



This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

Genesys Mobile Services Deployment Guide

Genesys Mobile Engagement 8.5.3

11/24/2024

Table of Contents

Genesys Mobile Services Deployment Guide	4
New in This Document	5
Planning Your Deployment	6
Prerequisites	7
Multi-site Deployment	11
Silent Setup	15
Installation	19
Create an Application Object	20
Install Genesys Mobile Services	46
Upgrade Genesys Mobile Services	64
Configuring Routing Dependencies	71
Setting ORS Dependencies	78
Configuring Basic GMS Services	81
Configuring Chat Version1	87
Configuring Digital Channels	97
Configuration	103
Configure an External Cassandra	104
Configuring Apache Load Balancer	122
Configure ORS Load Balancing	124
Configure URS Load Balancing	129
Configure DFMs in Load Balancing Deployment	132
Using Callbacks without DFM	136
Configuring and Starting a GMS Cluster	138
ORS Cookie Support	143
Mobile Push Notifications	144
Configure Historical Reporting	149
Implementing ADDP	175
Implementing IPv6	176
GMS Alarms	178
Configuring HTTP Caching	179
Enable Logging in UTF-8 Environment	180
Admin UI Internationalization	181
Configuration Options Reference	185
Service Options	217
Security	218

Secure GMS Access Control	219
Secure Connections to URS or ORS	238
Cassandra Security	240
Restricting Ports	245
Enabling Basic Authentication	249
Transport Layer Security for Third-Party Servers	260
Single Sign-On (SSO) (Deprecated)	263
Hiding Selected Data in Logs	267
Starting and Stopping GMS	271
Troubleshooting	273
Most Common System Errors	277
Testing Your Deployment	295
Configuring the GMS Built-in Services	296
Testing the GMS Built-in Services	302
Configuring the ORS-based Services	305
Testing the ORS-based Services	308

Genesys Mobile Services Deployment Guide

Welcome to the Genesys Mobile Services Deployment Guide! This deployment guide can be used to install Genesys Mobile Services on your system, configure basic settings, as well as configure more advanced settings. It includes chapters with the following information:

- **New in This Document** — Provides a document change history.
- **Planning information** — Details related to planning and preparation for your Genesys Mobile Services installation, including prerequisites and multi-site deployment. Genesys recommends reading the prerequisites page before you begin to ensure that your system meets the minimum requirements for Genesys Mobile Services.
- **Installation procedures** — Step-by-step guide to installing or upgrading Genesys Mobile Services and performing required configurations.
- **Configuration** — Includes additional configuration topics that you may wish to consider for your GMS installation.
- **Configuration Options Reference** — Provides a reference of the configuration options available for Genesys Mobile Services.
- **Service Options Reference** — Provides a reference of the options available for services.
- **Security** — Provides security configurations that can be used with GMS.
- **Starting and Stopping** — Describes how to start and stop GMS using the Solution Control Interface or Genesys Administrator, and provides information about possible alarms.
- **Troubleshooting** — Frequently asked questions.
- **Migrating GMS** — Migration instructions.
- **Testing your deployment** — Describes how to test your installation by configuring and using samples.

Important

The Admin UI page has been moved to the [Service Management Help](#).

New in This Document

The following topics have been added or changed in the GMS 8.5.230 release:

- Support for Cassandra 4.0

The following topics have been added or changed in the GMS 8.5.228 release:

- Mandatory iOS Device Settings in [Mobile Push Notifications](#)

The following topics have been added or changed in the GMS 8.5.208 release:

- Support for Oracle JDK 11 and Open JDK 11 was added to the [Prerequisites](#) section.
- A new strategy is available for download. Refer to the [Upgrade Notes](#) for further details.

The following topics have been added or changed in the GMS 8.5.206 release:

- Support for [HTTP Strict Transport Security \(HSTS\)](#).
- GMS no longer supports embedded Cassandra. Refer to the [Upgrade Notes](#) to migrate from embedded Cassandra to external Cassandra.
- Support for OpenJDK 8. See the Prerequisites section on the [Genesys Mobile Services](#) page in the [Genesys Supported Operating Environment Reference Guide](#) for more detailed information and a list of all prerequisites.

The following topics have been added or changed in the GMS 8.5.200 release:

- The [Configuration Options](#) page was updated.
- The [Service Options Reference](#) page was updated.
- The upgrade section in the [Installation](#) page was updated.
- The [Prerequisites](#) page was updated.

Planning Your Deployment

For the 8.x release, Genesys Mobile Services allows you to develop mobile applications that take advantage of Genesys capabilities. Every Genesys product also includes a Release Note that provides any late-breaking product information that could not be included in the manual. This product information can often be important. To view it, open the `read_me.html` file in the application home directory, or follow the link under the Release Notes section of the [product page](#) to download the latest Release Note for this product.

What You Should Know

This guide is written for software developers and application architects who intend to create mobile applications that interact with Genesys environments. Before working with Genesys Mobile Services, you should have an understanding of:

- computer-telephony integration (CTI) concepts, processes, terminology, and applications
- mobile concepts and programming
- network design and operation
- your own network configurations
- Genesys Framework architecture

In addition to being familiar with your existing Genesys environment, it is a good idea to be aware of some of the security issues that are involved with deploying a Genesys Mobile Services-based solution. That information and an overview of the deployment topology are discussed under [Security and Access Control](#).

Prerequisites

Modified in 8.5.2

To work with Genesys Mobile Services (GMS), you must ensure that your system meets the software requirements established in the Genesys Supported Operating Environment Reference Manual, as well as meeting the following minimum requirements:

Hardware Requirements

The following are minimum requirements:

- CPU: Quad core
- Memory: 4GB
- Disk: 160GB
- At least 2-3 nodes recommended for redundancy and availability

OS Requirements

- [Genesys Supported Operating Environment Reference Guide](#)

Important

For Linux installations, the Linux compatibility packages must be installed prior to installing the Genesys IPs.

Browser Support

- [Supported Operating Environment Reference Guide](#)

Antivirus

Antivirus software can affect system performance and call response time in some scenarios. Genesys recommends keeping antivirus software enabled on hosts where Genesys Mobile Services is running

and analyzing the performance of all applications on a particular host. If some applications are more vulnerable than GMS, consider moving them to a different host. If GMS seems to be related to significant overhead, Genesys does not recommend excluding GMS from the antivirus scanning. Instead, consider disabling the scanning of the following folders:

- The folder in which GMS is running.
- Any folder containing log files.

Important

The antivirus software must not restrict any ports that Genesys applications are using.

Java Requirements

Java 64 bits

- Before 8.5.206.04: Support for JDK 8 only
- Starting in 8.5.206.04: Support for Open JDK 8
- Starting in 8.5.208.09: Support for Open JDK 11 and for Oracle JDK 11
- Starting in 8.5.300.02: Support for Open JDK 17

Important

Starting from 8.5.300.02, GMS no longer supports JDK 8 and 11.

Tip

Edit JAVA_HOME to point to the JDK installation folder, for example, C:\Program Files\Java\<your JDK>. In some scenarios, the GMS installer may fail to find Open JDK 1.x. The workaround is to install Oracle JDK first, proceed with the installation, then once GMS is installed, point the JAVA_HOME variable to OpenJDK.

Cassandra Support on Linux

Modified in: 8.5.303+

- Cassandra 3.x: Tested version is 3.11

- Cassandra 4.x: Tested version is 4.0
- Cassandra 4.1.x: Tested version is 4.1

Tip

When you deploy Genesys Mobile Environment for Chat API V2, Email API V2, and Open Media API V2, Cassandra is required only if you enable mobile push notifications.

Genesys Environment

Modified in 8.5.200.07, 8.5.201.04

In addition to having a **Genesys Management Framework 8.1** environment installed and running, the following table lists the Genesys components that are used with a GMS installation.

Genesys Component	Minimum Version Required	Comments
Orchestration Server (ORS)	<ul style="list-style-type: none"> • 8.1.400.26 • 8.1.400.74 for GMS 8.5.201.04 and higher 	Optional, installed and running: <ul style="list-style-type: none"> • An HTTP port must be enabled in the related Application object. • The ORS server must use the Orchestration Server type in Configuration Manager. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Important You need a minimum of ORS 8.1.300.30 to be able to do Load Balancing with GMS.</p> </div>
Universal Routing Server (URS)	8.1.400.45	Mandatory, required for the GMS services, and if you plan to use URS-based dialing in Callback applications.
Interaction Routing Designer (IRD)	8.1.400.26	Mandatory, required for strategies running on URS.
SIP Server	8.1.100.67	<ul style="list-style-type: none"> • SIP or Inbound Voice is required for agents. • SIP Server is recommended for outbound calling for Callback.
Chat Server	8.1.000.26	Used for Chat support.
	8.5.105+	Required if you plan to use features related to file

Genesys Component	Minimum Version Required	Comments
		management.
	8.5.109+	Required if you plan to use Digital Channels Chat over CometD API feature.
Interaction Server	8.0.200.11	Used for Chat support.
Universal Contact Server (UCS)	8.5.200.19	Used for Digital Channel.
E-mail Server (ESJ)	8.5.103.01	Used for Digital Channel.
Stat Server	8.x	Used to obtain statistics.
Media Server	8.1.410.33	Used for Callback services, in order to play treatments and use Call Progress Detection (CPD) for outbound calls.
Resource Manager	8.1.410.13	Used for Callback services, in order to play treatments and use Call Progress Detection (CPD) for outbound calls.
Workspace Desktop Edition	(optional) 8.5.111.21	Support for Genesys Callback . This component is not mandatory.

Historical Reporting for Callback

Mandatory Genesys Components

Component	Minimum Version
Orchestration Server	8.1.400.24
Universal Routing Server	8.1.400.22
Interaction Concentrator	8.1.506.07
Genesys Info Mart	8.5.005 (GA)
Reporting and Analytics Aggregates (RAA)	8.5.000.02
Genesys Interactive Insights (GI2)	8.5.000.02

Real-time Statistics for Context Services

Component	Version	Comments
Genesys Pulse	<= 8.5.102	GMS Pulse option version = 1.
Genesys Pulse	>= 8.5.103 (and also 9.0.x.y)	GMS Pulse option version = 2.

Multi-site Deployment

For a multi-site deployment, you must consider the following questions:

- How are API requests from mobile devices going to be routed?
- How is the mobile/web client going to retry the request in case of network failure?
- How will the client find the addresses for the global load balancer or site load balancer?
- How is the telephony network going to route the call to the number the client is receiving through the data API call?
- Is the call origination direction user-originated or user-terminated?

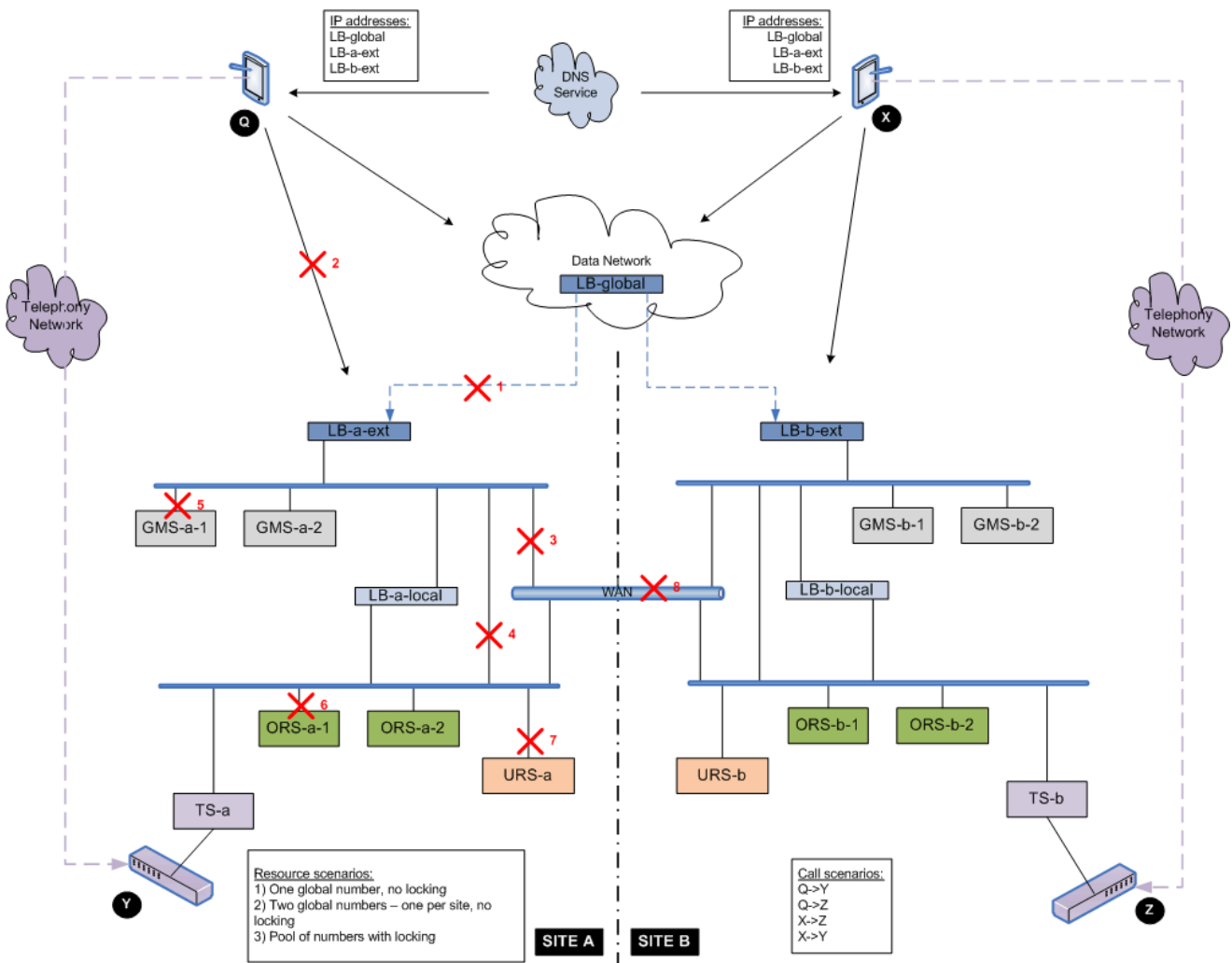
The general recommendation for user originated scenarios is to deploy an independent GMS cluster per site, and control call routing by ensuring that no intersecting pools of incoming phone numbers are configured. This way, if the client is requesting an access number through the data API call, it will be routed to Site A by the data network (internet), and then the call initiated by this client will land on the telephony switch/gateway located on (or associated with) Site A.

The matching API call will first be performed against the GMS cluster located on Site A. If Site A fails, then Site B is tried. This provides a good level of site isolation and simplifies maintenance, as well as day-to-day operations. Contact centers will be able to distribute telephony traffic between sites by controlling data API traffic (splitting between sites or directing to one site only) using standard IP load balancers, DNS records, and so on.

Using an approach of separate GMS clusters helps to avoid complex problems of "split brain" scenarios, where typically, some type of manual intervention is required either after the split, or later on when connection between sites is restored and clusters must be reconciled back together.

For user terminated scenarios, where call matching is not required, deploying a single GMS cluster across sites may provide some benefits. Still, it is not clear whether those benefits will outweigh the risks and be more beneficial as compared to the simplicity of a dedicated site cluster approach.

The following diagram shows major components of the solution and connectivity between them through local and wide area network segments. Possible network and/or component failures are numbered and described in more detail in the table below.



Failure Condition (see diagram)	Resource Locking Scenario	Two Separate GMS Clusters: One on Site A and One on Site B			
		Match ixn Scenario			
Create Service/Get Access Number		Same Site	Cross Site		
1 or 2	Global number, no locking.		Client has to retry the other site.	Not affected.	Request fails on the local site and has to be retried on the other site.
	Two global numbers - one per site, no locking.		Client has to retry the other site.	Not affected.	Will never happen because the site specific number was given out.
	Pool of numbers with locking.		Client has to retry the other site.	Not affected.	Request fails on the local site and

Failure Condition (see diagram)	Resource Locking Scenario	Two Separate GMS Clusters: One on Site A and One on Site B		
				has to be retried on the other site.
3	Global number, no locking.	Not affected.	Not affected.	Request fails. Retrying the other site is not possible because the GMS cluster cannot be reached on another site.
	Two global numbers - one per site, no locking.	Not affected.	Not affected.	Will never happen because the site specific number was given out.
	Pool of numbers with locking.	Not affected.	Not affected.	Request fails. Retrying the other site is not possible because the GMS cluster cannot be reached on another site.
4	Global number, no locking.	Not affected.	Request fails. Retrying the other site is not possible because separate clusters are being run.	
	Two global numbers - one per site, no locking.	Not affected.		
	Pool of numbers with locking.	Not affected.		
5	Global number, no locking.	Not affected.	Request does not fail. Another GMS instance will handle the calls.	
	Two global numbers - one per site, no locking.	Not affected.		
	Pool of numbers with locking.	Not affected.		
6	Global number, no locking.	Not affected.	Request does not fail. Another ORS instance will handle the calls.	
	Two global numbers - one per site, no locking.	Not affected.		
	Pool of numbers with locking.	Not affected.		
7	Global number, no locking.	Not affected.	Request fails if there is no URS backup set in the configuration. Retrying the other site is not possible because separate clusters are being run.	
	Two global numbers - one per site, no locking.	Not affected.		
	Pool of numbers	Not affected.		

Failure Condition (see diagram)	Resource Locking Scenario	Two Separate GMS Clusters: One on Site A and One on Site B		
	with locking.			
8	Global number, no locking.	Not affected.	Not affected.	Request fails. Retrying the other site is not possible because the GMS cluster cannot be reached on another site.
	Two global numbers – one per site, no locking.	Not affected.	Not affected.	Will never happen because the site specific number was given out.
	Pool of numbers with locking.	Not affected.	Not affected.	Request fails. Retrying the other site is not possible because the GMS cluster cannot be reached on another site.

Silent Setup

Genesys Silent Configuration allows for automated electronic software distribution, also known as a *silent setup*. With silent setup, you do not have to monitor the setup or provide input via dialog boxes. Instead, the setup parameters are stored in a response file, and the silent setup runs on its own, without any intervention by the end-user.

An installation procedure for a server application differs slightly from an installation procedure for a GUI application. Both, however, require that you update a response file with the necessary parameters and then use it for the actual installation.

Genesys Silent Configuration works on both UNIX and Windows operating systems.

The following Framework components support Silent Setup installation:

- Configuration Server
- Message Server
- Solution Control Server
- T-Server
- HA Proxy
- Stat Server

Creating the Response File

A template for the response file, called **genesys_silent.ini**, is included in the Installation Package (IP) for each supporting component. This template file guides you through the task of entering the required information, by providing the following information for each field:

- A full description of the field.
- If applicable, a description of valid values, either a range or a list.
- If applicable, any conditions in which the parameters may not be used.

Open this file and provide values for all required fields by replacing the text contained in angle brackets (<>)(see the examples). Then save the file. By default, it is saved as **genesys_silent.ini** in the installation folder.

Subsequently, you can use the same response file any time you need to install an application with the configured parameters.

Sample Response File Entries (**genesys_silent.ini**)

The following is an example of the Genesys Configuration Server information section in the **genesys_silent.ini** for Configuration Server, with values entered for the required fields.

[+] Show sample entries

```

=====
#       Genesys Configuration Server information section
#       NOTE:       If Genesys Configuration Wizard .ini file (GCTISetup.ini file) is
#                   detected in IP root directory, then Host, Port, User,
#                   Password Configuration Server parameters specified in
#                   Genesys Silent Configuration file are ignored.
=====
[ConfigServer]

#-----
#       Host name where Genesys Configuration Server is running.
#-----
Host=CShost

#-----
#       Port number of Genesys Configuration Server.
#-----
Port=2010

#-----
#       User name in Genesys Configuration Server.
#-----
User=User1

#-----
#       User's password in Genesys Configuration Server.
#       Password - is used to specify the non-encrypted password;
#-----
Password=*****

#-----
#       Application name in Genesys Configuration Server.
#       NOTE:       This parameter is ignored if only one application was defined in
#                   GCTISetup.ini file by Genesys Configuration Wizard (Setup reads
#                   application name from '[<ApplicatonName>]' section name
#                   of GCTISetup.ini file).
#                   This is a mandatory parameter if Installation uses application
#                   template in Genesys Configuration Server and GCTISetup.ini file
#                   does not exist or contains more then one defined application.
#-----
ApplicationName=config

```

Running the Silent Installation

The silent setup program does not display a message if an error occurs. The status information for the silent installation is recorded in a file called (by default) **genesys_install_result.log**.

Use the appropriate command line to launch the Genesys Silent Configuration, depending on your operating platform as follows:

On UNIX

```
./install.sh -s -fr <full path to the setup response file> -fl <full path to the setup log file>
```

where:

<full path to the setup response file>	
	The full path to the setup response file. By default, install.sh looks for a response file called genesys_silent.ini in the same directory as install.sh .
<full path to the setup log file>	
	The full path to the setup log file. By default, genesys_install_result.log is generated in the same directory as the response file being used.

Example

```
.\install.sh -s -fr /home/user/genesys_silent.ini -fl /home/user/genesys_install_result.log
```

On Windows

```
.\setup.exe /s /z"-s <full path to the setup response file> -sl <full path to the setup log file>"
```

where:

<full path to the setup response file>	
	The full path to the setup response file. By default, setup.exe looks for a response file called genesys_silent.ini in the same directory as setup.exe .
<full path to the setup log file>	
	The full path to the setup log file. By default, genesys_install_result.log is

generated in the same directory as the response file being used.

Important

- Enclose the entire string of parameters **-s <full path to the setup response file> -sl <full path to the setup log file>** in double quotation marks.
- Do not enter a space between the **/z** parameter and its value.

Example

```
.\setup.exe /s /z"-s c:\win\genesys_silent.ini -sl c:\win\genesys_install_result.log"
```

Silent Setup Log File

The silent setup program prints installation results into a setup log file. By default, the results file is named **genesys_install_result.log**, and is stored in the same folder as **genesys_silent.ini**.

Silent Uninstall

If you check your GMS configuration, you should find the `uninstall` option configured in the `sml` section. Use the command provided in this option to do a silent uninstallation.

Installation

This is the first GMS installation

Important

Before you begin with the installation process, make sure that your environment meets the minimum requirements specified in the [Prerequisites](#) section.

Installing Genesys Mobile Service is a process that consists of the following tasks:

1. [Creating a Genesys Mobile Services configuration object](#)
2. [Installing Genesys Mobile Services](#)
3. [Configuring ORS and Deploying DFM Files](#)

Once the installation is complete, additional configuration is required before your Genesys Mobile Services deployment is ready to use:

- [Basic Configuration](#)
- [Configuring Chat Support](#)

GMS Upgrade

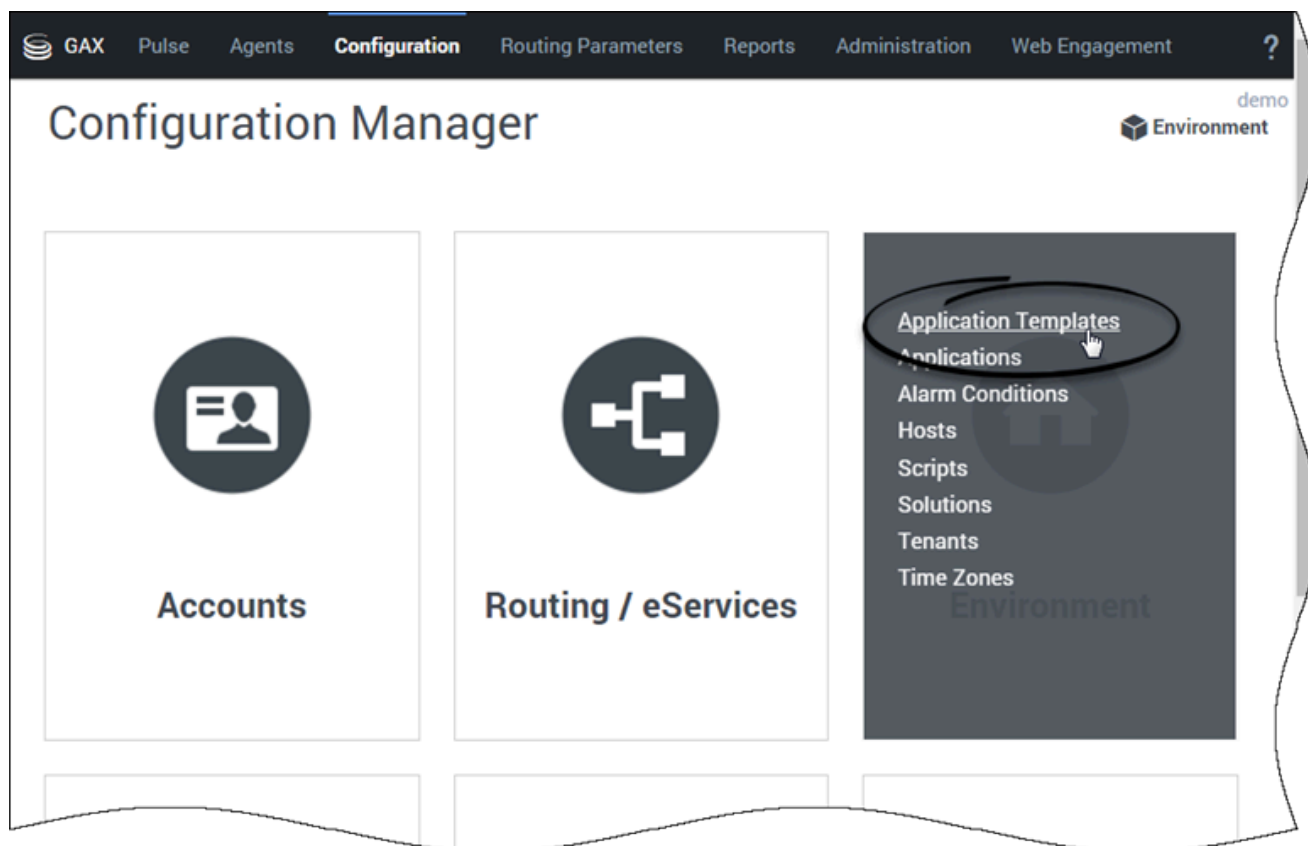
To upgrade GMS, run the Installation Package and follow the instructions detailed in the [Upgrade](#) page. Then, follow the [additional](#) upgrade steps introduced with the version that you are installing.

Create an Application Object

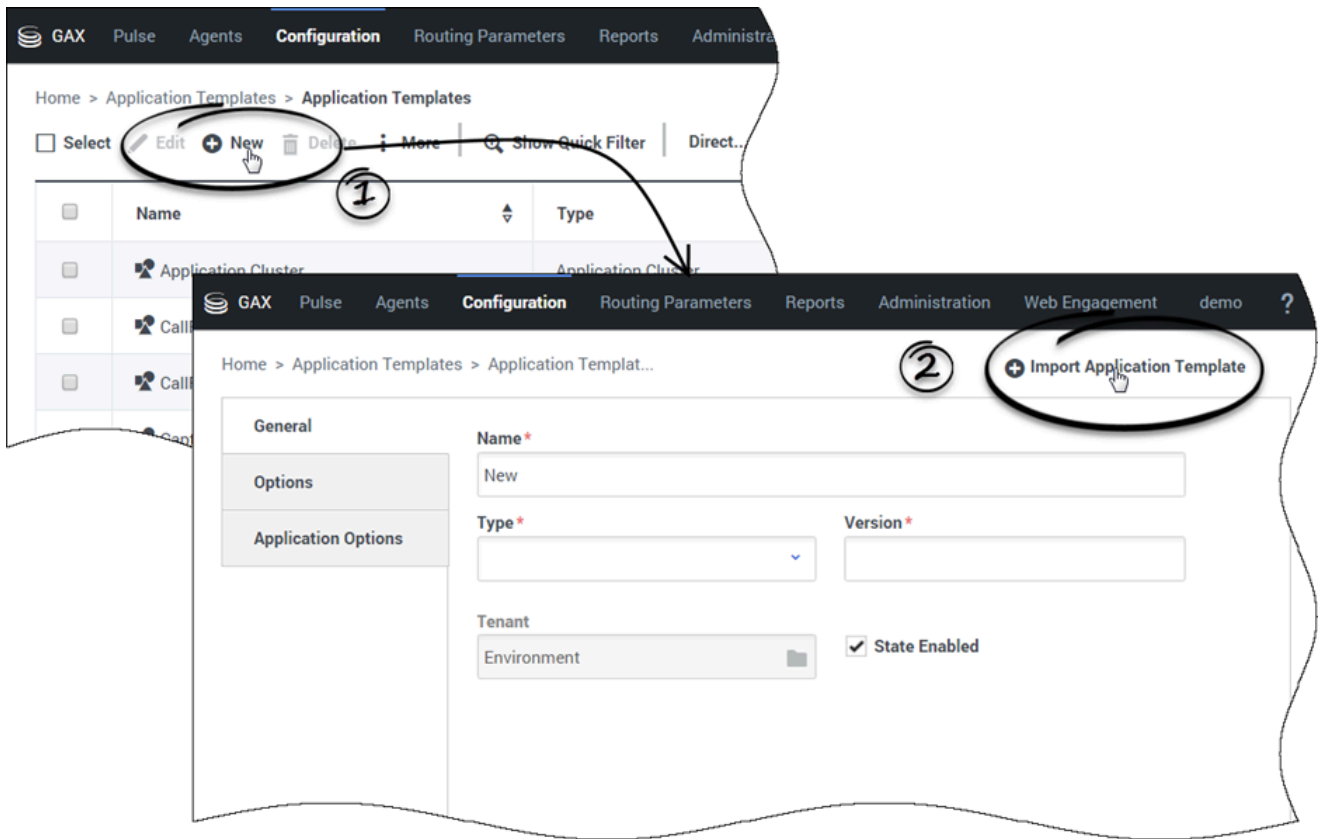
Important

Configuration objects can be created and configured in Genesys Administrator and Configuration Manager. To learn how to start Genesys Administrator, refer to the [Genesys Administrator Help](#).

Import the GMS Application Templates

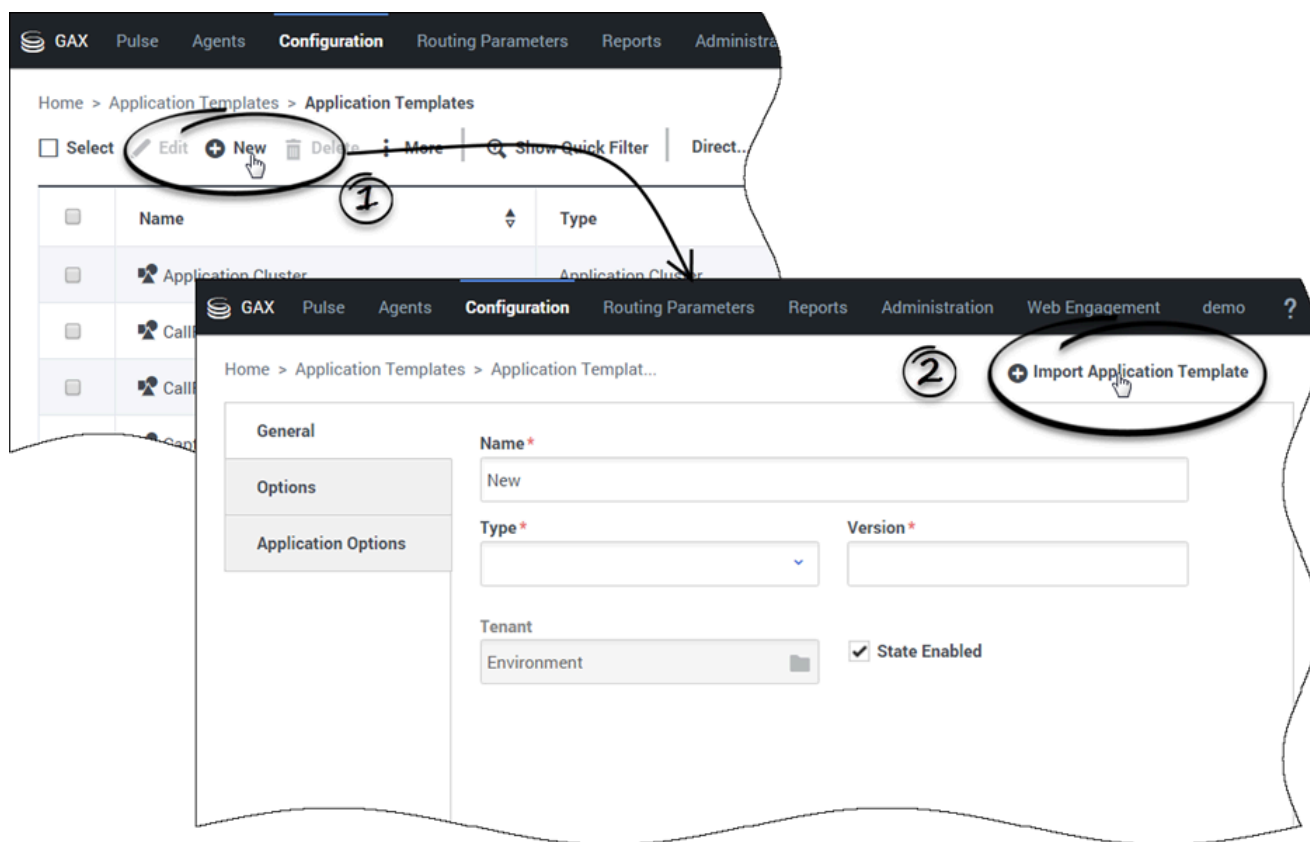


In Genesys Administrator Extension, find the **Configuration Manager > Environment** menu and click **Application Templates**.



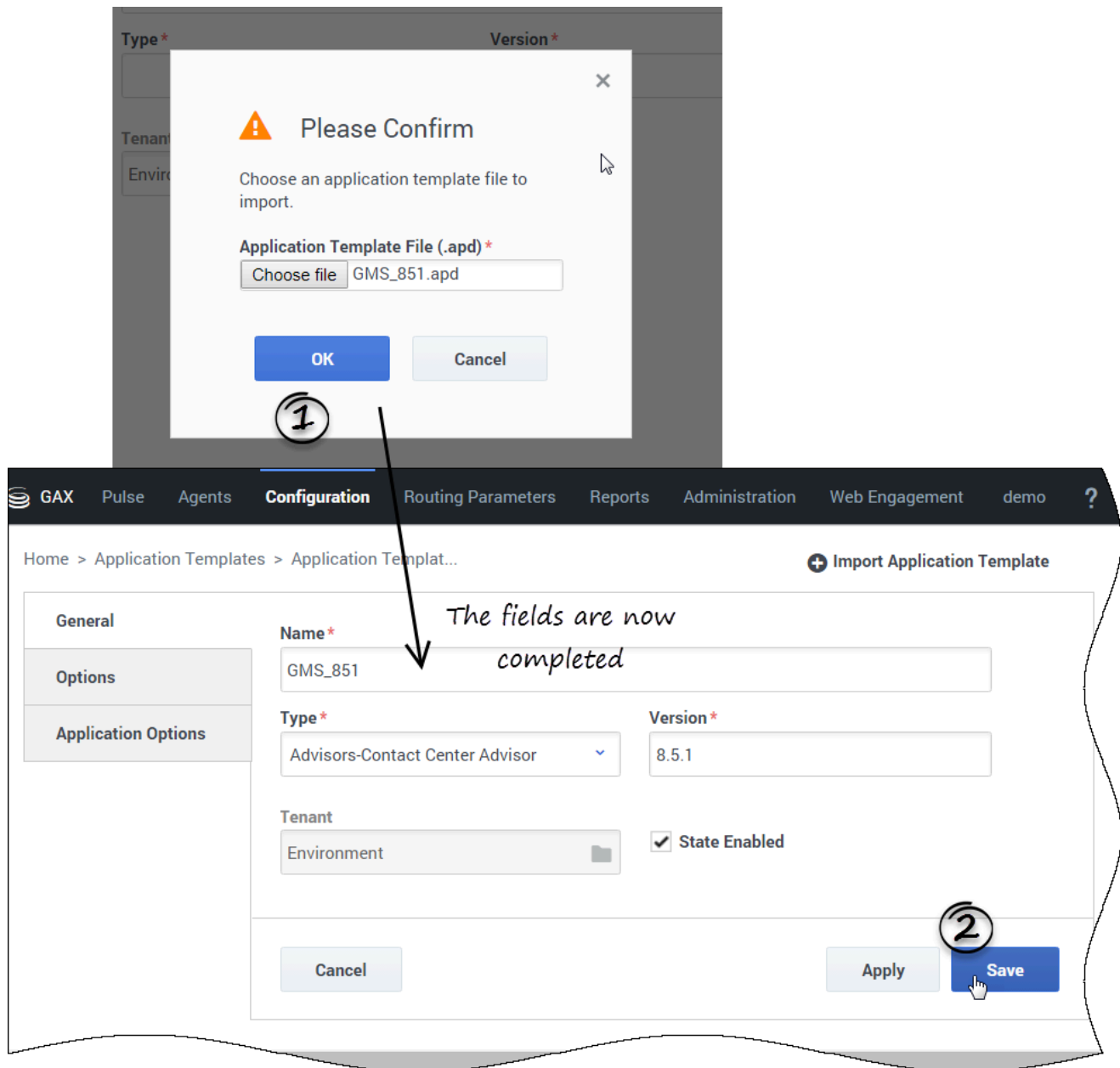
Click **+New**, then click **Import Application Template**. Navigate to the *Templates* directory of your Installation Package (IP) and add the following files:

- **GMS_851.apd** - This template is used to deploy GMS with default options.
- **ApplicationCluster_851.apd** - You do not need this template for a single node deployment. This template is used for deploying all GMS's into the same cluster. The Cluster Application will contain shared configuration for GMS nodes.



Click **+New**, then click **Import Application Template**. Navigate to the *Templates* directory of your Installation Package (IP) and add the following files:

- **GMS_851.apd** - This template is used to deploy GMS with default options.
- **ApplicationCluster_851.apd** - You do not need this template for a single node deployment. This template is used for deploying all GMS's into the same cluster. The Cluster Application will contain shared configuration for GMS nodes.



Confirm the Import action, then **Save** the template.

Create a GMS Administrator

[Documentation:GMS:Help>Login:8.5.3](#)

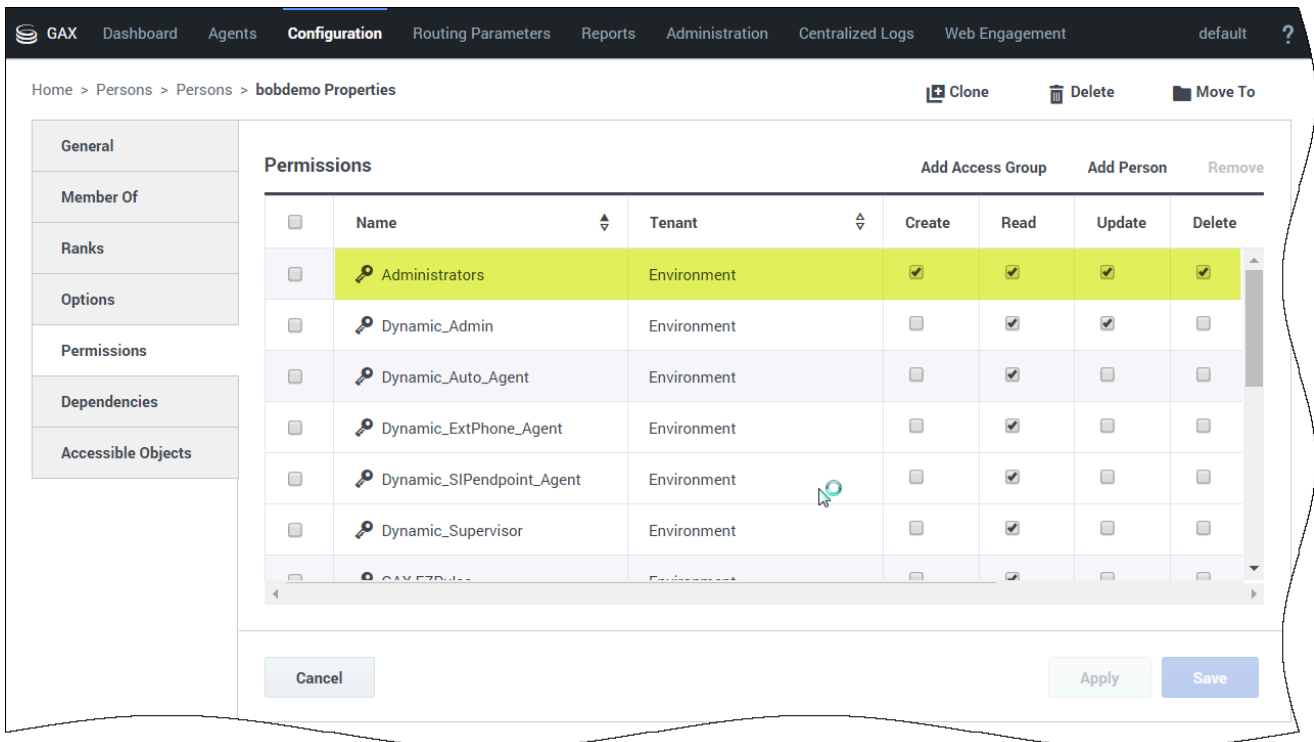
Important

If an Administrator changes a user's role during a Service Management UI session, the user will have to disconnect/reconnect for the new role to go into effect.

