes information about a detected alarm to the operating system command to be executed. For this purpose, SCS adds command-line arguments (listed in the table below) to the command line you specify in the **Command** property when you configure the alarm reaction.

> **Important**
>
> Some applications started as a result of the Execute OS Command alarm reaction may not recognize the command-line arguments added by SCS. This means that these applications might not work properly in this circumstance; for example, they might exit. To make them work, you can call such applications indirectly; for instance, from within a script that passes correct command-line parameters to these applications. You then specify name of this script in the **Command** property of the alarm reaction.

Additional Command-Line Parameters

The following table describes the additional command-line parameters that are added to the **Command** property when you configure an Execute OS Command Alarm Reaction.

[+] Show table

Additional Command-Line Parameters for Execute OS Command Alarm Reactions

Parameter	Description
-msgid	ID of the log event that resulted in the alarm
-msgtext	Text of the log event that resulted in the alarm
-condid	Alarm Condition ID
-condname	Alarm Condition Name
-conddesc	Alarm Condition Description
-appid	ID of the application that generated the log event that resulted in the alarm
-appname	Name of the application that generated the log event that resulted in the alarm

-hostname	Name of the application host that generated the log event that resulted in the alarm

Examples

The following are examples of how to use and configure Execute OS Command Alarm Reactions. **[+] Show examples**

Example 1 - Filtering by type of log event

To log occurrences of certain log events to a database other than the Genesys Log Database, create a ***.bat** file that provides logging in to your independent database. Name this file **process_alarm.bat**. Then using the Alarm Reaction Wizard, configure an Alarm Reaction of the Execute OS Command type and set the **Command** property to:

```
/home/Genesys/SCServer/scripts/process_alarm.bat
```

When a corresponding alarm is detected, SCS executes the following operating system command:

```
/home/Genesys/SCServer/scripts/process_alarm.bat -msgid 20002 -msgtext "CTI Link
disconnected" -condid 103 -condname "CTI Link Failure" -conddesc "Failure of
connection between any T-Server and a switch." -appid 120 -appname "T-
Server_Application" -hostname “NameOfHost”
```

> ### Important
> If a specified operating system command normally would result in a screen display, the alarm reaction is performed, but the screen output cannot be enabled and, therefore, cannot be seen.

Example 2 - Collecting information about alarms

A script called **react_script.sh**, for the bash UNIX shell, saves information about each alarm in the **reaction.log** text file located in the **/home/genesys/logs/** directory:

```
echo `date|cut -c4-16` : msgid=$2 : msgtext=$4 : condid=$6 : condname=$8 :
conddesc=${10} : appid=${12} : appname=${14} >> /home/genesys/logs/reactions.log
```

Example 3 - Filtering by the host that generated the alarm

To save information about alarms generated by a specific host, you must create a script that writes only those logs generated by a specific host. The following sample script, **HostA_alarms.sh** for the

bash UNIX shell, writes only those alarms generated by HostA to the **HostA_reactions.log** text file located in the **/home/genesys/logs/** directory.

```
while [ 0 -lt "$#" ]; do
case "$1" in
"-msgid") shift; MSGID="$1" ;;
"-msgtext") shift; MSGTEXT="$1" ;;
"-condid") shift; CONDID="$1" ;;
"-condname") shift; CONDNAME="$1" ;;
"-conddesc") shift; CONDDESC="$1" ;;
"-appid") shift; APPID="$1" ;;
"-appname")shift; APPNAME="$1" ;;
"-hostname")shift; HOSTNAME="$1" ;;
esac
shift
done
if [ $HOSTNAME = "HostA" ]; then
echo `date|cut -c4-16` : msgid=$MSGID : msgtext=$MSGTEXT : condid=$CONDID :
condname=$CONDNAME : conddesc=$CONDDESC : appid=$APPID : appname=$APPNAME >>
HostA_reactions.log
fi
```

Change application option

Select an application and specify its configuration options to be automatically set upon occurrence of an alarm event. With this type of alarm reaction, you can automatically change a specified configuration option of any given Daemon application on occurrence of same alarm event. Select an application whose configuration option is to be changed. If no application selected, the operation automatically applies to the application that triggered the alarm.

E-Mail Alarm Reactions

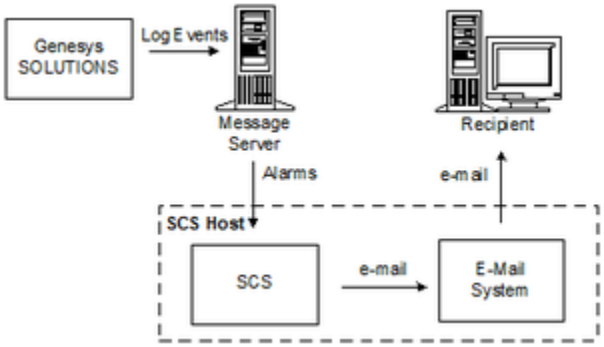
This chapter describes how the Management Layer processes alarm reactions of the E-Mail type and how to configure an email system for this function.

Alarm Reactions of the E-Mail Type

You can configure the Management Layer to send the content of an alarm as an email message to one or more email addresses. Simply create an alarm reaction of the Send an email type for a corresponding alarm condition. See the recommendations for configuring alarm reactions in How to Configure Alarm Conditions and Alarm Reactions.

> **Important**
>
> An *alarm* is a message generated by a Genesys application when a certain alarm condition is met. For more information, refer to Alarm-Signaling Functions.

The following diagram illustrates the message flow through the Management Layer when such an alarm is triggered. That flow includes the email system for your environment, which you must configure for the host running Solution Control Server.



E-Mail Alarm Reaction in Management Layer

Configuring E-Mail Systems

This section describes how to configure a UNIX- and a Windows-based email systems to send the content of an alarm message in an email.

On UNIX

On UNIX operating systems, Solution Control Server can use either the **sendmail** command or Simple Mail Transfer Protocol (SMTP) to send email messages. Depending on the protocol you prefer:

- You must correctly configure the **sendmail** command on the host computer running SCS.
- You must configure SMTP server host and port for SCS as values for the **smtp_host** and **smtp_port** configuration options. For more information about these options, refer to the "Solution Control Server" section of the *Framework Configuration Options Reference Manual*.

On Windows

On Windows operating systems, Solution Control Server uses Simple Mail Transfer Protocol (SMTP). To enable the operation of email alarm reactions via an SMTP email system, you must configure the `mailer` section. Specify the SMTP server host for SCS as the value for the **smtp_host** configuration option smtp_host, and specify the SMTP server port for SCS as the value for the option **smtp_port**.

For more information about Solution Control Server configuration options, refer to the *Framework Configuration Options Reference Manual*.

Predefined Alarm Conditions

Genesys software includes several predefined alarm conditions that are available immediately after you set up Framework. The conditions under which alarms are generated, the actions automatically taken by the system to cope with or recover from the failure, and the maintenance actions appropriate in each situation are discussed for each alarm condition.

The following alarm conditions are predefined:

Connection Failure

The Connection Failure Alarm Condition reports that the specified connection between any two applications has been lost. It is always reported by the client application and might indicate one of the following:

- The connection was intentionally closed by the server (for example, in response to an overload situation).
- The connection was closed by a networking software (for example, in response to a long interval without any data exchange through the given connection).
- The server terminated.
- The server stopped responding.
- The server host failed.
- A network connectivity problem occurred between the computers that run the given client application and the server.

The following table describes the fields of this Alarm Condition. **[+] Show fields**

Connection Failure Predefined Alarm Condition

Field Name	Value and Description
Name	Connection Failure
Description	The connection between any two Genesys components has been lost.
Category	Major
Detect Event	**00-04504**: `Connection to [server type] [server name] at host [host name] port [port number] lost`
Selection Mode	Select By Any
Cancel Event	**00-04503**: `Connected to [server type] [server name] at host [host name] port [port number]`
Cancel Timeout	48 hours
Reaction Scripts	None
Clearance Scripts	None
State Enabled	True

Refer to *Framework Combined Log Events Help* for full descriptions of the Detect and Cancel Events.

Automatic Recovery Actions

- If a backup server for the specified server is not configured, the client application that reported the connection failure periodically attempts to reconnect to the specified server.
- If a backup server for the specified server is configured, the client application that reported the connection failure attempts to connect interchangeably to the specified server and the backup server.

> **Important**
> The number of reconnect attempts is unlimited.

- After a successful reconnect attempt, the alarm condition is automatically cleared.

Suggested Maintenance Actions

1. Check the condition of the server host computer.
2. Check the condition of the server.
3. Check the server log to see if the given application has disconnected intentionally. Look for log events with ID **04523**.
4. Check the network connectivity between the computers that run the given application and the server.

Application Failure

The Application Failure Alarm Condition Reports that the specified application has either terminated or stopped responding. It might indicate one of the following:

- The application terminated because of an internal condition.
- The application was closed by means other than the Management Layer (for example, with an operating system command).
- The application entered a no-response condition.

The following table describes the fields of this Alarm Condition. **[+] Show fields**

Application Failure Predefined Alarm Condition

Field Name	Value and Description
Name	Application Failure
Description	Failure of any daemon Genesys component monitored by the Management Layer
Category	Major
Detect Event	**00-05064**: `Application terminated due to internal condition`
Selection Mode	Select by any
Cancel Event	**00-05090**: `Application start detected by Management Layer`
Cancel Timeout	48 hours
Reaction Scripts	None
Clearance Scripts	None
State Enabled	True

Refer to *Framework Combined Log Events Help* for full descriptions of the Detect and Cancel Events.

Automatic Recovery Actions

- If a backup application for the specified application is not configured and **Auto-Restart** is selected in the application's properties, the Management Layer attempts to restart the specified application.
- If a backup application for the specified application is not configured and **Auto-Restart** is not selected in the application's properties, no automatic recovery action takes place.
- If a backup application for the specified application is configured and **Auto-Restart** is selected in the application's properties, the Management Layer switches operations over to the backup application and attempts to restart the specified application in Standby mode.
- Upon a successful attempt to restart the specified application, the alarm is automatically cleared.

Suggested Maintenance Actions

1. Using Genesys Administrator, locate the exact source of the alarm and check the current status of the

application. It is likely that the fault has been eliminated through an automatic recovery action.

2. If the alarm is still active, check the status of the application through the operating system tools.
3. If the application is running but not responding, restart the application with an operating system command.
4. If the application is not running, start the application with an operating system command.
5. Ensure that SCI shows the status of the application correctly.
6. Verify that the Auto-Restart check box for this application is selected.

Licensing Error

The Licensing Error Alarm Condition reports that a licensing error has occurred. Possible violation types are as follows:

- License information is invalid.
- Licensable feature has expired.
- Feature usage level has been exceeded.
- Licensing system has experienced a general failure.

