Configurations Options Reference Manual

Common Log Options

5/5/2025

Common Log Options

Contents

This page describes common options used to create, view, and otherwise use the Centralized Log facility in Genesys software.

[log] Section

This section must be called **log**.

> ### Warning
> For applications configured via a configuration file, changes to log options take effect after the application is restarted.

buffering

Default Value: `true`
Valid Values:

`true`	Enables buffering
`false`	Disables buffering

Changes Take Effect: Immediately

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

> ### Important
> When buffering is enabled, there might be a delay before log messages appear at the console.

check-point

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

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

enable-thread

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

Specifies whether to enable or disable the logging thread. If set to `true` (the logging thread is enabled), the logs are stored in an internal queue to be written to the specified output by a dedicated logging thread. This setting also enables the log throttling feature, which allows the verbose level to be dynamically reduced when a logging performance issue is detected. Refer to the *Framework Management Layer User's Guide* for more information about the log throttling feature.

If this option is set to `false` (the logging thread is disabled), each log is written directly to the outputs by the thread that initiated the log request. This setting also disables the log throttling feature.

expire

Default Value: `10`
Valid Values:

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

Changes Take Effect: Immediately

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

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

keep-startup-file

Default Value: `false`
Valid Values:

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

`<number> MB`	Sets the maximum size, in megabytes, for a startup segment of the log.

Changes Take Effect: After restart

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

memory

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

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

> **Important**
>
> If the file specified as the memory file is located on a network drive, the application does not create a snapshot file (with the extension ***.memory.log**). Logging output to a file at a network location is not recommended and could cause performance degradation.

memory-storage-size

Default Value: `2 MB`
Valid Values:

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

Changes Take Effect: When memory output is created

Specifies the buffer size for log output to the memory, if configured. See also Log Output Options.

message-format

Default Value: `short`
Valid Values:

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

Changes Take Effect: Immediately

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

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

A log record in the full format looks like this:

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

A log record in the short format looks like this:

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

> **Important**
> Whether the full or short format is used, time is printed in the format specified by the **time_format** option.

messagefile

Default Value: As specified by a particular application
Valid Values: Any valid message file (**<filename>.lms**)
Changes Take Effect: Immediately, if an application cannot find its ***.lms** file at startup

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

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

no-memory-mapping

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

Specifies if memory-mapped files, including memory log output (with file extension **.memory.log**) and snapshot files (with file extension **.snapshot.log**) are disabled for file outputs.

print-attributes

Default Value: `false`
Valid Values:

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

Changes Take Effect: Immediately

Specifies whether the application attaches extended attributes, if any exist, to a log event that it sends to log output. Typically, log events of the Interaction log level and Audit-related log events contain extended attributes. Setting this option to `true` enables audit capabilities, but negatively affects performance.

Genesys recommends enabling this option for Solution Control Server and Configuration Server when using audit tracking. For other applications, refer to *Framework Combined Log Events Help* to find out whether an application generates Interaction-level and Audit-related log events; if it does, enable the option only when testing new interaction scenarios.

segment

Default Value: `100 MB`
Valid Values:

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

Changes Take Effect: Immediately

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

snapshot

Default Value: No value
Valid Values:

No value or not specified (default)	Snapshot is created in log output folder.
`<path>/<folder>`	Full or relative path and folder in which snapshot is created.

Changes Take Effect: Immediately

A snapshot file is created for each log output file to temporarily store logs that have not been flushed to the log file. This option specifies the folder, either a full path or a path relative to the application's working directory, in which the application creates the memory-mapped snapshot file associated with the log file. If this option is not configured, or a value is not specified (the default), the file is created in the log output folder.

> **Important**
> Do not write the snapshot file to a network drive, because disconnection of the network drive might cause application failure. If the application detects that the output folder is a network drive, the snapshot file will be disabled. However, this detection may not be possible for Storage Area Network (SAN) devices because of operating system limitations.

spool

Default Value: The application's working directory
Valid Values: Any valid folder, with the full path to it
Changes Take Effect: Immediately

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

throttle-period

Default Value: `30`
Valid Values: `0`–`3600`
Changes Take Effect: Immediately

Specifies, in seconds, how long to keep the throttled verbose level. When this period of time has

expired, the original log verbose level will be restored when the log queue size has decreased to less than 50% of the threshold.

> **Important**
> This option applies only if **enable-thread** is set to `true`.

throttle-threshold

Default Value: `5000`
Valid Values: `0–10000`
Changes Take Effect: Immediately

Specifies the size of the internal log queue at which the verbose level is to be reduced so as to lessen the load generated by logging. If this option is set to `0` (zero), throttling does not occur. For more information about log throttling, refer to the *Framework Management Layer User's Guide*.