Set Administrator Permissions

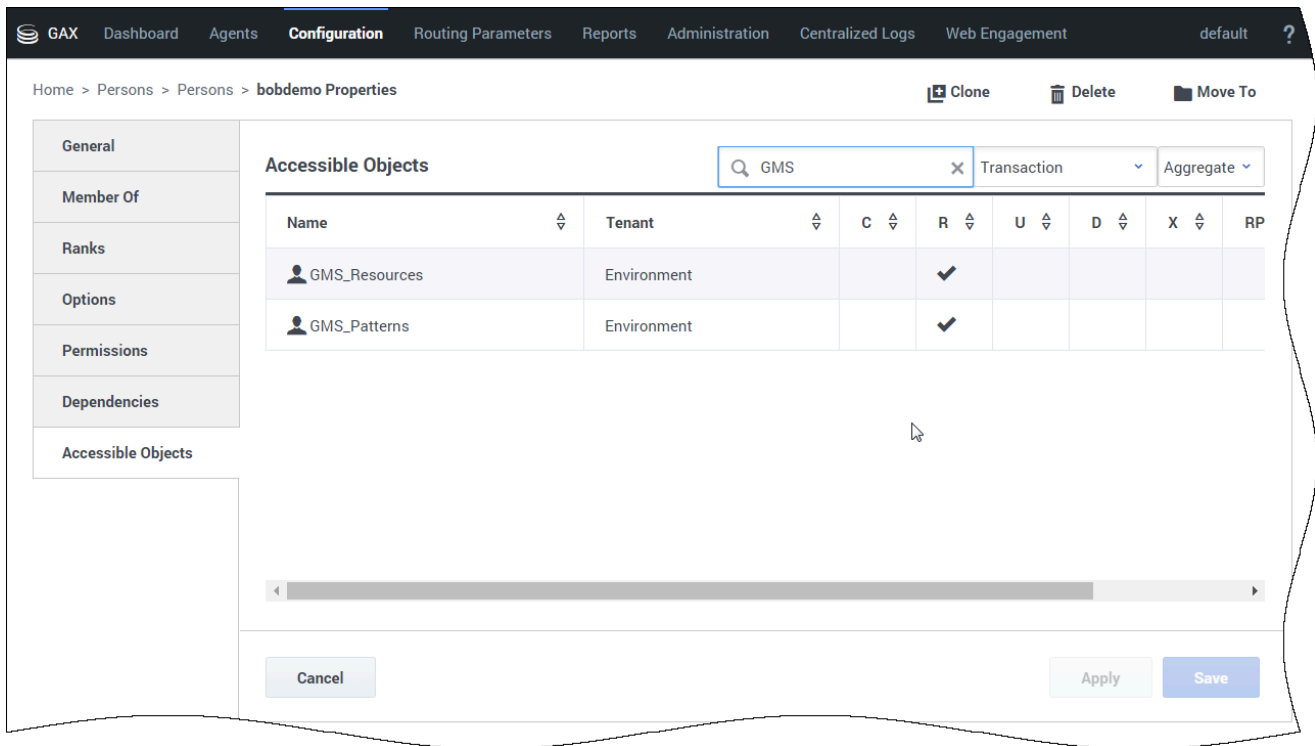
Warning

This step is required only for versions older than 8.5.102.11.



In the **Permissions** tab, make sure to add *Environment\Administrators*.

This step enables your GMS administrator to manage configuration objects that GMS reads and writes.



Select the **Accessible Objects** tab. For other GMS objects, you need to set the following permissions:

- Application (GMS/GMS Cluster): Read, Change
- Transactions: Read, Change
- Hosts: Read
- Persons: Read

Next Steps:

- For a GMS cluster deployment, go to [Cluster Deployment](#).
- For a single node deployment, go to [Single Node Deployment](#).

Single Node Deployment

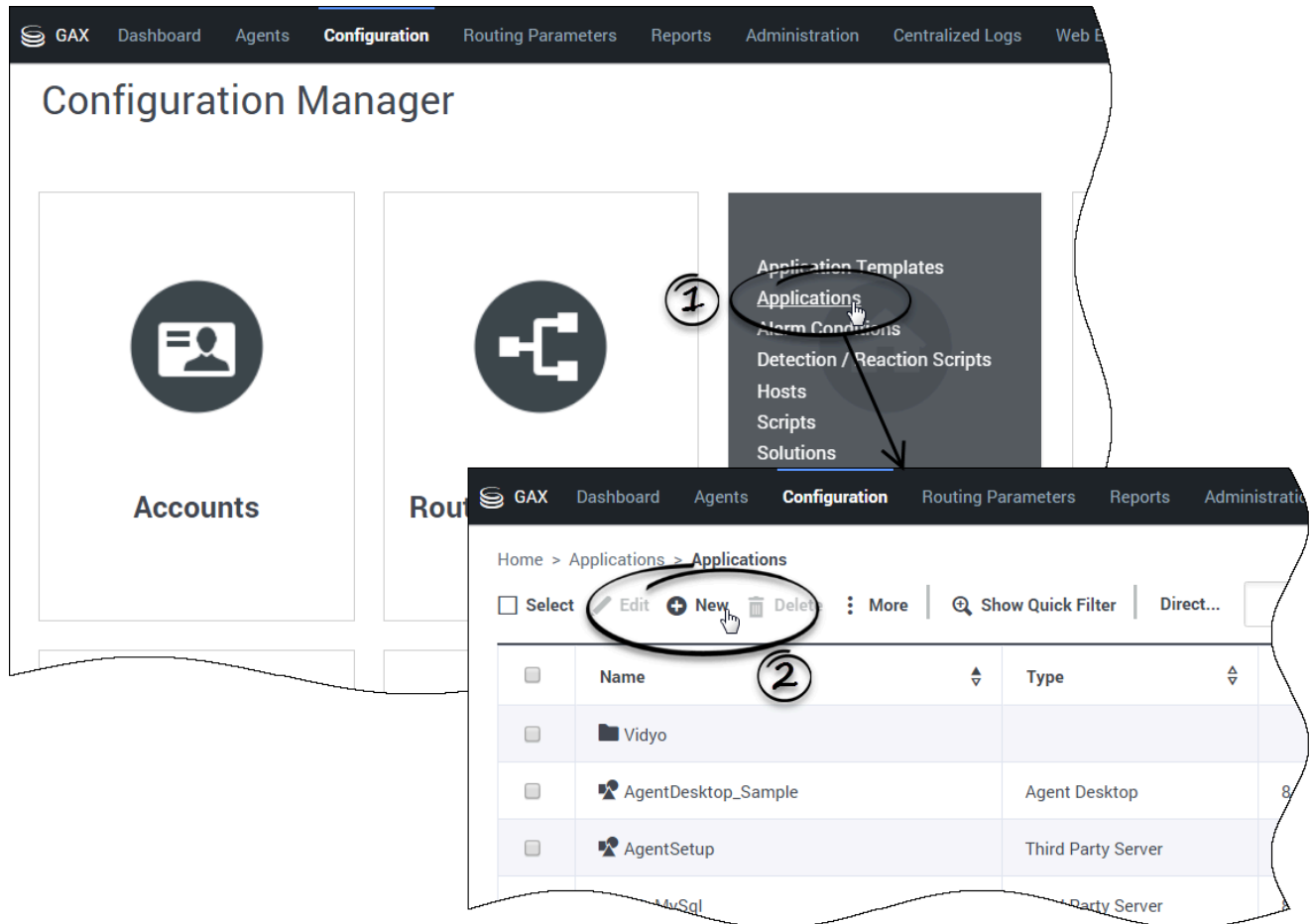
Purpose: To create and configure a GMS Application object for a single GMS node (no cluster).

Prerequisites:

- [Import the Application Templates](#)

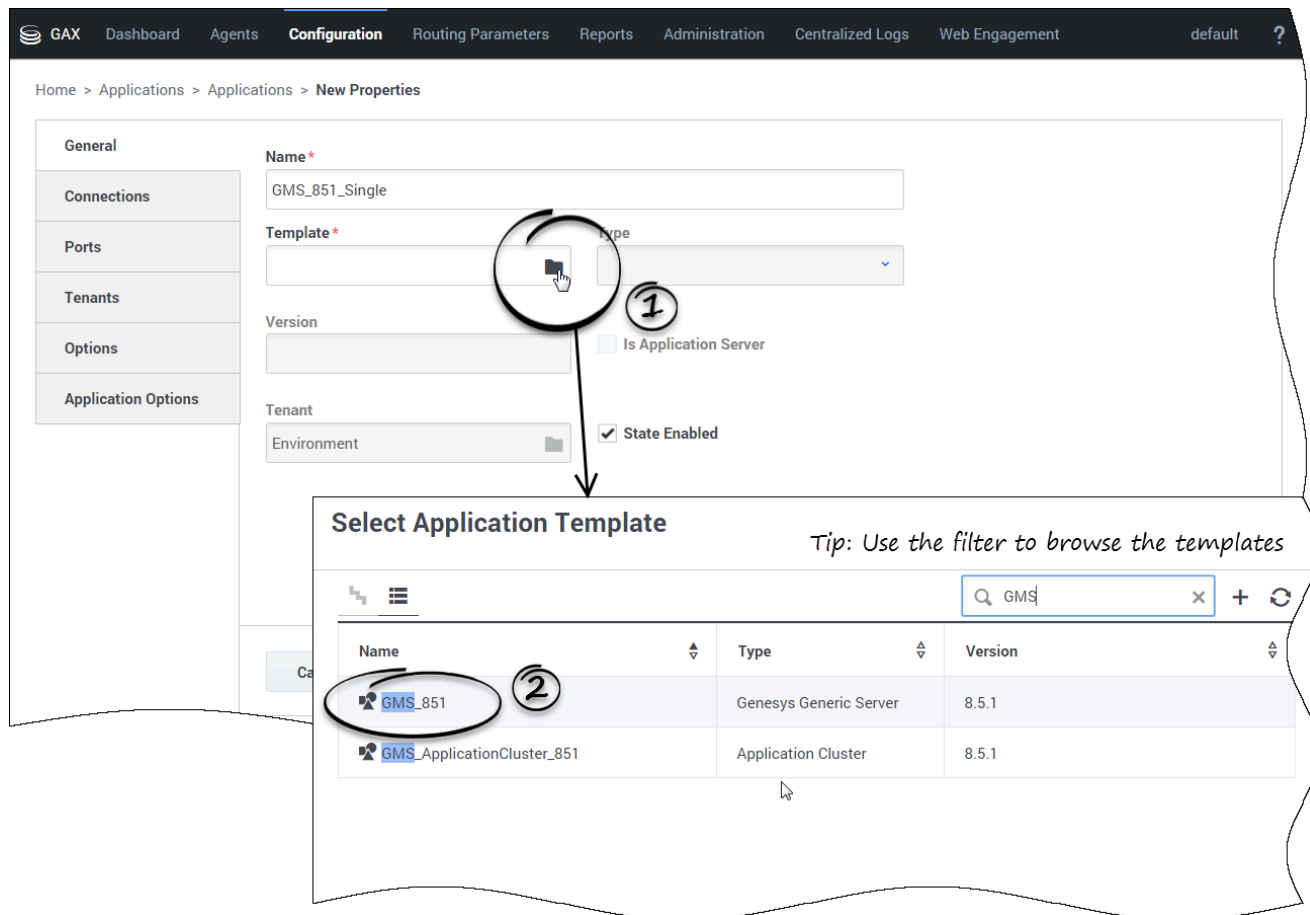
- [Create a GMS Administrator](#)

Create the GMS Application Object (Single Node)



In the Configuration Manager tab, click the *Applications* link in the Environment menu; then, click + **New** .

Browse your template



Enter a name for your GMS application (with no space). Click  to browse the GMS template. In **Select Application Templates**, click **Application Templates** and select the GMS_851 application template that you imported previously.

General Settings

The screenshot shows the 'New Properties' configuration page in the GAX interface. The page is titled 'Home > Applications > Applications > New Properties'. The left sidebar contains a menu with 'General' selected, and other options like 'Connections', 'Ports', 'Tenants', 'Options', and 'Application Options'. The main content area is divided into several sections:

- Component type:** A dropdown menu with 'Component' selected.
- Version:** A text input field containing '8.5.1'.
- Is Application Server:** A checked checkbox.
- Working Directory*:** A text input field containing '.'.
- Command Line*:** A text input field containing '.'.
- Command Line Arguments:** A text input field containing 'dummy'.
- Startup Timeout*:** A text input field containing '90'.
- Shutdown Timeout*:** A text input field containing '90'.

At the bottom of the form, there are three buttons: 'Cancel', 'Apply', and 'Save'. A handwritten annotation 'Scroll down' with an arrow points to the bottom of the form area.

- Select Component for **Component Type**.
- In the **Command Line Arguments** text box, enter a dummy value. It will be overwritten when Genesys Mobile Services is installed; however, having values in these fields is required to save the Application object.

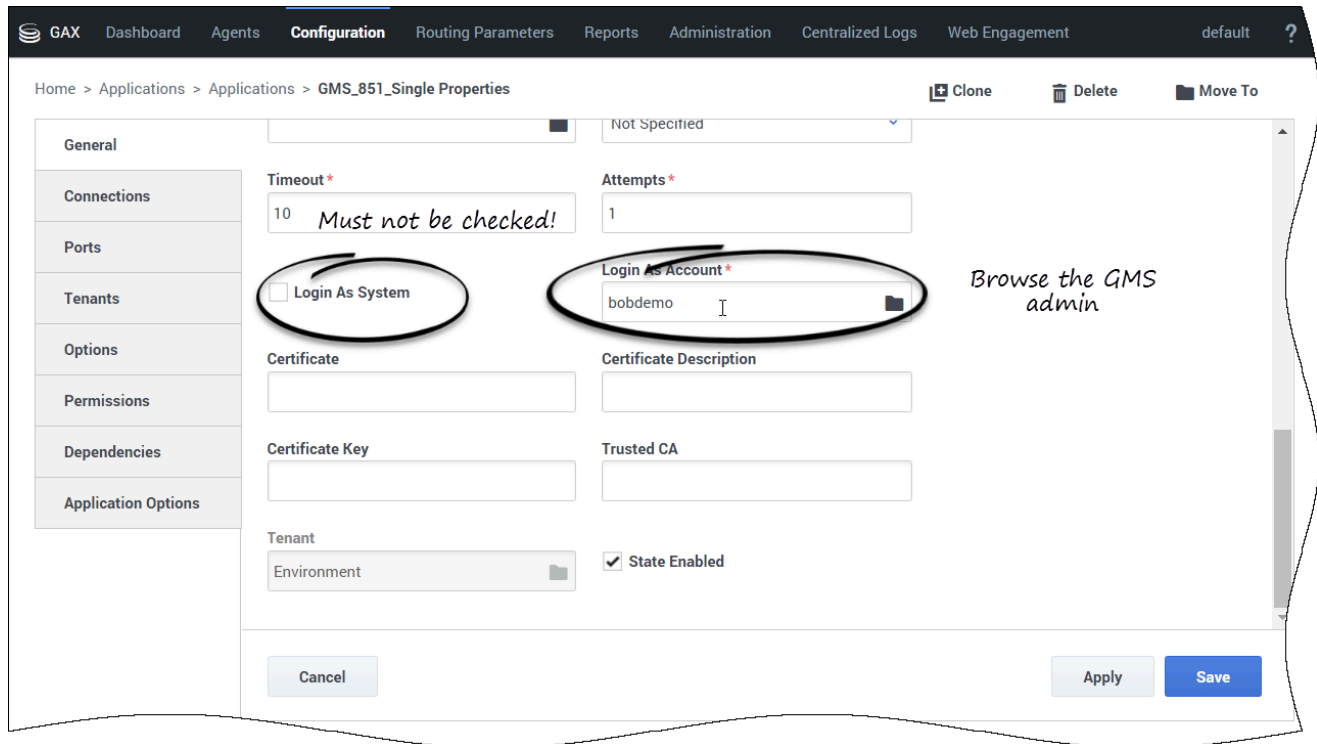
The screenshot shows the 'New Properties' configuration page in the GAX interface. The page has a dark header with navigation links: GAX, Dashboard, Agents, Configuration, Routing Parameters, Reports, Administration, Centralized Logs, Web Engagement, default, and a help icon. The breadcrumb trail is 'Home > Applications > Applications > New Properties'. On the left, there is a sidebar menu with options: General, Connections, Ports, Tenants, Options, and Application Options. The main content area contains several fields and checkboxes:

- Startup Timeout***: Input field with value 90.
- Shutdown Timeout***: Input field with value 90.
- Auto-Restart**: (unchecked).
- Primary**: (checked).
- Host***: Input field with value 135.39.45.125 and a file selection icon.
- Certificate**: Input field.
- Certificate Description**: Input field.
- Certificate Key**: Input field.
- Trusted CA**: Input field.
- Tenant**: A dropdown menu showing 'Environment'.
- State Enabled**: (checked), which is circled in black.

At the bottom of the form, there are three buttons: 'Cancel', 'Apply', and 'Save'.

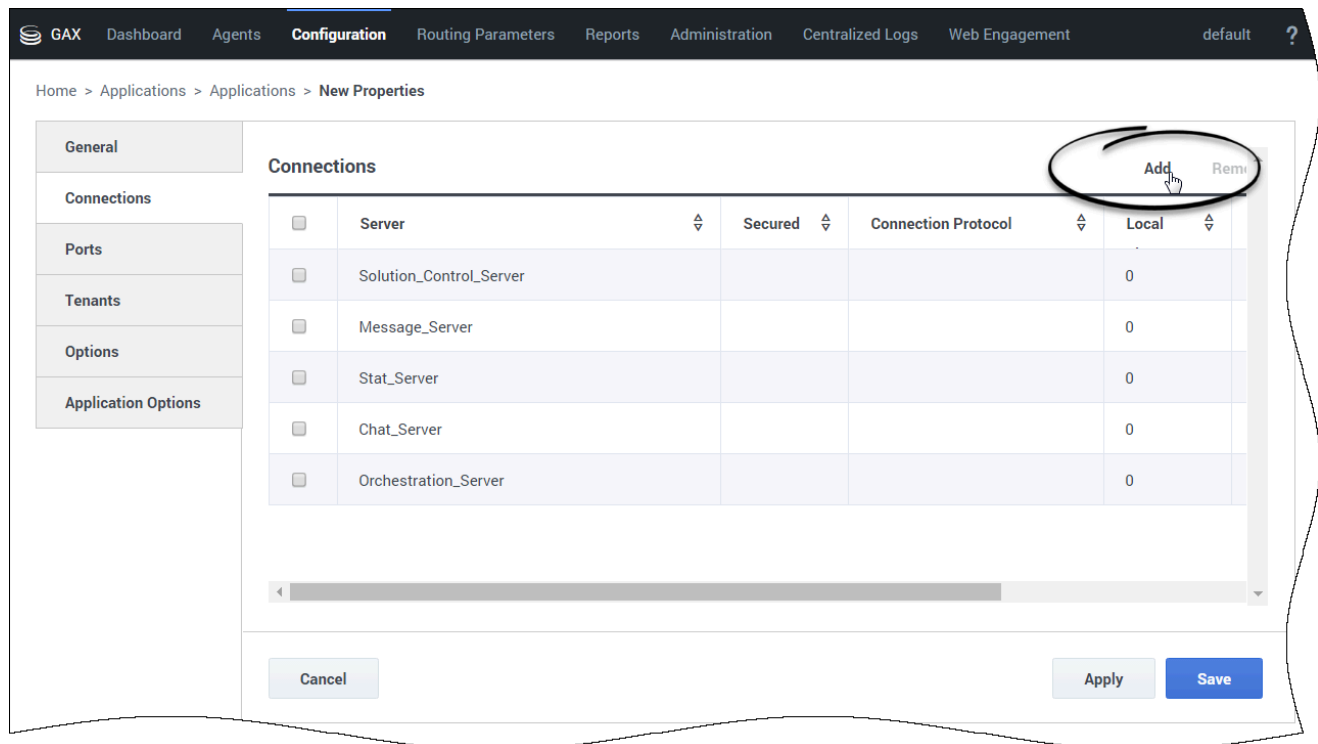
Check that the **State** is enabled, and browse your hostname for this application.
Click **Apply** or **Save**.
Your application is created.

Configure the GMS Application Object for Single Node



In the General tab, uncheck **Log On As System** option and browse your GMS Admin account to set up **Log On Account**. If you do not set up **Log On Account**, the GMS Admin UI will throw permission errors when you will try to create GMS Built-in services.

Set up your Connections



In the **Connections** tab, click **Add** to add the following connections:

- Orchestration Server (ORS) - optional: Add this connection (using HTTP) if you plan to use GMS Callback features.
- Solution Control Server - optional: Add this connection if you plan to use GMS Chat features.
- Chat Server - optional: Add this connection if you plan to use GMS Chat features.
- Stat Server - optional: Add this connection if you plan to use GMS Reporting features.
- Message Server - optional: Add this Connection if you plan to use Log features.
- URS Server - optional: Add this Connection if you plan to use EWT features.

In the **Ports** tab, check that a default dummy port is created.

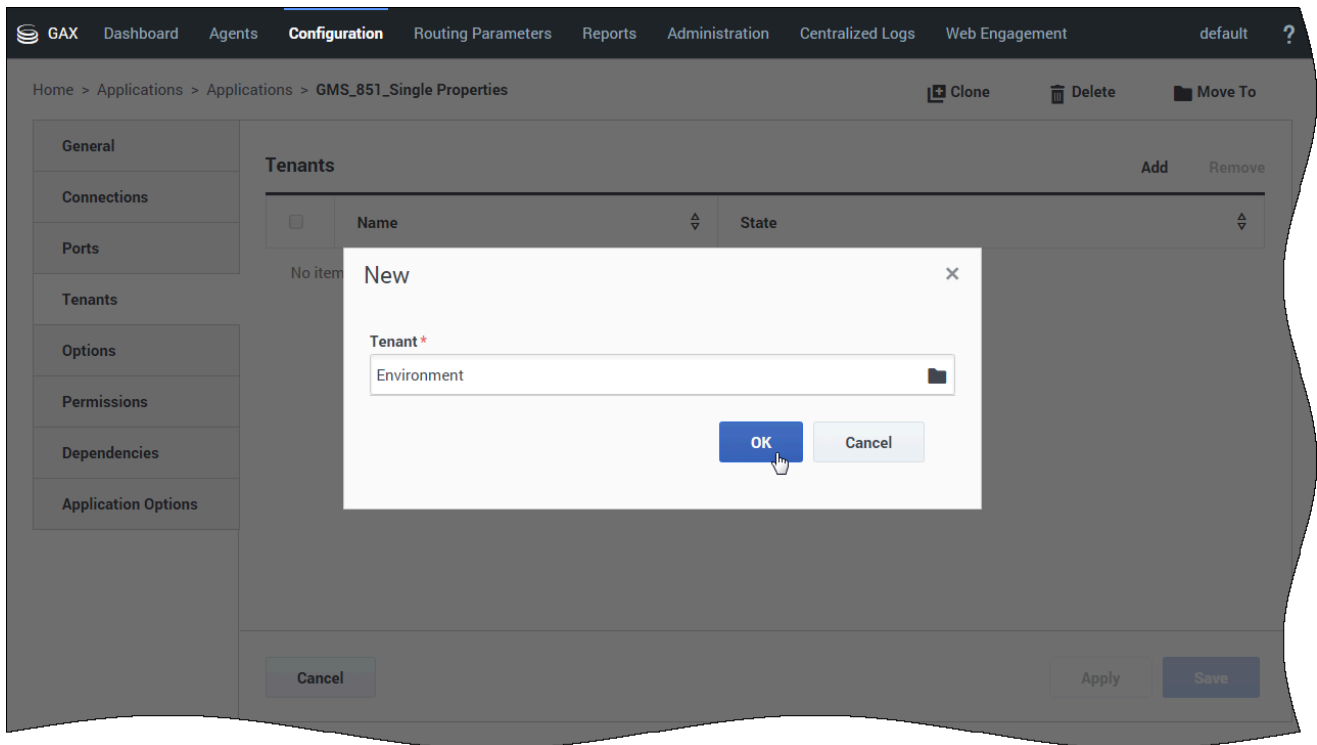
Important

If you do not plan to use GMS configuration defaults at startup or if you plan to use a Configuration Server proxy, refer to the following documentation:

- [Using the Management Layer](#)

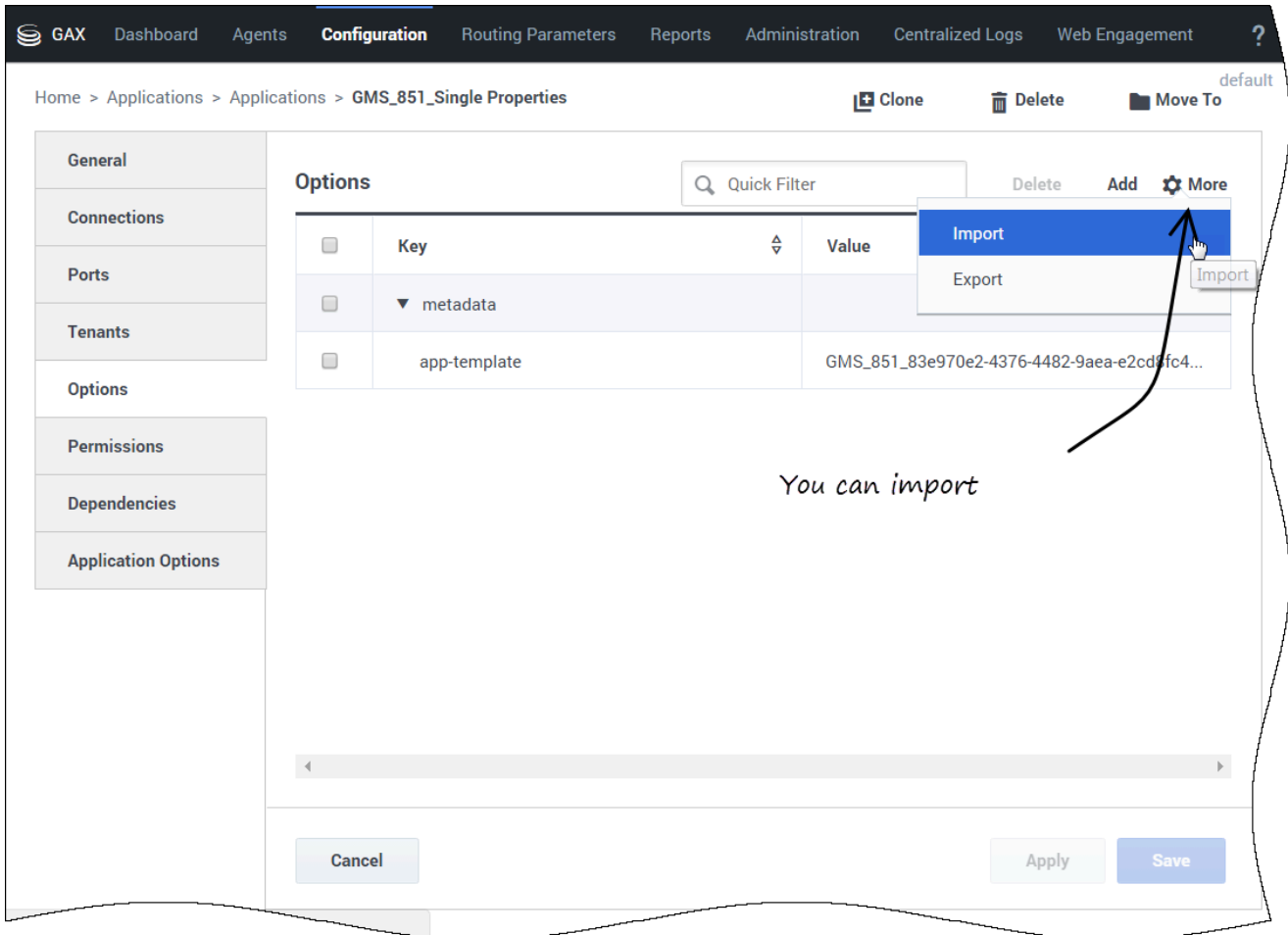
- [How to Use Startup Files](#)
- [Configuration Server Proxy](#)

Set up Server Information



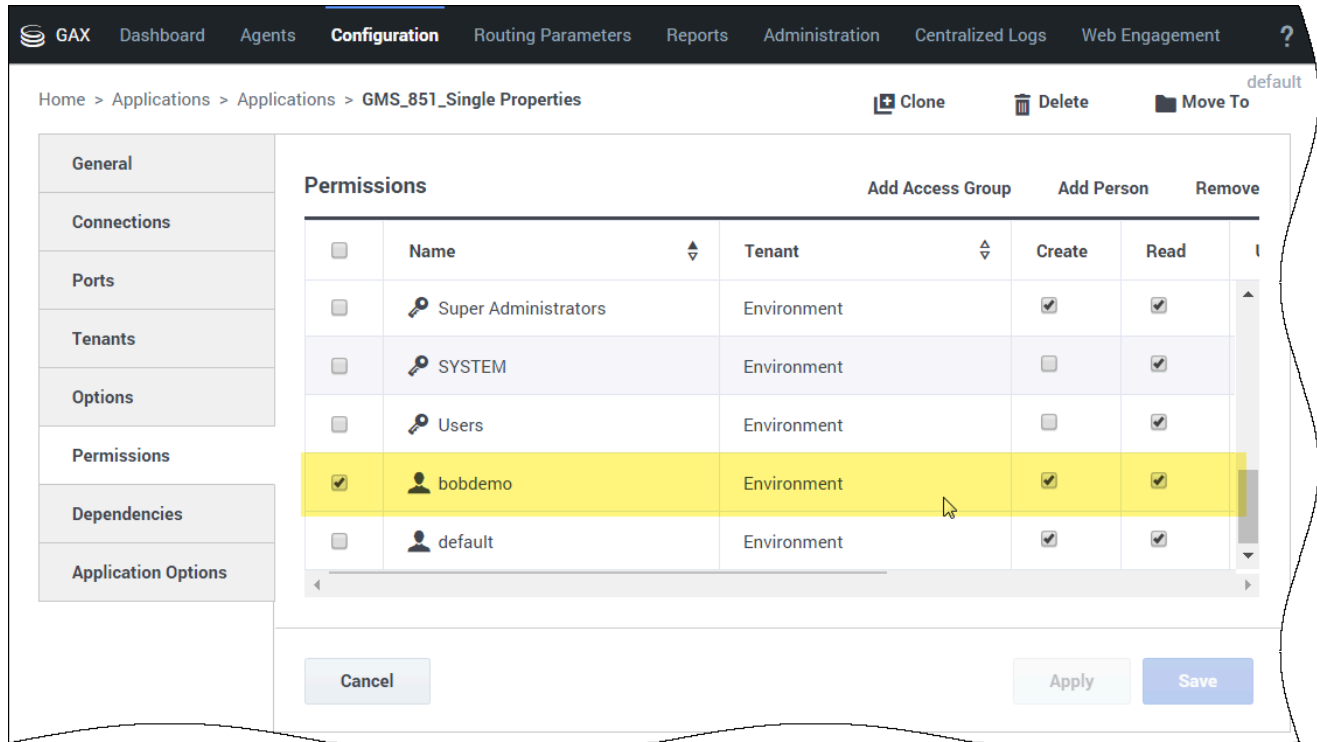
In the *Tenants* tab, click **Add** to set up the list of tenants that use your Genesys Mobile Services application.

Verify your Metadata



You may need to these metadata, for instance, if you are installing a Context Services application.

Set up Permissions for GMS Users

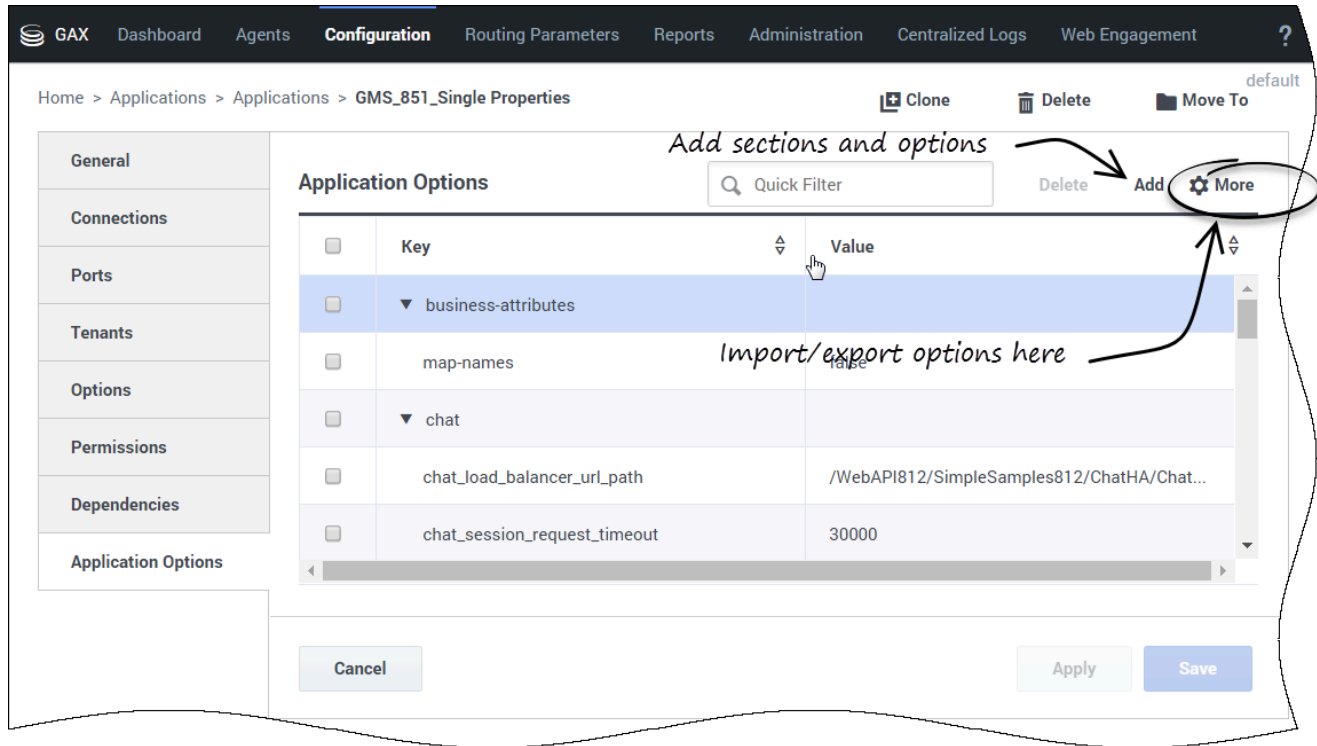


In the *Permissions* tab, you must add a user that is allowed to read/write data into GMS related configuration objects (for example, GMS Application, Transaction Lists for Resources/Patterns, and so on).

To do this, either click **Add Access Group** to browse the **Environment\Administrators** or **Add User** to browse the GMS Administrator set up previously.

Alternatively, you can simply use the *Default* user, which is already part of the Administrator Group.

Set up GMS options



In the **Application Options** tab, you can set up all your GMS options. Refer to the [Options reference](#) for details and check further chapters of this book. For Digital Channels API configuration options, refer to [Configuring the Digital Channels API](#).

You can now save and close your GMS application object.

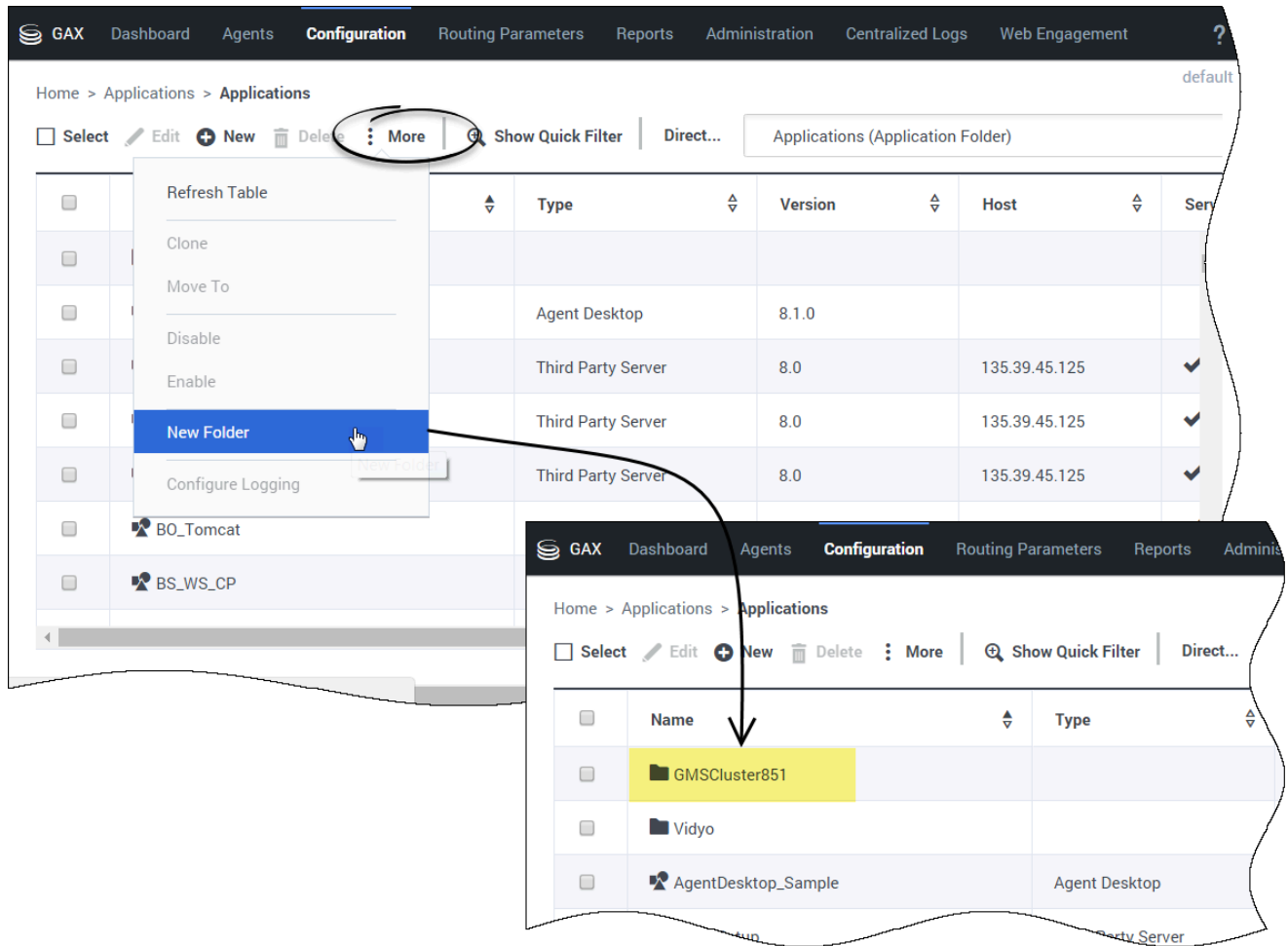
Next Steps: [Install Genesys Mobile Services](#).