The following table describes the fields of this Alarm Condition. **[+] Show fields**

Licensing Failure Predefined Alarm Condition

Field Name	Value and Description
Name	Licensing Failure
Description	Any licensing error identified by any Genesys component
Category	Critical
Detect Event	**00-07100**: `Licensing violation is identified, the violation type [type]`
Selection Mode	Select by any
Cancel Event	None
Cancel Timeout	48 hours
Reaction Scripts	None
Clearance Scripts	None
State Enabled	True

Refer to *Framework Combined Log Events Help* for a full description of the Detect Event.

Automatic Recovery Actions

None

Suggested Maintenance Actions

Depending on the value of the error code:

- Check the condition of License Manager. If the type of license you have requires License Manager, it should be running and accessible by the Genesys applications. Check that the host and port of License Manager are specified correctly.
- Make sure that the actual location of the license file or license server corresponds to the location specified in the command-line parameter used for application startup.
- Make sure that the specified license file is the exact copy of the license file received from Genesys.
- Locate the exact source of the alarm and apply to Genesys for an extension of the license.
- Locate the exact source of the alarm and check the current usage level against the usage level stipulated in the license. Either decrease the usage level or apply to Genesys for a new license that covers the increased usage needs.

CTI Link Failure

The CTI Link Failure Alarm Condition reports that the connection between the specified T-Server and its switch has been lost. It is always reported by T-Server and might indicate one of the following:

- The connection was intentionally closed on the switch side (for example, as an automatic defense action).
- The control system of the switch failed.
- A network connectivity problem occurred between the T-Server host and the switch.

The following table describes the fields of this Alarm Condition. **[+] Show fields**

CTI Link Failure Predefined Alarm Condition

Field Name	Value and Description
Name	CTI Link Failure
Description	Failure of connection between any T-Server and its switch
Category	Major
Detect Event	**01-20002**: `CTI Link disconnected`
Selection Mode	Select by application type T-Server
Cancel Event	**01-20001**: `CTI Link connected`
Cancel Timeout	48 hours
Reaction Scripts	None
Clearance Scripts	None
State Enabled	True

Refer to *Framework Combined Log Events Help* for full descriptions of the Detect and Cancel Events.

Automatic Recovery Actions

T-Server attempts to reconnect to the CTI link.

Suggested Maintenance Actions

- Check the condition of the control system of the switch and of its CTI link.
- Check the network connectivity between the control system of the switch and the computer running T-Server.

Host Inaccessible

The Host Inaccessible Alarm Reaction reports that the Management Layer cannot contact the Local Control Agent on the host on which Genesys daemon applications are running. It might indicate one of the following (in the order of probability of occurrence in a typical production environment):

- The connection between SCS and the LCA of the specified host failed.
- LCA is not started on the specified host.
- LCA is listening on a port that is different from the one specified in the configuration.
- The LCA of the specified host has terminated or stopped responding.

The following table describes the fields of this Alarm Condition. **[+] Show fields**

Host Inaccessible Predefined Alarm Condition

Field Name	Value and Description
Name	Host Inaccessible
Description	The Management Layer cannot access a host computer on which Genesys daemon applications run.
Category	Major
Detect Event	**00-08000**: `Host [host name] inaccessible. LCA is not listening on port [port number].`
Selection Mode	Select by any
Cancel Event	**00-08001**: `Host [host name] operates in normal condition.`
Cancel Timeout	48 hours
Reaction Scripts	None
Clearance Scripts	None
State Enabled	True

Refer to *Framework Combined Log Events Help* for full descriptions of the Detect and Cancel Events.

Automatic Recovery Actions

By default, this failure is treated by the Management Layer as a failure of every Genesys application running on the given host. For the applications located on the given host that have redundancy, the Management Layer makes their backup applications primary. After that, Solution Control Server makes repeated attempts to restore connection with the LCA of the specified host. Once the connection is restored, the Management Layer attempts to start all applications that were running before the alarm occurred.

Suggested Maintenance Actions

1. Check the condition of LCA. If LCA terminated or stopped responding, restart LCA. Notify Genesys Customer Care about the LCA failure.
2. Verify the LCA command line parameters and make sure that LCA listens on the same port as the one specified in the Configuration Database.

Service Unavailable

The Service Unavailable Alarm Condition reports that a Genesys component cannot provide service for some internal reasons.

The following table describes the fields of this Alarm Condition. **[+] Show fields**

Service Unavailable Predefined Alarm Condition

Field Name	Value and Description
Name	Service Unavailable
Description	A Genesys component is unable to provide service for some internal reasons.
Category	Major
Detect Event	**00-05094**: `Application is not able to provide service.`
Selection Mode	Select by any
Cancel Event	**00-05093**: `Application is ready to provide service.`
Cancel Timeout	48 hours
Reaction Scripts	None
Clearance Scripts	None
State Enabled	True

Refer to *Framework Combined Log Events Help* for full descriptions of the Detect and Cancel Events.

Automatic Recovery Actions

If a backup application for the specified application is configured, the Management Layer switches

operations over to the backup application.

Suggested Maintenance Actions

This alarm occurs because of internal application reasons. Examine the log of the application that signaled the alarm to determine and eliminate the source of the problem.

Host Unavailable

The Host Unavailable Alarm Condition reports that a host on which Genesys daemon applications are running is unavailable (turned off). It might indicate one of the following (in the order of probability of occurrence in a typical production environment):

- Specified host has failed or has been turned off.
- Network problems prevents SCS from connecting to LCA at the specified host.

The following table describes the fields of this Alarm Condition. **[+] Show fields**

Host Unavailable Predefined Alarm Condition

Field Name	Value and Description
Name	Host Unavailable
Description	A host on which Genesys daemon applications are running is unavailable (turned off).
Category	Major
Detect Event	**00-08002**: `Host [host name] unavailable.`
Selection Mode	Select by any
Cancel Event	**00-08001**: `Host [host name] operates in normal condition.`
Cancel Timeout	48 hours
Reaction Scripts	None
Clearance Scripts	None
State Enabled	True

Refer to *Framework Combined Log Events Help* for full descriptions of the Detect and Cancel Events.

Automatic Recovery Actions

This failure may occur when an SCS attempt to connect to the LCA at the specified host fails. This failure is determined based on the error code returned by the networking subsystem. No automatic recovery actions are performed when this failure occurs.

Suggested Maintenance Actions

1. Check the condition of the host. If the host failed, take measures to restore its normal operating

condition. Once the normal condition is restored, the Management Layer automatically brings up all Genesys applications that are supposed to be running.

2. Check the condition of the network. Make sure that it is possible to reach the host of interest from the host on which SCS is running.

Host Unreachable

The Host Unreachable Alarm Condition reports that the Management Layer cannot reach the host on which Genesys daemon applications are running (no route to the host). It might indicate the following:

- Network configuration is incorrect: there is no route to the host of interest from the host on which SCS is running.

The following table describes the fields of this Alarm Condition. **[+] Show fields**

Host Unreachable Predefined Alarm Condition

Field Name	Value and Description
Name	Host Unreachable
Description	The Management Layer cannot reach the host on which Genesys daemon applications are running (no route to the host).
Category	Major
Detect Event	**00-08003**: `Host [host name] unreachable.`
Selection Mode	Select by any
Cancel Event	**00-08001**: `Host [host name] operates in normal condition.`
Cancel Timeout	48 hours
Reaction Scripts	None
Clearance Scripts	None
State Enabled	True

Refer to *Framework Combined Log Events Help* for full descriptions of the Detect and Cancel Events.

Automatic Recovery Actions

This failure may occur when an SCS attempt to connect to the LCA at the specified host fails. This failure is determined based on the error code returned by the networking subsystem. No automatic recovery actions are performed when this failure occurs.

Suggested Maintenance Actions

Check the condition of the network. Make sure that routing is configured correctly in the network and that it is possible to reach the host of interest from the host on which SCS is running.

Unplanned Solution Status Change

The Uplanned Solution Status Change alarm Condition reports that a solution status has changed from Started to Pending without any requests to stop the solution. It might indicate the following:

- Failure of one or more of the solution components.

The following table describes the fields of this Alarm Condition. **[+] Show fields**

Unplanned Solution Status Change Predefined Alarm Condition

Field Name	Value and Description
Name	Unplanned Solution Status Change
Description	Solution status has changed from Started to Pending without any requests to stop the solution. This may indicate a failure of one of the solution components.
Category	Major
Detect Event	**43-10385**: `Solution [solution name] nonplanned change of state from Started to Pending.`
Selection Mode	Select by any
Cancel Event	**43-10370**: `Solution [solution name] is started.`
Cancel Timeout	48 hours
Reaction Scripts	None
Clearance Scripts	None
State Enabled	True

Refer to *Framework Combined Log Events Help* for full descriptions of the Detect and Cancel Events.

Automatic Recovery Actions

If this alarm occurred because of the failure of one or more of the solution components, the Management Layer performs the same automatic recovery actions for each failed application as described for the Application Failure alarm condition.

Suggested Maintenance Actions

For each failed solution component, perform the same Maintenance Actions as suggested for the Application Failure alarm condition.

Message Server Loss of Database Connection

The Message Server Loss of Database Connection reports that Message Server has lost connection to

the Centralized Log Database. It might indicate one of the following (in the order of probability of occurrence in a typical production environment):

- Failure of the DB Server used by Message Server to access the Centralized Log Database.
- Failure of the DBMS that stores the Centralized Log Database.

The following table describes the fields of this Alarm Condition. **[+] Show fields**

Message Server Loss of Database Connection Predefined Alarm Condition

Field Name	Value and Description
Name	Message Server Loss of Database Connection
Description	Message Server has lost connection to the Centralized Log Database.
Category	Major
Detect Event	**42-11051**: `Connection with DB Cluster lost.`
Selection Mode	Select by application type Message Server
Cancel Event	**42-11050**: `Connection with DB Cluster established ([host name]:[port number]).`
Cancel Timeout	48 hours
Reaction Scripts	None
Clearance Scripts	None
State Enabled	True

Refer to *Framework Combined Log Events Help* for full descriptions of the Detect and Cancel Events.

Automatic Recovery Actions

If this alarm occurred because of the failure of DB Server used by Message Server to access the Centralized Log Database, the Management Layer performs the same automatic recovery actions for DB Server as described for the Application Failure alarm condition.

Suggested Maintenance Actions

1. In the case of Log DB Server failure, perform the same Maintenance Actions as suggested for the Application Failure alarm condition.
2. Otherwise, make sure that the DBMS that stores the Centralized Log Database is operating in normal condition.

SNMP Support

This section describes Management Layer support for network management systems (NMS) that comply with the Simple Network Management Protocol (SNMP). It also describes how to activate this function. In particular, it focuses on how the Management Layer distributes SNMP commands from an NMS and how it processes alarm reactions of the SNMP Trap type. It also describes the layout of the Genesys Management Information Base (MIB) file and the format of the SNMP traps, including the abbreviations for the Genesys application types.