> **Important**
> This option applies only if enable-thread is set to `true`.

time_convert

Default Value: `Local`
Valid Values:

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

Changes Take Effect: Immediately

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

time_format

Default Value: `time`
Valid Values:

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

ISO8601	The date in the time string is formatted according to the ISO 8601 format. Fractional seconds are given in milliseconds.

Changes Take Effect: Immediately

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

A log record's time field in the ISO 8601 format looks like this: 2001-07-24T04:58:10.123

verbose

Default Value: all
Valid Values:

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

Changes Take Effect: Immediately

Specifies if log output is created, and if so, the minimum level of log events generated. Log event levels, starting with the highest priority level, are Standard, Interaction, Trace, and Debug. See also Log Output Options.

> **Important**
> For definitions of the Standard, Interaction, Trace, and Debug log levels, refer to the *Management Layer User's Guide* or *Framework Genesys Administrator Help*.

Log Output Options

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

You can use:

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

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

Warning

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

Important

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

all

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

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

Changes Take Effect: Immediately

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

> **Important**
>
> To ease the troubleshooting process, consider using unique names for log files that different applications generate.

alarm

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

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

Changes Take Effect: Immediately

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

standard

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

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

Changes Take Effect: Immediately

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

interaction

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

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

Changes Take Effect: Immediately

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

trace

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

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

Changes Take Effect: Immediately

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

debug

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

`stdout`	Log events are sent to the Standard output (stdout).
`stderr`	Log events are sent to the Standard error output (stderr).
`network`	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.

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

Changes Take Effect: Immediately

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

> **Important**
>
> Debug-level log events are never sent to Message Server or stored in the Log Database.

Log File Extensions

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

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

  > **Important**
  >
  > Provide `*.snapshot.log` files to Genesys Customer Care when reporting a problem.

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

Examples

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

Production Mode Log Section

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

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

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

Lab Mode Log Section

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

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

Failure-Troubleshooting Log Section

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

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

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

Debug Log Options

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

x-conn-debug-all

Default Value: `0`
Valid Values:

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

Changes Take Effect: After restart

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

> **Important**
>
> Use this option only when requested by Genesys Customer Care.

x-conn-debug-api

Default Value: `0`
Valid Values:

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

Changes Take Effect: After restart

Generates Debug log records about connection library function calls.

> **Warning**
> Use this option only when requested by Genesys Customer Care.

x-conn-debug-dns

Default Value: `0`
Valid Values:

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

Changes Take Effect: After restart

Generates Debug log records about DNS operations.

> **Warning**
> Use this option only when requested by Genesys Customer Care.

x-conn-debug-open

Default Value: `0`
Valid Values:

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

Changes Take Effect: After restart

Generates Debug log records about “open connection” operations of the application.

> **Warning**
> Use this option only when requested by Genesys Customer Care.

x-conn-debug-security

Default Value: `0`
Valid Values:

`0`	Log records are not generated.

1	Log records are generated.

Changes Take Effect: After restart

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

> **Warning**
> Use this option only when requested by Genesys Customer Care.

x-conn-debug-select

Default Value: 0
Valid Values:

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

Changes Take Effect: After restart

Generates Debug log records about “socket select” operations of the application.

> **Warning**
> Use this option only when requested by Genesys Customer Care.

x-conn-debug-timers

Default Value: 0
Valid Values:

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

Changes Take Effect: After restart

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

> **Warning**
> Use this option only when requested by Genesys Customer Care.

x-conn-debug-write

Default Value: `0`
Valid Values:

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

Changes Take Effect: After restart

Generates Debug log records about “write” operations of the application.

> **Warning**
> Use this option only when requested by Genesys Customer Care.

[log-extended] Section

This section must be called **log-extended**.

level-reassign-disable

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

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

level-reassign-<eventID>

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

`alarm`	The log level of log event `<eventID>` is set to `Alarm`.
`standard`	The log level of log event `<eventID>` is set to `Standard`.
`interaction`	The log level of log event `<eventID>` is set to `Interaction`.
`trace`	The log level of log event `<eventID>` is set to `Trace`.

debug	The log level of log event <eventID> is set to Debug.
none	Log event <eventID> is not recorded in a log.

Changes Take Effect: Immediately

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

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

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

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

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

Example

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

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

Before the log levels of the log are changed:

- Log event 1020, with default level `standard`, is output to `stderr` and `log_file`, and sent to Message Server.
- Log event 2020, with default level `standard`, is output to `stderr` and `log_file`, and sent to Message Server.
- Log event 3020, with default level `trace`, is not generated.
- Log event 4020, with default level `debug`, is not generated.

Extended log configuration section:

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

After the log levels are changed:

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