Cluster Deployment

Purpose: To create and configure a GMS Application Cluster object for Genesys Mobile Services.

Prerequisites: [Import the Application Templates](#).

Create a folder for the GMS cluster (Best practice)

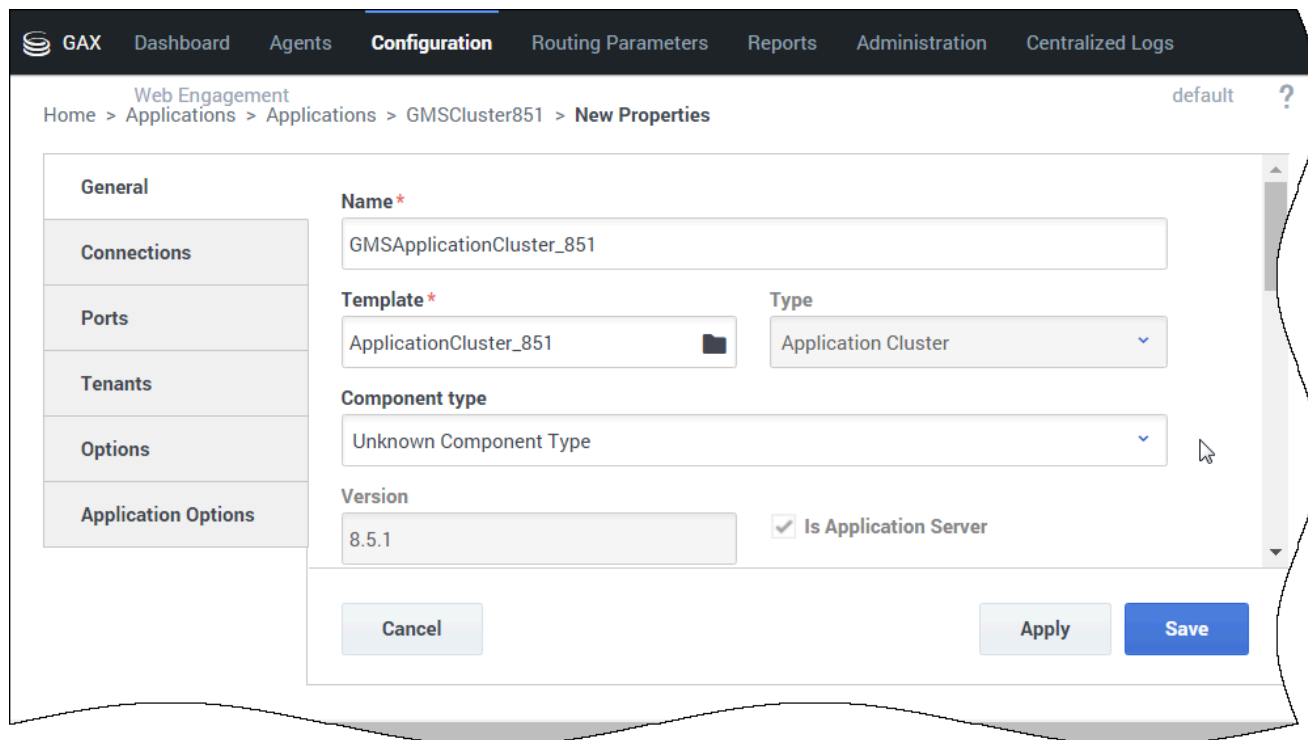


In the Configuration Manager tab, click the Applications link in the Environment menu; then, click More > New Folder to create a folder that will contain all of your application nodes. Set the new folder to a meaningful name, such as *GMScluster851*.

Important

Do not use spaces in the folder name.

Create a GMS cluster application



The screenshot shows the GAX Configuration page for creating a new application. The breadcrumb trail is: Home > Applications > Applications > GMSCluster851 > New Properties. The page title is "Web Engagement" and the user is logged in as "default". The left sidebar contains a menu with the following items: General, Connections, Ports, Tenants, Options, and Application Options. The main content area is titled "New Properties" and contains the following fields:

- Name ***: GMSApplicationCluster_851
- Template ***: ApplicationCluster_851
- Type**: Application Cluster
- Component type**: Unknown Component Type
- Version**: 8.5.1
- Is Application Server**

At the bottom of the form are three buttons: Cancel, Apply, and Save.

Create a new application with the wizard (as detailed for a single node), but use the **ApplicationCluster_851.apd** template instead of the regular template.

- Select **Unknown Component Type** in Component type.

Important

Do not use spaces in the application cluster name.

Set the Cluster General Settings

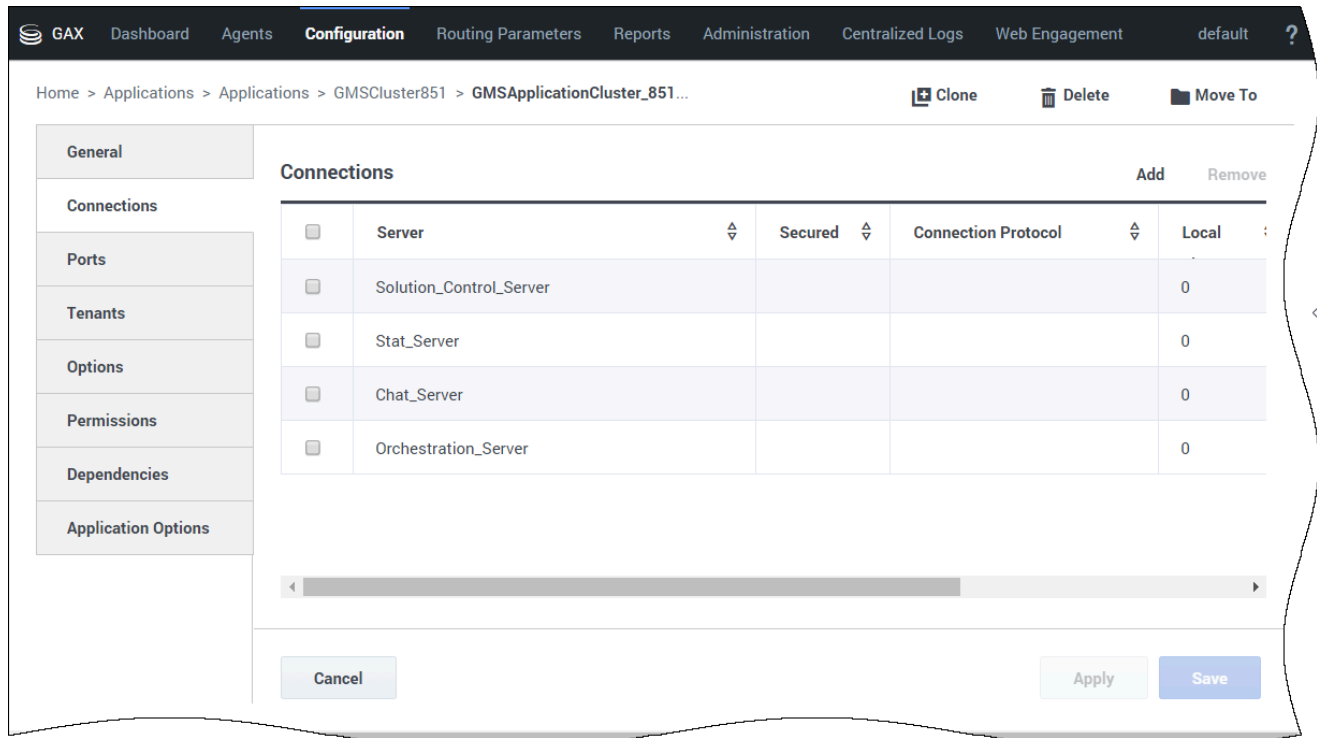
The screenshot shows the GAX Configuration page for creating a new application object. The breadcrumb navigation is: Home > Applications > Applications > GMSCluster851 > New Properties. The page is titled "Web Engagement" and has a "default" language setting. The left sidebar contains a menu with "General" selected, and other options: "Connections", "Ports", "Tenants", "Options", and "Application Options". The main content area is titled "New Properties" and contains the following fields and options:

- Working Directory ***: A text input field containing ".".
- Command Line ***: A text input field containing ".".
- Command Line Arguments**: A text input field containing "dummy".
- Startup Timeout ***: A text input field containing "90".
- Shutdown Timeout ***: A text input field containing "90".
- Auto-Restart**
- Primary**
- Host ***: A text input field containing "135.39.45.125".

At the bottom of the form are three buttons: "Cancel", "Apply", and "Save".

- Type dummy values in *Working Directory* field; this option will be overwritten when Genesys Mobile Services is installed; however, having values in these fields is required to save the Application object.
- Uncheck **Log On As System** option and browse to your GMS Admin account to set up **Log On Account**. If you do not set up **Log On Account**, the GMS Admin UI will throw permission errors when you will try to create GMS Built-in services.
- Check that the State is enabled, and browse a dummy hostname for this application (mandatory to be able to save the application).

Click **Apply** or **Save**. Your application is created.



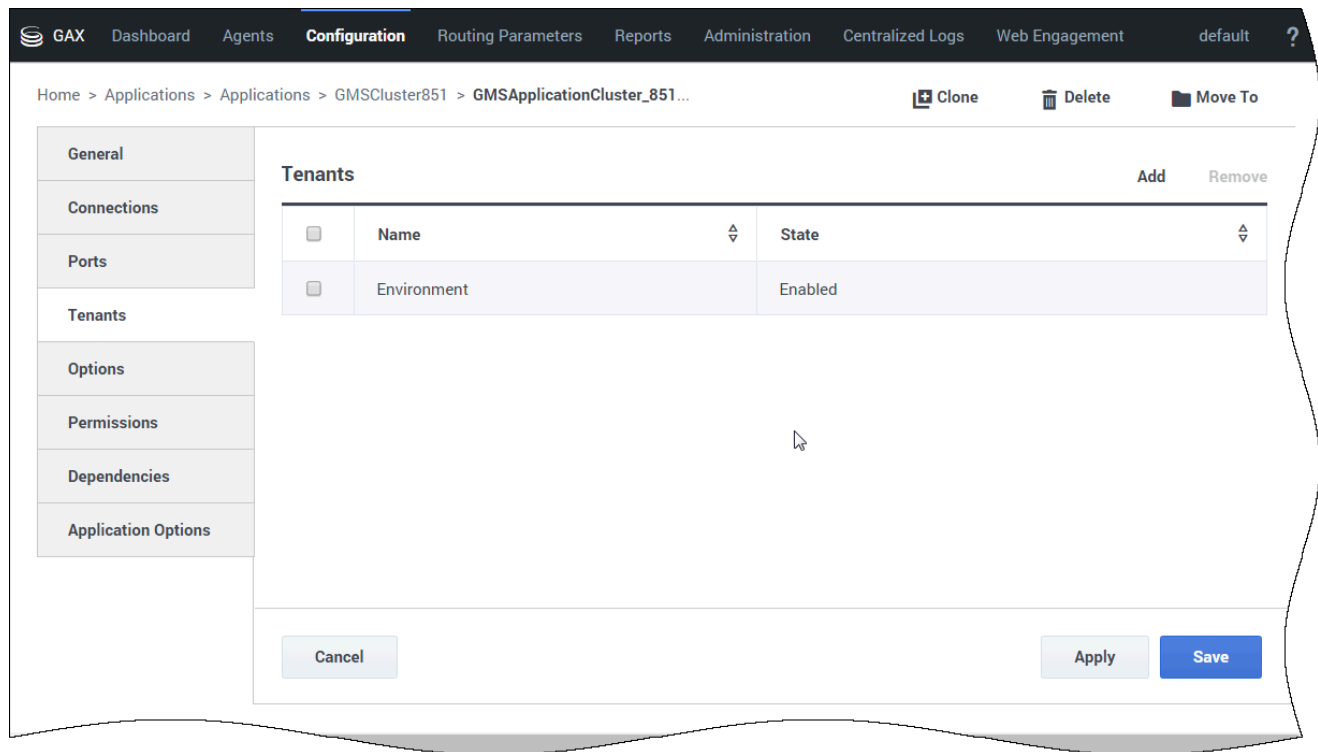
Set the *Connections* list and specify all of the servers to which Genesys Mobile Services must connect:

- Orchestration Server (ORS) - optional: Add this connection (using HTTP) if you plan to use GMS Callback features.
- Solution Control Server - optional: Add this connection if you plan to use GMS Chat features.
- Chat Server - optional: Add this connection if you plan to use GMS Chat features.
- Stat Server - optional: Add this connection if you plan to use GMS Reporting features.
- Message Server - optional: Add this Connection if you plan to use Log features.
- URS Server - optional: Add this Connection if you plan to use EWT features.

Important

If you do not plan to use GMS configuration defaults at startup or if you plan to use a Configuration Server proxy, refer to the following documentation:

- [Using the Management Layer](#)
- [How to Use Startup Files](#)
- [Configuration Server Proxy](#)



- In the Tenants section, click **Add** to set up the list of tenants that use your Genesys Mobile Services applications (one or more in a multi-tenant environment).

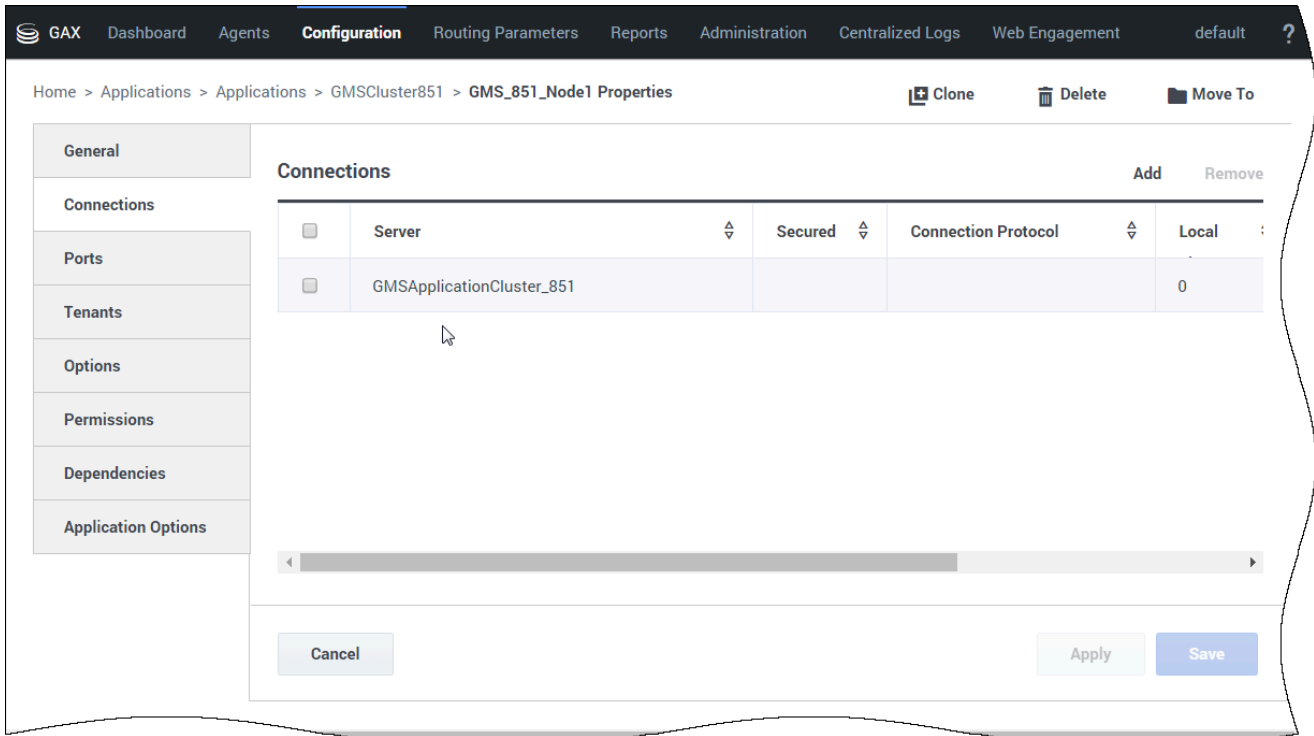
Create and Configure a GMS Application Node (Cluster)

Purpose: To create and configure a GMS Application object for a GMS node that is part of a cluster.

Prerequisites:

- [Import the Application Templates](#)
- [Create a GMS Administrator](#)
- [Create the GMS Application Cluster Object](#)

Create your nodes

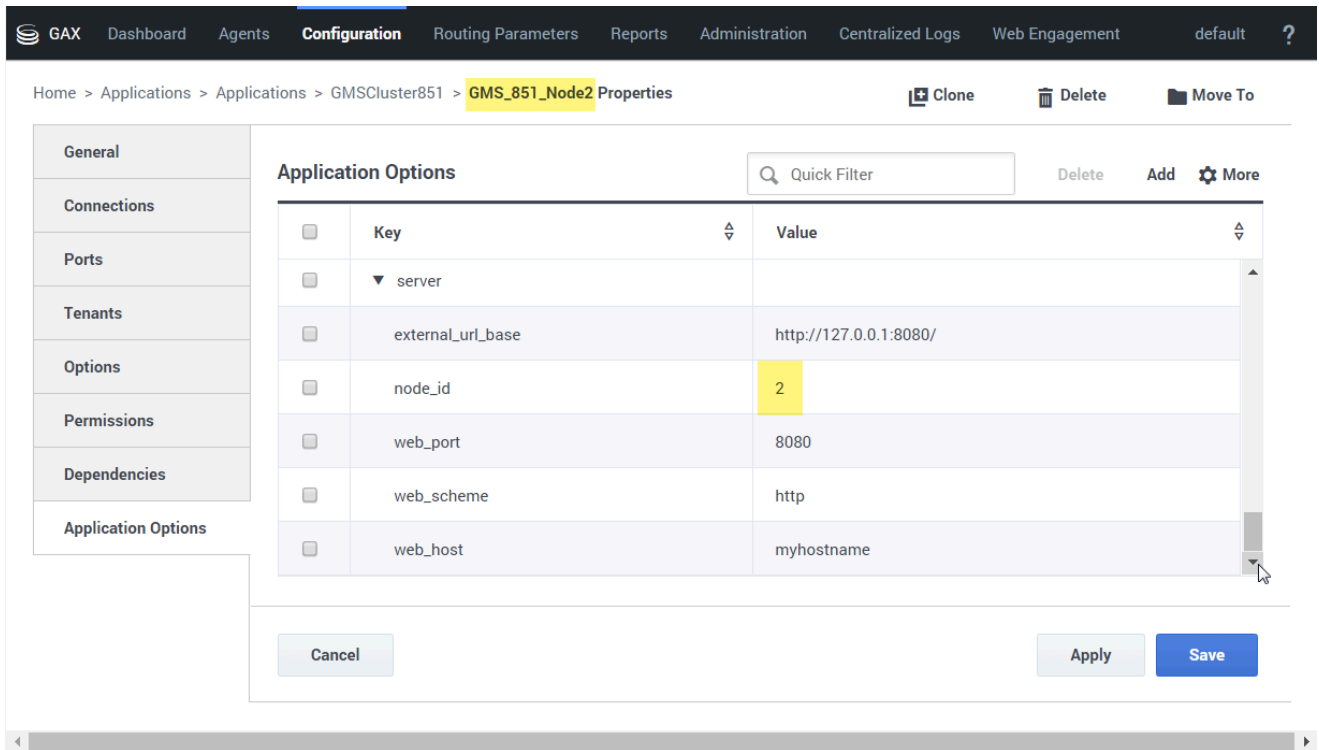


To create a GMS Node, you must create a **GMS application** in your newly created Application Cluster folder.

Important

Make sure to **set up permissions** for all nodes.

When setting your node connections, add the GMS Application Cluster that you just created previously.



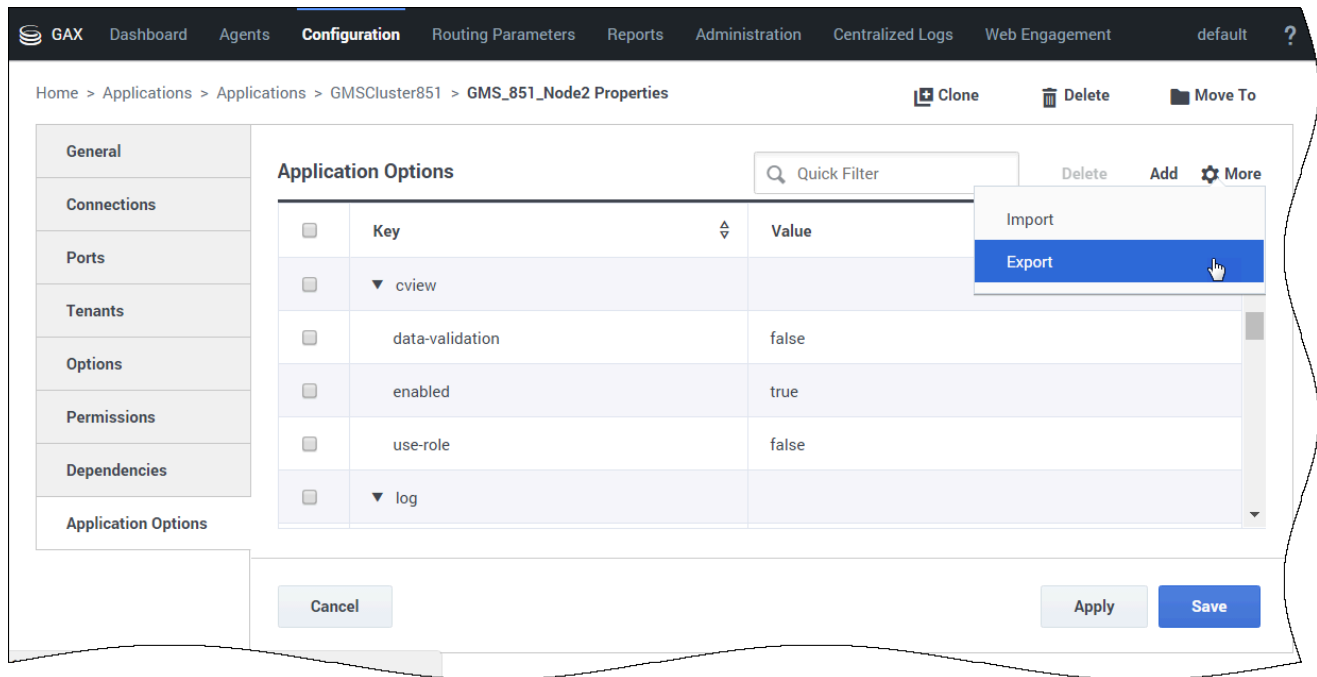
For each node, select the **Application Options** tab and configure `node_id` in the **server** section. The node ID is used to identify each GMS node and will be used for resource allocation, access code, and more. `node_id` must be unique and must be defined in each GMS application, and not at Cluster level.

Configure the following optional parameters in the **server** section if you need to force GMS to discover your nodes:

- `web_host`
- `web_port`
- `web_scheme`

`web_host` is used for GMS communications between other GMS nodes and other servers like ORS and URS. You can use `web_port` and `web_scheme` with `web_host` to build the GMS address for other servers that need to communicate with GMS.

Share Options Across the GMS Cluster



Set up additional configuration options in one of your nodes (as detailed in further chapters), and then click **Export** to save these options to a file.

Then:

1. Remove all these options from your node, and save.
2. Edit the GMS Application Cluster and click **Import** in the **Options** tab to load the options file.
3. Make sure to remove the `node_id` setting from your cluster's configuration.

Tip

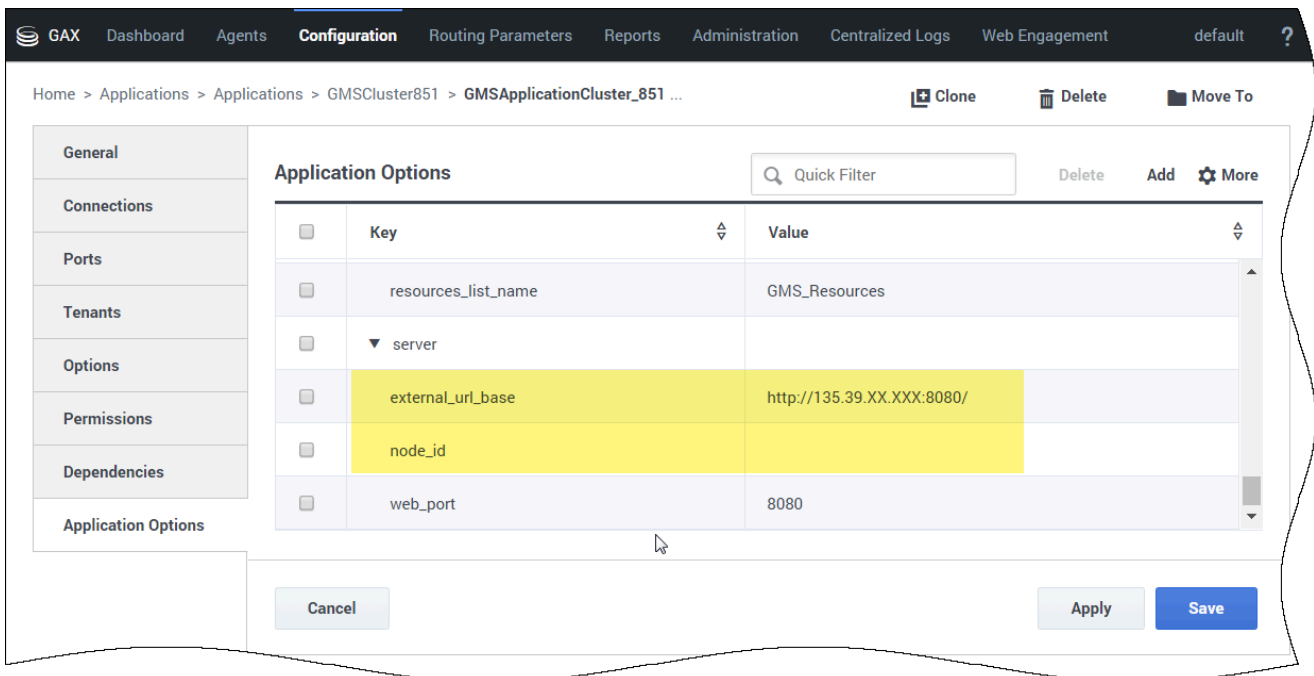
You can set up options in your GMS node, but these options will not be shared. This is useful to set specific settings for Cassandra storage, logs, and more.

If your GMS version is lower than 8.5.202, and if you configure a section in one of the GMS Nodes, its content overwrites the same section defined in your cluster application object. Starting in 8.5.202, only the options that you define in your GMS node supersede those configured in the same section of your cluster. This means that the options that are defined in your cluster apply to your node even if they are not configured in this node.

GMS Cluster	GMS Node	Result
-------------	----------	--------

<pre>[cview] allow-custom-ids=true data-validation=true enabled=true use-role=false expiration=45s</pre>	<pre>[cview] allow-custom-ids=true data-validation=true enabled=true use-role=false</pre>	<p>Prior to 8.5.202, the expiration parameter does not apply to the GMS node, because the cview section of the GMS node overwrites the one defined in the GMS Cluster.</p> <p>Starting in 8.5.202, the expiration parameter applies to the GMS node, because it is configured in the cluster and not overwritten in the node's section.</p>
--	---	---

Enable Load Balancing in your Cluster Application



In the Options tab of your GMS Cluster application, section server, select external_url_base.

Set this option to the external URL of your cluster's load balancer, which will receive the requests submitted by your client applications.

Important

This step ensures that all GMS Nodes will share the same load balancer. See [the Options reference for more details](#).

You can now save and close your GMS application cluster object.

Next Steps: You must now install Genesys Mobile Services for each node of the cluster.

Install Genesys Mobile Services

Purpose: To install Genesys Mobile Services in your environment. In a cluster deployment, you must run the installation for each application node of the cluster.

Prerequisites: [Create a Genesys Mobile Services Application Object](#).

With basic Configuration Server details in place, you are ready to complete the installation process.

Important

- Genesys does not recommend the installation of its components via a Microsoft Remote Desktop connection. You should perform the installation locally.
- Starting in 8.5.206.04, GMS no longer supports embedded Cassandra. If you are using an older version and if you need additional instructions, refer to [8.5.1 installation instructions](#).

Important

Managing primary/backup config server at GMS startup.

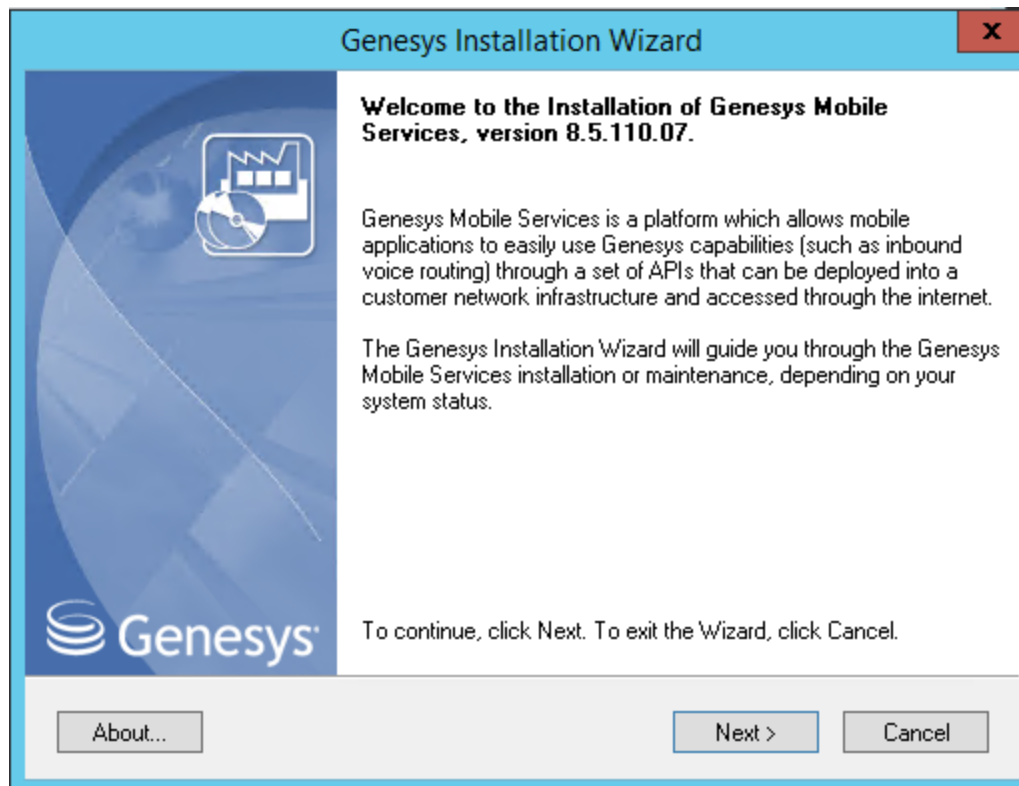
Now, GMS is able to manage a backup config server at startup using two new command line parameters:


- `-backuphost`: hostname/IP address for backup config server
- `-backupport`: port for backup config server

For example, `./launcher -host 135.39.45.14 -port 2020 -backuphost 135.39.45.14 -backupport 3020 -app "GMS236"`.

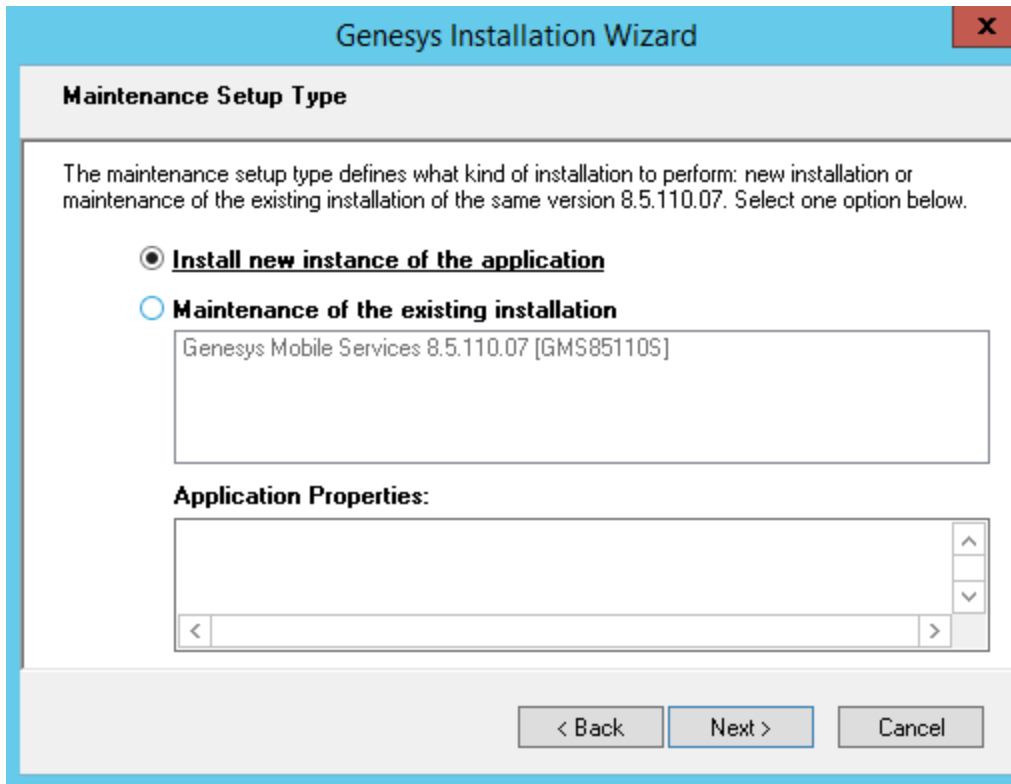
Installing Genesys Mobile Services on Windows

Start the Installation Wizard



In your installation package, locate and double-click the  `setup.exe` icon in windows or `install.sh` in Linux.

The installation starts. Click **Next** and follow the instructions below.



Select **Install new instance of the application** if you are installing GMS for the first time on this server. Select **Maintenance of the existing application** to upgrade Genesys Mobile Services and follow the steps detailed in the [Upgrade Genesys Mobile Services](#) section of this guide.

Genesys Installation Wizard

Connection Parameters to the Configuration Server

The parameters in the Host and User fields are required to establish a connection to Configuration Server.

Host

Specify the host name and port number for the machine on which Configuration Server is running.

Host name: 135.39.45.14

Port: 2020

User

Specify your Configuration Server user name and password.

User name: default

Password: ●●●●●●●●●●

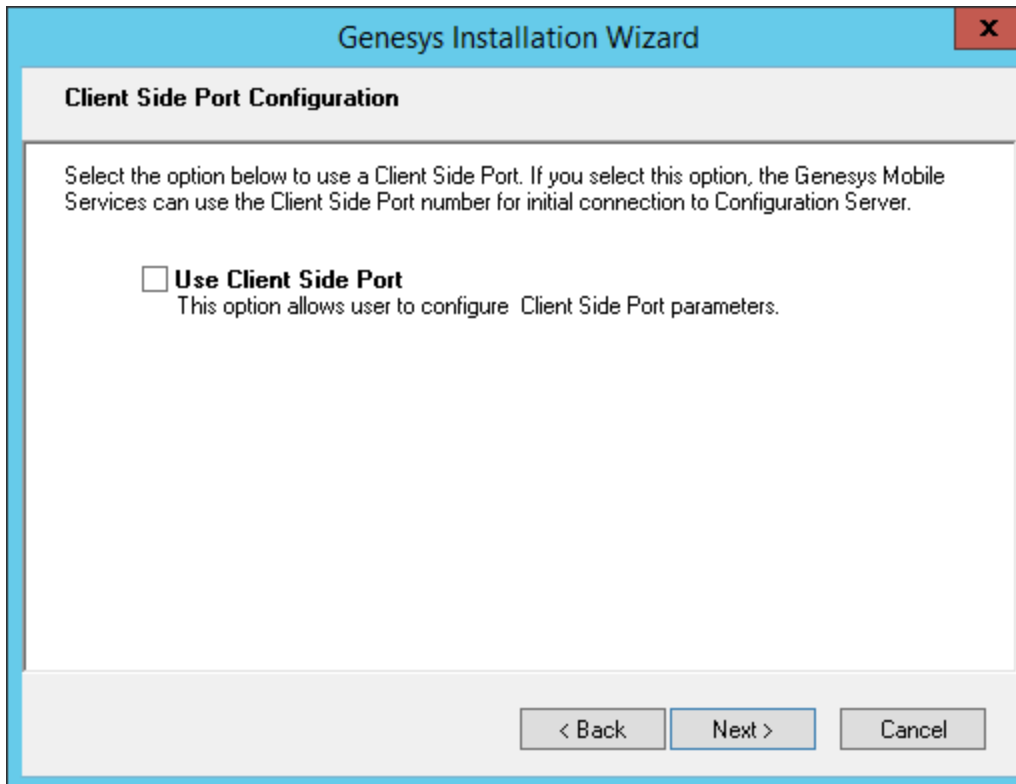
< Back Next > Cancel

In **Connection Parameters to the Configuration Server**, specify the host name and port number where Configuration Server is running. (This is the main "listening" port entered in the **Server Info** tab for Configuration Server, which is also used for authentication in the Configuration Manager login dialog box.)

Click **Next**.

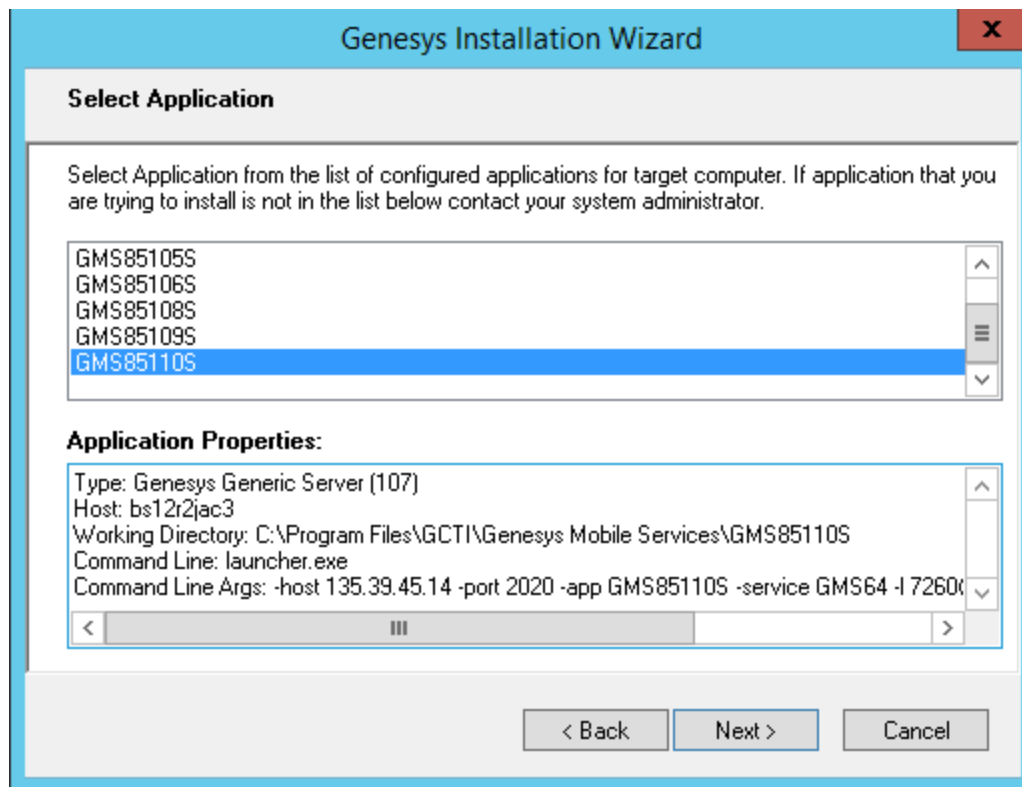
Important

When setting a user to connect to the Configuration Server, do not choose a user with an empty password. This would make your installation fail.