SNMP Interface

The Management Layer provides support for SNMP-compliant third-party NMS. Solution Control Server (SCS) processes various NMS commands and generates SNMP traps based on changes in the current status of an individual application. With this support for SNMPv1–v3, you can access Management Layer functions through your existing NMS interface.

> **Important**
>
> The Genesys built-in SNMP implementation, provided by the Genesys SNMP Master Agent component, for SNMPv1 passed all the tests developed and published by CERT/CC for this sort of application. For information about tests with which you can check your system against vulnerability to SNMPv1 malformed SNMP packets, go to http://www.cert.org/advisories/CA-2002-03.html.

Architecture

The Management Layer provides you, as a network administrator, with three ways to monitor and control Genesys products via an NMS user interface:

- You can start, stop, and monitor the status of any Genesys or third-party application that the Management Layer monitors and controls. In addition, you can modify log options for Genesys server applications.
- You can retrieve application-specific SNMP statistics and data as defined in the MIB file for those Genesys server applications that support application-specific SNMP requests.
- You can receive alarms from any Genesys server application in the form of SNMP traps.

With all three options, the communications between SCS and NMS require an SNMP Master Agent application that is compliant with the AgentX protocol. If your NMS does not contain such an application, you can use Net-SNMP (release 8.5.1 and later) and/or Genesys SNMP Master Agent to integrate the Management Layer into your NMS. The Genesys MIB file, which the NMS uses, defines the communication interface between the Management Layer and the NMS.

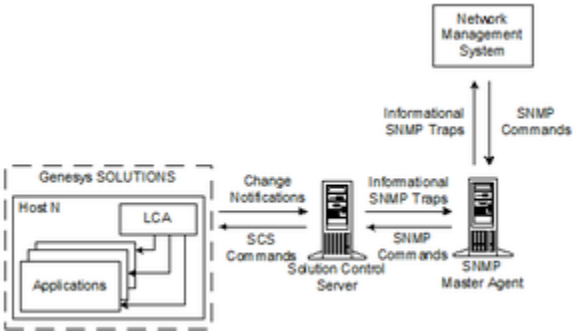
SNMP Command Processing

The figure below illustrates how the Management Layer processes the SNMP commands it receives from an NMS. The commands include:

- Start and stop commands for any Genesys or third-party application that the Management Layer monitors and controls.
- Change of log options settings for Genesys server applications.

With this architecture, you can also:

- View the configuration of any Genesys or third-party product that the Management Layer monitors and controls.
- Monitor the current status of an application (see if it is running or not) and, for redundant configurations, view the current redundancy mode (Primary or Backup) of a running application.
- View the configuration and status of any host registered as a Host object in the Configuration Database, including the LCA configuration of that host.
- View configured solutions and their statuses (see if solutions are running).



Management Layer Processing of an SNMP Command from an NMS

Requesting SNMP Data

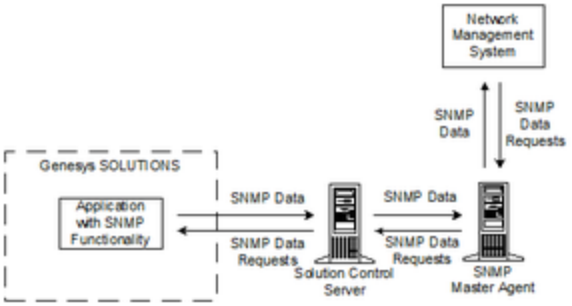
In addition to its application-monitoring functions, you can use the Management Layer to retrieve some SNMP data particular to applications of a given type. For example, you can request from T-Server the number of calls it is currently handling.

You can only retrieve SNMP data and prompt application-specific SNMP traps for applications built with the Genesys management library, such as:

- Call Concentrator
- Configuration Server
- Universal Routing Server
- T-Server

The following diagram illustrates how these applications interact with an NMS. All requests from the NMS and data and traps from the applications come through Solution Control Server.

Consult product-specific documentation to see if your product supports SNMP.

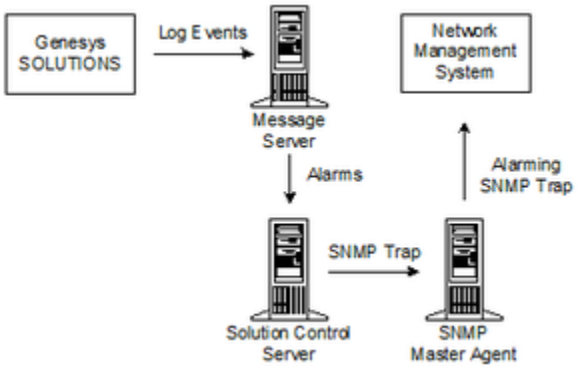


SNMP Information Exchange Between Some Servers and NMS

Alarms and SNMP Trap Processing

To transmit the content of an alarm message to an SNMP-compliant third-party NMS, the Management Layer converts that information into an SNMP trap. An *alarm* is a message generated by a Genesys application when a certain Alarm Condition is met. For more information about alarm signaling, see Alarm-Signaling Functions and *Genesys Administrator Help*.

The following diagram illustrates how the Management Layer reacts to an alarm of type Send an SNMP trap.



Management Layer Processing of an SNMP Trap Alarm Reaction

How to Activate SNMP Support

You must make some changes in your Genesys installation to enable SNMP communications between the Management Layer and your Network Management System.

As already mentioned, the communications between SCS and NMS require an AgentX-compliant SNMP Master Agent application:

- If your NMS already contains such an application, configure an Application object for it in the Configuration Database. See the *Genesys Administrator Help* for instructions.
- If you want to use Genesys SNMP Master Agent or Net-SNMP, deploy it as described in the *Framework*

Deployment Guide.

For either configuration, order licenses that enable the SNMP functionality of the Management Layer and modify the licensing system as needed. Refer to the *Genesys Licensing Guide* for more information.

Stand-alone or Redundant SNMP Master Agents

The Management Layer supports two types of configuration—stand-alone and redundant. Stand-alone configuration consists of a single SNMP Master Agent. Redundant configuration consists of two SNMP Master Agent applications, one primary and one backup. When Solution Control Server loses a connection with the primary SNMP Master Agent, SCS switches all NMS communications to the backup SNMP Master Agent.

> **Important**
>
> You must use 8.5.1 or later versions of SCS and LCA to be able to configure multiple SNMP Master Agent applications as HA pairs that SCS will recognize and connect. With earlier versions of Management Layer components, you must configure each SNMP Master Agent as standalone and make sure autorestart is enabled to allow the automated restart of SNMP in case of failure. In this situation, SCS will always use a single configured SNMP MA to report its status over AgentX protocol.

Both Net-SNMP and Genesys SNMP Master Agent implementations of SNMP support redundant configuration when used with SCS and LCA 8.5.1 or newer. A redundant configuration consists of two SNMP Master Agent Application objects, one primary and one backup. When SCS loses a connection with the primary SNMP Master Agent, SCS switches all NMS communications to the backup. The only difference is in the redundancy type set in the primary Genesys SNMP Master Agent Application object—`Hot Standby` for the Genesys SNMP Master Agent implementation, and `Not Specified` for Net-SNMP.

> **Important**
>
> The mode for both the primary and backup SNMP Master Agents in the HA pair is displayed as Primary.

Refer to the *Framework Deployment Guide* for detailed instructions about deploying both stand-alone and redundant SNMP Master Agents.

Setting Configuration Options

The *Framework Configuration Options Reference Manual* describes configuration options and their values for SNMP Master Agents. You can alter trap configuration (target host, port, community, and multiple trap destinations) by modifying `SNMP_TARGET_MIB`, which you maintain externally with SNMP commands through SNMP Master Agent. Refer to RFC 2273 (SNMPv3 Applications) for further information about how to configure traps via standard management MIBs.

Solution Control Server reads the configuration settings of the SNMP Master Agent Application object and uses option values from the **[agentx]** configuration section to connect to SNMP Master Agent. This is true for both Genesys SNMP Master Agent and a third-party SNMP Master Agent. Therefore, if you are using a third-party SNMP Master Agent, make sure that the option values configured for the SNMP Master Agent Application object in the Configuration Database match the actual configuration settings in your third-party SNMP Master Agent.

How to Use Contact-Center Graceful Shutdown Script

Management Layer provides authorized users with capability to gracefully shut down contact-center software. This functionality, implemented in the form of a PERL script, operates through the Management Layer SNMP Interface.

The Contact-Center Graceful Shutdown script (called `ccgs.pl`) does the following:

1. Enumerates all currently running T-Servers.
2. Determines if there are ongoing interactions in the contact center by querying the number of active calls from each T-Server.
3. If there are active calls, waits one minute and then checks again.
4. When there are no more active calls, shuts down T-Servers.

Installing the Script

If you installed Solution Control Server, `ccgs.pl` is already installed, and is located in the same folder in which Solution Control Server was installed.

Starting in 8.1.2, you can install the Solution Control Server utilities without installing Solution Control Server itself. If you have not installed the utilities, use the procedure “Installing Solution Control Server Utilities” in the *Framework Deployment Guide*. After the utilities are installed, `ccgs.pl` is stored in the location you specified during the installation.

The SCS installation package also includes all additional PERL modules necessary to utilize the Management Layer SNMP Interface with PERL scripts.

Using the Script

Start the script file with the command line in this format:

```
ccgs.pl [<parameter> <value>] [<parameter> <value>] ...
```

Separate parameters from their values with a space.

The script accepts the following command-line parameters:

Contact-Center Graceful Shutdown Script Parameters

Parameter	Description	Default Value
-h	SNMP Master Agent host name or	localhost

	IP address	
-p	SNMP Master Agent SNMP port	161
-c	Community string	public
-v	SNMP version (v1 or v2c)	v2c
-pt	Polling timeout—Time, in seconds, the script waits for the end of data tables refresh; should be equal to or greater than 5.	60
-mic	Maximum number of idle polling cycles the script waits before exit. An idle polling cycle is one when no shutdown requests are sent. Should be equal to or greater than 1.	100
-st	SNMP timeout—Timeout, in seconds, during which the script waits for a response after a request is sent. Note that when a request is retried, the timeout is increased by the SNMP backoff factor (`-sb`). Should be equal to or greater than 1.	2
-sr	SNMP retries—Number of attempts that the script prompts for a reply to the SNMP request. If no response is received within the timeout specified in value -st, the request is resent and a new response awaited with a longer timeout. Should be equal to or greater than 1.	5
-sb	SNMP backoff factor—Used to increase the timeout every time an SNMP request is retried. Should be equal to or greater than 1.	1

SNMP Managed Objects

This section describes the MIB file and the SNMP objects and tables used in the Management Layer to support SNMP.

MIB File

The Genesys MIB file, located in the root of the directory in which SCS is installed, contains all SNMP objects available to the NMS. Genesys MIB utilizes the SMI-v2 Row-Status mechanism and the control/data-tables concept to facilitate management of multiple Genesys servers simultaneously.

RowStatus—a TEXTUAL-CONVENTION defined in IETF's SNMPv2-TC—is commonly used to control the dynamic creation and deletion of rows in SMIv2 defined tables. A conceptual SMIv2 table using RowStatus also acts as a control table or configuration table. Using the control table, the NMS application configures the information to be monitored. A separate data table holds the information that is gathered. One control entry (one row) is linked to the data that is gathered. Each control entry contains control parameters that specify which data or statistics you want to access and collect.

Genesys MIB uses one control table and several data tables. Data tables are organized based on their functional areas and divided into two main groups: *server-generic* data tables and *server-specific* data tables. Each data table, whether it is server-generic or server-specific, is assigned a unique identifier. (See `TableID ::= Textual Convention` in the Genesys MIB file, for the complete list of tables and their identifiers.) This table identifier, along with the Genesys server identifier (server DBID number), is used as an index in the control table. Thus, each row in the control table gathers particular data from a particular Genesys server.

You can enable an automatic refresh of MIB tables. When you set up a row to gather data from a particular table, you have a choice of automatic or manual refresh for the table. If you select Automatic Refresh mode, specify the time period, in seconds, after which Solution Control Server is to refresh the table.

In addition to control and data tables, you can access a group of server standard objects independently of the control/data-table mechanism.

The following sections present information about each object set:

- Standard SNMP Objects describes the standard Genesys SNMP objects.
- gServerControlTable describes the Control Table objects.
- T-Server-Specific SNMP Objects describes the SNMP objects specific to T-Server.

For information about supported traps, see SNMP Traps.

Standard SNMP Objects

The tables in this section describe standard Genesys SNMP objects by object group, as follows:

- **gServersTable**—Contains information about server environments. The following table describes objects that belong to gServersTable. **[+] Show table**

gServersTable Table of Standard SNMP Objects

Object Name	Value Type	Access Level	Description
gsCleanupTimeout	Unsigned 32	read/write	The time, in minutes, the agent should keep rows in the gsControlTable and consequently in related data tables if there were no requests to objects of this row or corresponding rows from data table(s). After the timeout, the agent should automatically delete unattended rows. A value of `0` (zero) specifies that MIB clean up should not be performed.
gServersTable	sequence	read	Specifies a sequence of the following objects.
gServerId	integer	read	Uniquely identifies a Genesys server. Corresponds to the number assigned to an object in the Configuration Database to identify the object among all objects of the same type. The gpServerCurrent object uses this value to switch from one Genesys server to another.
gServerName	string	read	Specifies the application name of a server application as configured in the Configuration Database.
gServerStatus	string	read	Specifies the current operational status of a server. The possible

			settings are UP or DOWN, which indicates if the server is running or not.
gServerType	string	read	Indicates the type of a server; that is, the application type specified for this application in the Configuration Database.
gServerVersion	string	read	Specifies the current version of the running server; that is, the application version specified for this application in the Configuration Database.
gServerWorkDir	string	read	Specifies the server's working directory; that is, the working directory specified for this application in the Configuration Database.
gServerCommandLine	string	read	Indicates the full command line used to start this server, as specified in the Configuration Database. For example: `scs -host host1 -port 4135 -app SCS_Primary`
gServerPID	string	read	Specifies the process ID of the server that is currently running.
gServerCommand	integer	read/write	Specifies the command to start, shut down or gracefully shut down a server. Accepts the following values: • 1 start • 2 shutDown • 3 shutDownGracefully
gServerDeleteClient	integer	read/write	Sends a delete-client

			command to the server. Enter the socket number of a client as the value.

- **gServerControlTable**—Configures the information to be monitored and controls the data-refresh process. The following table describes objects that belong to gServerControlTable. **[+] Show table**

gServerControlTable Table of Standard SNMP Objects

Object Name	Value	Access Level	Description
gServerControlTable	sequence	read	Specifies a sequence of the following objects.
gsCtrlServerID	integer	not-accessible	An index. Specifies the DBID of the server to be managed by this control row. The valid DBID number used to set rows in this control table is retrieved from gServersTable.
gsCtrlTableID	integer	not-accessible	An index. Specifies the data to be gathered for this server. Valid values and the tables they represent are as follows: • 1 – gsLogTable • 2 – gsInfoTable • 3 – gsClientTable • 4 – gsPollingTable • 5 – tsInfoTable • 6 – tsCallTable • 7 – tsDtaTable • 8 – tsLinkTable • 9 – tsCallFilterTable • 10 – tsCallInfoTable • 11 – tsLinkStatsTable Follow the links for detailed information about these tables.
gsCtrlRefreshStatus	integer	read-only	Indicates refresh status of corresponding data table as specified by gsCtrlTableID. The following refresh statuses are reported:

			• 1 – dataNotReady • 2 – dataRefreshInProgress • 3 – dataReady • 4 – mgmtIsNotAvailable • 5 – dataRefreshFailed Refer to the Genesys MIB file for detailed descriptions of these statuses.
gsCtrlLastRefreshed	timetick	read-only	Specifies the time in hundredths of seconds since the row was last successfully refreshed.
gsCtrlRowStatus	RowStatus	read/create	Controls and manages row creation and row deletion. Initiates data-refresh process for a data table managed by this control row, and reports the status of this row. Refer to the Genesys MIB file for a detailed description of the way this object is manipulated.

- **gsInfoTable**—Contains miscellaneous statistics and data about a managed server. The following table describes objects that belong to gsInfoTable. **[+] Show table**

gsInfoTable Table of Standard SNMP Objects

Object Name	Value	Access Level	Description
gsInfoTable	sequence	read	Specifies a sequence of the following objects. Indexed by gsCtrlServerID.
gsClientsExistNum	Unsigned32	read	Indicates the number of clients currently connected to a server.
gsClientsTotalNum	Unsigned32	read	Indicates the total number of clients connected so far to a server.
gsServerConfigFile	string	read	Indicates configuration file name, if any, used to start a server.

- **gsPollingTable**—Specifies a heart-beat feature; that is, a periodic signal sent over a network to the NMS. The following table describes objects that belong to gsPollingTable. **[+] Show table**

gsPollingTable Table of Standard SNMP Objects

Object Name	Value	Access Level	Description
gsPollingTable	sequence	read	Specifies a sequence of the following objects. Indexed by gsCtrlServerID.
gsPollingID	Unsigned32	read/write	Specifies the amount by which each polling signal increases over the last one. The initial polling event equals the same integer. For more information about this variable, see Table 19 on page 106.
gsPollingInterval	Unsigned32	read/write	Specifies the interval, in seconds, between two subsequent polling signals sent from a server. Can be set to any integer.
gsPollingLastTrap	string	read	Specifies the last trap value of a polling signal sent to the NMS. For more information about this variable, see Table 19 on page 106.
gsPollingStatus	string	read/write	Activates or deactivates the server polling feature. Values are ON and OFF. Value ON causes a server to send periodic SNMP signals to SCS, which, in turn, converts these signals into SNMP traps.

- **gsLogTable**—Contains information about log option settings. Values of the SNMP objects in this table correspond to values of configuration options specified in the log section in an Application object's Options. The following table describes objects that belong to gsLogTable; check information in the Description column for the name of the particular option to which an object corresponds. **[+] Show table**

gsLogTable Table of Standard SNMP Objects

Object Name	Value	Access Level	Description
gsLogTable	sequence	read	Specifies a sequence of the following objects. Indexed by gsCtrlServerID.
logVerbose	string	read/write	Log level. Filters output of messages by their priorities. Corresponds

			to the `verbose` log option.
logTrace	string	read/write	Lists the set of log outputs for the log messages of the Trace level. Corresponds to the **trace** log option.
logStandard	string	read/write	Lists the set of log outputs for the log messages of the Standard level. Corresponds to the `standard` log option.
logDebug	string	read/write	Lists the set of log outputs for the log messages of the Debug level. Corresponds to the `debug` log option.
logAll	string	read/write	Lists the set of log outputs for the log messages of all levels. Corresponds to the `all` log option.
logBuffering	string	read/write	Turns on/off OS file buffering. Buffering increases performance of file output; however, log messages may appear in the log with a delay after they have been logged. Corresponds to the `buffering` log option.
logSegment	string	read/write	Sets the mode of log output segmentation. Currently implemented only for the file output. When a currently opened log segment exceeds the size set by this option, the current segment is closed and a new one is created (an empty new segment). Corresponds to the **segment** log option.
logExpire	string	read/write	Sets the expiration mode for old files (segments); that is, specifies whether to remove old files when new ones are created.

			Corresponds to the `expire` log option.
logMessageFile	string	read/write	Sets the name of the file that defines log messages specific to applications of this type. Corresponds to the `messagefile` log option.
logMessageFormat	string	read/write	Specifies the format of log record headers that an application uses when writing logs in the log file. Corresponds to the `message_format` log option.
logTimeFormat	string	read/write	Specifies how to represent in a log file the time when an application generates log records. Corresponds to the `time_format` log option.
logTimeConvert	string	read/write	Specifies in which system an application calculates the log record time when generating a log file. Corresponds to the `time_convert` log option.

Warning

To change log option settings for a particular application via SNMP, you must first associate both the Application object and the Solution Control Server Application object with the account in the Configuration Database having permissions to modify configuration object properties. In other words, the account must have Change permissions for Application objects. Do the following steps:

1. Associate Solution Control Server with an account that has permissions to change Application objects. This association will enable you to change log option settings via SNMP. **[+] Show steps**

Prerequisites

- Management Layer components are installed and running.
- The Solution Control Server Application object has been installed and configured.
- Genesys Administrator is started.

Start

a. Log into Genesys Administrator under a user account having Full Control permissions.
b. Go to Provisioning tab > Applications, and click the SCS application to open its properties.
c. In the Server Info section of the Configuration tab:
   i. If the Log On As SYSTEM checkbox is checked, clear it.
   ii. In the Log On Account field, click the Search icon and select any user account (Person object) from the Browse window. This can be one of the following:
      - An account that belongs to the Administrators default access group.
      - An account that belongs to the role-specific Access Group with the Change permissions you have created for this purpose.
      - An individual account to which you will grant Change permissions in any Access Group.
d. Click Save or Save and Close to save configuration changes.

2. Associate applications with the same account you designated in Step 1 for SCS. **[+] Show steps**

Prerequisites

- Management Layer components are installed and running.
- The Solution Control Server Application object has been associated with an account with application change permissions. See step 1 above.
- You are logged in to Genesys Administrator.

Start

a. Select the Application object which you will be associating with the account you selected in Step 1. You can also select a folder or subfolder that contains Application objects, in which case permissions change for all Application objects in the selected folder.
b. Open the properties of the Application object and click the Permissions tab.
c. Add the user account you designated as This Account for SCS as follows:
   a. Click Add User in the toolbar, and select the user from the Browse window.
   b. Click the entry in the Access column for this user and select Change.
d. Click Save or Save and Close to save your changes.