Select **Client Side Port** if GMS must communicate with the Configuration Server through specific ports, for example, if the Configuration Server is behind a firewall. Otherwise, click **Next**.

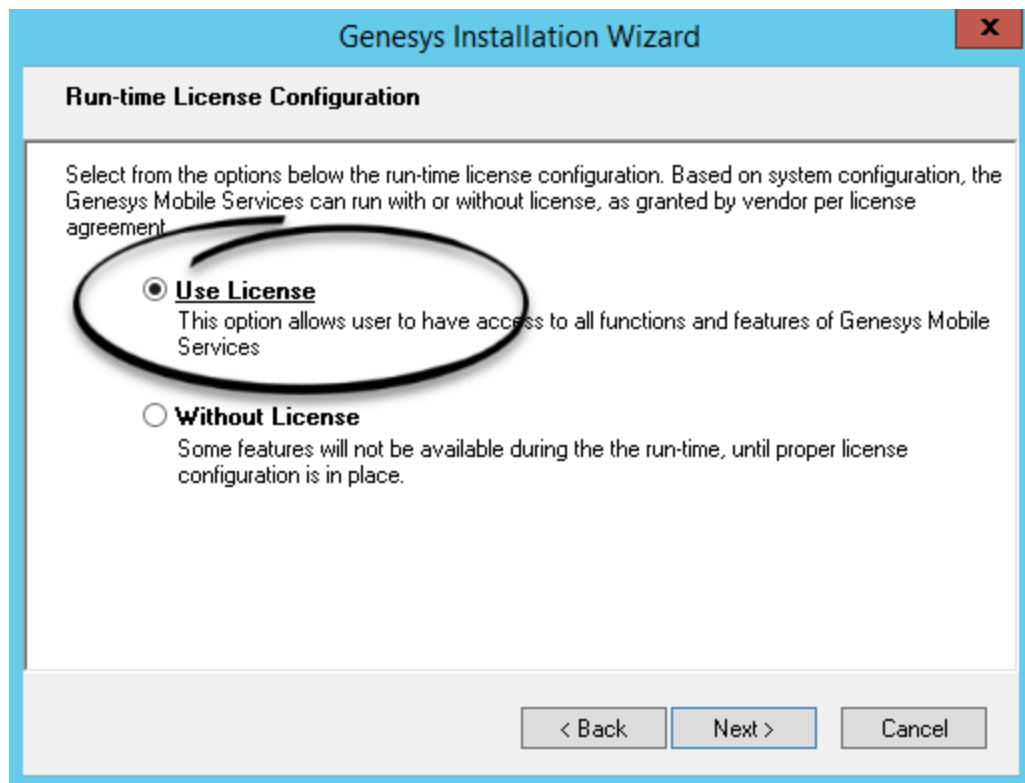
Select your application



Select the Genesys Mobile Services Application object that you created before the installation. The **Application Properties** area shows information previously entered in the **Server Info** and **Start Info** tabs of the selected Application object.

You cannot select a cluster application. You must install each node in a cluster deployment.

Select Use Licence



Select **Use Licence** if you want to use Context Services with GMS.

Important

You must purchase this specific license to use Context Services.

Genesys Installation Wizard

Access to License

Select the license access type and parameters from the options below.

- License Manager**
Option requires information about parameters for the server where License Manager installed and running.
- License File**
Option requires full path to the License File location.

License Manager

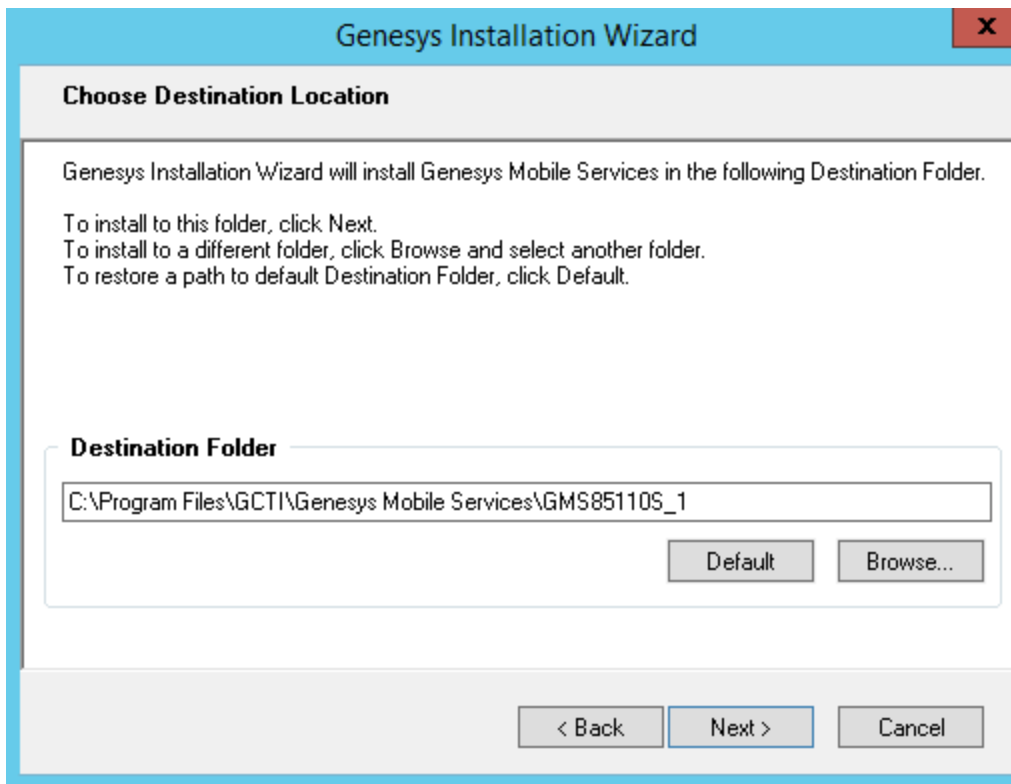
Specify Host name and Port for the machine where the FLEXlm License Manager is running.

Host name:

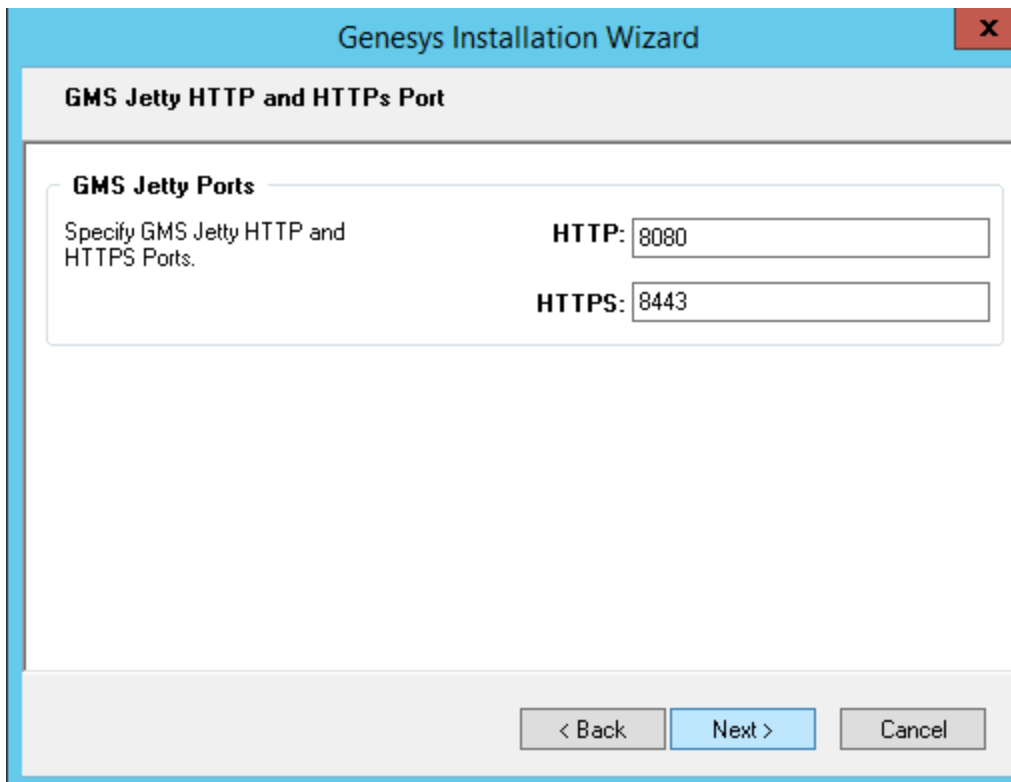
Port:

< Back Next > Cancel

Enter your License parameters and click **Next**.



In **Choose Destination Location**, keep the default destination or browse for the desired installation location under **Destination Folder**.



The screenshot shows a window titled "Genesys Installation Wizard" with a close button (X) in the top right corner. The window has a light blue header and a light gray footer. The main content area is titled "GMS Jetty HTTP and HTTPS Port" and contains a section labeled "GMS Jetty Ports". Below this section, there is a text prompt: "Specify GMS Jetty HTTP and HTTPS Ports." To the right of this prompt are two input fields. The first field is labeled "HTTP:" and contains the value "8080". The second field is labeled "HTTPS:" and contains the value "8443". At the bottom of the window, there are three buttons: "< Back", "Next >" (highlighted in light blue), and "Cancel".

In **GMS Jetty Ports**, enter the Jetty ports that will be used later to generate the [Jetty XML configuration snippet](#).

Select the Installation Type

Genesys Installation Wizard

Server Installation Mode Parameters

Select the server type and parameters from the options below.

Regular GMS
This option allows user to install a regular GMS application and access to all application's functions and features.

Chat Server only
Chat Server feature only will be available during the the application run-time.

GMS/WebAPI Server RAM Size
Specify RAM size (GB) for Cassandra database.

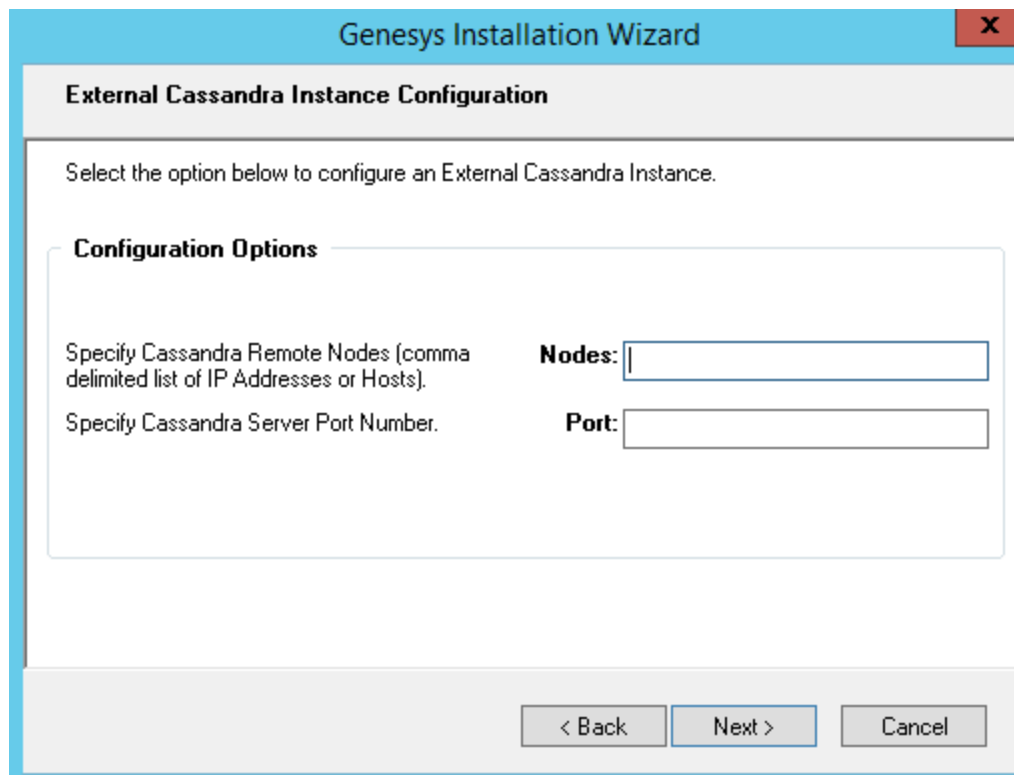
RAM size: 2

< Back Next > Cancel

Choose **Regular GMS** if you wish to implement voice scenarios, Callback, Context Services. This option enables the full GMS functionality—including the Digital Channels API. It also enables the samples for Chat API Version 1, Chat API Version 2, and GMS Chat. **Note:** This mode requires Cassandra.

Choose **Chat Server Only** to enable only the Digital Channels API, that is, Chat API Version 2, Chat API Version 2 via CometD (since GMS version 8.5.230.06, see release notes), Email API Version 2, and Open Media API. When you choose this mode, you will not see the Cassandra installation panels, and none of the other GMS features, including the Chat API Version 1 and Chat samples, will work. Click **Next**.

Configure an External Cassandra



The screenshot shows a window titled "Genesys Installation Wizard" with a close button (X) in the top right corner. The main title of the dialog is "External Cassandra Instance Configuration". Below the title, there is a text prompt: "Select the option below to configure an External Cassandra Instance." Underneath this is a section titled "Configuration Options" which contains two rows of input fields. The first row is labeled "Specify Cassandra Remote Nodes (comma delimited list of IP Addresses or Hosts)." and has a text box labeled "Nodes:". The second row is labeled "Specify Cassandra Server Port Number." and has a text box labeled "Port:". At the bottom of the dialog, there are three buttons: "< Back", "Next >", and "Cancel".

Important

Only for **Regular GMS** installation.

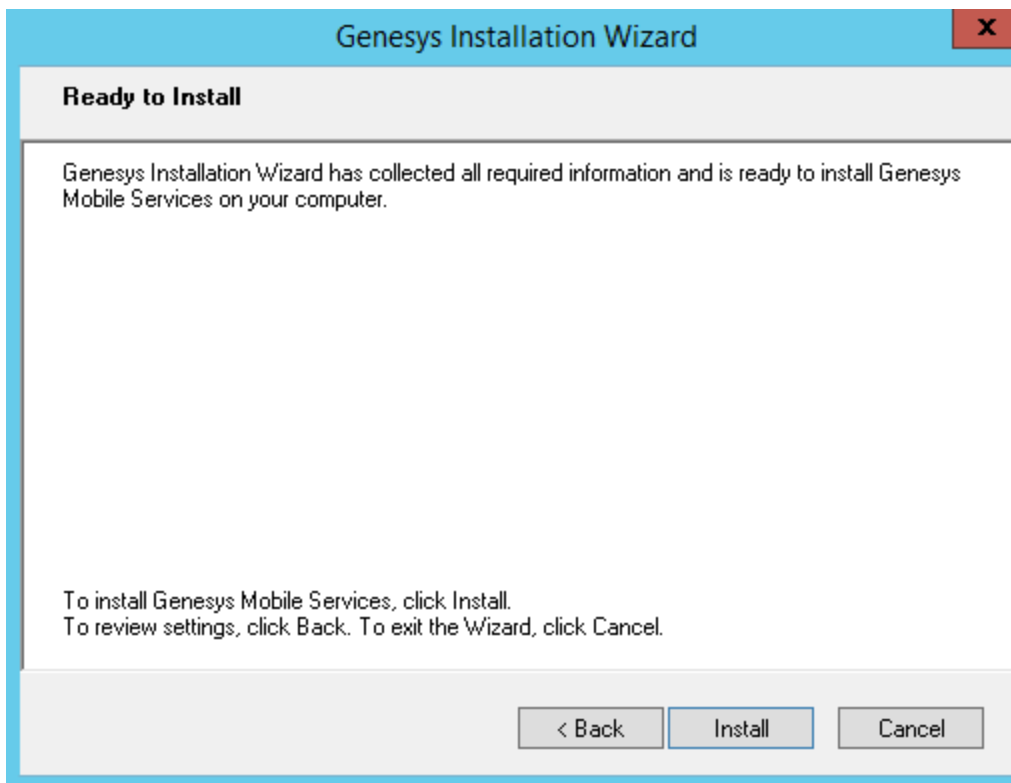
Enter the comma-separated list of Cassandra nodes, and for the port value, enter the **native_transport_port** value (9042 by default). Note: **native_transport_port** is the thrift port for client connections in the `cassandra.yaml` configuration file of your external Cassandra instance.

Click **Next**.

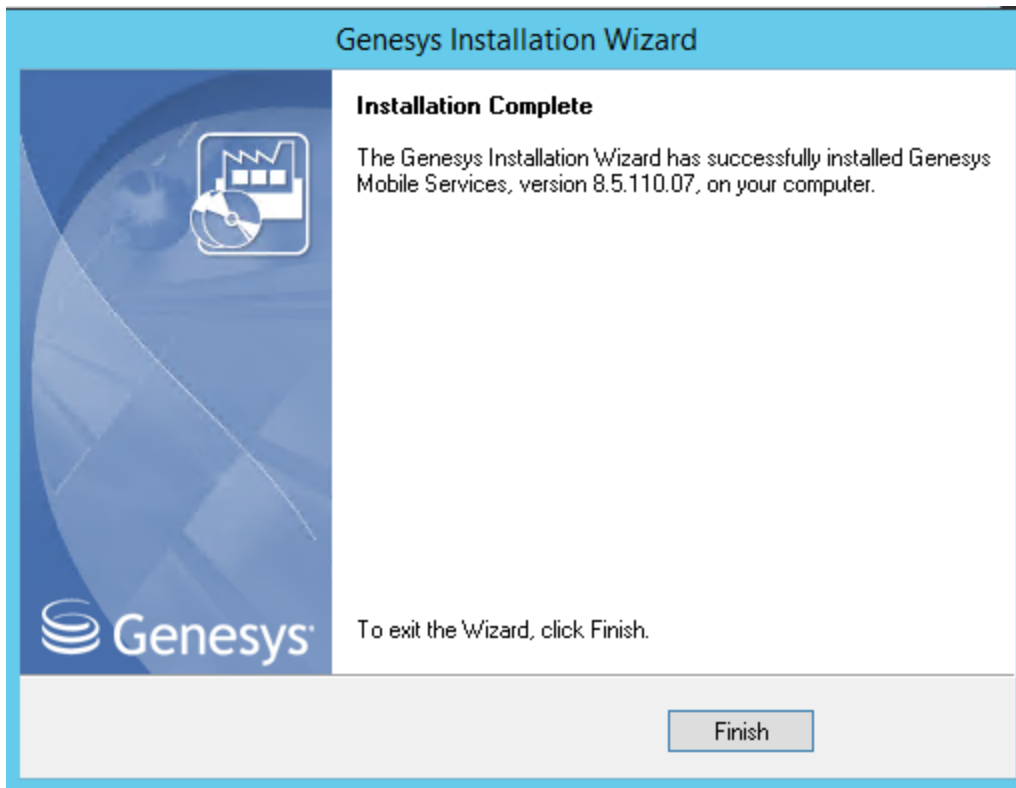
Warning

If you are installing GMS on a production server or on a Windows host, choose an external Cassandra instance which is installed on a Linux host.

Complete the Installation



The **Ready to Install** screen appears. Click **Install**. The Genesys Installation Wizard indicates that it is performing the requested operation.



When through, the **Installation Complete** screen appears. Click **Finish** to complete your installation of Genesys Mobile Services.

Repeat this procedure to install GMS on other hosts.

Installing Genesys Mobile Services on Linux

```
[genesys@bsjacentos7 gms-8.5.110.07]$ sh install.sh
*****
* Welcome to the Genesys 8.5 Installation Script *
*****

Installing Genesys Mobile Services, version 8.5.110.07

Please enter the hostname or press enter for "bsjacentos7" =>

Please enter the following information about your Configuration Server:

Configuration Server Hostname =>135.39.45.14
Network port =>2020
User name =>default
Password =>

Client Side Port Configuration
Select the option below to use a Client Side Port. If you select this option,
the application can use Client Side Port number for initial connection to
Configuration Server.

Do you want to use Client Side Port option (y/n)?n
```

To install Genesys Mobile Services on Linux, run the `sh install.sh` command to start the Installation Script. This step-by-step script will request the same inputs than detailed previously in the Windows Installation Wizard.

Optional - Test your installation

*** Server Info**

Tenants:

Name	State
Environment	Enabled

* Host: 192.168.10.98

* Listening Ports:

ID	Port
No objects to display	

* Working Directory: C:\Program Files\GCTI\Genesys Mobile Services\GMS_Node1_851

* Command Line: launcher.exe

Command Line Arguments: -host demosrv.genesyslab.com -port 2020 -app GMS_Node1_851 -service GMS64_2 -l 7

* Startup Timeout: 90

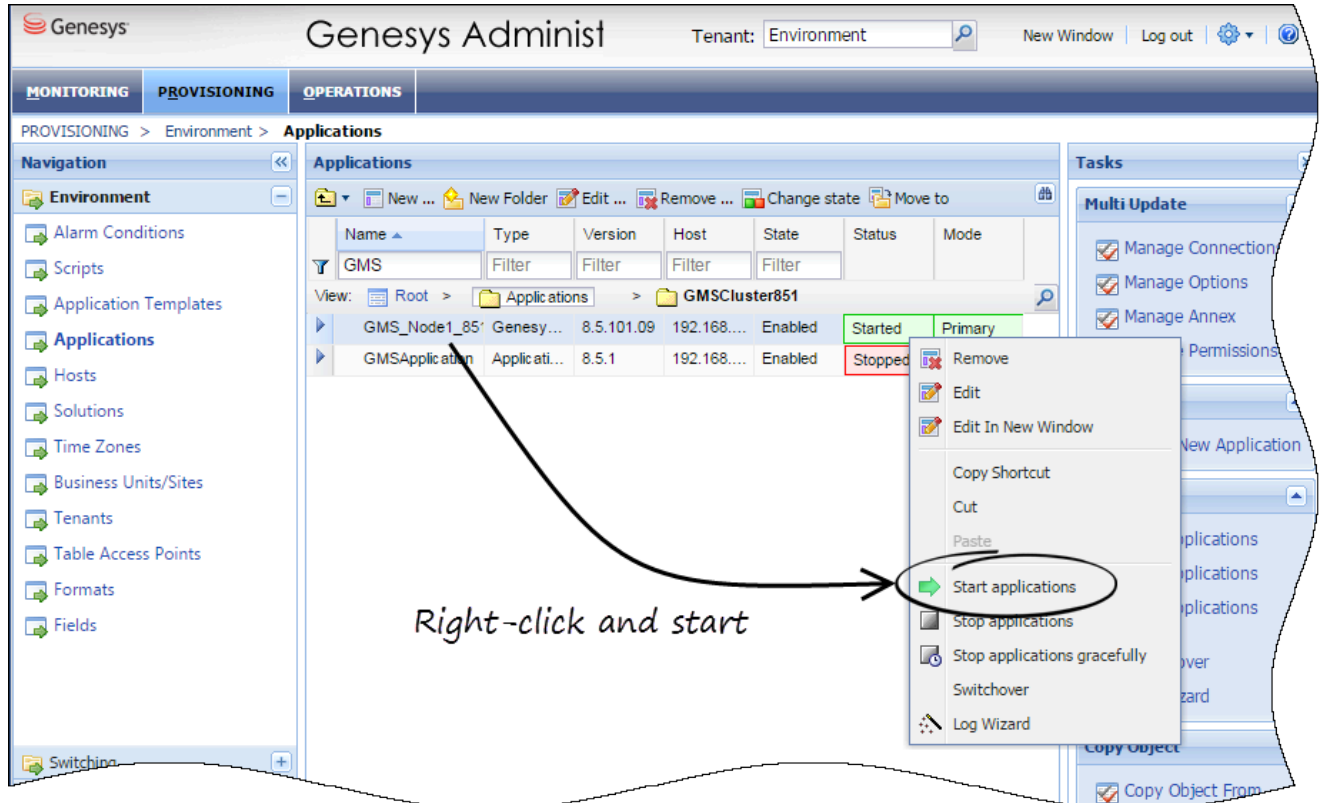
* Shutdown Timeout: 90

Backup Server: [Unknown Backup Server]

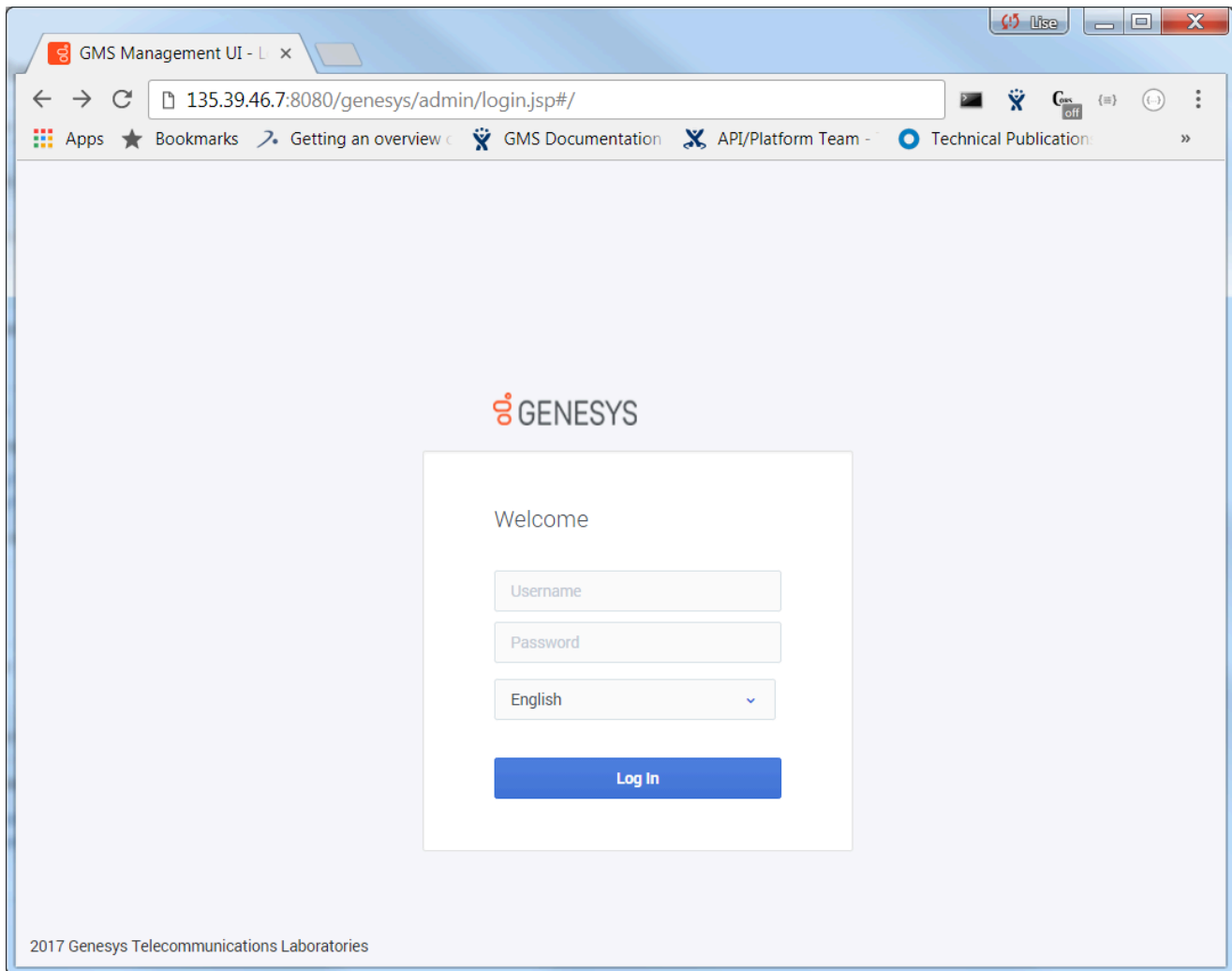
* Redundancy Type: Not Specified

Edit your GMS application in Genesys Administrator. In the **Server Info** section, you should be able to verify that your installation updated the **Working directory** and the **Command Line** fields.

Start your GMS application and browse the Service Management UI



In Genesys Administrator, start your GMS application.



Then, open your web browser for the following URL: `<GMS Local Host>:8080/genesys`

You should see the Service Management UI of your GMS application.

Note that, to work properly, this UI requires access to the following URLs:

`http://<GMS Local Host>:8080/genesys/1/admin/*`

Make sure to enable this access through your firewall and security if needed.

Next Steps: [Setting ORS dependencies.](#)

Upgrade Genesys Mobile Services

To upgrade GMS, run the Installation package and follow the instructions detailed below. Then, follow the **additional** upgrade steps introduced with the version that you chose for upgrading.

Linux Upgrade

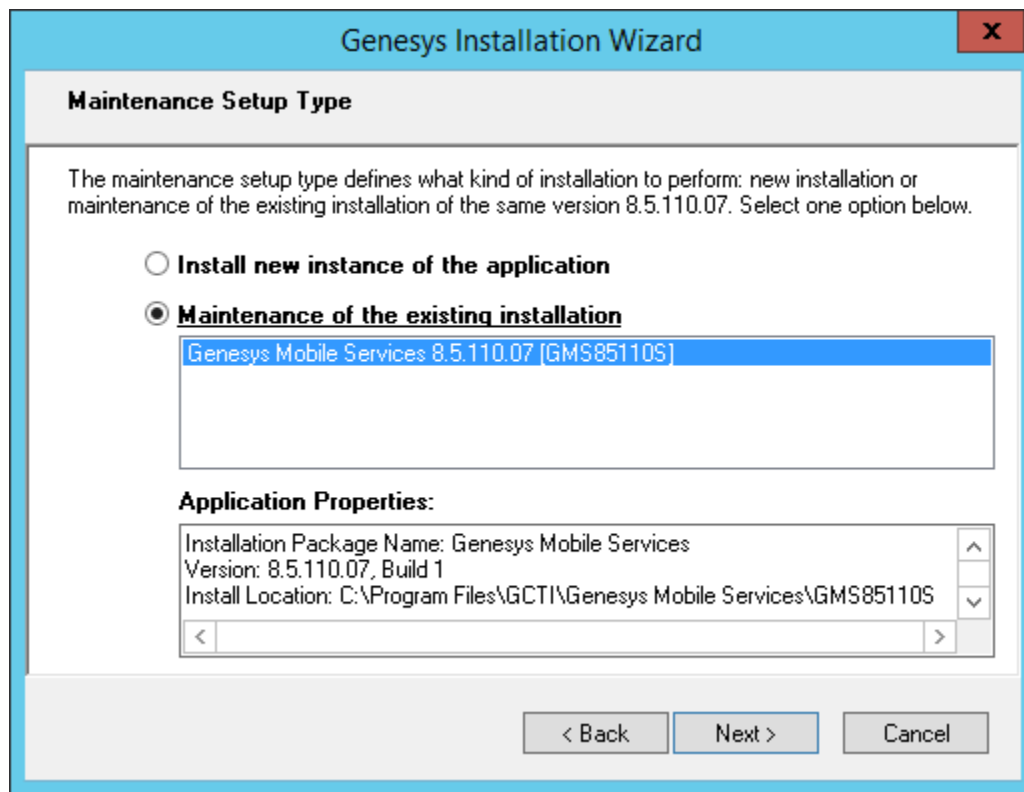
If you want to upgrade GMS on Linux, just run the installation command as if you were installing GMS for the first time.

```
sh install.sh
```

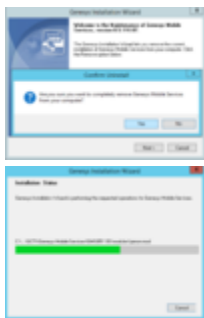
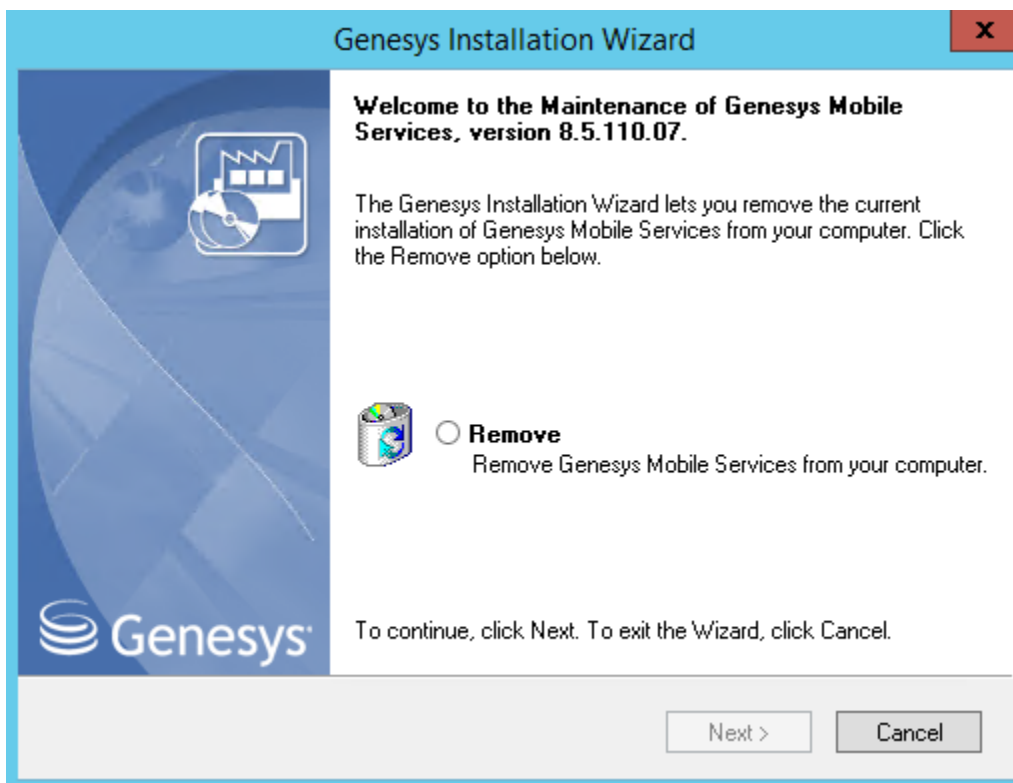
There is no specific step required for upgrading GMS.

Windows Upgrade

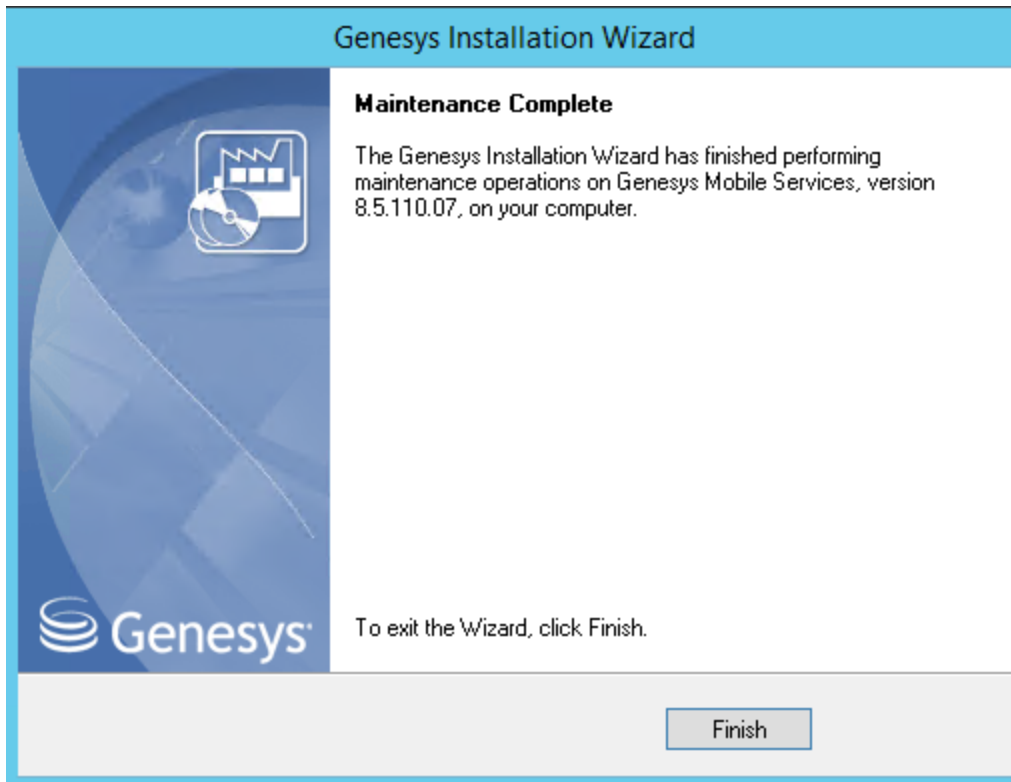
The **Installation Wizard** will prompt you to select the maintenance setup type on your GMS application.



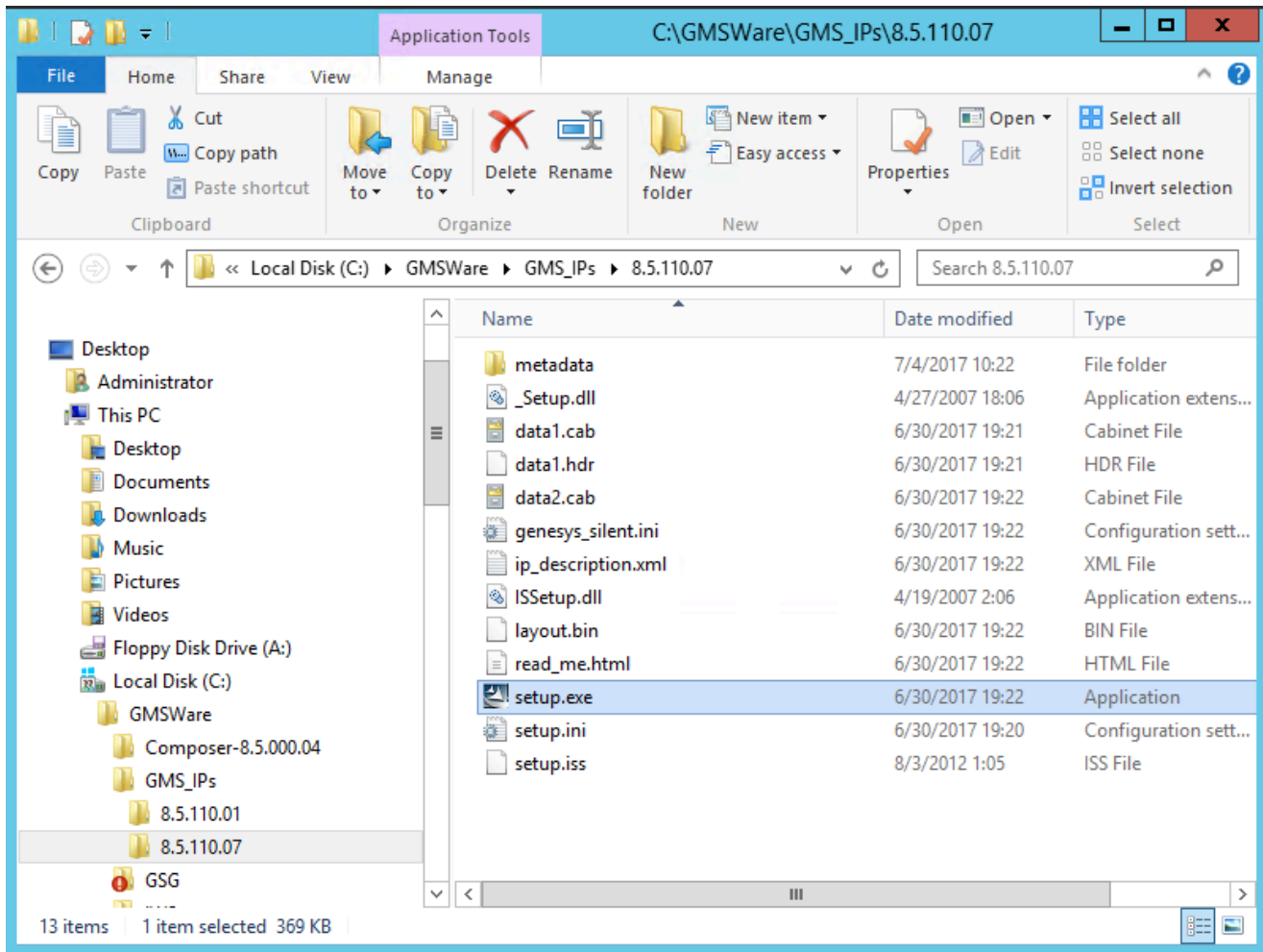
Select **Maintenance of the existing application** to upgrade Genesys Mobile Services.