You could grant change permissions for Application objects to the SYSTEM account, but doing so (and making all servers connect to Configuration Server with change permissions) might impact data security.

- **gsClientTable**—Gathers statistics about server clients. The following table describes objects that belong to gsClientTable. **[+] Show table**

gsClientTable Table of Standard SNMP Objects

Object Name	Value	Access Level	Description
gsClientTable	sequence	read	Specifies a sequence of the following objects. Indexed by gsCtrlServerID.
gsClientAppName	string	read	Specifies the client's application name.

gsClientAuthorized	string	read	Specifies the client's level of authorization.
gsClientGotEvents	Unsigned32	read	Specifies the number of events the client has received.
gsClientSentReqs	Unsigned32	read	Specifies the number of requests the client has sent.
gsClientSocket	Unsigned32	read	Specifies the socket number through which the client is connected to the server.
gsClientType	integer	read	Specifies the client's type.

- **gsAlarmObjectsTable**—Specifies how the Management Layer converts the alarms it generates to SNMP traps and sends them to the NMS. The following table describes objects that belong to gsAlarmObjectsTable. The following lists Genesys application types as they appear in alarm-related traps. **[+] Show table**

gsAlarmObjectsTable Table of Standard SNMP Objects

Object Name	Value	Access Level	Description
gsServersLastAlarm	string	read	Specifies the last trap value sent to the NMS. For information about traps that use this variable, see the table in Application-Generated SNMP Traps.
gsServersLastTrap	string	read	Specifies the last server-status (server up or down) trap sent to NMS.
gsAlarmID	Unsigned32	read	The unique identifier of the Alarm Condition name as configured in the Configuration Database.
gsAlarmLogText	string	read	The text of the log event that triggered this alarm.
gsAlarmMessagesIds	string	read	The unique identifier of the log event that triggered or removed this alarm.
gsAlarmApplicationName	string	read	The name of the application that reported this alarm as specified in the Configuration Database.
gsAlarmApplicationType	string	read	The type of application

			that reported this alarm as specified in the Configuration Database.
gsAlarmCategory	string	read	The alarm category as specified in the Configuration Database: Critical, Major, or Minor.

T-Server-Specific SNMP Objects

The tables in this section summarize the SNMP objects specific to T-Server. These objects give you access to internal T-Server tables that contain information about call states, addresses, and CTI links.

- **tsInfoTable**—Collects miscellaneous data and statistics specific to T-Server. The following table describes objects that belong to tsInfoTable. **[+] Show table**

tsInfoTable Table of T-Server-Specific SNMP Objects

Object Name	Value	Access Level	Description
tsInfoTable	sequence	read	Specifies a sequence of the following objects. Indexed by gsCtrlServerID.
tsCallsExistNum	Unsigned32	read	Specifies the current number of calls being handled by T-Server.
tsCallsTotalNum	Unsigned32	read	Specifies the number of calls T-Server has handled since it started.
tsLinksCommand	string	read/write	Specifies the command to be sent to T-Server. **Note:** Reserved for future use.
tsLastChangedLink-Status[a]	string	read	Specifies the server name, link name, and link's new status. **Note:** Reserved for future use.

[a] This object is specific to environments with X.25 links.

- **tsCallFilterTable**—Supports stuck calls functionality by providing the interface for setting call filter criteria for the tsCallInfoTable table, and for cleaning calls by their ConnectionID. The following table describes objects that belong to tsCallFilterTable. **[+] Show table**

tsCallFilterTable Table of T-Server-Specific SNMP Objects

Object Name	Value	Access Level	Description
gsCtrlServerId	ServerDBID	not-accessible	Uniquely identifies a T-Server application.
fltCallCreatedBefore	Unsigned32	read-write	Reports the calls that were created earlier than a specified number of seconds counting from the time of the request. A 0 (zero) value means the filter is not used.
fltCallCreatedAfter	Unsigned32	read-write	Reports the calls that were created later than a specified number of seconds counting from the time of the request.
fltCallUpdatedBefore	Unsigned32	read-write	Reports the calls that were last time updated earlier than a specified number of seconds counting from the time of the request.
fltCallUpdatedAfter	Unsigned32	read-write	Reports the calls that were last time updated later than a specified number of seconds counting from the time of the request.
clearCallByConnId	DisplayString	read-write	Connection ID (converted to string by `connid_to_str` function) of the call to be cleared.

- **tsCallInfoTable**—Supports stuck calls functionality by storing the latest snapshot of active calls from a given T-Server, and contains a set of attributes that facilitates the discovery of stuck calls. The following table describes objects that belong to tsCallInfoTable. **[+] Show table**

tsCallInfoTable Table of T-Server-Specific SNMP Objects

Object Name	Value	Access Level	Description
gsCtrlServerId	ServerDBID	not-accessible	Uniquely identifies a T-Server application.
callInfoInstanceID	Unsigned32	not-accessible	Reports the call instance ID.
callInfoType	Unsigned32	read-only	Reports a call type.
callInfoCreationTimestamp	Unsigned32	read-only	Reports a call creation timestamp.
callInfoLastUpdatedTimestamp	Unsigned32	read-only	Reports a timestamp of the last update on this

			call.
callInfoInternalParties	DisplayString	read-only	Reports an Internal DN.

- **tsLinkStatsTable**—Stores information about the current statistics for messages sent and received over a T-Server link. The following table describes objects that belong to tsLinkStatsTable. **[+] Show table**

tsLinkStatsTable Table of T-Server-Specific SNMP Objects

Object Name	Value	Access Level	Description
gsCtrlServerId	ServerDBID	not-accessible	Uniquely identifies a T-Server application.
LinkID	Unsigned32	read-only	Uniquely identifies a T-server link to which these statistics apply.
timeElapsedSec	Unsigned32	read-only	The time (in seconds) that have elapsed since the last statistics were measured.
numberMessagesTx	Unsigned32	read-only	The number of CTI messages that have been sent by T-Server over this link since the last statistics were measured.
rateMessagesTx	Unsigned32	read-only	The rate at which CTI messages are sent by T-Server over this link. Literally, the ratio of numberMessagesTx to timeElapsedSec.
numberMessagesRx	Unsigned32	read-only	The number of CTI messages that have been received by T-Server over this link since the last statistics were measured.
rateMessagesRx	Unsigned32	read-only	The rate at which CTI messages are received by T-Server over this link. Literally, the ratio of numberMessagesRx to timeElapsedSec.

- **tsCallTable**—Contains data about telephony calls being processed by T-Server. The following table describes objects that belong to tsCallTable. **[+] Show table**

tsCallTable Table of T-Server-Specific SNMP Objects

Object Name	Value	Access Level	Description
tsCallTable	sequence	read	Specifies a sequence of the following objects.

			Indexed by gsCtrlServerID and callInstanceID.
callANI	string	read	Automatic Number Identification. Provides calling party information (typically, the telephone number or billing account number) to the called party.
callCallID	string	read	Specifies the current call identifier that the switch has assigned to a call.
callConnID	string	read	Specifies the identifier that T-Server has assigned to a call.
callCustomerID	string	read	Specifies the Customer (Tenant) identifier used when a call was initiated.
callDNIS	string	read	Directory Number Identification Service. Identifies to the called system the last three or four digits of the number actually dialed by the caller.
callFirstTransferDN	string	read	Specifies the DN on a remote T-Server from which a call was first made.
callFirstTransfer-Location	string	read	Specifies the location of the remote T-Server from which a call was first transferred.
callInstanceID	counter	read	Specifies the instance number for each call.
callLastTransferDN	string	read	Specifies the DN on a remote T-Server from which a call was last transferred.
callLastTransfer-Location	string	read	Specifies the location of the remote T-Server from which a call was last transferred.
callNumParties	string	read	Specifies the number of parties currently involved in a call.
callPartiesList	string	read	Specifies a list of

			parties involved in a call.
callReferenceID	string	read	Specifies the reference ID of a call.
callTimeStamp	string	read	Specifies the timestamp of when the call was created, in seconds starting from January 1, 1970.
callType	string	read	Specifies the type of a call.
callState	string	read	Specifies the current state of the call in question.

- **tsDtaTable**—Stores information about all registered DNs. The following table describes objects that belong to tsDtaTable. **[+] Show table**

tsDtaTable Table of T-Server-Specific SNMP Objects

Object Name	Value	Access Level	Description
tsDtaTable	sequence	read	Specifies a sequence of the following objects. Indexed by gsCtrlServerID and tsDtaInstanceID.
tsDtaDigits	string	read	Specifies the digits field of the DTA structure.
tsDtaInstanceID	counter	read	Specifies the instance field of the DTA structure.
tsDtaMode	string	read	Specifies the mode field of the DTA structure.
tsDtaState	string	read	Specifies the state field of the DTA structure.
tsDtaType	string	read	Specifies the type field of the DTA structure.

- **tsLinkTable**—Contains information about the CTI links that exist between T-Server and switches, and the attributes of those links. The following table describes objects that belong to tsLinkTable. **[+] Show table**

tsLinkTable Table of T-Server-Specific SNMP Objects

Object Name	Value	Access Level	Description
tsLinkTable	sequence	read	Specifies a sequence of the following objects. Indexed by gsCtrlServerID and tsLinkID.
tsLinkAddress	string	read	Specifies the address of the link.

tsLinkDelay[a]	string	read	Specifies a link reconnect delay in the case of an unsuccessful attempt to reconnect to the line.
tsLinkDTEClass[a]	string	read	Specifies the DTE class for the X.25 connection.
tsLinkName	string	read	Specifies the name of the link.
tsLinkID[a]	integer	read	Specifies the link identifier. **Note:** Reserved for future use.
tsLinkMode[a]	string	read	Specifies the mode of the link.
tsLinkPID[a]	string	read	Specifies the link's process ID.
tsLinkPort	string	read	Specifies the physical port number of the link.
tsLinkProtocol	string	read	Specifies a protocol type.
tsLinkSocket[a]	string	read	Specifies the socket number through which the server is connected to the link. **Note:** Reserved for future use.
tsLinkStatus	string	read	Specifies the current status of the link. The status property is used in fault monitoring, providing the NMS with information about which links are currently established and which links have lost physical or logical connection. Valid values: • 0 – Link is not configured properly. • 1 – No physical or TCP/IP connection between T-Server and the switch. • 2 – Both physical

			and logical connections are OK. • 3 – Physical connection exists between T-Server and switch, but the logical connection is missing.
tsLinkTemplate	string	read	Specifies a template for the connection.
tsLinkX25Device[a]	string	read	Specifies the X.25 device connected to the link.
tsLinkX25LocalAddress[a]	string	read	Specifies the local address for an X.25 connection.

a. This object is specific to environments with X.25 links.

SNMP Traps

There are two types of SNMP trap messages you can receive from Genesys applications:

- Application-generated—SNMP traps that only those server applications listed here and built with the management library can generate.
- Alarm-related—SNMP traps that SCS generates on behalf of any Genesys server it monitors.

This section describes both types of SNMP traps, and lists the application types for server applications as they are displayed in SNMP traps.

Application-Generated SNMP Traps

Application-generated SNMP traps are generated only by those server applications that are listed here and that are built with the management library.

Application-Generated SNMP Traps

Trap	Variable Name	String Format and Valid Values	Description
gsServerUpTrap	gsServersLastTrap	String format: `<server name>:server is UP`	Reports that a server was down but is running again. The `<server name>` in the message is the server's application name in the Configuration Database. For example: `tserver_1:server is UP`
gsServerDownTrap	gsServersLastTrap	String format: `<server name>:server is DOWN`	Reports that a server has gone down. The `<server name>` in the message is the server's application name in the Configuration Database. For example: `tserver_1:server is DOWN`
gsAlarm	gsServersLastAlarm	String format: `<server name>:0:<alarm code>:<alarm message>`	Reports that a server has encountered an alarm situation. A server itself generates this trap. Do not confuse with `gSmlSCserverAlarm` which is generated by Solution Control Server

			based on log events it receives from other servers.
tsLinkStatusTrap	tsLastChangedLinkStatus	String format: `server <server name> (<server number>) - link <link name> status changed -` <0,1> (<DOWN,UP>)	Reports that T-Server link status has been changed.
gsPollingSignal	gsPollingLastTrap	String format: `<server name>:<value>` Valid Value: Any positive integer	Sent in response to a polling signal from the server. If the polling ID status is set to ON (see gsPollingTable), the server's polling ID increases by a set amount every time the server sends the polling signal. The server then sends the new value to SCS. The new value becomes the value inside of the variable gpPollingLastTrap, which SCS sends to NMS as the trap gpPollingSignal. The server name in the message is the server's application name in the Configuration Database. For example: **tserver_1:122** This message means that the last SNMP polling signal, which the T-Server application named `tserver_1` sent to the SCS, was number 122.

Alarm-Related SNMP Traps

Alarm-related SNMP traps are SNMP traps that SCS generates on behalf of any Genesys server it monitors.

Application-Generated SNMP Traps

Trap	Variable Name	String Format and Valid Values	Description
gsMLAlarm	gsAlarmId	integer	The unique identifier of the Alarm Condition name as configured in the Configuration Database.

	gsAlarmLogText	string	The text of the log event that triggered this alarm.
	gsAlarmMessagesIds	string	The unique identifier of the log event that triggered or removed this alarm.
	gsAlarmApplicationName	string	The name of the application that reported this alarm as specified in the Configuration Database.
	gsAlarmApplicationType	string	The type of application that reported this alarm as specified in the Configuration Database.
	gsAlarmAppHostName	string	The host name of the application that reported this alarm.
	gsAlarmCategory	string	The alarm category as specified in the Configuration Database: Critical, Major, or Minor.
	gsAlarmGUID	string	The unique identifier of the Alarm Clearance Trap with Alarm Creation Trap.

Application Types in SNMP Traps

The following tables lists application types for server applications—their database IDs and abbreviations—as they are displayed in alarm-related SNMP traps. The table also presents the application type names as displayed by the Configuration Layer.

Application Type Representations

Type ID	Type Abbreviation	Type Name
01	TServer	T-Server
01	N/A	Programmable Gateway Framework
02	StatServer	Stat Server
03	BillingServer	Billing Server
06	VoiceTreatmentServer	Voice Treatment Server
08	DBServer	Database Access Point
09	CallConcentrator	Call Concentrator
10	SDialer	CPD Server

11	ListManager	List Manager
12	OSServer	Outbound Contact Server
15	RouterServer	Universal Routing Server
21	ConfigurationServer	Configuration Server
23	ThirdPartyServer	Third Party Server
26	DARTServer	Obsolete
28	CustomServer	Custom Server
29	ExternalRouter	External Router
31	VirtualRP	Virtual Routing Point
32	Database	Obsolete
33	NetVector	Web Option
34	DetailBiller	Detail Biller
35	SummaryBiller	Summary Biller
36	NACD	Network Overflow Manager
37	BackUpControlClient	Backup Control Client
38	InfomartStatCollector	CC Analyzer Data Sourcer
40	IVRInterfaceServer	IVR Interface Server
41	IServer	I-Server
42	MessageServer	Message Server
43	SCS	Solution Control Server
45	SNMPAgent	SNMP Agent
46	RealDBServer	DB Server
48	WFMDataAggregator	WFM Data Aggregator
50	WFMScheduleServer	WFM Schedule Server
52	ETLProxy	ETL Proxy
54	GVCServer	GVP-Voice Communication Server
55	VSSSystem	VSS System
58	CCAnalyzerDataMart	CC Analyzer Data Mart
59	ChatServer	Chat Server
60	CallbackServer	Callback Server
61	CoBrowsingServer	Co-Browsing Server
62[a]	SMSServer	SMS Server
63	ContactServer	Contact Server
64	EmailServer	E-Mail Server
65	MediaLink	MediaLink
66	WebInteractionRequestsServer	Web Interaction Requests Server
67	WebStatServer	Web Stat Server
68	WebInteractionServer	Web Interaction Server

69	WebOptionRoutePoint	Web Option Route Point
74	VoIPController	Voice over IP Controller
77	HAProxy	High Availability Proxy
78	VoIPStreamManager	Voice over IP Stream Manager
79	VoIPDMXServer	Voice over IP DMX Server
80	WebAPIServer	Web API Server
81	LoadBalancer	Load Balancer
82	ApplicationCluster	Application Cluster
83	LoadDistributionServer	Load Distribution Server
84	GProxy	G-Proxy
85	GIS	Genesys Interface Server
86	AgentDesktopDeliveryServer	GCN Delivery Server
88	IVRDT	IVR DirectTalk Server
89	GCNThinServer	GCN Thin Server
90	ClassificationServer	Classification Server
91	TrainingServer	Training Server
92	UniversalCallbackServer	Universal Callback Server
93	CPDServerProxy	CPD Server Proxy
94	XLinkController	XLink Controller
95	KWorkerPortal	K-Worker Portal
96	WFMServer	WFM Server
97	WFMBuilder	WFM Builder
98	WFMReports	WFM Reports
99	WFMWeb	WFM Web
100	KnowledgeManager	Knowledge Manager
101	IVRDriver	IVR Driver
102	IVRLibrary	IVR Library
103	LCSAdapter	LCS Adapter
104	DesktopNETServer	Desktop NET Server
105	Siebel7ConfSynchComponent	Siebel7 ConfSynchComponent
106	Siebel7CampSynchComponent	Siebel7 CampSynchComponent
107	GenericServer	Generic Server
108	GenericClient	Generic Client
109	CallDirector	Call Director
110	SIPCommunicationServer	SIP Communication Server
111	InteractionServer	Interaction Server
112	IntegrationServer	Integration Server
113	WFMDaemon	WFM Daemon

114	GVPPolicyManager	GVP Policy Manager
115	GVPCiscoQueueAdapter	GVP Cisco Queue Adapter
116	GVPTextToSpeechServer	GVP Text To Speech Server
117	GVPASRLogManager	GVP ASR Log Manager
118	GVPBandwidthManager	GVP Bandwidth Manager
119	GVPEventsCollector	GVP Events Collector
120	GVPCacheServer	GVP Cache Server
121	GVPASRLogServer	GVP ASR Log Server
122	GVPASRPackageLoader	GVP ASR Package Loader
123	GVPIPCommunicationServer	GVP IP Communication Server
124	GVPResourceManager	GVP Resource Manager
125	GVPSIPSessionManager	GVP SIP Session Manager
126	GVPMediaGateway	GVP Media Gateway
127	GVPSoftSwitch	GVP Soft Switch
128	GVPCoreService	GVP Core Service
129	GVPVoiceCommunicationServer	GVP Voice Communication Server
130	GVPUnifiedLoginServer	GVP Unified Login Server
131	GVPCallStatusMonitor	GVP Call Status Monitor
132	GVPReporter	GVP Reporter
133	GVPH323SessionManager	GVP H323 Session Manager
134	GVPASRLogManagerAgent	GVP ASR Log Manager Agent
135	GVPGenesysQueueAdapter	GVP Genesys Queue Adapter
136	GVPIServer	GVP IServer
137	GVPSCPGateway	GVP SCP Gateway
138	GVPSRPServer	GVP SRP Server
139	GVPMRCPTTSServer	GVP MRCP TTS Server
140	GVPCCSServer	GVP CCS Server
141	GVPMRCPASRServer	GVP MRCP ASR Server
142	GVPNetworkMonitor	GVP Network Monitor
143	GVPOBNManager	GVP OBN Manager
144	GVPSelfServiceProvisioningServer	GVP Self Service Provisioning Server
145	GVPMediaControlPlatform	GVP Media Control Platform
146	GVPFetchingModule	GVP Fetching Module
147	GVPMediaControlPlatformLegacyInterpreter	GVP Media Control Platform Legacy Interpreter
148	GVPCallControlPlatform	GVP Call Control Platform
149	GVPResourceManager	GVP Resource Manager
150	GVPRedundancyManager	GVP Redundancy Manager

151	GVPMediaServer	GVP Media Server
152	GVPPSTNConnector	GVP PSTN Connector
153	GVPReportingServer	GVP Reporting Server
154	GVPSSG	GVP SSG
157	InteractionWorkspace	Interaction Workspace
158	Advisors	Advisors
159	ESSExtensibleServices	ESS Extensible Services
160	CustomerView	Customer View
161	OrchestrationServer	Orchestration Server
162	Reserved	
163	CapturePoint	Capture Point
164	RulesESPServer	Rules ESP Server
165	GenesysAdministrator	Genesys Administrator
166	iWDManager	iWD Manager
167	iWDRuntimeNode	iWD Runtime Node
168	BusinessRulesExecutionServer	Business Rules Execution Server
169	BusinessRulesApplicationServer	Business Rules Application Server
170	VPPolicyServer	VP Policy Server
171	SocialMS	Social Messaging Server
172	CSTAConnector	CSTA Connector
173	VPMRCPProxy	VP MRCP Proxy
174	UCMConnector	UCM Connector
175	OTICSServer	OT ICS Server
176	OTICSOMPInfra	OT ICS OMP Infrastructure
177	Advisors-CCAdviser	Advisors–Contact Center Advisor
178	Advisors-FAAdviser	Advisors–Frontline Advisor
179	Advisors-AdvisorsPlatform	Advisors–Advisors Platform
180	Advisors-AdvisorsGenesysAdapter	Advisors–Advisors Genesys Adapter
181	Advisors-AdvisorsCiscoAdapter	Advisors–Advisors Cisco Adapter
182	FederationServer	Federation Server
183	FederationStatProvider	Federation Stat Provider
184	GenesysAdministratorServer	Genesys Administrator Server
185	WebEngagementBackendServer	Web Engagement Backend Server
186	WebEngagementFrontendServer	Web Engagement Frontend Server
187	WebRTCGateway	WebRTC Gateway
188	LRMServer	LRM Server