The next prompt will start the removal of the current GMS installation.



Once the previous version is removed, the Wizard indicates that the Maintenance operation is done. Click **Finish**. This action will end the Installation Wizard.



Then, you can run the Installation Wizard again.

Genesys Installation Wizard

Connection Parameters to the Configuration Server

The parameters in the Host and User fields are required to establish a connection to Configuration Server.

Host

Specify the host name and port number for the machine on which Configuration Server is running.

Host name: 135.39.45.14

Port: 2020

User

Specify your Configuration Server user name and password.

User name: default

Password: ●●●●●●●●

< Back Next > Cancel

The Installation Wizard will start the installation of the new version.

Additional Upgrade Instructions Per Release

If GMS has external Cassandra configured, when you upgrade GMS, you need to import the Callback Template from the new GMS installation directory. Start the Service Management UI, [upload](#) the <GMS Installation Directory>/service_templates/callback.zip file, and restart ORS.

Starting in 8.5.102, Cassandra schemas are compatible with GMS 8.5.105+ and do **not** require any upgrade. But if you upgrade from GMS versions older than 8.5.102, you will need to manually update the Cassandra schemas in **one** of your nodes (not all). See details [here](#).

The following subsections detail the additional upgrade instructions (if any) that were introduced with a given version number.

8.5.208.09

Import the GMS_URS_Stategy_85208_v2.67 strategy available for download in the [Callback Solution guide](#).

8.5.206.04

GMS no longer supports embedded Cassandra. To migrate from embedded Cassandra to external Cassandra, proceed as follows:

1. Deploy a new GMS Cluster and new GMS nodes which use external Cassandra.
2. Reroute requests to the new GMS cluster.
3. Wait for in-progress customer operations to finish in the embedded Cassandra.
4. Stop the GMS node which uses embedded Cassandra.

8.5.201.04

Import the `GMS_URS_Strategy_85200_v2.64.1.zip` strategy available for download in the [Callback Solution guide](#).

8.5.200.07

- The minimum required version for URS is now 8.1.400.47.

8.5.114.09

- GMS now requires Interaction Routing Designer (IRD) 8.1.400.26 and Universal Routing Server (URS) 8.1.400.39. Upgrade these servers, and then import the latest `GMS_URS_Strategy_85114_v2.63.zip` strategy as detailed in the [Callback Solution guide](#).
- The GMS configuration options of the `lab` section were moved to the `features` section.

8.5.112.05

To use the new `_urs_vq_priority_boost_on_connect` option, import the `GMS_URS_Strategy_85109_v2.58.zip` strategy available for download in the [Callback Solution guide](#).

8.5.111.04

If you enabled the Bulk and Cancel feature in 8.5.110, rename the `enable-bulk-cancel-and-export-callback` option to `disable-bulk-cancel-and-export-callback` and set its value to `false`.

8.5.109.08

Release 8.5.109.08 or later requires the [download and update](#) of the following strategies:

- `WaitForTarget` version 2.5+
- `SetRouteDelay` version 2.1+

If you are upgrading from a version older than 8.5.109.05, you may need to modify the configuration

option `_enable_in_queue_checking`. By default, this option blocks all the Callback requests issued from a customer number that has already appeared twice in a queue. If you wish to keep the previous behavior (with no automatic blocking), set this option to `false`.

Older Versions

If you are upgrading from 8.5.104 or earlier, install GMS as usual, and then **update the DFM files** to ensure correct callback processing.

Configuring Routing Dependencies

Before getting started with your GMS services, you must first ensure that external dependencies are configured properly. The following outline will guide you through each dependency.

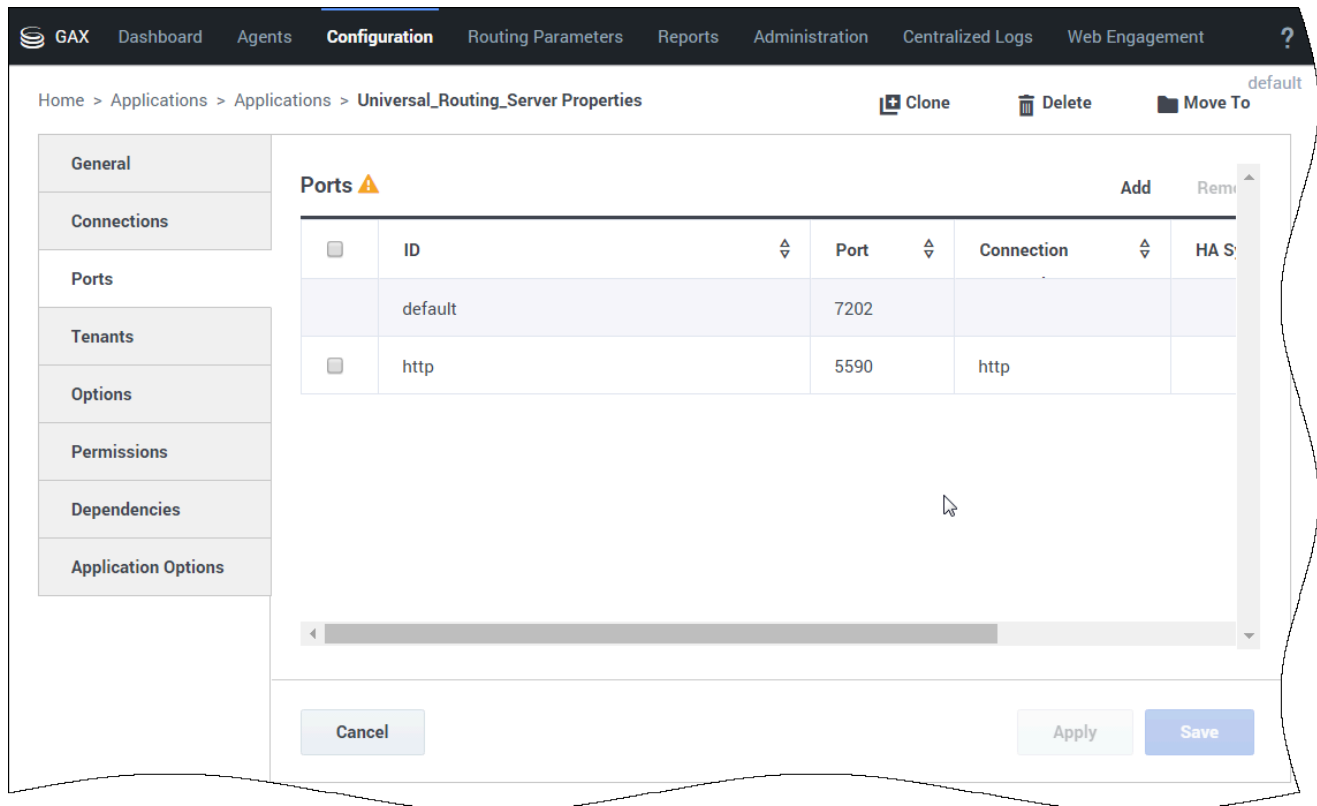
Note: These procedures assume a multi-tenant configuration and Tenant = Environment.

Universal Routing Server

GMS requests URS to start strategies by HTTP, and GMS receives asynchronous Callbacks from URS by HTTP. To enable the HTTP interface:

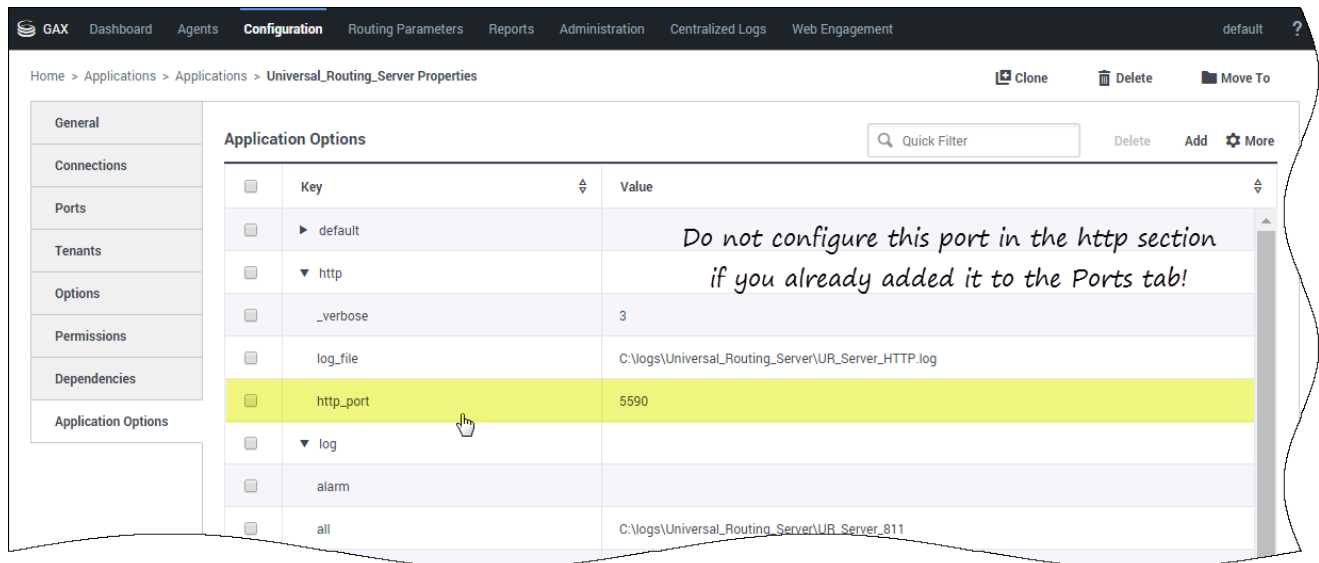
Create a Listening HTTP Port in URS

URS will listen on this port for incoming HTTP requests. Basically, this steps turns URS into an HTTP server.



In Genesys Administrator Extension, edit your URS application.

Add an HTTP listening port with a port ID `http` in the **Ports** tab. Make a note of this port number as you will need it later when configuring GMS and ORS-based services.



You can also do this by creating the `http_port` option in the `http` section of your **Application Options** tab.

Warning

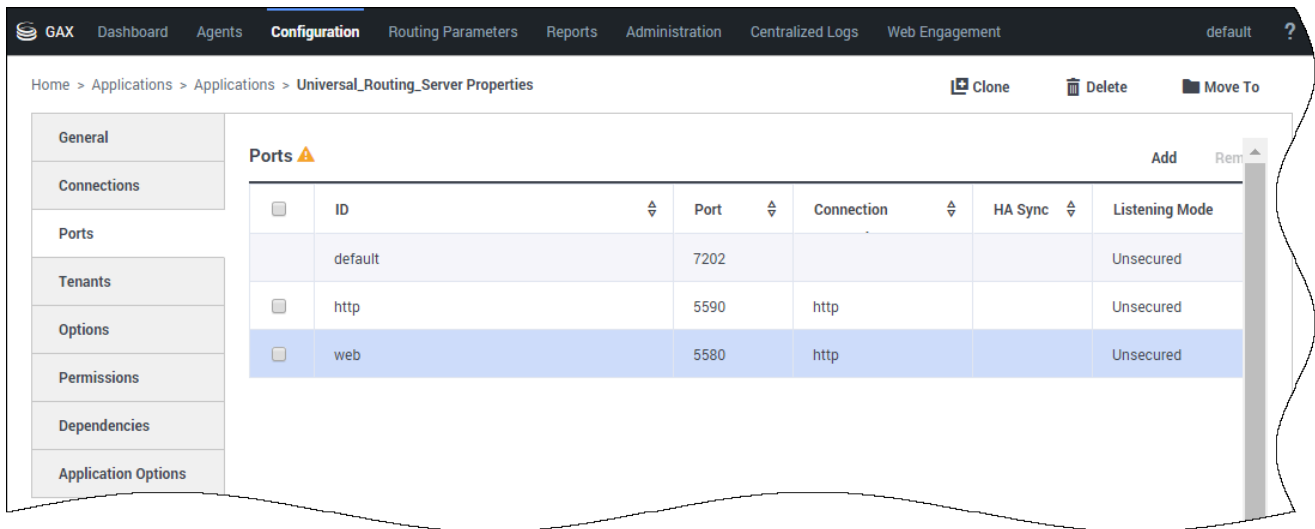
This HTTP port needs to be created in one place only.

Enable Web HTTP Replies in URS

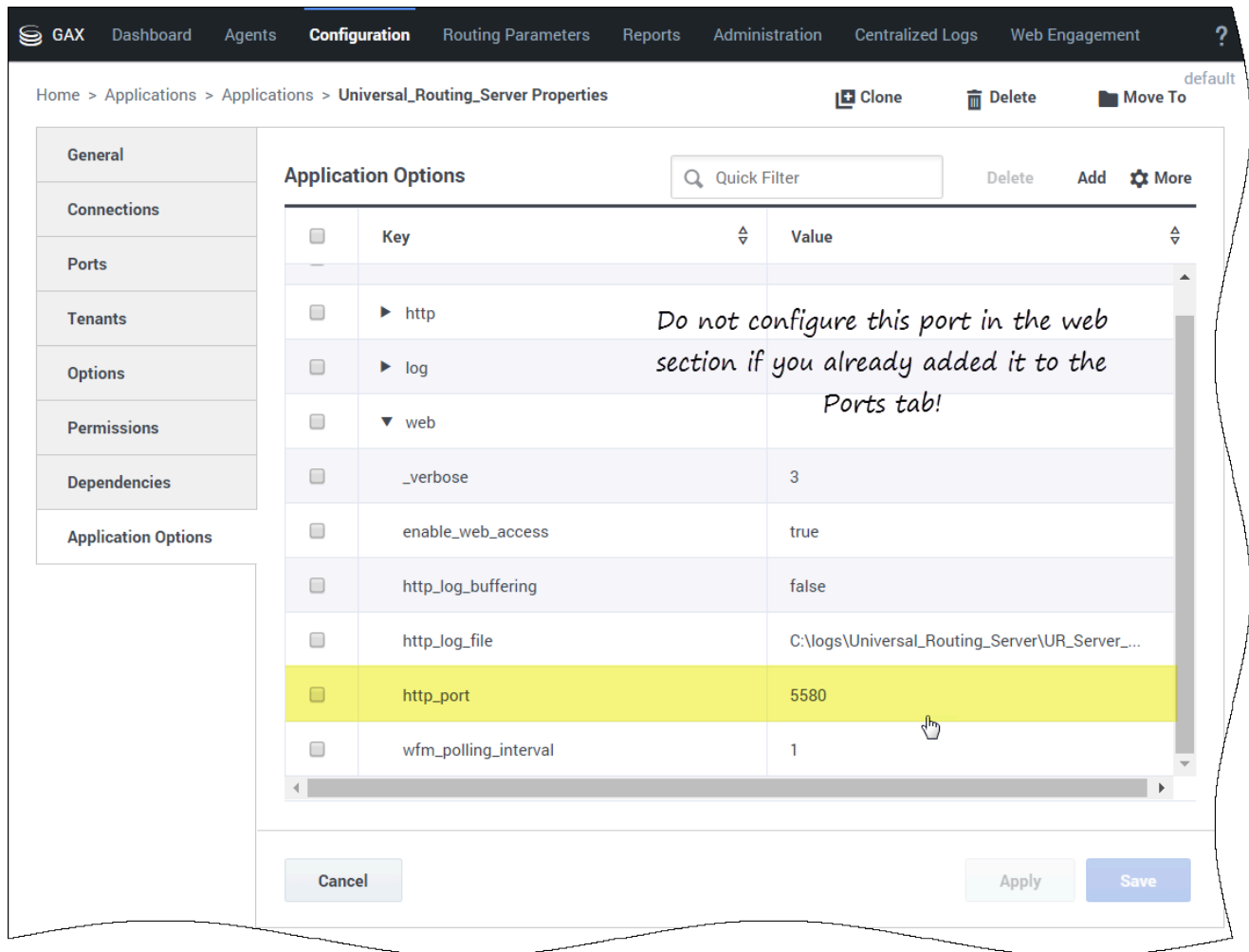
URS uses the `httpbridge` module to send target information back to GMS. To make this possible, create a web HTTP port that will be used to reply. URS will be able to perform external HTTP requests, for example, to submit `timetodial` events to GMS, and so on.

Important

The listening **http** port created in the previous section and the **web** port defined below **MUST** have different values.



In Genesys Administrator Extension, edit your URS application. Add an HTTP port with a port ID web in the **Ports** tab.



You can also do this by creating the http_port option in the **web** section of your **Application Options** tab.

- http_port = 5580 (or some other port, used internally)

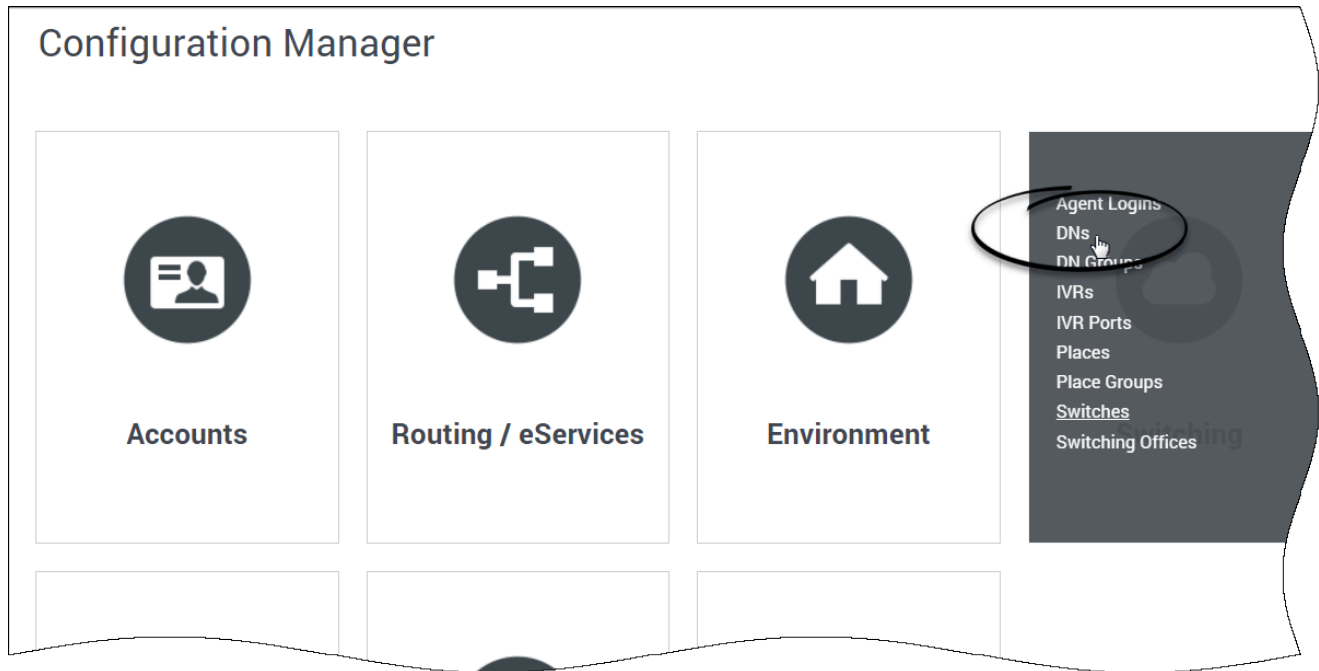
Warning

This HTTP port needs to be created in one place only.

Configure Strategies

You must deploy URS delay strategies. This step is required because when a service request is received by GMS, the request is sent to ORS for execution. ORS then sends a request to URS to create

a virtual interaction and to place it in the specified virtual queue. When an agent is available, URS sends an asynchronous response containing the selected target information to GMS, via a URL specified at the time of creation of the virtual interaction. For samples, you will create a new virtual queue in which to place the interactions, however, for a real-world scenario, the virtual queue must be selected appropriately.



Deploy URS Delay Strategies

1. Go to **Configuration Manager > Switching > DNs > Switches > SIP_Switch > DN > Virtual Queue**.
2. Create a virtual queue GMS_VQ with alias GMS_VQ_SIP_Switch.
3. Download the URS Strategies and import them into IRD. See the procedure [URS Strategy](#) to access the downloadable files and for more details.

Important

When you upgrade GMS, you need to import the Callback Template from your GMS installation directory. Start the Service Management UI, upload the <GMS Installation Directory>/service_templates/callback.zip file, and restart ORS.

GMS version	ZIP	Instructions
8.5.200.09 and higher	GMS_URS_Strategy_85200_v2.64.1.zip	<ol style="list-style-type: none"> 1. Download and unzip the zip file containing the URS strategies. 2. Open Interaction Routing Designer (IRD). 3. Import the strategy WaitForTarget.zcf, and subroutine SetRouteDelay.zcf, using <i>File > Import From File</i> on the respective tabs. 4. Open the strategy and subroutine. 5. Compile and save.
8.5.207.05 and higher	GMS_URS_Strategy_85200_v2.66.zip	
8.5.208.09 and higher	GMS_URS_Strategy_85208_v2.72.zip	
8.5.230.06 and higher	GMS_URS_Strategy_85230_v2.73.zip	

- Starting in 8.5.109.08, the URS Dial Success Rate is set to 85% when new callbacks are created to improve the callback performance.
- Starting in 2.64.1, the DialOutSuccessRate function of the WaitForTarget strategy is no longer invoked to allow the enhanced VCB algorithm within Universal Routing Server to work properly. If your application requires the legacy VCB algorithm to work, change the strategy to invoke the DialOutSuccessRate function as in earlier versions of the strategy.

Important

You do not need to load the strategy in ORS because ORS will request it when needed. See the [Interaction Routing Designer help file](#) for information about using IRD.

Enable ORS to pull interactions

1. Go to **Configuration Manager > Applications > Universal Routing Server Application**
2. Set the option Strategy=ORS.

SIP Server

1. Enable the answering machine connection, which is required for user-terminated scenarios with Call Progress Detection (CPD) capability. To do this:
 - Go to **Configuration Manager > Applications > SIP Server Application**, and set TServer/am-detected = connect.

2. Enable MSML, which is required so SIP Server can communicate with GVP as a Media Server to delegate outbound calls, play treatments, and CPD. To do this, set the following:
 - TServer/msml-support=true
 - TServer/refer-enabled=true

Media Server (GVP)

Note: See the [Genesys Voice Platform Deployment Guide](#) for additional details.

1. GMS Callback uses Media Server via SIP Server:
 - To play treatments.
 - For CPD (Call Progress Detection).
 - To make outbound calls.
2. SIP Server talks to Media Server using MSML and requires the following configuration to enable:
 - Go to Configuration Manager > SIP_Switch > DN > VOIP Service > MSML_Service.
 - Make sure that the following options are configured for MSML_Service to enable outbound:

```
make-call-rfc3725-flow=1
refer-enabled=false
ring-tone-on-make-call=false
userdata-map-filter=*
```
 - Configure the Routing Point for outbound source DN. To do this, go to Configuration Manager > Switches > SIP_Switch > DN > Routing Point.
 - Create a Routing Point object with name 8999 and alias 8999_SIP_Switch.

Setting ORS Dependencies

You must perform the following configuration steps to enable the Orchestration Server (ORS) to work with your Genesys Mobile Services installation.

Important

You do not need ORS for chat scenarios.

Setting ORS Options

Start

1. **Start Configuration Manager.**
2. In Configuration Manager, select *Environment > Applications*.
3. Locate and open the Application object for your Orchestration Server.

Important

You must use the same Application object that the one in your **GMS connections**.

4. Select the *Options* tab.
5. Open the *orchestration* section.
6. Set the value of the option *parse-start-params* to *false*.
7. Set the value of the option *mcr-pull-by-this-node* to *true*.
8. If you installed your Orchestration Server without Cassandra, disable the Cassandra persistence as detailed in **Recovery of Voice Calls Without Persistence** in ORS documentation.
Set the value of the option *sessionid-with-nodeid* to *true*.
Set the value of the option *cassandra-nodes* option to *unknown*.
9. Click *OK* to save your changes.

End

Deploying DFM Files

Included with your installation are special configuration files, called DFM, which are required for Orchestration Server-based services. These files define Genesys Mobile Services-specific SCXML constructs that are required for the execution of SCXML applications used within Orchestration Server-based Services. For the Orchestration Server-based Services to function correctly, the following DFM files need to be configured in your Orchestration Server Application object:

- Storage
- Notification
- Callback
- Genesys Mobile-Based Services

The latest DFM definition files are included with the GMS installation and are available for download through the [Service Management User Interface](#).

Important

Starting 8.5.104, you must update the DFM files deployed locally with the latest version provided in the GMS Admin UI.

These files are default configuration files which need to be edited before being deployed. See [Configuring DFMs](#). Details about deploying these DFM files in your environment are provided below.

After deploying these DFM files, you can use either an actual device with the demo application or an HTTP client (such as RestClient) to send API requests to Orchestration Server-based services. Refer to the Genesys Mobile Services API Reference for syntax of the requests.

Important

You must restart Orchestration Server after deploying DFM files for the changes to take effect.

Start

1. Download the DFM files from the [Service Management User Interface](#).
2. Copy the DFM files onto your local file system where Orchestration Server (ORS) is running.
3. [Start Configuration Manager](#).
4. In Configuration Manager, select *Environment > Applications*.
5. Locate and open the Application object for your ORS. Note: This should be the same Application object you created a connection to when [configuring your Genesys Mobile Services Application object](#).

6. Select the *Options* tab.
7. Click *Add* to create the *dfm* section.
8. In the *dfm* section, create and configure one option for each DFM, using the option value to specify the file path. Details are provided in the table below.
9. Click *OK* to save your changes.
10. Restart Orchestration Server (ORS). ORS reads the DFM configuration on startup.

End

List of DFM Options for Orchestration Server (Modified in: 8.5.201.04)

Service	Option Name	Option Value
Storage	gsgStorage	file://c:\dfms\storage.jsp
Notification	gsgNotification	file://c:\dfms\notification.jsp
Genesys Mobile-Based Services	gsgBasedServices	file://c:\dfms\services.jsp
Callback	gsgCallback	file://c:\dfms\callback.jsp
Statistics	gsgStatistics	file://c:\dfms\statistic.jsp

Additional ORS Configuration

See the [Orchestration Server Deployment Guide](#) to perform the following additional configuration steps.

1. Set up the Cluster object in the transactions list object.
2. Set up the Application object in the transactions list object.
3. Valid customer numbers should include a + sign if needed. If true, edit the valid-digits option in the **gts** section of your SIP Switch object:

```
[gts] valid-digits = +0123456789
```

Refer to [ORS](#) documentation for further details.

Configuring Basic GMS Services

This page details the basic configuration steps required before you can use your Genesys Mobile Services installation. For a more general look at the configuration options available, refer to the [Configuration Options Reference](#).

Basic Configuration Overview

Genesys Mobile Services provides a set of services or APIs that require configuration before the product can be used. The configuration of services is stored in the Configuration Server. To get the list of basic services, refer to the [Basic GMS Service Options](#) page.

Working in Configuration

Genesys Mobile Services is represented by an Application object in the Configuration Server database. This Application object is based on the "Genesys Generic Server" template and contains typical settings for a Genesys application including Server Info, Start Info, and Connections to other servers. It also includes options that correspond to configuration details for sub-services of Genesys Mobile Services. [Genesys Mobile Engagement Configuration Options](#) describes the available options and sections for this product.

Important

By design, settings in Genesys Administrator Extension for any configured service override the matching request parameters. This means that if `_provide_code` is set to `true` then this service will always respond with an access code, even if the `_provide_code` parameter received with the request is set to `false`.

Configuration settings are grouped for different service types, and stored in the Option sections described below:

- **gms Section**—Configuration settings used across different services.
- **push Section**—Notification service parameters. Not monitored at run time, so the Genesys Mobile Server instance must be restarted for changes to take effect.
- **resource Section**—Details about how resource groups are handled. Runtime configuration changes are supported, so changes take effect immediately.
- **server Section**—Cluster sub-service configuration details. Includes URL representation of this node of the cluster, consisting host, port and application name formatted in the following way: `http://web_host:web_port/app_name`. (Example: `http://yourHostName:8080/gms`). Runtime configuration changes are supported, but due to tight logical connection to the web-container configuration, a restart is needed in most cases.

- [service.<servicename> Section](#)—Additional configuration options for customized services.

Some services also rely on configuration details from a Transaction object that must be created and configured in your Genesys environment.

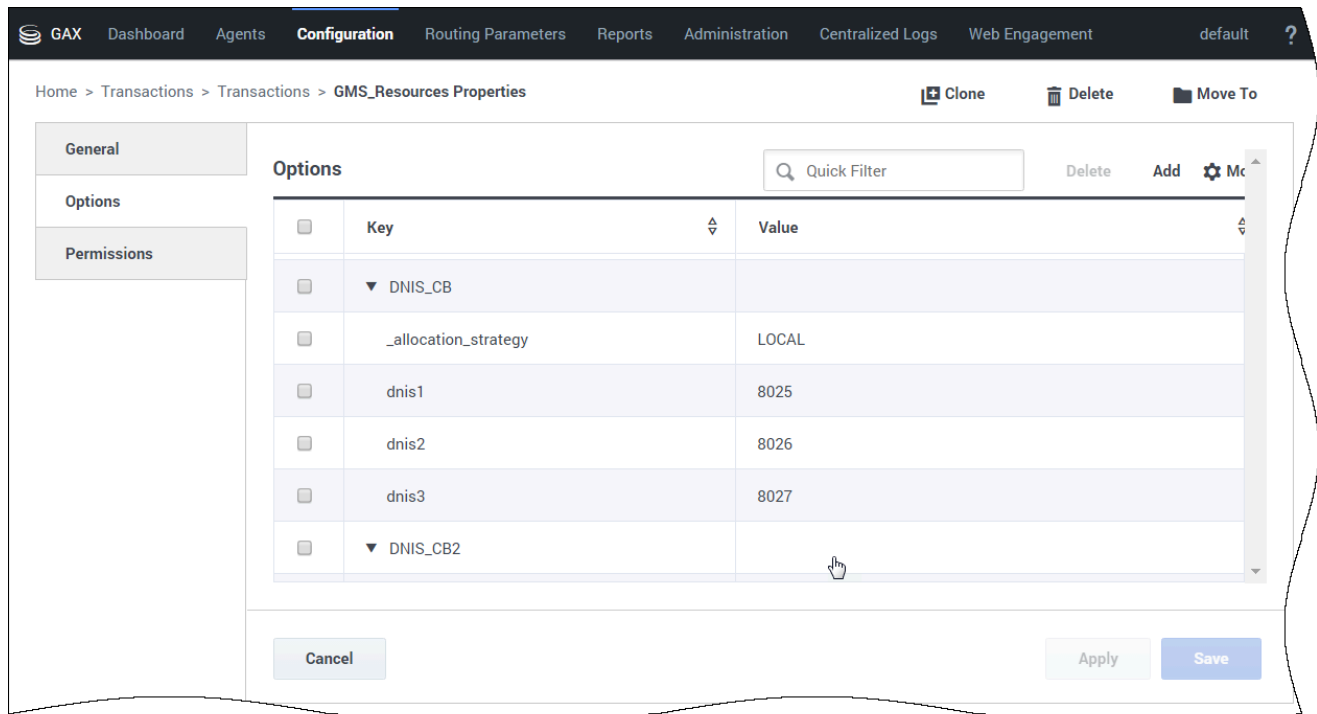
Important

- If you are setting up multiple Genesys Mobile Services nodes, the configuration options specified in the Application object must be the same for each instance.
- You can conveniently configure services by using the [Service Management Admin UI](#).
- Option services are described in the [Basic GMS Service Options](#) and in [Callback Service Options](#) reference pages.

If you are familiar with working in a Genesys environment, this type of configuration should be second nature. If you require additional information about how to work with Genesys Administrator Extension to edit these configuration options, refer to the Help file included with that product.

Creating and Configuring a Resource List

Some services included with Genesys Mobile Services require a list of resources, such as a list of access numbers that can be managed. Such lists are held in a Transactions object, which is then referenced by Options set in the Genesys Mobile Services Application object. The steps required to create and populate a resource list are provided below. You can also configure these services through the [GMS Service Management User Interface](#).



1. In Genesys Administrator Extension, find **Configuration Manager > Routing / eServices > Transactions**.
2. Click **+ New**.
3. On the **General** tab, configure the following fields:
 - Name—This name must match the resources > resource_list_name option value from your Genesys Mobile Services Application object. The default value is GMS_Resources.
 - Type—Select **List** from the drop down box.
 - Alias—Enter an alias of your choice.
4. On the **Options** tab, create a new section. The section name used here must match the value of the resource_group option, located in the service.servicename section of your Genesys Mobile Services Application object.
5. Add options to the newly created section to create your resource list. The Resource name should not start with an underscore (_).
6. Add and set an _allocation_strategy option for this group. The valid values are RANDOM, LOCAL or CLUSTER.

Creating and Configuring a Pattern List

Callback services included with Genesys Mobile Services require a list of patterns (for exceptions, and so on) to compare parameter values to a list of defined patterns. These lists are held in a

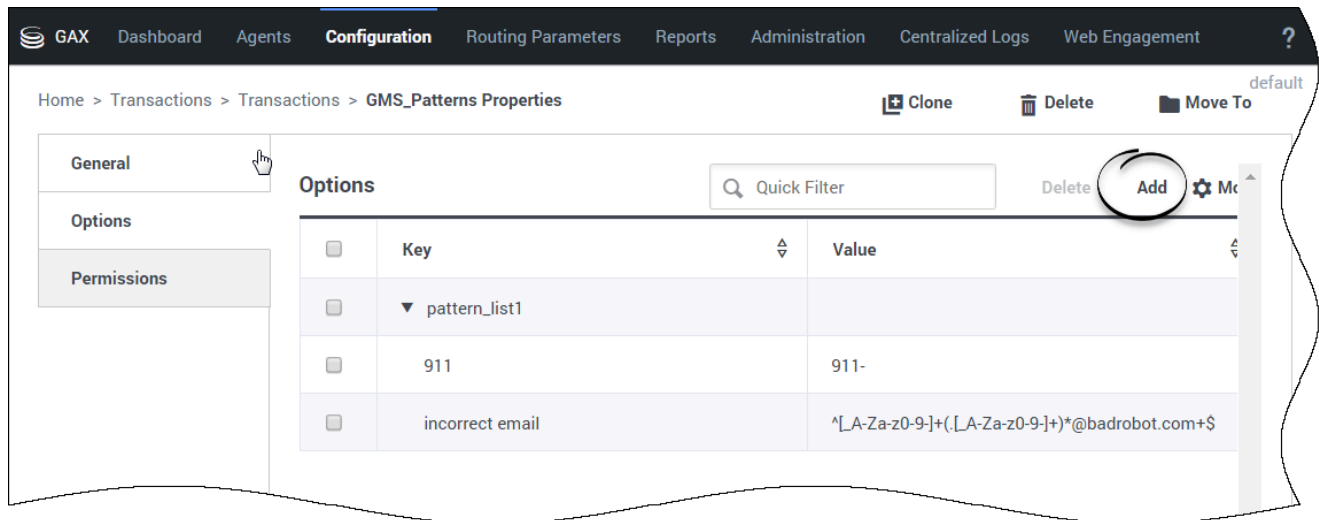
Transactions object (GMS_Patterns by default), which is referenced by the patterns_list_name option set in the Genesys Mobile Services Application object. Each section of this object defines a list of patterns that you will later use in your Callback services. You can also configure these list of patterns through the [GMS Service Management User Interface](#).

The screenshot shows the Genesys Administrator Extension Configuration Manager interface. The top navigation bar includes 'GAX', 'Dashboard', 'Agents', 'Configuration', 'Routing Parameters', 'Reports', 'Administration', and 'Centralized Logs'. The breadcrumb trail is 'Home > Transactions > Transactions > GMS_Patterns...'. Action buttons for 'Clone', 'Delete', and 'Move To' are visible. The 'General' tab is selected, showing the following fields:

- Name***: Text input field containing 'GMS_Patterns'.
- Alias***: Text input field containing 'GMS_Patterns'.
- Transaction Type***: Drop-down menu set to 'List'.
- Recording Period (min.)**: Text input field containing '0'.
- Format**: Large empty text area.
- Tenant**: Drop-down menu set to 'Environment'.
- State Enabled**: Checkmark is checked.

At the bottom, there are 'Cancel', 'Apply', and 'Save' buttons.

1. In Genesys Administrator Extension, find **Configuration Manager > Routing / eServices > Transactions**.
2. Click **+ New**.
3. On the **General** tab, configure the following fields:
 - **Name**—This name must match the patterns_list_name option value from your Genesys Mobile Services Application object. The default value is GMS_Patterns.
 - **Type**—Select List from the drop-down box.
 - **Alias**—Enter an alias of your choice.



On the **Options** tab, click **Add** to create as many patterns than you need.

- The section name defines a list of patterns and must match the value of the `_exceptions` option that will be used in your services.
- The key defines the name of a pattern.
- The value is a regular expression used to filter strings and numbers. It must match the [Java Regular Expression Lesson by Oracle](#).

Configuring GMS Built-in Services

To complete your deployment, the following GMS-based services need to be configured:

1. request-interaction
2. match-interaction
3. request-access

Required options are outlined below, with some sample values to help you get started. You can configure these services through the [GMS Service Management User Interface](#).

request-interaction

[Documentation:GMS:Help:SimpleVoiceInImmediate:8.5.3](#)

match-interaction

[Documentation:GMS:Help:Match-interaction:8.5.3](#)

request-access

[Documentation:GMS:Help:Request-access:8.5.3](#)

Setting the timezone for your GMS Host

Make sure that all of the servers hosting your GMS nodes and connected Genesys servers are using the same timezone.

Single Tenant Support and Chat Service

For chat services ([Chat API Version1](#)), the default chat endpoint value in the `default_chat_endpoint` option can be used for all services. Or by service, you can customize the `_chat_endpoint` option. These options, `default_chat_endpoint` and `_chat_endpoint`, are composed of `<tenant name>:<chat endpoint name>`.

Configuring Chat Version1

This page details the specific configuration steps required to use **Chat API Version 1** included with Genesys Mobile Services. This API includes two types of chat services:

- The basic [request-chat service](#).
- The ORS-Chat services described in the [Chat Immediate](#) and [Chat Delayed](#) scenarios.

For further details about the usage of this API, refer to the [Chat API Version1](#) page.