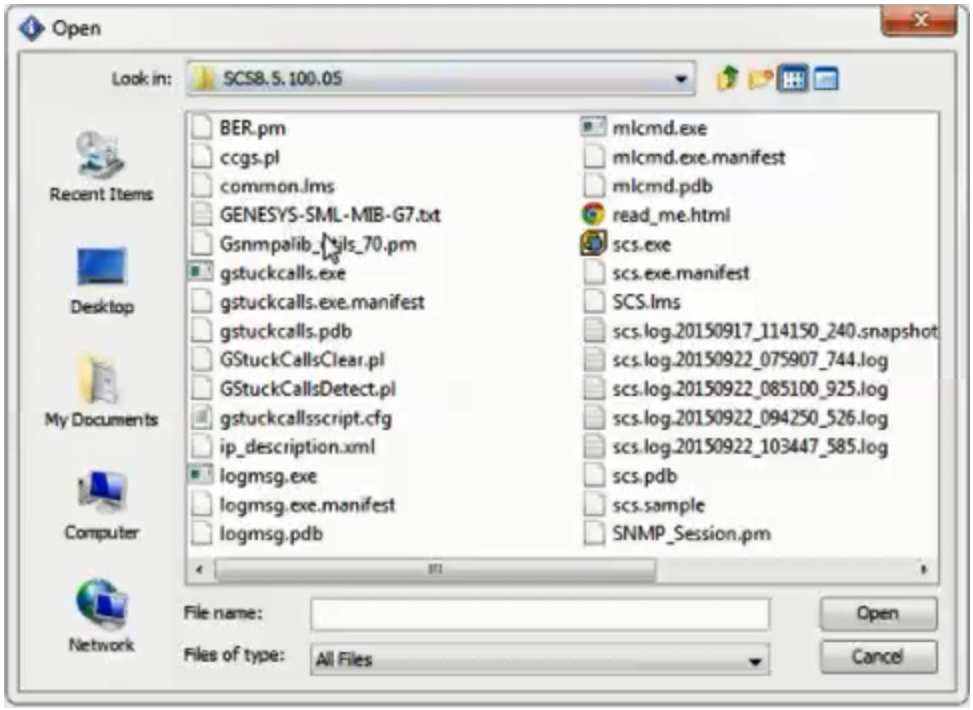
189	RecordingCryptoServer	Recording Crypto Server
190	GenesysKnowledgeCenterServer	Genesys Knowledge Center Server
191	GenesysKnowledgeCenterCMS	Genesys Knowledge Center CMS

[a.] Prior to release 8.0, this value was used for the Application Type IS Transport Server. In release 8.0 and later, this value is used for the new application type SMS Server.

Configure SNMP Monitoring

Configuring SNMP Monitoring for T-Servers and SIP Server requires the skills and knowledge of a system administrator, and knowledge of the Genesys installation that is running on your site. Ideally, you or a current colleague installed the Genesys software, so you have—or can easily find—information that you need, such as the location of certain files. You also need an MIB Editor; ideally, one that you have used before.

Before you configure



Verify or perform these requirements:

- Your Genesys software installation must include a current Media Layer SNMP license. Verify with your Genesys marketing representative.
- The applications SNMP Master Agent and Solution Control Server must be installed, configured and running. **[+] Show steps**

  The Framework Deployment Guide describes how to deploy and configure the Solution Control Server (SCS) and the SNMP Master Agent. See steps 5 and 6 in Deployment Summary.
- The SIP Server(s)/TServer(s) that you wish to monitor must be provisioned with *<TServer>*\management-port and restarted.
  For each SIP Server and T-Server, configure <TServer>\management-port. Follow the steps on page

653, and the option settings on page 657, of the SIP Server 8.1 Deployment Guide.

[+] Show steps

You can change T-Server common configuration options in the Options section of the Application object (for example: SNMP-MA, for the master SNMP).

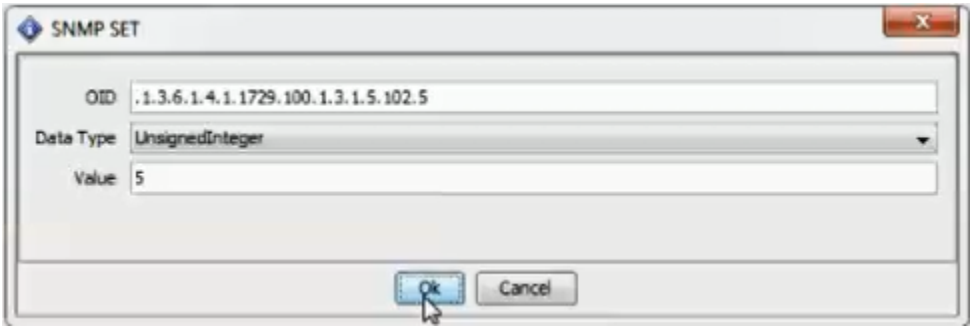
Find the Application object in this directory:

- **If you use Genesys Administrator:** `Application object > Options tab > Advanced View (Options)`
- **If you use Configuration Manager:** `Application object > Properties dialog box > Options tab`

Option: management-port
Default Value: `0`
Valid Values: `0` or any valid TCP/IP port
Changes Take Effect: after T-Server is restarted
This option specifies the TCP/IP port that management agents use to communicate with T-Server. If set to `0` (zero), this port is not used.

- The Network Management System (NMS) must be connected to the SNMP Master Agent. See How to Activate SNMP Support.
- Import the file **GENESYS-SML-MIB-G7.txt** to NMS. Find this file in the working directory of the Solution Control Server application; the person(s) who installed Genesys for your company should know the location. This file contains Genesys-specific SML, MIB, and G71 definitions.

Summary



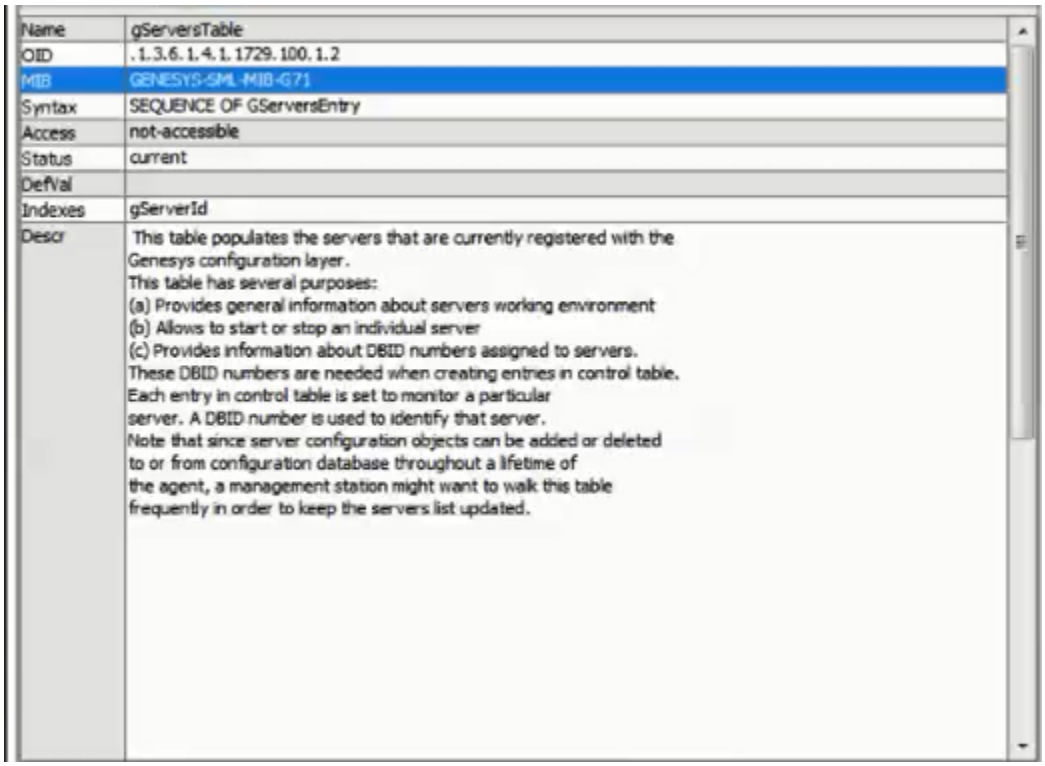
For each SIP Server / TServer...

1. Query the gServersTable and record the value of: gServerId (<*index*>).
2. Set the value 4 (createAndGo) for: Name/OID: `gsCtrlRowStatus.<index>.5;`
3. Set the refresh timeout. For example, 5 seconds for: Name/OID: `gsCtrlAutomaticRefresh.<index>.5 for tsInfoTable`

Consider the three steps above as general directions which you can execute in the most efficient way for you. The Summary is intentionally generic because every MIB editor is different.

Consider the five steps below as an example of how one system administrator might configure SNMP Monitoring for T-Servers and SIP Server. The real steps that you use to access SNMP data will depend on your site's NMS implementation and on your MIB editor.

Example of configuring SNMP monitoring



Your actual steps will be similar:

1. With the MIB Editor, **Load** the imported file GENESYS-SML-MIB-G7.txt.
2. Inside GENESYS-SML-MIB-G7.txt, find and read the DESCRIPTION for gServerControlTable and gsCtrlRowStatus.
   The description contains information about a table's contents, purpose, and behavior—and how to request data from NMS.
   [+] Show table description

```
gServerControlTable OBJECT-TYPE
...
DESCRIPTION
"Control table containing a set of parameters to set up and control data collection from Genesys server(s). This table facilitates the monitoring of multiple Genesys servers.

The following data tables defined in this MIB module are controlled by the gsControlTable:

        gsLogTable

        gsInfoTable
```

gsClientTable

gsPollingTable

tsInfoTable

tsCallTable

tsDtaTable

tsLinkTable

tsCallFilterTable

tsCallInfoTable

tsLinkStatsTable

Entries in the above tables are created on behalf of an entry in the gServerControlTable. If a control row is destroyed, then corresponding row in the respective data table is destroyed too.

Some tables defined in this MIB module may just be used to perform some management function on a server (for example,gsPollingTable). When a management station creates a row in the control table for such a table, the agent will create a row in the corresponding table thus making it ready to be used to perform a management function for a selected server. This way, we are allowed to create multiple instances of a management object that can be used to perform a management function for multiple servers.

For example, in case of gsPollingTable a manager will be able to perform reconfig command for multiple Stat Servers."

3. Get the value of gServerId(s) of SIP Server(s) / TServer(s) from the gServersTable.
   [+] Show steps

   Scan the Table view for T-Servers or SIP Servers (in the column gServerType) that have "start" in the gServerCommandLine column. These should appear in the topmost rows of the Table View.