Prerequisites for Chat V1 Dependencies

Prerequisites: Before you can start, you must complete the previous installation steps which include:

- **Create** a GMS application object, which includes connections to the Solution Control Server and to the Chat Server.
- **Install GMS** in your system.
- Set the mandatory **GMS dependencies**.

To use the Chat API Version 1, you must specify configuration options in the Application objects for the following objects:

- Genesys Mobile Services
- Chat Server

Important

The Chat Server must be installed and configured. Refer to the [eServices Deployment Guide](#), which include details on how to install a Chat Server when deploying eServices.

The following sections provide details about configuration changes required to use chat with your Genesys Mobile Services deployment. Procedures and illustrations on this page use Genesys Administrator, although the configuration can also take place using Configuration Manager.

Setting Chat Options in GMS Configuration

The following configuration options must be specified in your Genesys Mobile Services Application object:

1. In Genesys Administrator Extension, find **Configuration Manager > Environment > Applications**.
2. Locate and edit the Genesys Mobile Services Application object that you previously created and configured.
3. Under the *Application Options* tab, in the *chat* section, include the following mandatory **configuration options**:

chat_session_request_timeout

Section: chat

Default Value: 30000

Valid Values: Any positive integer

Changes Take Effect: Immediately upon notification.

Duration in milliseconds after which the chat interaction gets deleted.

default_chat_endpoint

Section: chat

Default Value: Environment:default

Valid Values: <tenant_name>:<chat_endpoint>

Changes Take Effect: Immediately upon notification.

This option is used for all chat services in order to define the queue (URS) where the chat session initiated by GMS will enter. The value of this option is the tenant name on which the service(s) will proceed, and the chat endpoint as defined in the ChatServer option. For example, the section endpoints for the tenant Environment in the chat options is written as endpoints:1. This section contains the endpoint options (for example, default=queue). The chat endpoint value to use this default endpoint in the Environment tenant is Environment:default.

Tip

You can supersede this option for each chat service using the `_chat_endpoint` option with the same `<tenant_name:chat_endpoint>` value.

_client_timeout

Section: chat

Default Value: 900

Valid Values: Any positive integer
Changes Take Effect: After restart

Client timeout in seconds for Cometd chat sessions.

If the client does not interact with the Chat service (refresh, send message, send event), GMS stops to poll the Chat server, and the Chat session is closed. This option applies only to chat sessions implemented using Cometd connections. For non-Cometd implementation, Chat server timeout parameters apply. The default value for this option is 15 minutes.

Single Tenant Support and Chat Service

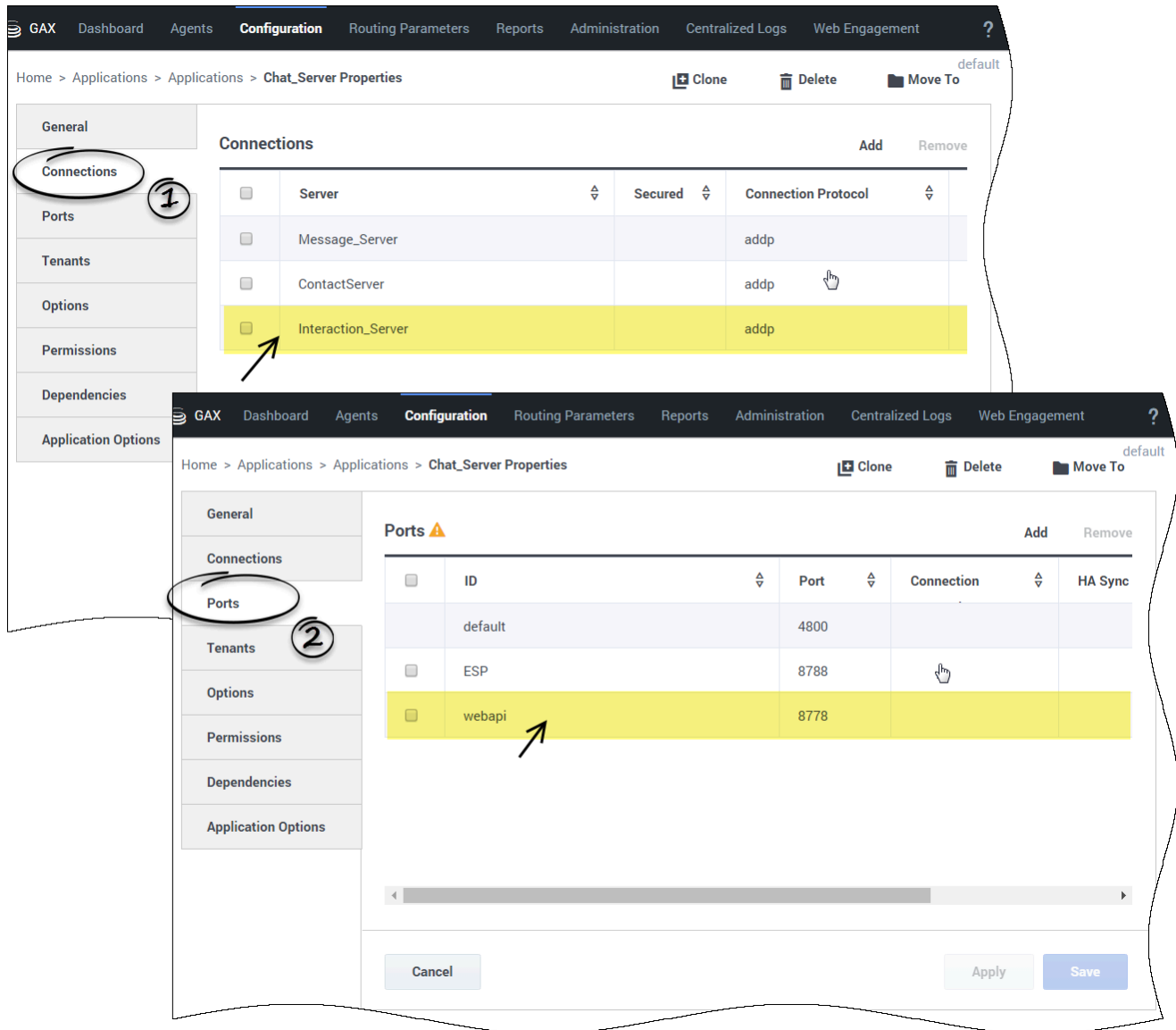
For chat services ([Chat API Version1](#)), the default chat endpoint value in the `default_chat_endpoint` option can be used for all services. Or by service, you can customize the `_chat_endpoint` option. These options, `default_chat_endpoint` and `_chat_endpoint`, are composed of `<tenant name>:<chat endpoint name>`.

Chat Server Configuration in Primary/Backup mode

The Chat Server Application object being used by your Genesys Mobile Services deployment should have the following configuration updates:

- Add a connection to Interaction Server.
- Set a backup server and specify the redundancy type.

Configuring your Chat Server's Connections



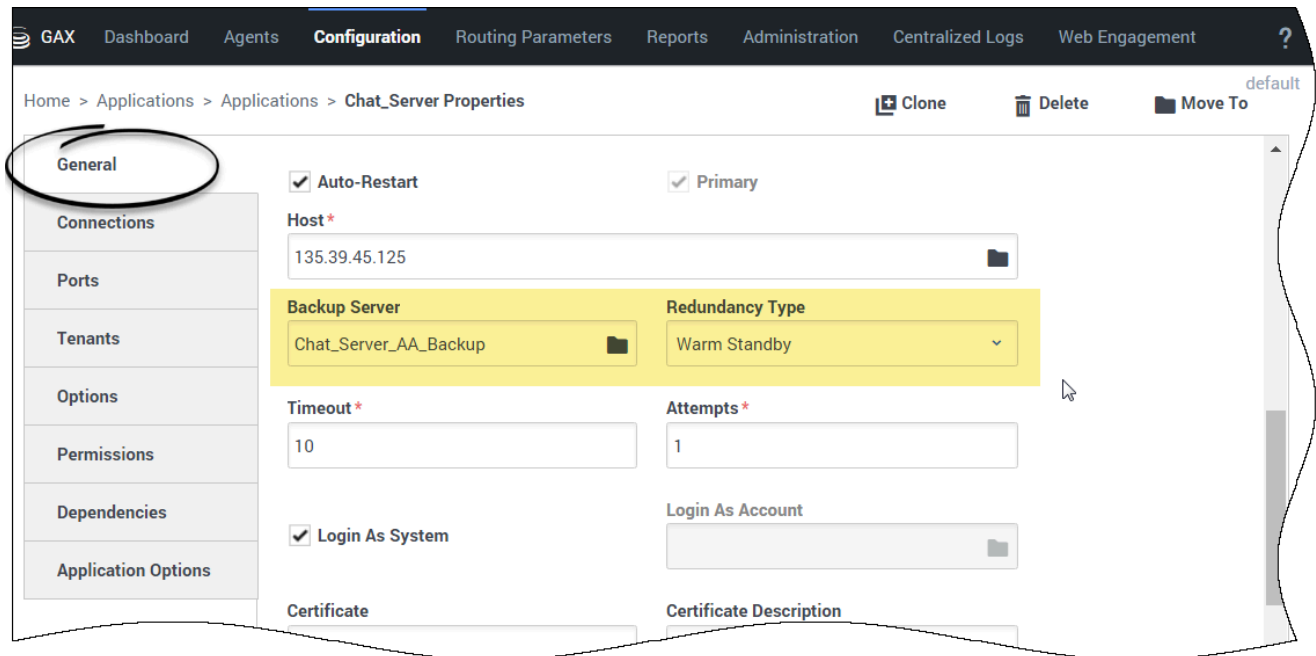
1. In Genesys Administrator Extension, find **Configuration Manager > Environment > Applications**.
2. Locate and edit the Chat Server Application object associated with your Genesys Mobile Services deployment.
3. View the **Connections** tab and click **Add**.
4. Locate and select the Interaction Server Application object that you want to use.
5. Make sure to use either port with id webapi or port with chat-flex protocol of the Chat Server.

Repeat this procedure for each Chat Server associated with your Genesys Mobile Services application.

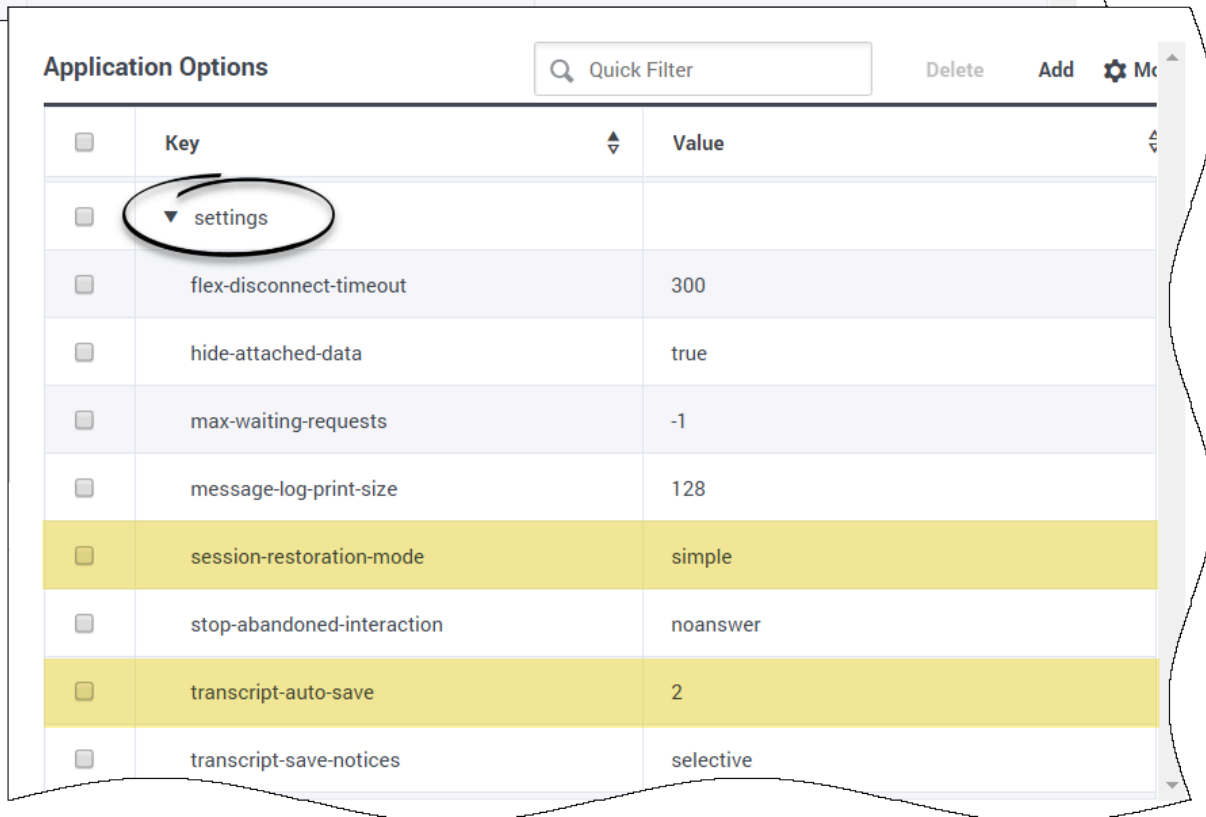
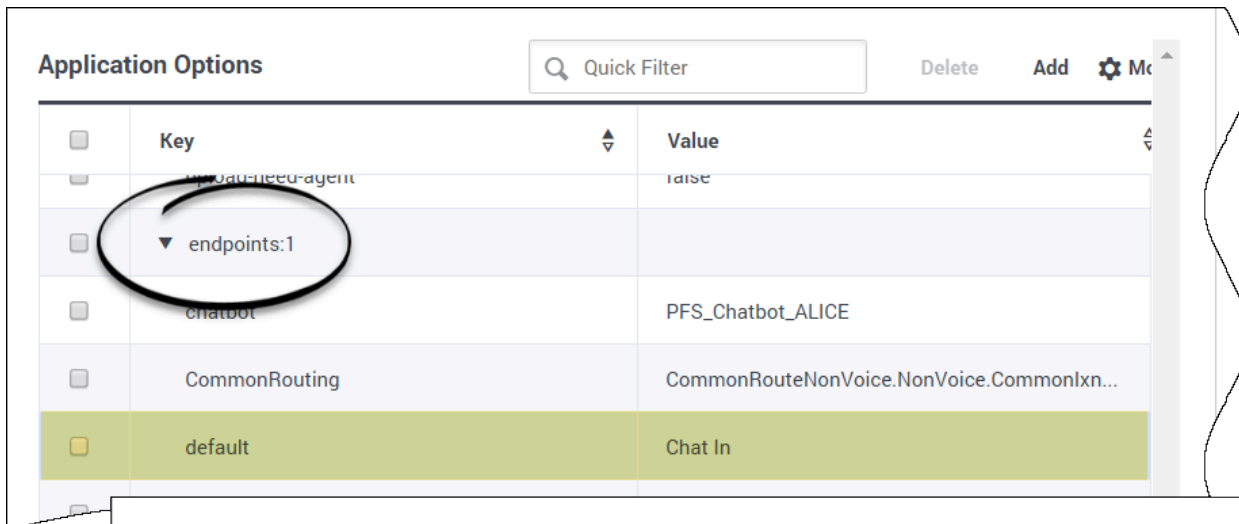
Setting Chat Server HA-Specific Options

Important

Enabling High Availability requires Chat Server **8.1.000.20 or higher**.



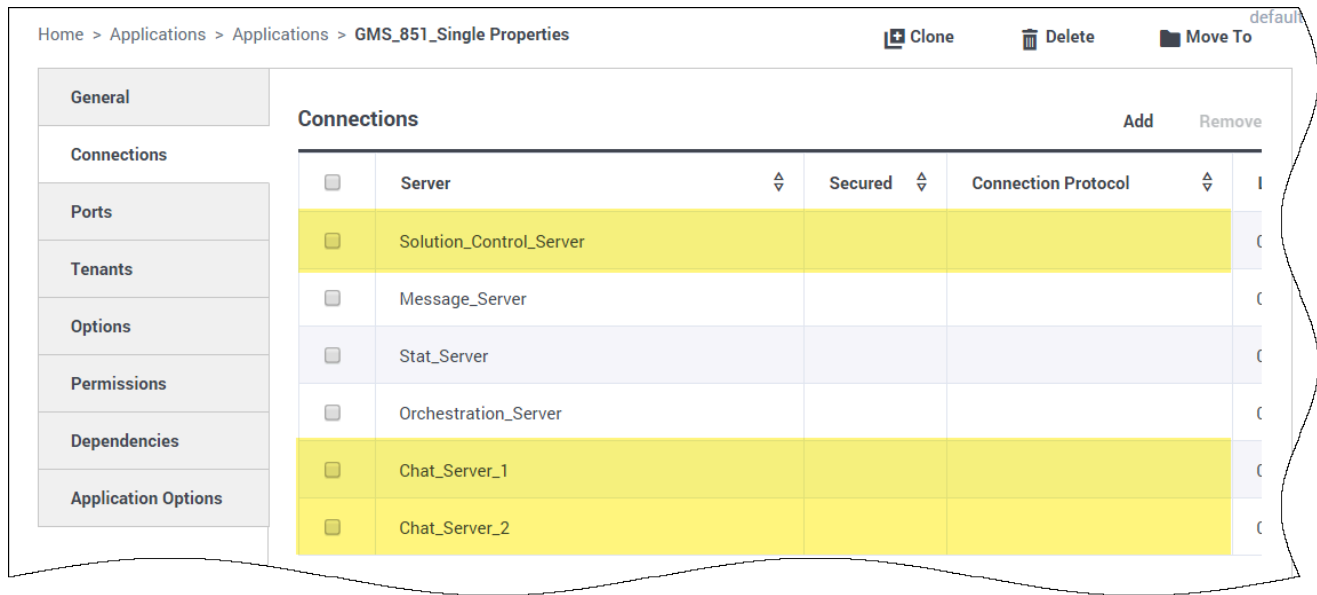
1. View the *Backup Server Info* section on the *General* tab.
2. Specify a *Backup Server* value.
3. Set the *Redundancy Type* to *Warm Standby*.



Under the **Application Options** tab, include the mandatory configuration options required for High Availability

- Section: **endpoints:1**
default = Chat In
- Section: **settings**
session-restoration-mode = simple
transcript-auto-save = 2

Chat Server Configuration in N+1 Mode



Important

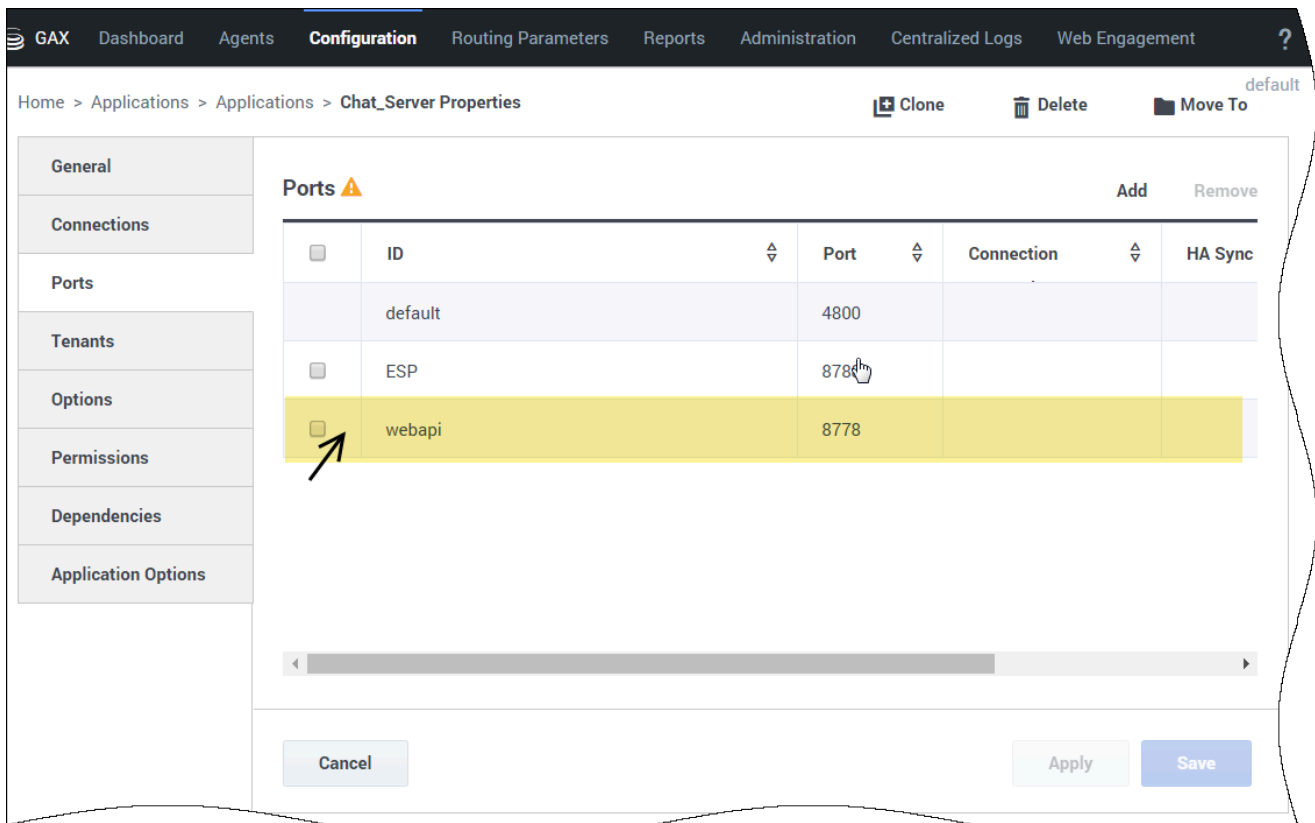
This configuration is recommended to support Load Balancing with your GMS application.

Prerequisites: Your Chat Servers must not be configured in Primary/Backup mode—That is, their configuration must not include a backup server.

Edit your GMS configuration and add all Chat Servers and a Solution Control Server to the list of connections.

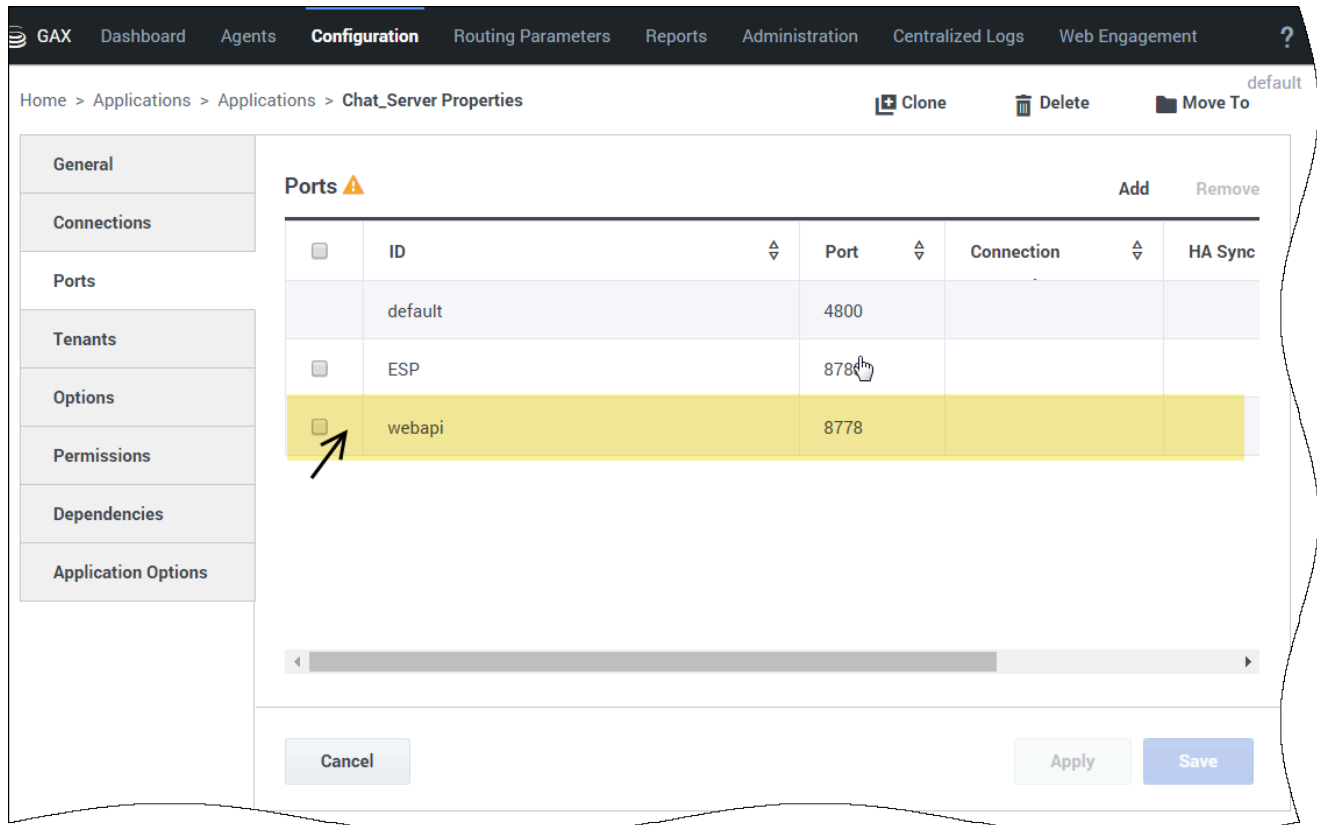
Note: Make sure to use either port with id `webapi` or port with `chat-flex` protocol of Chat Server in the **Connections** tab of GMS.

Add an HTTP listening port to your Chat Server Configuration



Make sure to have an HTTP listening port (called `webapi` in our sample) in the configuration of each of your Chat Server.

Add an HTTP listening port to your Chat Server Configuration



Make sure to have an HTTP listening port (called `webapi` in our sample) in the configuration of each of your Chat Server.

Tune Chat Session Refresh

GMS nodes are responsible for refreshing chat sessions (that cover the refresh of the chat customer transcript using comet notification). However, a given node may sometimes be unable to refresh the previous chat sessions even if it can accept new ones. In that case, another GMS node will retrieve and refresh the suspended chat sessions. This is possible only if the Chat Server keeps these sessions alive for long enough.

To enable this mechanism, configure the `flex-disconnect-timeout` option in the **settings** section of the Chat Server application. Increase the value to **300** seconds so that the Chat Server keeps the chat sessions alive long enough for GMS to refresh. Then, GMS can again refresh uncompleted chat sessions after 60 seconds.

flex-disconnect-timeout

Section: settings

Default Value: 45

Valid Values: Any integer from 1-1728000

Changes Take Effect: Immediately

Modified: 8.5.301.06

Specifies timeout (in seconds) after which Chat Server disconnects an inactive web (flex) chat client. Inactivity is defined as the absence of protocol requests, not the absence of messages.

Next Steps

With basic configurations now complete, you can start loading and managing your services, using the GMS Service Management User Interface.

- [Service Management User Interface](#)

You can also configure additional, advanced settings that are outlined in the following section:

- [Configuration](#)

Configuring Digital Channels

Digital Channel APIs provide the following capabilities:

- Email
- Open Media
- Chat
 - Chat API Version2
 - Chat API Version2 with CometD

You can use **Genesys Administrator** to configure several types of configuration options for your Genesys Mobile Engagement Digital Channels API application. This topic shows how to do that, and also discusses several third-party options.

Prerequisites for Digital Channels

Before you can start, **create** a GMS application object, which includes **connections** to the following servers:

Solution Control Server

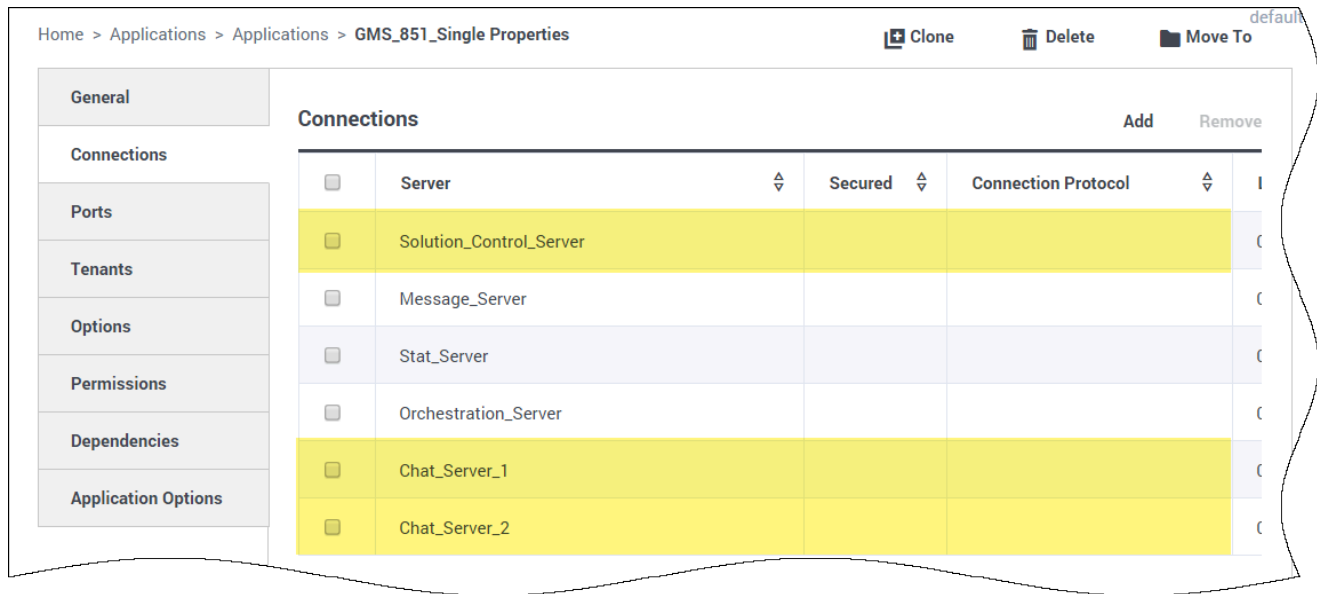
- This connection is mandatory

The Mobile Engagement node monitors the status of each connected media server and will load balance requests between currently running media servers. If Mobile Engagement does not have a connection to the Solution Control Server, or if the node cannot connect to it, the Digital Channels API functionality will be disabled or may not work properly.

Chat Server

- You need this connection only if you plan to use Chat V2 features.
- For the chat file transfer feature, use Chat Server version 8.5.104 or later.
- Genesys recommends that you use an N+1 deployment (with standalone Chat Servers), as opposed to using warm standby pairs.

Your Chat Servers must not be configured in Primary/Backup mode—That is, their configuration must not include a backup server.



Edit your GMS configuration and add all Chat Servers and a Solution Control Server to the list of connections in the **Connections** tab of GMS.

Note: Make sure to use either port with id webapi or port with chat-flex protocol when adding the Chat Server. Make sure to create these ports in the Chat server before you add connections. See [Deploying the Chat Solution](#) for additional Chat Server instructions.

Universal Contact Server

- You need this connection only if you plan to use chat file transfer feature.
- For the chat file transfer feature, use Chat Server version 8.5.104 or later.

Email Server

- You need this connection only if you plan to use email features.
- To implement an email service, add a connection to every Email Server Java instance that you plan to use.

Interaction Server





- You need this connection only if you plan to use open media features.

Load Balancing

The Digital Channels API is fully capable of working behind a load balancer. To configure load

balancing, refer to the [related instructions](#) for creating GMS nodes. As long as at least one node within the Mobile Engagement cluster is up and running, all of the other nodes can fail—and the cluster will continue to provide all of the functionalities of the Digital Channels API.

Use the table below to determine if you need to configure session stickiness on your Load Balancer.

Digital Channel API	Sticky session required	No sticky session
Chat		
Chat with CometD		
Email		
Open Media		

Third-party Configuration and Considerations

Genesys Mobile Engagement is distributed with the Jetty application container. Note that various versions of Genesys Mobile Engagement can run on various versions of Jetty. As such, some of the application container options may affect or interfere with your Genesys Mobile Engagement application options. In addition to this, Genesys Mobile Engagement is built on top of web frameworks that may also have conflicting options. Because of this, you may need to adjust your Jetty configuration. Genesys recommends that you review the following parameters, editing them, as needed:

Jetty default application port

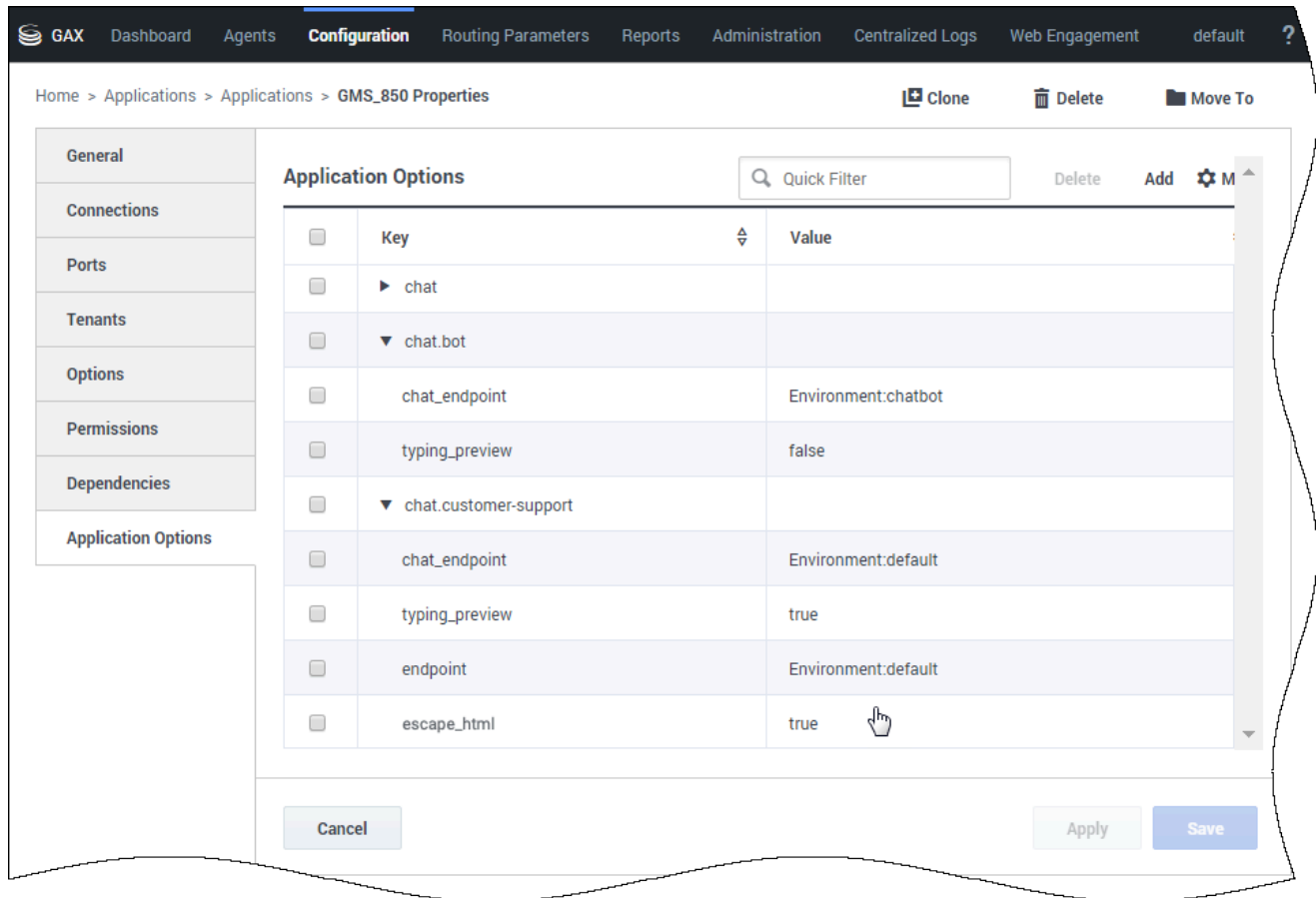
jetty.port defines the port used in the endpoint URL. You might need to adjust this port if other applications running on the same host already use this port. This setting appears in **etc/jetty-http.xml** file.

```
<Set name="port"><Property name="jetty.port" default="18080"/></Set>
```

Adding a Digital Channel

To configure a digital channel, you need to create at least one service for this digital channel. If you have more than one service for this channel, you can configure all these services in the related media section, as described below.

Service-Related Section



You can configure Genesys Mobile Engagement to support multiple services for the same media type. This allows you to provide multiple API endpoints for each channel, each of which can be uniquely configured.

For each service that you want to enable, create a new configuration section in your GMS Configuration. The name of this new section consists of the name of the media type and the service name, separated by a period: `<media>.<service-name>` where media can be chat, email, or openmedia. For example, create **chat.customer-support**.

For example, to configure customer support services for chat and for email, create sections called **chat.customer-support** and **email.sales-inquiry**. Or to create an sms-details service for open media, call the new section **openmedia.sms-details**.

Each of these service-related sections enables a Genesys Mobile Engagement Digital Channels API endpoint with a name that corresponds to the service name, as shown in these examples:

- `http://myhost.domain/genesys/2/chat/customer-support`
- `http://myhost.domain/genesys/2/email/sales-inquiry`

- <http://myhost.domain/genesys/2/openmedia/sms-details>

Then, configure your service by creating options. The options that you can use are listed in the in the GMS Configuration Options Reference Guide:

- [Chat service options](#)
- [Email service options](#)
- [Open Media service options](#)

If you create more than one service for a given media, you can decide to configure all of them in the media-related section, as detailed below. In that case, since you cannot save an empty configuration section, you must configure at least one option in each service-related section. If you do not need to configure any service-related options, you can create a dummy option, for example, **###dummy###**. This will allow you to save that section.

Media-Specific Section

Each of the digital channels supported by Genesys Mobile Engagement has its own medium-specific configuration section, which provides application-wide options for all of the configurable services offered for that medium.

This section, which is optional, is named after its media type. Since Mobile Engagement supports chat, email, and open media channels, the relevant sections would be called **chat**, **email**, and **openmedia** respectively. All the options are documented in the GMS Configuration Options Reference Guide:

- [Chat service options](#)
- [Email service options](#)
- [Open Media service options](#)

Configuration Option Precedence

Genesys Mobile Engagement uses the following rules to determine which configuration options to use when it initializes a service:

1. Look for the option in the **service**-related section: if present, use this value.
2. If the option is **not** present in the **service**-related section, look for it in the **media**-related section: if present, use this value.
3. If the option is **not** present in the **media**-related section, use the **default** value.

The following table should help you to check which value applies:

Use-Case Description	Service section	Media section	Applied Value
The option is defined at the service level	yes	yes/no	Service
The option is defined at the media level, but not defined at the service	no	yes	Media

Use-Case Description	Service section	Media section	Applied Value
level.			
The option is not defined for either both the service or media level.	no	no	Default

Refer to the configuration options page for each media type to get the list of all options with their default values.

Important

When establishing precedence, Genesys Mobile Engagement considers an instance of a configuration option with an empty value to be present and stops when it encounters it. Because of this, you can override the default value for an option by creating an instance of the option in the service-related or media-related sections and leaving its value empty.

Configuration

The following table lists additional configurations that can be used with your GMS installation.

Page	Summary
Configuring an External Cassandra	Provides configuration steps in order to use external Cassandra instances in your GMS deployment.
Load Balancer Configuration Examples	Recommendations for load balancing.
Configuring a GMS Service for ORS Load Balancing	Describes how to set up a GMS Service for ORS Load Balancing.
Configuring and Starting a GMS Cluster	Describes the process for initializing a GMS cluster.
ORS Cookie Support	Provides information about the cookie-based management that GMS supports.
Mobile Push Notifications	Details configuration for push notification service.
Custom Reporting	Basic configuration for Real-Time and Historical Reporting based on T-Server's UserEvent mechanism.
Implementing ADDP	Discusses the Advanced Disconnect Detection Protocol (ADDP) protocol.
Implementing IPv6	Provides information about Internet Protocol version 6 (IPv6).
Starting and Stopping GMS	Instructions to start and stop GMS, and configure a delay for a graceful shutdown.
GMS Alarms	Describes alarms that can be raised by GMS based on system status/configuration.
Configuring HTTP Caching	Describes how to configure HTTP Caching.