   In the partial screen shot above, Line 2 reads "T-Server" in the gServerType column and "start" in the gServerCommandLine column. So that line contains the information that you need.

   The T-Server in this row of data has an identifying number that appears in both the gServerId column and the Index column. You will need that value later.

   In the example, that value is: 102.

4. Then set the value 4 (createAndGo) for Name/OID: `gsCtrlRowStatus.<index>.5`;
   ...where <index> is the value of `gServerId` from the step above, to initiate the data collection for `tsInfoTable`.

   Set other tables in the same way. For example: gsCtrlRowStatus.<index>.6 for tsCallTable(6).

   [+] Show steps

   Go to this directory: `MIB Tree > iso.org.dod.internet > private > enterprises > genesys > servers > genericServer > gsCleanupTimeout > gServerControlTable > gServersEntry > gsCtrlRowStatus`

   Select Set from the Operations drop-down. In the SNMP Set dialog box, find the OID field and append the number in it with this: ".102.5" (102 is the gServerId for the T-Server that you found in the previous step.

   In the value field, enter 4 (for and ".4" (for createAndGo).

5. Set the refresh timeout to 5 seconds for Name/OID: `gsCtrlAutomaticRefresh.<index>.5` for tsInfoTable

   Use a similar method to set other tables, for example: `gsCtrlAutomaticRefresh.<index>.6` for tsCallTable(6).

Genesys MIB File Changes

The Genesys 7 MIB (Management Information Base) file used for SNMP support is built into the latest release of Management Layer. This file is installed automatically, and resides in the root directory where Solution Control Server is installed.

This section lists the differences in Genesys MIB files between releases.

Changes in 8.5

There are no changes in the Genesys MIB file in release 8.5.

Changes in 8.1

There were no changes in the Genesys MIB file in release 8.1.

Changes in 8.0

In release 8.0, the Genesys MIB file was modified as follows:

- `apps OBJECT IDENTIFIER ::= { genesys 200 }` was added to provide an extension point for applications. This enables an application, such as Genesys Voice Platform, to register additional MIB regions that are controlled by the application itself (not by Solution Control Server).
- The `gServerStatus` object was extended with the following new status levels:
  - `statusSuspending(7)`
  - `statusSuspended(8)`
- The table **tsLinkStatsTable** was added to provide current statistics about the T-Server Link.

Changes in 7.6

There were no changes in the Genesys MIB file in release 7.6.

Troubleshooting

This section contains suggestions on how to identify and handle the most common mistakes made when you are enabling the Management Layer functionality.

Major Checkpoints

The Management Layer must be configured correctly to function properly. When you are configuring the Management Layer, ensure that it operates properly by using the following checklist:

- SQL server is running and configured properly.
- Configuration Server is running.
- Solution Control Server (SCS) is running.
- Local Control Agent (LCA) is running with sufficient permissions on each monitored host.
- At least one instance of Message Server is running.
- Message Server, used for centralized logging, has the **db_storage** configuration option set to `true`. (Check the `messages` section on the **Options** tab of the Message Server Application object.)
- The Log Database scripts have been executed successfully.
- The user account that is specified in the **DB Info** section of the Database Access Point Application object, and is used for accessing the Log Database, has Write permissions configured in the Database Management System (DBMS).
- All monitored applications have the verbose configuration option set to a value other than none. (Check the `log` section in the **Options** tab of the Application object.)
- All monitored applications have the network output type specified. (Check the `log` section in the **Options** tab of the Application object.)
- A connection to Message Server is configured in the Application object's **Connections**.
- A connection to Message Server is configured in the SCS Application object's **Connections**.
- In Genesys Administrator:
  - A connection to the correct SCS is configured in the Configuration Manager Application object's **Connections**.
  - The same Database Access Point Application object is specified in the **Connections** of both Configuration Manager and Message Server Application objects.

> **Important**
>
> Refer to the *Framework Configuration Options Reference Manual* for configuration option descriptions and information about their valid values.

Application Start/Stop

This section suggests actions to take if you have difficulty enabling Management Layer's control functionality.

Applications Cannot Be Started

- Make sure that LCA is running on the host on which the application is installed.
- Check the SCS log to make sure that connection to the LCA running on the application's host is established.
- Make sure that the command-line parameters are specified in the Application object's properties.
- Make sure LCA has sufficient permissions to start an application.
- Make sure that the application is installed on the host specified in the Application object's **Start Info** section in Genesys Administrator.

Applications Cannot Be Stopped

- Make sure LCA has sufficient permissions to stop an application.

Connectivity Failure and Microsoft's Media-Sense Feature

In the Microsoft Windows XP operating system, when a host is disconnected from the network, LCA may start second instances of Solution Control Server and Configuration Server even though these components are already running. That in itself is harmless because Framework detects and terminates the additional CS instance and the additional SCS instance never connects to Framework.

The probable cause of this effect is Microsoft's media-sense feature, included with Windows XP system, which enables a NIC (Network Interface Card) to detect if a network cable is connected to it. The default setting of this feature (`on`) has these effects:

- If any cable is disconnected, Windows disables the protocols on the adapter, which affects TCP/IP (although loopback of 127.0.0.1 in your HOSTS file still works). This affects applications that require IP connectivity to remain constant; for example, laptops.
- If a network cable is disconnected, Windows also disables the entire network protocol stack, which means that you cannot reach network addresses on your own system.

If you find these effects undesirable, you should disable media-sense. Use the following steps to do this for systems that use TCP/IP.

1. Start the registry editor (`regedit`).
2. Move to `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters`
3. From the **Edit** menu select `New - DWORD value`.
4. Name the new item `DisableDHCPMediaSense` and press **Enter**.
5. Double click the new value and set it to 1.

6. Click **OK**.
7. Reboot the computer.

Point your browser to www.microsoft.com, and search for *media-sense* to read more information about disabling this feature.

Alarming

This section suggests actions to take if you have difficulty enabling Management Layer's alarm-signaling functionality.

No Active Alarms in Genesys Administrator

- Check the configuration of the Alarm Condition Application object and make sure that the correct Detect Log Event ID is specified.
- Make sure that the log message appears in a local log file.
- Make sure that all monitored applications have the network output type. (Check the `log` section in the Application object's **Options** tab.)
- Make sure that the **verbose** configuration option is set to send log messages of the needed level.
- Check the Message Server log to make sure that Message Server receives log messages.
- Make sure that the correct Message Server is configured in the SCS Application object's **Connections**.
- Check the Message Server log to make sure that Message Server sends messages to SCS.
- In Genesys Administrator, make sure that the correct SCS is configured in the Configuration Manager Application object's **Connections**.

No Alarm Reactions Executed

- Make sure that the alarm is triggered (see No Active Alarms in Genesys Administrator).
- Make sure that a Script object of the Alarm Reaction type is specified on the **Reaction Scripts** tab of the Alarm Condition's properties.

No Alarm Reactions "Send SNMP Trap" Executed

- Make sure that the alarm is triggered (see No Active Alarms in Genesys Administrator).
- Make sure that a Script object of the Alarm Reaction type is specified on the **Reaction Scripts** tab of the Alarm Condition's properties.
- Make sure that the Genesys or a third-party SNMP Master Agent is installed and configured as required. See SNMP Support.

No Alarm Reactions “Send E-Mail” Executed

- Make sure that the alarm is triggered (see No Active Alarms in Genesys Administrator).
- Make sure that a Script object of the Alarm Reaction type is specified on the **Reaction Scripts** tab of the Alarm Condition’s properties.
- Make sure that the email system is installed and configured as required. (See Configuring E-mail Systems.)
- Make sure that the correct email address is specified for the Alarm Reaction Script object.

Alarm Reaction

This section suggests actions to take if you have difficulty enabling alarm reactions of the Send an email and Send an SNMP Trap types.

E-Mail

- Make sure the host on which SCS is running has an email system, which is installed, configured, and running correctly.

See also E-Mail Alarm Reactions.

SNMP Traps

- Make sure that the SNMP Master Agent Application object is configured in the SCS Application object’s **Connections**.
- Make sure that the host and port parameters are specified correctly on the SNMP Master Agent Application object’s **Server Info** section in Genesys Administrator).
- Make sure that the configuration options are set correctly in the SNMP Master Agent Application object’s **Options** tab.

See also SNMP Traps.

Logging

This section suggests actions to take if you have difficulty enabling Management Layer’s logging functionality.

No Application Logs

- Check the `log` section in the Application object’s **Options** tab and make sure that the **verbose** configuration option is set to a value other than `none`.

- Check the `log` section in the Application object's **Options** tab and make sure that at least one output type is specified.

No Log Messages in Genesys Administrator

- Check the `log` section in the Application object's **Options** tab and make sure that the **verbose** configuration option is set to a value other than `none`. Then check the value of the configuration option that corresponds to the value of **verbose** and make sure that it is set to **network**.
- Make sure that Message Server is configured in the Application object's **Connections**.
- Check the `messages` section in the Message Server Application object's **Options** tab and make sure that the **db_storage** configuration option is set to `true`.
- Make sure that a user with Write permission is configured in the DBMS and that the same user account is specified in the **DB Info** section of the Database Access Point.
- In Genesys Administrator, make sure that the same Database Access Point Application object is specified in the **Connections** of both Configuration Manager and Message Server.

Distributed SCS Functionality

This section suggests actions to take if you have difficulty enabling Distributed mode for Solution Control Servers that control your environment. Refer to the *Framework Deployment Guide* for detailed instructions and information about configuring distributed Solution Control Servers.

Incorrect Message Server Configuration

- Make sure you have a Message Server dedicated to support communications among Distributed Solution Control Servers. Verify that the **signature** configuration option is set to the **scs_distributed** value in the `MessageServer` section in that Message Server Application object's Options tab.

Incorrect SCS Configuration

- Make sure that the **distributed_mode** configuration option is set to the `ON` value in the `general` section in each configured SCS Application object's Options.
- Make sure that the Message Server Application object that you dedicated to support Distributed SCS communications is specified on each configured SCS Application object's **Connections**.

Incorrect SCS Role Configuration

- If you decide not to have a main Distributed SCS control all unassigned configuration objects, make sure that the **distributed_rights** configuration option is either not configured or not set to the `MAIN` value in the `general` section in each configured SCS Application object's Options.
- If you decide to have a main Distributed SCS control all unassigned configuration objects, make sure that the **distributed_rights** configuration option in the general section is set to:
  - The `MAIN` value in the Options of the SCS Application object you designate as the main Distributed

SCS.

- The DEFAULT value in the Options for the rest of the SCS Application objects.

Incorrect Configuration of Controlled Objects

- If you decide not to have a main Distributed SCS control all unassigned configuration objects, make sure that you assign a particular SCS to each Host, Application, and Solution object:
  - For Host objects, specify a Distributed SCS for the Host object.
  - For Application objects, specify a Distributed SCS for the Host object with which the Application is associated.
  - For Solution objects, specify a Distributed SCS for the Solution object.