Configure an External Cassandra

Modified in: 8.5.207.05, 8.5.206.04, 8.5.203.02, 8.5.230.06

Version	Description
8.5.303+	<ul style="list-style-type: none"> Support for Cassandra 4.1. Deprecated option: The create-tables option in the cassandra section will be removed. As a result, the GMS schemas and tables cannot be modified by the GMS application. You need to use the GMS CQL scripts to create or update the GMS schemas/tables on the Cassandra server during deployment. If any tables are missing, GMS will not create them and will fail, causing the GMS instance to stop.
8.5.230+	<ul style="list-style-type: none"> Support for Cassandra 4.0. See Supported Operating Environment Reference for GMS.
8.5.207+	<ul style="list-style-type: none"> New deployment for GMS schemas in the Cassandra cluster. Create and deploy GMS schemas in the Cassandra cluster before starting GMS nodes. This change improves the management of the Cassandra authorization for a specific user; as a result, the user no longer needs to own super-user authorizations to create keyspaces. If, later, new tables might be created during an update or a migration, configure create-tables = true to allow GMS to create the missing tables in GMS keyspaces. Genesys simplifies GMS deployments with Cassandra by deprecating old options and introducing new ones. See the cassandra Section section of the <i>Genesys Mobile Engagement Configuration Options</i> guide. The minimum supported version of Cassandra is now 2.1. Refer to the Prerequisites section.
8.5.206+	<ul style="list-style-type: none"> GMS no longer supports installing embedded Cassandra (in lab or production). Change of Cassandra driver. The new driver used by GMS to connect to Cassandra supports setting multiple Cassandra instances. Other

	<p>Cassandra instances that belong to same Data Center are automatically detected. If one of them dies, the driver connects to other detected Cassandra instances, including those that are not defined in the GMS options.</p>
<p>8.5.203+</p>	<p>GMS only supports deployments with an external Cassandra. An external Cassandra might be used in the following scenarios:</p> <ul style="list-style-type: none"> • You already have a Cassandra/DataStax deployment. • You are securing your data in segregated networks, cages, and racks. • You want multiple redundancy features, such as distinct data centers, rack, and chassis awareness. • You are installing GMS on a production server or on a Windows host. <ul style="list-style-type: none"> • Starting with 8.5.206, GMS no longer supports installing embedded Cassandra (in lab or production). • Genesys does not recommend installing Cassandra on Windows.

Configuring GMS for an external Cassandra is a multi-step process to enable connection, TLS connection, authentication, and authorization. The steps include setting configuration options in Configuration Manager (or Genesys Administrator), changing configuration settings in the `cassandra.yaml` file, and executing Cassandra Query Language (CQL) commands.

Important

- All external Cassandra nodes must be of the same version.
- Cassandra is not required if you deploy Genesys Mobile Environment for Chat API V2, Email API V2, and Open Media API V2.
- When you are deploying Cassandra and GMS instances, ensure perfect time synchronization between hosts. NTP or another daemon should be installed on each host to ensure only a few milliseconds difference between hosts at maximum. Over time, data in a Cassandra replica can become inconsistent with other replicas due to the distributed nature of the database. In this scenario, you need to repair them as described in the [Official Cassandra documentation](#).
- GMS does not support multi-datacenter Cassandra simultaneous connections. A GMS Cluster must connect only to a single and local datacenter Cassandra Cluster.

Install External Cassandra Nodes

If you have not already installed Cassandra, then follow these guidelines.

Prerequisites

- GMS supports Cassandra 2.1 and higher (≤ 4.1):
 - The latest tested version is 4.1.
 - For Cassandra 2.1, use CQL 3.1.
 - For Cassandra 3.0 and 3.1, use CQL 3.3.
 - For Cassandra 4.0, use CQL 6.0.0.
 - For Cassandra 4.1, use CQL 6.1.0.
- For the Network Topology, use one or several clusters.

Install Cassandra on All Nodes

Download Apache Cassandra. Use the right Unix user to install the application on all of your nodes (or host):

```
$ wget https://dlcdn.apache.org/cassandra/4.1.7/apache-cassandra-4.1.7-bin.tar.gz
```

Extract

```
$ tar xf apache-cassandra-4.1.7-bin.tar.gz
```

Configure Cassandra for Network Topology

1. Edit the `conf/cassandra.yaml` file for all of the nodes that you installed:

- Modify the `cluster_name` value:

```
cluster_name: '<the same name for all the cluster/datacenter>'
```

- Modify the `seeds` value to specify a comma-separated list of host IP addresses.

```
- seeds:
```

```
"<cassandra_seed_host_ip_address_cluster_datacenter1>,<cassandra_seed_host_ip_address_cluster_datacenter2>"
```

For example DC1 and DC2 (see below):

```
- seeds: "192.168.1.172,10.100.198.143"
```

- Modify the `listen_address` value:

```
listen_address: <externally reachable cassandra host ip address>
```

- Modify the `native_transport_port` value to specify the port matching the value of the GMS Configuration option `native-port` :

```
native_transport_port: 9042 # default port
```

```
native_transport_port_ssl: 9142 # default TLS port if you need
native_transport_port for non-TLS connections
```

Note: If you enable TLS, use **native_transport_port** for TLS connections. If you need to keep **native_transport_port** for non-TLS connections, use **native_transport_port_ssl** to enable client encryption on the secured TLS port. See [Client-to-node encryption](#) in the *Official Apache Cassandra* documentation.

- Modify the `rpc_address` value:

```
rpc_address: <externally reachable cassandra host ip address>
```

- Modify the `endpoint_snitch` value:

```
endpoint_snitch: PropertyFileSnitch
```

Important

Depending on your topology, you may use other values such as `GossipingPropertyFileSnitch` for the snitch endpoint.

2. Edit the `conf/cassandra-topology.properties` file for all of the nodes:

```
# Cassandra Node IP=Data Center:Rack
# DC1
192.168.1.172=DC1:RAC1 # seed node DC1
192.168.1.215=DC1:RAC1
192.168.1.234=DC1:RAC1

# DC2
10.100.198.143=DC2:RAC1      # seed node DC2
10.100.198.94=DC2:RAC1
10.100.198.136=DC2:RAC1
# default for unknown nodes
default=DC1:RAC1
```

Start all of the cassandra nodes. Now, your cluster of cassandra nodes is functional.

Synchronize Cassandra Hosts

To avoid synchronization issues, synchronize the clock of all GMS nodes. On Windows, use following the command:

```
net time \\<ComputerName> /set /yes
```

Where `\\<ComputerName>` specifies the name of a server you want to check or with which you want to synchronize.

Deploy GMS schemas in Cassandra

Starting in 8.5.207, it is mandatory to create and deploy GMS schemas in the Cassandra cluster before starting GMS nodes. This change improves the management of the Cassandra authorization for a specific user; as a result, the Cassandra user no longer needs to own super-user authorizations to create keyspaces.

Create Schemas for your First Deployment

In a new fresh deployment, you need to create GMS keyspaces and tables in **one** of your nodes (not in all of them). Before you run the CQL scripts that perform these operations, edit the following files:

- <GMS Home>/scripts/cassandra_gsg_schema.cql
- <GMS Home>/scripts/cassandra_gsg_dd_schema.cql

Replace the text placeholders [ToBeChanged:<keyspace_name>] with your keyspace names and set the replication factor to NetworkTopologyStrategy.

For example, if you edit the `cassandra_gsg_schema.cql` file, here are the first lines that you will see:

```
CREATE KEYSPACE [ToBeChanged:<keyspace_name>] WITH replication =
{'class': 'SimpleStrategy', 'replication_factor': '2'}
AND durable_writes = true;

CREATE TABLE [ToBeChanged:<keyspace_name>].gsg_principal_roles (
    key blob,
    column1 blob,
    value blob,
    PRIMARY KEY (key, column1)
)
```

If your GMS keyspace name is `gsg`, you should edit these lines as follows:

```
CREATE KEYSPACE gsg WITH replication =
{ 'class' : 'NetworkTopologyStrategy', 'DC1' : 2, 'DC2' : 2 }
AND durable_writes = true;

CREATE TABLE gsg.gsg_principal_roles (
    key blob,
    column1 blob,
    value blob,
    PRIMARY KEY (key, column1)
)
```

After you have edited the CQL scripts, launch them to create the GMS schemas in the Cassandra node:

```
$ <CASSANDRA_HOME>/bin/cqlsh <cassandra_host> <cassandra_port> -f <GMS_HOME>/scripts/
cassandra_gsg_schema.cql
$ <CASSANDRA_HOME>/bin/cqlsh <cassandra_host> <cassandra_port> -f <GMS_HOME>/scripts/
cassandra_gsg_dd_schema.cql
```

Important

Create these schemas in one node only.

Update Schema after Upgrading

Starting in 8.5.102, Cassandra schemas are compatible with GMS 8.5.105+ and do **not** require any upgrade. But if you upgrade from GMS versions older than 8.5.102, you will need to manually update the Cassandra schemas in **one** of your nodes (not all).

Before running the CQL scripts that perform these operations, edit the following files:

- <GMS Home>/scripts/update_cassandra_gsg_schema.cql
- <GMS Home>/scripts/update_cassandra_gsg_dd_schema.cql

Replace the text placeholders [ToBeChanged:<keyspace_name>] with your keyspace names and set the replication factor to NetworkTopologyStrategy.

For example, if you edit the update_cassandra_gsg_schema.cql file, here are the first lines that you will see:

```
CREATE KEYSPACE IF NOT EXISTS [ToBeChanged:<keyspace_name>] WITH replication = {'class':
'SimpleStrategy', 'replication_factor': '2'}
AND durable_writes = true;

CREATE TABLE IF NOT EXISTS [ToBeChanged:<keyspace_name>].gsg_principal_roles (
    key blob,
    column1 blob,
    value blob,
    PRIMARY KEY (key, column1)
)
```

If your GMS keyspace name is gsg, you should edit these lines as follows:

```
CREATE KEYSPACE IF NOT EXISTS gsg WITH replication = {'class' : 'NetworkTopologyStrategy',
'DC1' : 2, 'DC2' : 2}
AND durable_writes = true;

CREATE TABLE IF NOT EXISTS gsg.gsg_principal_roles (
    key blob,
    column1 blob,
    value blob,
    PRIMARY KEY (key, column1)
)
```

After you edited the CQL scripts, launch them to update the schemas:

```
$ <CASSANDRA_HOME>/bin/cqlsh <cassandra_host> <cassandra_port> -f <GMS_HOME>/scripts/
update_cassandra_gsg_schema.default.cql
$ <CASSANDRA_HOME>/bin/cqlsh <cassandra_host> <cassandra_port> -f <GMS_HOME>/scripts/
update_cassandra_gsg_dd_schema.default.cql
```

Important

Update the schemas in one node only.

Configuration Options

The `cassandra` and `cassandra-authentication-security` sections list the configuration options applicable to an external Cassandra deployment. Changes take effect after restart.

Connection to an External Cassandra

The following steps are required to enable GMS to connect to an external Cassandra.

1. In Configuration Manager, locate and open your GMS Application object.
2. On the Options tab, **[cassandra]** section, configure the following options:
 - `nodes` = <your Cassandra hosts or IP addresses for the local datacenter (comma-separated list)>
 - `native-port` = <your Cassandra port: default is 9042>
 - (Deprecated in 8.5.207) `strategy-class` = `NetworkTopologyStrategy`
 - (Deprecated in 8.5.207) `strategy-option` = `DC1:2;DC2:2`
 - `keyspace-prefix` = <if you need to select a prefix other than the default one (gsg) for your schemas>
3. Restart GMS.

Important

(Deprecated in 8.5.207) The `strategy-class` and `strategy-option` options must match the replication factor that you set when creating or updating your Cassandra schemas.

Configuring TLS

Before starting your Cassandra nodes, you must configure the TLS options.

Cassandra Side

Create Cassandra node certificate

You can either create one certificate per Cassandra node or create only one certificate for all of your Cassandra nodes.

Use **keytool**, which is part of the JDK toolset, to create your certificate:

Example

```
$ keytool -genkey -keyalg RSA -alias cassandranode -keystore keystore.node -storepass
cassandra -keypass cassandra -validity 36500 -dname "CN=192.168.2.1, OU=None, O=None, L=None,
C=None"
# will create keystore.node file
$ keytool -export -alias cassandranode -file cassandranode.cer -keystore
keystore.node # password to use: cassandra
# will create cassandranode.cer file
$ keytool -import -v -trustcacerts -alias cassandranode -file cassandranode.cer -keystore
truststore.node
# new password: cassandra
# will create create truststore.node file
```

Prepare keystore file for cassandra configuration file (cassandra.yaml)

Copy and secure the keystore file, **keystore.node**, in an appropriate path in your system

Example

```
# copy keystore.node file in <cassandra home path>/conf/cassandra.yaml
$ cp keystore.node .keystore
# for Cassandra server side (client_encryption config)
# restrict access to this file
$ chmod 600 .keystore
```

Set up Cassandra client encryption in cassandra.yaml file

```
# enable or disable client/server encryption.
client_encryption_options:
  enabled: true ## to be changed
  # If enabled and optional is set to true encrypted and unencrypted connections are
  handled.
  optional: false
  keystore: conf/.keystore ## to be changed
  keystore_password: cassandra ## to be changed
  # require_client_auth: false
  # Set trustore and truststore_password if require_client_auth is true
  # truststore: conf/.truststore
  # truststore_password: cassandra
  # More advanced defaults below:
  # protocol: TLS
  # algorithm: SunX509
  # store_type: JKS
  # cipher_suites:
  [TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256_CBC_SHA,TLS_DHE_RSA_WITH_AES_128_CBC_SHA,TLS_DHE_RSA_WITH_AES_256_CBC_SHA]
```

You can now start all of your Cassandra nodes and use CQLSH to provision them.

CQLSH TLS configuration

If you set up both an unsecured and a secured Cassandra port, you can use CQLSH on the unsecured Cassandra port as is. However, if you need to use CQLSH on a secured Cassandra port, you must export the cassandra nodes certificate to another format:

```
$ keytool -importkeystore -srckeystore keystore.node -destkeystore cassandranode.p12
-deststoretype PKCS12 -srcstorepass cassandra -deststorepass cassandra
# will create cassandranode.p12 file
$ openssl pkcs12 -in cassandranode.p12 -nokeys -out cassandranode.cer.pem -passin
pass:cassandra
# will create cassandranode.cer.pem file
$ openssl pkcs12 -in cassandranode.p12 -nodes -nocerts -out cassandranode.key.pem -passin
pass:cassandra
# will create cassandranode.key.pem file
```

CQLSH TLS setup

Using a cqlshrc file

Create a cqlshrc file in **/home/user home/.cassandra/cqlshrc**:

```
[ssl]
certfile = <path to certificate>/cassandranode.cer.pem
validate = false ;; Optional, true by default.
```

Note: If **validate** is enabled, the host in the certificate is compared to the host of the machine to which it is connected, to verify that the certificate is trusted.

Using an environment variable

You must set the **SSL_CERTFILE** and **SSL_VALIDATE** environment variables:

```
$ export SSL_CERTFILE=<path to certificate>/cassandranode.cer.pem
$ export SSL_VALIDATE=true
```

CQLSH TLS connection

To start CQLSH, you must log in with user **cassandra** (default password: *cassandra*):

```
<apache-cassandra-4.x home>$ bin/cqlsh --ssl -u cassandra -p cassandra <cassandra node IP
address> <native port, default is 9042>
```

GMS side

GMS TLS configuration

Import the Cassandra node certificate

You can import the **cassandranode.cer** file created above in one of two ways:

- Import the Cassandra node certificate into a specific GMS file. **Note:** If you use this method, you won't be able to use some features of the notification API that needs to read other certificates.
- Import the certificate into the Java JDK/JRE security file that resides in the JDK you use to start GMS:
Java JDK home/jre/lib/security/cacerts

Using a specific GMS Certificate file

To import the Cassandra certificate to a specific Java client truststore file (for GMS), use **keytool**, which is part of the JDK toolset:

```
<GMS client side>$ keytool -import -v -trustcacerts -alias cassandranode -file
cassandranode.cer -keystore client.truststore
# password: cassandra
```

Modify launcher.xml

Add the following parameters to **launcher.xml** (note that the **debugtls** parameter, which is used for debugging, is optional):

```
<parameter name="cassandranodes_truststore" displayName="cassandratruststore" mandatory="true"
hidden="true" readOnly="true">
  <description><![CDATA[Certificates trustStore for Cassandra nodes]]></description>
  <valid-description><![CDATA[]]></valid-description>
  <effective-description/>
  <format type="string" default="-Djavax.net.ssl.trustStore=client.truststore" />
  <validation></validation>
</parameter>
<parameter name="cassandranodes_trustStorePassword" displayName="cassandratrustStorePassword"
mandatory="true" hidden="true" readOnly="true">
  <description><![CDATA[Certificates trustStore password for Cassandra
nodes]]></description>
  <valid-description><![CDATA[]]></valid-description>
  <effective-description/>
  <format type="string" default="-Djavax.net.ssl.trustStorePassword=cassandra" />
  <validation></validation>
</parameter>
<parameter name="debugtls" displayName="Debug TLS" mandatory="true" hidden="true"
readOnly="true">
  <description><![CDATA[Add debug mode for SSL]]></description>
  <valid-description><![CDATA[]]></valid-description>
  <effective-description/>
  <format type="string" default="-Djavax.net.debug=ssl" />
  <validation></validation>
</parameter>
```

Using a global Certificate file

For Windows: Add **Certificates trustStore for Cassandra nodes** to the main JRE **lib/security/cacerts** file:

```
$ sudo keytool -importcert -file cassandranodes.cer -alias cassandranodes -keystore
$JAVA_HOME/jre/lib/security/cacerts -storepass changeit
```

You do not need to change **launcher.xml** when using a global Certificate file.

Set the TLS-related GMS options

Set the GMS native-port and secured application options to the values shown here:

```
[cassandra]
nodes = <your Cassandra hosts or IP addresses for the local datacenter (comma-separated list)>
native-port = 9142
secured = true
keyspace-prefix = <if you need to select a prefix other than the default one (''gsg'') for
your schemas>
```

For versions prior to GMS 8.5.207, you also need to configure the following options:

```
strategy-class = NetworkTopologyStrategy
strategy-option = DC1:2;DC2:2
```

Authentication on External Cassandra

Important

Supports Cassandra version 2.1.x and higher (≤ 4.1).

The following steps are prerequisites prior to enabling authentication.

Configure `cassandra.yaml` File

1. Stop the Cassandra nodes.
2. Edit the `conf/cassandra.yaml` file for all nodes.
3. Ensure that `cluster_name` is identical for all nodes.
4. Locate the seed nodes. This is the field for all Cassandra nodes; change it accordingly:
 - For the seed node, this will be its own port.
 - For the non-seed nodes, this will be the IP address of the seed node.
5. Ensure that `listen_address` is changed from `127.0.0.1` to the current IP address.
6. Ensure that `rpc_address` is changed from `127.0.0.1` to the current IP address.
7. Locate the authenticator field.
8. Change the value from `AllowAllAuthenticator` to `PasswordAuthenticator`.

```
authenticator: PasswordAuthenticator
```

Note: The full classname is `org.apache.cassandra.auth.PasswordAuthenticator`.
9. Save the file.
10. Repeat these steps on each external Cassandra instance.
11. Start all the cassandra nodes.

Execute CQL Commands

1. On the external Cassandra, using the `cqlsh` utility (included with Cassandra), create your username and password for **one** of the nodes (not all). The following example shows the creation of a `genesys` user with `genesys` password.

```
$ cqlsh -u cassandra -p cassandra cassandra_host cassandra_port
> CREATE USER genesys WITH PASSWORD 'genesys';
> LIST USERS;
  name | super
-----+-----
 genesys | False
 cassandra | True
```

Important

The default superuser is `cassandra` with password `cassandra`. This step is required to be completed on only one external Cassandra instance. It will then be replicated to the other nodes.

2. On Windows OS / Cassandra versions 2.1 or higher, replace:

```
$ cqlsh -u cassandra -p cassandra cassandra_host cassandra_port
```

with:

```
{path_to_cassandra}\bin>{path_to_python}\python.exe cqlsh cassandra_host -u cassandra -p cassandra
```

3. Set the options of `cqlsh` before parameters or set your python 2.7 path in `PATH` environment variable like this:

```
PATH={path_to_python};%PATH%
```

Therefore, you can launch the `cqlsh` script using the `cqlsh.bat` command:

```
cqlsh.bat -u cassandra -p cassandra cassandra_host
```

Using the default `cassandra` port of `native_transport_port` (default is 9042). Otherwise you will need to add the port parameter to the `cqlsh` script.

4. Change the consistency level of the `system_auth` table and apply the `CREATE` command above according to the Cassandra version:

```
$ cqlsh -u cassandra -p cassandra cassandra_host cassandra_port
> ALTER KEYSPACE system_auth WITH REPLICATION = { 'class' : 'NetworkTopologyStrategy',
'DC1' : 2, 'DC2' : 2 };
```

Change the Replication Factor or Strategy

If you use authentication with Cassandra, the `system_auth` Cassandra system table has a replication factor of 1. In this scenario, only one node is aware of authentication).

If you change the strategy class or the replication factor, you must also report this change for all nodes of the cluster, as described here:

- Start all Cassandra nodes and change the replication factor to 3.

```
cqlsh -u cassandra -p cassandra
ALTER KEYSPACE "system_auth" WITH REPLICATION = {'class' : 'SimpleStrategy',
'replication_factor' : 3};</source>
```

- Start repair task on all nodes:

```
nodetool repair
```

- Select a node (other than the first one) for testing:

```
cqlsh -u cassandra -p cassandra
```

Set Configuration Options

1. In Configuration Manager, locate and open your GMS Application object.
2. On the Options tab, **[cassandra-authentication-security]** section, set the following options with the same username and password that you just created on the external Cassandra.
 - **username**, for example, genesys
 - **password**, for example, genesys
3. **Restart GMS**. The Pelops and Hector clients connect to the external Cassandra using the login and password.

Authorization on External Cassandra

Updated in 8.5.207

Important

Supports Cassandra version 2.1.x and higher (≤ 4.1).

GMS 8.5.207+

Starting in 8.5.207, you only need to grant permissions in **one** of the nodes (not all) for the **genesys** user.

```
$ cqlsh -u cassandra -p cassandra cassandra_host cassandra_port
> LIST USERS; // or LIST ROLES; on cassandra 4.x versions
      name | super
-----+-----
      genesys | False
      cassandra | True
> LIST ALL PERMISSIONS OF genesys;
```

```

(0 rows)

> CREATE KEYSPACE gsg WITH REPLICATION = { 'class' : 'NetworkTopologyStrategy', 'DC1' : 2,
'DC2' : 2 };
> CREATE KEYSPACE gsg_dd WITH REPLICATION = { 'class' : 'NetworkTopologyStrategy', 'DC1' : 2,
'DC2' : 2 };
> GRANT CREATE ON KEYSPACE gsg TO genesys;
> GRANT MODIFY ON KEYSPACE gsg TO genesys;
> GRANT SELECT ON KEYSPACE gsg TO genesys;
> LIST ALL PERMISSIONS OF genesys;
username | resource | permission
-----+-----+-----
genesys | <keyspace gsg> | CREATE
genesys | <keyspace gsg> | SELECT
genesys | <keyspace gsg> | MODIFY
(3 rows)
> GRANT CREATE ON KEYSPACE gsg_dd TO genesys;
> GRANT MODIFY ON KEYSPACE gsg_dd TO genesys;
> GRANT SELECT ON KEYSPACE gsg_dd TO genesys;
> LIST ALL PERMISSIONS OF genesys;
username | resource | permission
-----+-----+-----
genesys | <keyspace gsg> | CREATE
genesys | <keyspace gsg> | SELECT
genesys | <keyspace gsg> | MODIFY
genesys | <keyspace gsg_dd> | CREATE
genesys | <keyspace gsg_dd> | SELECT
genesys | <keyspace gsg_dd> | MODIFY
(6 rows)

```

Versions Prior to 8.5.207

After creating the authentication, you must enable authorization and create keyspaces.

Configure the cassandra.yaml Files

1. Edit the conf/cassandra.yaml file for all nodes. Locate the authorizer field.
2. Change the value from AllowAllAuthorizer to CassandraAuthorizer.

```
authorizer: CassandraAuthorizer
```

Note: The full classname is org.apache.cassandra.auth.CassandraAuthorizer.

3. Save the file.
4. Repeat these steps on each external Cassandra instance.
5. Start all the cassandra nodes.

Execute CQL Commands

To authorize actions on the keyspace, you must first create the keyspace(s), then grant them permissions in **one** of the nodes (not all).

1. On the external Cassandra, using the cqlsh utility (included with Cassandra), create your keyspaces in one of the nodes. The following example shows the gsg and gsg_dd keyspaces.

Important

This step is required to be completed on only one external Cassandra instance. It will then be replicated to the other nodes.

- Set permissions to GMS keyspaces in Cassandra using CQLSH (example for cassandra 2.1). Change the consistency level of GMS keyspaces and apply the CQL commands shown below according to the Cassandra version:

```
$ cqlsh -u cassandra -p cassandra cassandra_host cassandra_port
> LIST USERS;
  name | super
-----+-----
 genesys | False
 cassandra | True
> LIST ALL PERMISSIONS OF genesys;
(0 rows)

> CREATE KEYSPACE gsg WITH REPLICATION = { 'class' : 'NetworkTopologyStrategy', 'DC1' :
2, 'DC2' : 2 };
> CREATE KEYSPACE gsg_dd WITH REPLICATION = { 'class' : 'NetworkTopologyStrategy', 'DC1'
: 2, 'DC2' : 2 };
> GRANT ALTER ON KEYSPACE gsg TO genesys;
> GRANT CREATE ON KEYSPACE gsg TO genesys;
> GRANT DROP ON KEYSPACE gsg TO genesys;
> GRANT MODIFY ON KEYSPACE gsg TO genesys;
> GRANT SELECT ON KEYSPACE gsg TO genesys;
> LIST ALL PERMISSIONS OF genesys;
username | resource | permission
-----+-----+-----
 genesys | <keyspace gsg> | CREATE
 genesys | <keyspace gsg> | ALTER
 genesys | <keyspace gsg> | DROP
 genesys | <keyspace gsg> | SELECT
 genesys | <keyspace gsg> | MODIFY
(5 rows)
> GRANT ALTER ON KEYSPACE gsg_dd TO genesys;
> GRANT CREATE ON KEYSPACE gsg_dd TO genesys;
> GRANT DROP ON KEYSPACE gsg_dd TO genesys;
> GRANT MODIFY ON KEYSPACE gsg_dd TO genesys;
> GRANT SELECT ON KEYSPACE gsg_dd TO genesys;
> LIST ALL PERMISSIONS OF genesys;
username | resource | permission
-----+-----+-----
 genesys | <keyspace gsg> | CREATE
 genesys | <keyspace gsg> | ALTER
 genesys | <keyspace gsg> | DROP
 genesys | <keyspace gsg> | SELECT
 genesys | <keyspace gsg> | MODIFY
 genesys | <keyspace gsg_dd> | CREATE
 genesys | <keyspace gsg_dd> | ALTER
 genesys | <keyspace gsg_dd> | DROP
 genesys | <keyspace gsg_dd> | SELECT
 genesys | <keyspace gsg_dd> | MODIFY
(10 rows)
```

- Add the user to Cassandra by using the same CQLSH commands than for Authentication.
- Restart GMS. The Pelops and Hector clients connect to the external Cassandra and are authorized to manage the GMS keyspaces (gsg and gsg_dd).

Final Steps

- In the GMS Application > Security section > Log On As SYSTEM Account.
- The time zone for all nodes must be identical. Make sure that you synchronize the time before testing.

Repairing Cassandra Nodes

Important

For detailed procedures about repairing nodes, refer to the [Official DataStax documentation](#).

According to the DataStax documentation, repairing nodes is part of your housekeeping. Schedule this operation regularly, as detailed in [When to run anti-entropy repair](#).

For detailed information about commands, see [Manual repair: Anti-entropy repair](#).

Do not restart a C* instance down for a few hours

If a C* instance is down for a few hours, do not restart it to join the cluster. If you do so, you will create some inconsistencies. If you do not mind creating a few inconsistencies and if the number of involved instances is 1-2 C* max, a full cluster repair will resolve the inconsistencies. If you don't perform a full cluster repair, some issues will remain.

Do not repair C* instances that are down for too long

If the C* instances are down for too long, assuming that enough C* instances are part of the cluster, the cluster should survive the death of some C* instances. For example, if the replication factor is 4, written as RF=4, then having 5+ instances allows 1 instance to die or to be decommissioned safely without impacting the cluster dataset. This allows intermittent issues on a C* instance as long as the QUORUM is available, which is $RF/2 + 1 = 3$ in this example.

If the quorum is available, you should remove the dead nodes from the cluster in a clean way, before eventually rejoining.

To remove the dead nodes cleanly, you can try the following procedure:

```
# check if some removal in progress (removenode status)
$ nodetool removenode status
...

# check if there are dead nodes (describecluster)
$ nodetool describecluster
...

# check token id for dead nodes (status)
```

```

$ nodetool status
Datacenter: DC2
=====
Status=Up/Down |/ State=Normal/Leaving/Joining/Moving
-- Address      Load          Tokens  Owns    Host ID                               Rack
UN 192.168.XXX.XXX 155.97 KB    256     ?       a5303996-e61d-4463-a5cd-bde36c8762f5 RAC2
UN 192.168.XXX.XXX 165.72 KB    256     ?       17c8c1d8-08bc-489b-85eb-92bc9bc0dedb RAC1
Datacenter: DC1
=====
Status=Up/Down |/ State=Normal/Leaving/Joining/Moving
-- Address      Load          Tokens  Owns    Host ID                               Rack
UN 192.168.XXX.XXX 116.01 KB    256     ?       2c75bf48-66ea-47fb-a4f4-0664c82060cb RAC1
UN 192.168.XXX.XXX 101.8 KB     256     ?       3a61a518-b6e1-42f1-90f5-f8349f66fd3d RAC2

# remove the dead token (removenode) using the *Host ID* field of the down node to be removed
$ nodetool removenode <token>
...
# If this command hangs for more than a minute cancel it (for example press Ctrl-C) and run:
$ nodetool removenode force

```

From this point, the removed C* instance is no longer part of C* cluster. You can now delete its data on disk. To add the C* instance back into the cluster like a new node, restart it.

Important

By default, the data folder contains all data related to the instance replica (data, commitlog, and more). Other folders might be found under the root data folder). You can delete all the content of the data folder.

Frequently Asked Questions

The 8.5.206 release introduces an important change of Cassandra driver.

What is the significance of adding all the nodes/IP addresses in the node section of the Cassandra option?

The driver used by GMS to connect to Cassandra supports setting multiple Cassandra instances.

- For versions older than 8.5.206, these instances are used as Cassandra coordinators. If one instance dies, the driver automatically switches to the next Cassandra instance that is used as coordinator.
- Starting in 8.5.206, other Cassandra instances that belong to same Data Center are automatically detected. If one of them dies, the driver connects to all the detected Cassandra instances, including those that are not defined in the GMS options.

Can I add only one seed node in place of all the nodes/IP addresses in the node

section of the Cassandra option?

Yes. However, adding only one seed node means that GMS will not be able to start if this instance is down.

- For versions older than 8.5.206, this also means that, after a correct GMS start, if your Cassandra node dies, GMS will die too because it can reach known Cassandra instances only.
- Starting in 8.5.206, GMS will try connection to other detected Cassandra instances that belong to same Data Center.

My GMS Application is not starting up when one node/host server is in hung or downstate; what can be a permanent fix for this issue?

Genesys recommends that you upgrade to 8.5.206+. This scenario has been tested and it does not reproduce starting in 8.5.206.04. Note that this release also adds other features, like a secured connection from GMS to Cassandra.

Configuring Apache Load Balancer

The following is an example of how to configure Apache load balancer that can be positioned in front of GMS nodes for API requests distribution. For configuration of other load balancing solutions, please refer to their documentation.

Warning

You cannot change the `/genesys` path used for the Genesys Mobile Services applications. The base URL must remain `/genesys`.

See also the [Configuration Sample Chat API v2 CometD](#) page for some load-balancer/proxy configuration examples that show how to make a load-balancer proxy work with GMS WebSockets.

Configuration of `mod_proxy_balancer`

Install Apache 2.2 or higher, starting from this version `mod_proxy` is able to use the extension `mod_proxy_balancer`. Make sure the following modules are present in your Apache "modules\" folder, upload them in case they are absent:

- `mod_proxy.so`
- `mod_proxy_balancer.so`

Add the following strings to your Apache configuration "httpd.conf" file in order to load the modules:

- `LoadModule proxy_module modules/mod_proxy.so`
- `LoadModule proxy_balancer_module modules/mod_proxy_balancer.so`
- `LoadModule proxy_http_module modules/mod_proxy_http.so`

Basically you need to add node based configuration, for this add the following to the `httpd.conf` file:

```
ProxyPass / balancer://my_cluster/ stickysession=JSESSIONID nofailover=0n

<Proxy balancer://my_cluster>
  BalancerMember http://yourjetty1:8080 route=jetty1
  BalancerMember http://yourjetty2:8080 route=jetty2
</Proxy>
```

`Proxy balancer://` - defines the nodes (workers) in the cluster. Each member may be a `http://` or `ajp://` URL or another `balancer://` URL for cascaded load balancing configuration. If the worker name is not set for the Jetty servers, then session affinity (sticky sessions) will not work. The `JSESSIONID` cookie must have the format `<sessionID>.<worker name>`, in which worker name has the same value as the route specified in the `BalancerMember` above (in this case "jetty1" and "jetty2").

As an example, add the following to the start.ini file to set the worker name in case you need session affinity (sticky sessions):

```
jetty.sessionIdManager.workerName=jetty1  
--module=...,sessions
```

Load Balancer Management Page

Apache provides balancer manager support so that the status of balancer can be viewed on a web page. The following configuration enables this UI at /balancer URL:

```
<Location /balancer>  
SetHandler balancer-manager  
  
Order Deny,Allow  
Deny from all  
Allow from all  
</Location>
```

Configure ORS Load Balancing

Important

Starting in version 8.5.219, GMS requires Orchestration Server (ORS) version 8.1.300.30 and higher to send requests to the ORS /heartbeat URI.

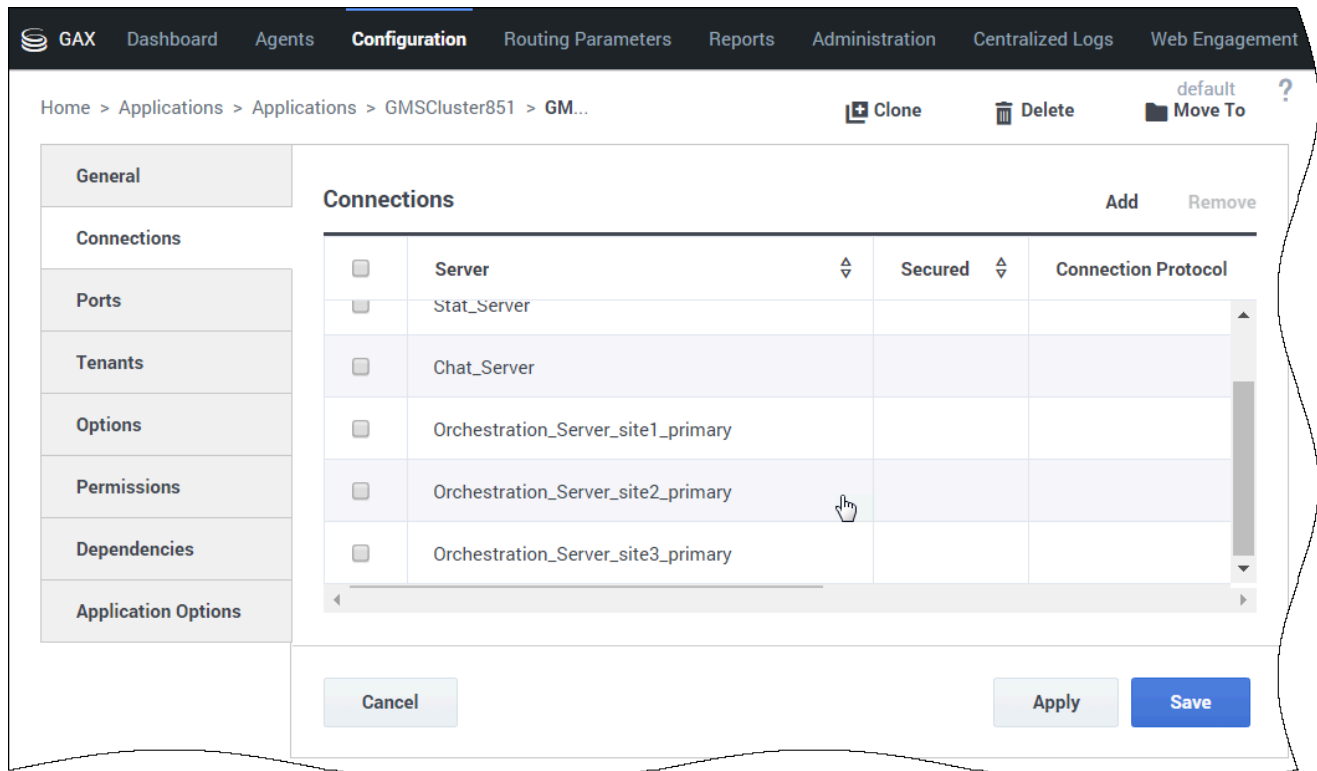
Supported ORS Load Balancing Features

Genesys Mobile Services (GMS) supports the following load balancing features:

- Ability to configure a list of service URLs to access a given list of nodes.
- **Linear** hunt strategy where requests are always delivered to the first available node in the list.
- **Circular** hunt strategy where requests are delivered in a round-robin fashion to the list of nodes/URLs. Note that the Circular strategy automatically takes into account ORS and ORS backups.
- Ability to configure a linear hunt strategy or a circular hunt strategy in the service configuration. The default hunt mode is **circular**.

How to set up GMS connections to ORS?

Genesys recommends that you add a list of ORS primary servers to the **Connections** tab in case GMS should fall back to other nodes on failure.



GMS will automatically find the ORS backup servers and add them to the load balancing strategy; however, when failing on one node, GMS will try the next ORS node in the list, regardless if it is a primary or a backup server.

This default hunt strategy is **circular**. To modify this strategy, edit the `_ors_lb_strategy` option in the **ors** section.

When to Disable ORS Load-Balancing?

If the ORS is too old—check the **Prerequisites**, you can use the `enable_ors_loadbalancer` option to disable load-balancing in the **ors** section.

More ORS Load Balancing Settings

The following settings can be defined at the cluster or node level in the **ors** section of your GMS application. You cannot supersede these settings at the service level:

- `max_ors_idle_connection_time`
- New in 8.5.219

- healthcheck-ping-url
- healthcheck-ping-interval
- Deprecated in 8.5.219
 - ors_loadbalancer_refresh_rate

Important

If you do not update your configuration with the new options, GMS uses the deprecated options for backward compatibility.

Configure ORS Request Attempts

In a circular ORS load balancing strategy, `max_ors_request_attempts` provides you with the ability to select the next ORS Server in the list of ORS Servers defined for the service when the request to the first ORS fails.

For example, if you set the value of `max_ors_request_attempts` to 1, the first ORS Server in the list will be used only one time, and in case of ORS failure, the request will fail. If you set the value to 3, the first ORS Server in the list will be used and if the request fails, the next ORS Server in the list will be used, and so on, until the third server. After the third server fails to respond, the request returns a failure. A linear ORS load balancing strategy follows the same process with the `max_ors_request_attempts` option for retry on failure.

Advanced Setup Scenarios for ORS Connections

For any advanced setup, the default ORS Load Balancing strategy is **circular** and **enabled** by default.

Important

Unless you have specific needs, using the **Connections** tab to add GMS connections to ORS should cover most use cases.

Overwrite General Server Settings

To define a list of ORS servers at the GMS application level, you can set a list of ORS URLs in the **server** section by using the `_ors` option.

- Include all of the ORS URLs in the `_ors` comma-separated list, regardless if they are backup or primary servers.
- To modify the strategy mechanism related to this list, edit the `_ors_lb_strategy` option of the **server** section.
- This configuration applies to the application's services by default.

Overwrite Settings per Service

Important

Your service must be of type ors.

To overwrite ORS settings at the service level, you can use the [Service Management User Interface](#) to set a comma-separated list of ORS URLs in the `_ors` option of your service.

- For each service, you can also specify a load balancing strategy in the `_ors_lb_strategy` option. This ensures the possibility to define as many load balancing strategies as services with a distinct list of Orchestration Servers.
- By default, the load balancing strategy is **circular**.

How to Manage ORS Settings in a Cluster

If you define an application cluster for your GMS applications and if you wish to have specific ORS settings, you can edit your service configuration through Configuration Manager, Genesys Administrator, or Genesys Administrator Extension instead of using the [Service Management User Interface](#).

- If you define your load balancing properties for your service in this cluster configuration, these service options apply to all of the GMS nodes.
- If you modify these service options in one of the GMS nodes, the new options apply to this given node and cluster options are superseded in this node.

This mechanism lets you define common service options for your cluster, with the possibility to fine-tune the service options in one or more GMS nodes.

1. Open your application's configuration and select the **Options** tab. Edit your service.<service_name> section.
2. Click Create New Section/Option.
3. Enter `_ors` for the Option Name, and then enter the list of URLs, separated with commas, for the Option Value. Click OK.
4. (Optional) Click Create New Section/Option again. Enter `_ors_lb_strategy` for the Name, and then enter `circular` or `linear` for the Value. If not specified, the default `_ors_lb_strategy` value is `circular`.

Section	Option	Default	Description
service.<service name>	_ors	empty	Comma-separated list of ORS URLs. http://host1:port1,http://host2:port

Section	Option	Default	Description
			Overrides the <code>_ors</code> option of the server section.
service.<service name>	<code>_ors_lb_strategy</code>	<code>circular</code>	Strategy for the ORS load balancer. This option overrides the <code>_ors_lb_strategy</code> option of the server section. Supported values are: <code>circular</code> or <code>linear</code> .

Configure URS Load Balancing

Introduced in 8.5.107

URS provides Genesys Mobile Engagement with the data used to calculate Estimated Wait Time and submit Callback interactions at the appropriate time. If you have configured a URS backup, or more than one URS instance, you need to set up load balancing features.

Prerequisites

- URS version 8.1.400.12+

Supported URS Load Balancing Features

Genesys Mobile Services (GMS) supports the following load balancing features:

- Ability to configure a list of service URLs to access a given list of nodes.
- **Linear** (default) hunt strategy where requests are always delivered to the first available node in the list.
- **Circular** hunt strategy where requests are delivered in a round-robin fashion to the list of nodes/URLs. Starting in 8.5.107, the Circular strategy automatically takes into account URS and URS backups if the URS is specified in the GMS Connections tab.
- Ability to configure a linear hunt strategy or a circular hunt strategy in the service configuration. The default hunt mode is **linear**.

How to set up GMS connections to URS?

Genesys recommends that you add a list of URS primary servers to the **Connections** tab in case GMS should fall back to other nodes on failure.

GMS will automatically find the URS backup servers and add them to the load balancing strategy; however, when failing on one node, GMS will try the next URS node in the list, regardless if it is a primary or a backup server.

This default hunt strategy is **linear**. To modify this strategy, edit the `_urs_lb_strategy` option in the **urs** section.

When to Disable URS Load-Balancing?

If the URS is too old—check the [Prerequisites](#), you can use the `enable_urs_loadbalancer` option to disable load-balancing in the [reporting](#) section.

If you upgraded to 8.5.107:

- You should set `enable_urs_loadbalancer` to `true` even if you were not using URS load balancing previously;
- Set this option to `false` only if you need to deactivate load balancing.

More URS Load Balancing Settings

The following settings can be defined at the cluster or node level in the `urs` section of your GMS application. You cannot supersede these settings at the service level:

- `max_urs_idle_connection_time`
- `urs_loadbalancer_refresh_rate`

Configure URS Request Attempts

In a circular URS load balancing strategy, the `max_urs_request_attempts` option provides you with the ability to select the next URS Server in the list of URS Servers defined for the service when the request to the first URS fails.

For example, if you set the value of `max_urs_request_attempts` to 1, the first URS Server in the list will be used only one time, and in case of URS failure, the request will fail. If you set the value to 3, the first URS Server in the list will be used and if the request fails, the next URS Server in the list will be used, and so on, until the third server. After the third server fails to respond, the request returns a failure. A linear URS load balancing strategy follows the same process with the `max_urs_request_attempts` option for retry on failure.

Advanced Setup Scenarios for URS Connections

Starting in 8.5.107, for any advanced setup, the default URS Load Balancing strategy is **linear** and **enabled** by default.

Important

Unless you have specific needs, using the **Connections** tab to add GMS connections to URS should cover most use cases.

Overwrite General Server Settings

To define a list of URS primary servers at the GMS application level, you can set a list of URS URLs in the **reporting** section by using the `_urs_url` option.

- You need to provide URS primary servers only, not their backup. In 8.5.107, GMS will automatically find URS backup servers and add them to the Load Balancing strategy.
- To modify the strategy mechanism related to this list, edit the `_urs_lb_strategy` option of the **urs** section.
- This configuration applies to the application's services by default.

Overwrite Settings per Service

To overwrite URS settings at the service level, you can use the **Mobile Engagement User Interface** to set a comma-separated list of URS URLs in the `_urs_url` option of your service.

- By default, starting in 8.5.107, the load balancing strategy is **linear**.
- The `_urs_url` option is not available in all services.
- You cannot modify the strategy per service.

How to Manage URS Settings in a Cluster

If you define an application cluster for your GMS applications and if you wish to have specific URS settings, you can edit your service configuration through Configuration Manager, Genesys Administrator, or Genesys Administrator Extension instead of using the **Mobile Engagement UI**.

- If you define your load balancing properties for your service in this cluster configuration, these service options apply to all of the GMS nodes.
- If you modify these service options in one of the GMS nodes, the new options apply to this given node and cluster options are superseded in this node.

This mechanism lets you define common service options for your cluster, with the possibility to fine-tune the service options in one or more GMS nodes.

1. Open your application's configuration and select the **Options** tab. Edit your `service.<service_name>` section.
2. Click **Create New Section/Option**.
3. Enter `_urs_url` for the Option Name, and then enter the list of URLs, separated with commas, for the Option Value. Click **OK**.
 - By default, starting in 8.5.107, the load balancing strategy is **linear**.

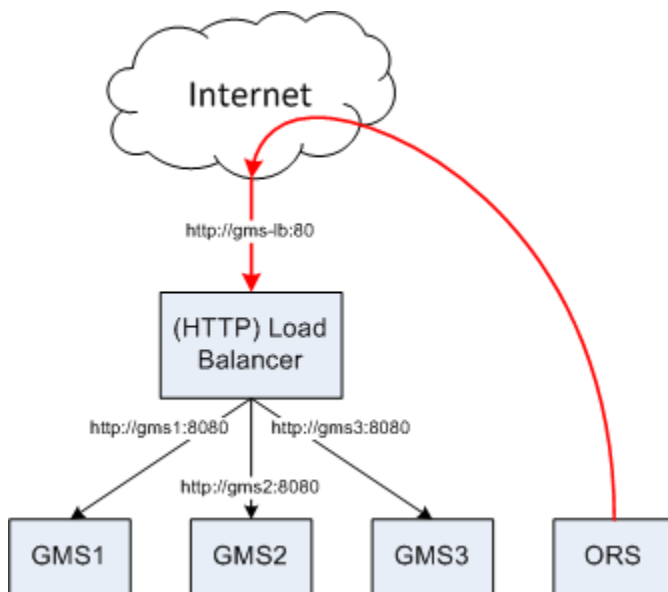
Configure DFMs in Load Balancing Deployment

Included in your Installation Package (IP) are special configuration files, called DFMs, which are required for Orchestration Server-based services. These files define Genesys Mobile Services-specific SCXML constructs that are required for the execution of SCXML applications used within Orchestration Server-based services.

Important

Starting 8.5.104, you must update the DFM files deployed locally with the latest version provided in the GMS Admin UI.

Simple HTTP Connection to a Load Balancer



HTTP_simple connections can be set up in two cases:

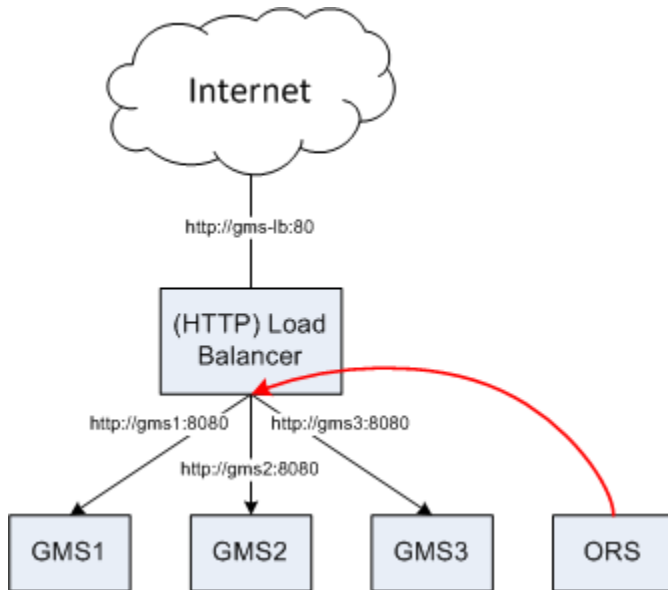
- Simple connection to an internal load balancer.
- Simple connection to an external load balancer:
 - If you use a load balancer address external to the company network in the simple connection, the Orchestration Server (ORS) must be able to reach to the outside world (Internet) before it can “back” to the load balancer and further on to GMS instances.

```
<dfm name='gsgBasedServices' namespace='http://www.genesyslab.com/modules/dfm/gsgBasedServices/v1'>
  ...
  <connections>
  <connection name='HTTP_simple' type='http' selectionmode='simple'>
  <servers>
  <server name='gsg' host='gms-lb' port='80' />
  </servers>
  </connection>
  ...
</dfm>
```

Important

Using a load balancer address external to the company network implies extra hops to the Internet. Longer access times and security policies that allow ORS to go outside of the network might create downturns.

Round Robin Connections



In the absence of an external load balancer or if the load balancer is external and you want to add extra hops to the internet (and/or deny ORS access to the outside internet), you can configure round robin connections. The round-robin connection type basically works as an internal load balancer whenever GMS services are being invoked by ORS. To implement this configuration, create round robin connections to each GMS node, as follows:

```

<connection name='HTTP_round_robin' type='http' selectionmode='round-robin'>
  <servers>
    <server name='gsg1' host='gms1' port='8080' />
    <server name='gsg2' host='gms2' port='8080' />
    <server name='gsg3' host='gms3' port='8080' />
  </servers>
</connection>

```

If you set up round robin connections, you can use the `conntype` attribute to control how ORS reaches GMS. You must set the `conntype` attribute for HTTP transports to `HTTP_round_robin` for POST, and DELETE requests:

```

<transport name='HTTP_GET' conntype='HTTP_round_robin'>

```

Important

ORS is not designed as a soft load balancer, thus does not handle GMS node failure in an optimal way. In addition, you must modify DFM files again, in case additional GMS nodes are added to the cluster, it does not make sense to use only a partial GMS cluster in the DFM configuration.

Modifying URL in HTTP transports

You might need to modify the `url` parameter of various HTTP transports if you use an external load balancer that is configured to “hide” the GMS Admin UI (`/genesys/admin`) part of the URL.

```
<transport name='HTTP_GET' conntype='HTTP_round_robin'>
  <param name='method' expr='GET' />
  <param name='enctype' expr='application/x-www-form-urlencoded' />
  <param name='url' expr='/genesys/1/service/' />
</transport>
```

Using Callbacks without DFM

To use the Callback strategy without using DFM, the GMS API can be used with the Session/Fetch requests (HTTP requests) using the URL configured in the **external_url_base** configuration option in the **server** section.

The GMS API will be called from the Callback strategy using the Session/Fetch requests (HTTP request), the URL used for these HTTP operations is the GMS application option **external_url_base** in **server** section. The following options are available to configure this feature.

- `_enable_gms_dfm`
- `_fetch_timeout`
- `_fetch_retry_attempt`
- `_fetch_retry_delay`
- `_services_credentials`

```
[service.<callback service name>]
...
_enable_gms_dfm=false
...
```

You can also create a GMS Services Credentials Transaction List when accessing the GMS API with credentials:

When GMS API are using credentials, create GMS Services Credentials Transaction List (called for example `GMS_Credentials`) with the following annex:

```
[server]
username=username
password<password>
```

The server username is the same username as the **username** option in GMS Application options in **server** section and the server password is the same password as the **password** option in GMS Application options in **server** section.

Add an `_services_credentials` option in Callback service section:

```
[service.<callback service name>]
...
_enable_gms_dfm=false
...
_services_credentials=GMS_Credentials
...
```

Limitations

When using the Callback strategy without using DFM, the following limitations apply:

- The Session/Fetch HTTP request uses the URL from GMS: `external_url_base` option value.
- There are no round-robin hosts as in the DFM definition.
- Credentials on GMS API are activated on server section only (not per service) via a Transaction List.

Configuring and Starting a GMS Cluster

Important

Even if GMS multi-node deployment requires a Load Balancing solution, Genesys does not provide support for third-party software. The steps in this page refer to a sample deployment of HAProxy that should be used only as an example. You must deploy an adequate flavor of Load Balancing software in front of GMS.

Prerequisites

- GMS version 8.5.x
- Red Hat Linux version 5.0 (32 bit and 64 bit), 4Gb RAM or Higher (for the HA Proxy Load Balancer)
- Before 8.5.206.04: Support for JDK 8 only
- Starting in 8.5.206.04: Support for Open JDK 8
- GMS nodes and cluster are **configured**.
- External Cassandra is **configured**.

Introduction

The process for initializing a GMS cluster (whether it is a single node, multiple nodes, or multiple data center cluster) is to first correctly configure the **Node and Cluster Initialization Properties** in each node's **cassandra.yaml** configuration file, and then start each node individually, starting with the seed node(s). Configuration file **cassandra.yaml** is automatically generated by GMS Installation Package, you don't need to update the file until you need specific settings. Installation Package proposes to choose between the type of node "seed node/Not a seed node". The following section details the GMS cluster setup.

Initializing a Single-Node Cluster

GMS is intended to run on multiple nodes, however, you may want to start with a single node cluster for evaluation purposes.

To start GMS on a single node, set the following required properties in the **cassandra.yaml** file:

```
cluster_name: GMS Cluster
initial_token:
```

(Optional) The following properties are already correctly configured for a single node instance of Cassandra. However, if you plan on expanding to more nodes after your single-node evaluation, setting these correctly the first time you start the node is recommended.

```
seeds: <IP of GMS node>
listen_address: <IP of GMS node>
rpc_address: <IP of GMS node>
```

Start GMS on the node.

Initializing a Multi-Node or Multi-Data Center Cluster

To correctly configure a multi-node or multi-datacenter cluster you must determine the following information:

- A name for your cluster
- How many total nodes your cluster will have, and how many nodes per data center (or replication group)
- The IP addresses of each node
- The token for each node (see [Calculating Tokens](#)). If you are deploying a multi-datacenter cluster, make sure to assign tokens so that data is evenly distributed within each data center or replication grouping (see [Calculating Tokens for Multiple Data Centers](#)).
- Which nodes will serve as the seed nodes. If you are deploying a multi-datacenter cluster, the seed list should include a node from *each* data center or replication group.

This information will be used to configure the [Node and Cluster Initialization Properties](#) in the `cassandra.yaml` configuration file on each node in the cluster. Each node should be correctly configured before starting up the cluster, one node at a time (starting with the seed nodes). For example, suppose you are configuring a 4 nodes cluster spanning 1 rack in a single data center. The nodes have the following IPs, and one node in the rack will serve as a seed:

- GMS node 172.25.157.171 (seed)
- GMS node1 172.25.157.177
- GMS node2 172.25.157.179
- GMS node3 172.25.157.185

The `cassandra.yaml` files for each node would then have the following modified property settings.

node0

```
cluster_name: 'GMS Cluster'
initial_token:
seed_provider:
  - seeds: '172.25.157.171'
listen_address: 172.25.157.171
```

```
rpc_address: 172.25.157.171
```

node1

```
cluster_name: 'GMS Cluster'  
initial_token:  
seed_provider:  
  - seeds: '172.25.157.171'  
listen_address: 172.25.157.177  
rpc_address: 172.25.157.177
```

node2

```
cluster_name: 'GMS Cluster'  
initial_token:  
seed_provider:  
  - seeds: '172.25.157.171'  
listen_address: 172.25.157.179  
rpc_address: 172.25.157.179
```

node3

```
cluster_name: 'GMS Cluster'  
initial_token:  
seed_provider:  
  - seeds: '172.25.157.171'  
listen_address: 172.25.157.185  
rpc_address: 172.25.157.185
```

When the installation and configuration are done for all GMS's, you can start each instance.

Load Balancing Between GMS Instances

Load balancing is a computer networking methodology to distribute workload across multiple computers or a computer cluster, network links, central processing units, disk drives. In a GMS Cluster, Load Balancing is used to distribute the workload across multiple GMS instances. The software is installed on a separate Red Hat host. The installation of HAProxy is described [here](#). See also [How to setup HAProxy as Load Balancer for Nginx on CentOS 7](#). Once installed, you have to create a configuration file for HAProxy "haproxy-gms.cfg" and copy the following in the file:

```
global  
  daemon  
  maxconn 256  
defaults  
  mode http  
  timeout connect 5000ms  
  timeout client 50000ms  
  timeout server 50000ms  
frontend http-in  
  bind *:8080  
  default_backend cluster_gms  
listen admin  
  bind *:9090  
  stats enable  
backend cluster_gms  
  balance roundrobin # Load Balancing algorithm  
  #following http check, is used to know the status of a GMS (using NodeService from
```

GMS)

```
option httpchk GET /genesys/1/node
option forwardfor # This sets X-Forwarded-For
## Define your servers to balance
server server1 172.25.157.171:8080 weight 1 maxconn 512 check
server server2 172.25.157.177:8080 weight 1 maxconn 512 check
server server3 172.25.157.179:8080 weight 1 maxconn 512 check
server server4 172.25.157.185:8080 weight 1 maxconn 512 check
```

Once done, you can start HAProxy using the following command:

```
[root@bsgenhaproxy haproxy]# ./haproxy -f haproxy-gms.cfg
```

GMS Service Management UI

Cluster view in the [GMS Service Management User Interface](#), Home page:

Last Updated: 7/31/2013 12:59:25

IP: ✕

Token: 75046021690165490968853874128439747963

Status: Down

Load: ?

Data Center: datacenter1

Rack: rack1

Own: 14.69%

Running Since: Wed Jul 31 2013 12:59:25 GMT+0300

IP:

Token: 50057505283674829242183335979434942266

Status: Up

Load: 151.69 MB

Data Center: datacenter1

Rack: rack1

Own: 85.31%

Running Since: Tue Jul 30 2013 14:36:18 GMT+0300

HAProxy Statistics Report page

The following page is available at: http://<haproxy_host>:9090/haproxy?stats

HAProxy version 1.4.22, released 2012/08/09

Statistics Report for pid 4043

> General process information

pid = 4043 (process #1, nproc = 1)
 uptime = 0s 0h0m0s
 system limits: memmax = unlimited; ulimitn = 528
 maxsock = 528; maxconn = 250; maxpipes = 0
 current conns = 1; current pipes = 0/0
 Running tasks: 1/3

active UP

active UP, going down

active DOWN, going up

active or backup DOWN

active or backup DOWN for maintenance (MAINT)

Note: UP with load-balancing disabled is reported as "NOLE"

backup UP

backup UP, going down

backup DOWN, going up

not checked

Display option:

- [Hide DOWN servers](#)
- [Refresh now](#)
- [CSV export](#)

External resources:

- [Primary site](#)
- [Updates \(v1.4\)](#)
- [Online manual](#)

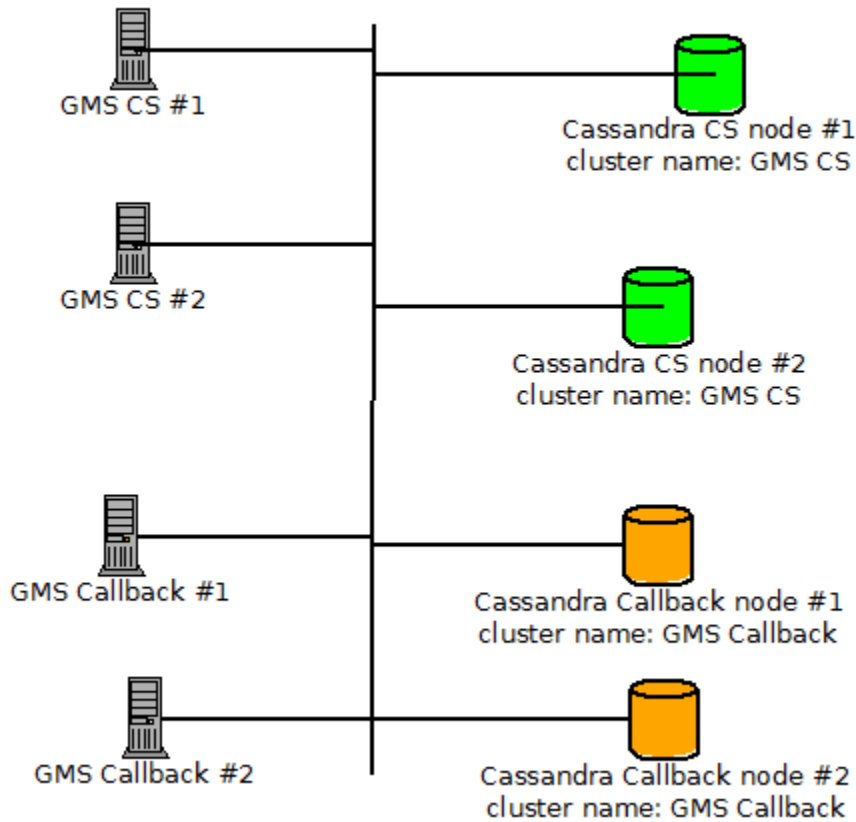
http_in		Queue		Session rate			Sessions			Bytes		Denied		Errors		Warnings		Server													
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Downtme	Thrtle		
Frontend	0	0	-	0	0	-	0	0	0	2 000	0	0	0	0	0	0	0	0	0	0	0	OPEN									
admin																															
Frontend	0	0	-	1	1	-	1	1	1	2 000	1	0	0	0	0	0	0	0	0	0	0	OPEN									
Backend	0	0	-	0	0	-	0	0	0	2 000	0	0	0	0	0	0	0	0	0	0	3s UP		0	0	0	0	0	0	0	0	
cluster_gms																															
server1	0	0	-	0	0	-	0	0	0	512	0	0	0	0	0	0	0	0	0	0	3s DOWN	LACON in 0ms	1	Y	-	0	0	1	3s	-	
server2	0	0	-	0	0	-	0	0	0	512	0	0	0	0	0	0	0	0	0	0	3s UP	L7OK:200 in 3ms	1	Y	-	0	0	0	0	0	
Backend	0	0	-	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3s UP		1	1	0	0	0	0	0	0	

You can now use the HAProxy endpoint `http://<haproxy_host>:<haproxy_port>` as the main entry point for the Cluster.

Limitations

If you set up different clusters of GMS for different purposes (Callback, and so on), you must change the cluster name to be able to start GMS (this applies to embedded, and external Cassandra clusters if they are set up for different purposes).

Example of GMS architecture with Cassandra clusters for different purposes:



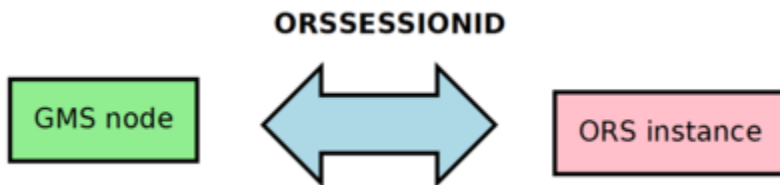
ORS Cookie Support

GMS now supports Cookie-based management, with the ability to store ORS cookies for a given service ID. GMS will use ORS-generated cookies in subsequent requests to the same service.

The ORS cookie prefix is ORSSESSIONID.

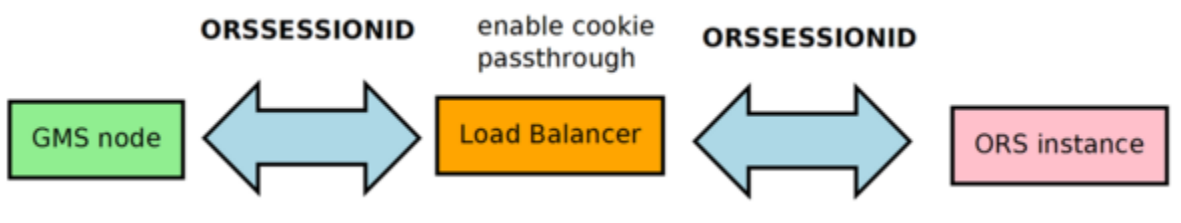
Basic Cookie Usage

When GMS nodes are directly connected to ORS instances, that is, without a load balancer between GMS and ORS, the ORS cookies are managed and stored in GMS for subsequent requests. When another request for the same service session ID is processed and also requires ORS processing, the GMS request to ORS will contain the previously stored cookie.



Load Balancer Cookie Usage

When the environment contains a load balancer between GMS nodes and ORS instances, the load balancer should take into account that ORS cookies must be transferred from and to the GMS nodes. In the load balancer configuration, make sure that cookie management is enabled in order for the ORS cookies to pass back and forth.



Mobile Push Notifications

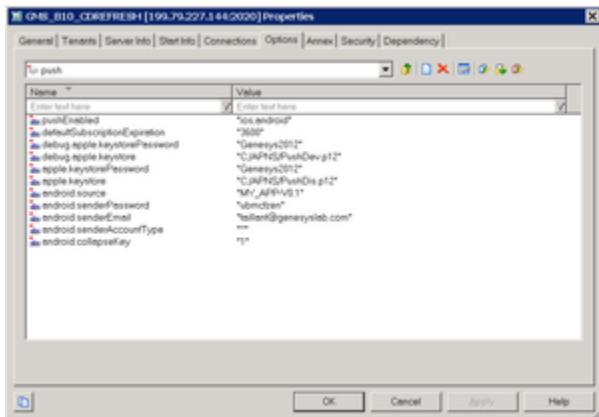
If you are using push notification service, the configuration detailed here should be implemented; however, these steps are not mandatory to complete your **GMS** installation. See also [Push Notification Service](#) for detailed information and examples.

Some services send native push messages to the mobile device. For this to work, both general and device-specific settings need to be configured correctly in the *push* section of your Genesys Mobile Services Application object.

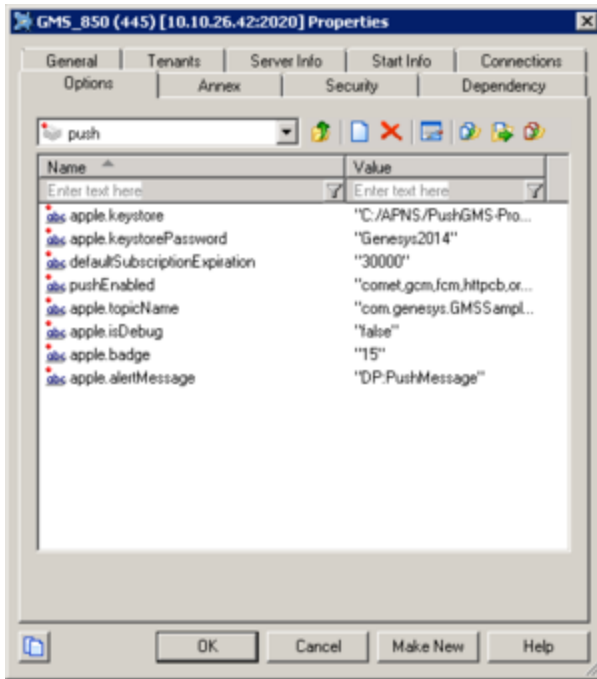
Important

Options set in the *push* section determine how all push notifications are handled by Genesys Mobile Services, regardless of which service is sending the notification.

Note that it is possible to configure this native push notification service for more than one type of device by using a comma-delimited string in the *pushEnabled* option. In this case, be sure to configure the mandatory options for all available device types.



GMS multi-device notification.png



GMS iOS notification.png

Common Device Settings

- pushEnabled - Device operating system.
- defaultSubscriptionExpiration
- customhttp.url

Mandatory iOS Device Settings

Updated in 8.5.228

To establish a TLS session with APNs, install an Entrust Secure CA root certificate on the provider's server. If the server is running on macOS, this root certificate is already part of the keychain. On other systems, if the certificate is not available, download this certificate from the [Entrust SSL Certificates](#) website.

To configure Apple Push Notification, configure the following options in the [push] section your GMS application:

- pushEnabled=ios in the Push section
- apple.keystore

- `apple.keystorePassword`
- `apple.topicName`, where `apple.topicName` is the bundle ID of your apple app.
For example, `com.genesys.GMSSampleApp` is the topic name of the `gmssample` app.

```
[push]
apple.keystore=<path of the downloaded p12 certificate>
apple.keystorePassword=<keystorePassword>
apple.topicName=<Bundle ID of your app>
pushEnabled=ios
```

If you do not specify provider information in your API queries or service configuration, GMS uses the above options defined in the push section by default.

You can also define these options for non-default providers in the `[push.provider.{ProviderName}]` section, where `{ProviderName}` is the string that you provide in the user data of your API queries or in your service configuration.

Important

To route push notification calls through a proxy, update the `proxy.pac` file and specify the file's location in your GMS application, under the `[gms]` section as shown here:

```
[gms]
http.proxy-auto-config-file=temp/ProxyPac.js
```

Mandatory Android Device Settings

Firebase Cloud Messaging Notification

Introduced in: 8.5.114

Due to recent changes in Google Cloud Messaging, GMS now supports Firebase Cloud Messaging (FCM).

- Refer to the official documentation to [Set Up a Firebase Cloud Messaging Client App on Android](#).
- You will need to retrieve an API Key through the [Firebase Console](#).
- If GMS is deployed behind a firewall, edit your rules to allow the server `fcm.googleapis.com` and range 5228-5230 for ports, as detailed in the [FCM ports and your firewall](#) section of the Firebase Cloud Messaging documentation.

To configure Native Push Notification through Firebase Cloud Messaging, you can either specify an `apiKey` or create a dedicated section to secure passwords (recommended for production environments) in the push section of your GMS application.

Development

```
[push]
fcm.apiKey=<serverKey>
pushEnabled=fcm
```

Production

```
[push]
fcm=fcmsection
pushEnabled=fcm
```

```
[fcmsection]
password=***** (<serverKey>)
```

- If you define these options in the push section, they will be used as default settings if you do not specify provider information in your API queries or service configuration.
- You can also define these options for non-default providers in the `[push.provider.{ProviderName}]` section, where `{ProviderName}` is the string you need to provide in the user data of your API queries or in your service configuration.

For example, you can configure `_provider_name={ProviderName}` for a given Callback service, or in a Chat V2 scenario, you could pass this provider name in the `push_notification_provider` user data of the `requestChat` operation.

GMS allows the following options for the provider-level configuration:

- `fcm.apiKey`
- `debug.fcm.apiKey`

Use `fcm.title` and `fcm.body` options to extend your configuration by creating an event-level `[push.provider.{ProviderName}.event]` section and then, a specific service name and event:

- `[push.provider.{ProviderName}.event.{ServiceName}]`
- `[push.provider.{ProviderName}.event.{ServiceName}.{EventName}]`

Deprecated —Mandatory Android GCM Device Settings

- `android.gcm.apiKey` — A valid Google API key value (Notifications are sent on behalf of this API key, see <http://developer.android.com/guide/google/gcm/gs.html>).
- `android.gcm.retryNumber` — The number of retries in case of service unavailability errors.

For additional detail about these options and the allowed values, see the [push Section](#) documentation.

Deprecated — Mandatory Android C2DM Device Settings

Important

Google has deprecated the C2DM Service, and no new users are being accepted. Please use the Firebase Cloud Messaging Service, described above).

- `android.senderEmail` — Name of a valid mail account. (Notifications are sent on behalf of this account.)
- `android.senderPassword` — Password of mail account specified in `android.senderEmail`.
- `android.senderAccountType` — Specified when initializing C2DM push service.
- `android.source` — Specified when sending push notifications.
- `android.collapseKey` — An arbitrary string used to collapse a group of like messages when the device is offline so that only the last message gets sent to the client.

Mandatory customhttp Settings

- `customhttp.url` — `http://xxxx/xx`
- `pushEnabled` — `comet,gcm,customhttp,orscb,ios`

Configure Historical Reporting

Genesys Mobile Engagement can work together with other Genesys products to provide Callback Historical reporting features. This page details how you can configure Callback to implement reporting.

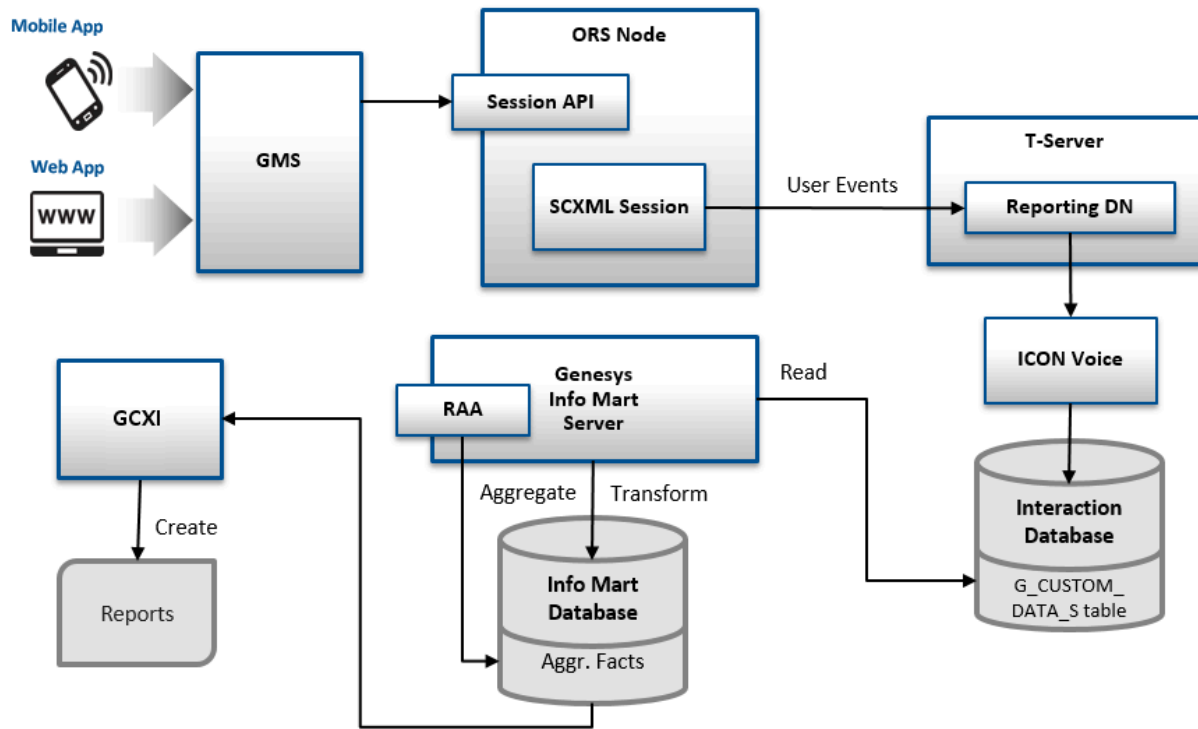
Prerequisites

Mandatory Genesys Components

Component	Minimum Version
Orchestration Server	8.1.400.24
Universal Routing Server	8.1.400.22
Interaction Concentrator	8.1.506.07
Genesys Info Mart	8.5.005 (GA)
Reporting and Analytics Aggregates (RAA)	8.5.000.02
Genesys CX Insights (GCXI)	9.0.007.03

Historical Reporting Architecture

Reporting on Genesys Callback relies on the user-event mechanism to provide Callback-related metrics and requires Interaction Concentrator, Genesys Info Mart, and [Reporting and Analytics Aggregates \(RAA\)](#) to collect and organize data to produce a database from which Genesys CX Insights (GCXI) can rapidly extract the needed data.



1. Genesys Callback reports callback metrics through UserEvents to the configured DN. SCXML strategies that you load through the **templates** in the Service Management UI collect metrics and then pass the metrics as user data (KVPs) with two UserEvent events, one sent at the start of the session and another, at the end of the session. Genesys Info Mart has certain minimum requirements for the KVPs that must be sent. The out-of-box templates include these KVPs, as well as other KVPs that Genesys Info Mart requires for meaningful reporting. See **Genesys Info Mart KVP Requirements** for details.
2. Interaction Concentrator (ICON) stores the user data (KVPs) attached to these events into the G_CUSTOM_DATA_S table of the Interaction Database (IDB).
3. Genesys Info Mart transforms the data into the CALLBACK_FACT table of the Info Mart database; this format can be more quickly loaded into reports.
4. Reporting and Analytics Aggregates (RAA) aggregates the data; in other words, RAA summarizes and organizes the data from Genesys Info Mart in such a way that Genesys CX Insights (GCCI) can extract meaning.
5. Genesys CX Insights (GCCI) then presents two **out-of-box callback** reports: Callback Summary Report and Callback Details Report.

Callback Summary Report												
REPORT INFO												
Report Date(s):	1/1/2014 to 12/31/2014											
Queue:	ALL											
Callback Type:	ALL											
Channel:	ALL											
Tenant:	ALL											
Tenant: Environment												
Queue: Callback_VQ												
Callback Type	Channel	Day	Offered	Accepted	Declined		Attempted	Customer Connected		% Cancelled	% Abandoned	Success
					Count	%		Count	%			
IMMEDIATE	IVR	2014-09-18	1	1	0	0.00%	1	1	100.00%	0.00%	100.00%	0
SUB TOTAL:			1	1	0	0.00%	1	1	100.00%	0.00%	100.00%	0
Queue: Performance_VCB_VQ												
Callback Type	Channel	Day	Offered	Accepted	Declined		Attempted	Customer Connected		% Cancelled	% Abandoned	Success
					Count	%		Count	%			
IMMEDIATE	IVR	2014-09-19	391	364	27	6.91%	364	364	100.00%	0.00%	0.27%	363
SUB TOTAL:			391	364	27	6.91%	364	364	100.00%	0.00%	0.27%	363

For example, for reporting purposes, the following are some of the keys that GMS sends in UserEvents related to Outbound calls:

- _CB_T_SERVICE_START
- _CB_SERVICE_ID
- _CB_D_CALLBACK_OFFER
- _CB_N_CALLBACK_OFFERED
- _CB_T_CALLBACK_OFFERED
- _CB_T_CALLBACK_ACCEPTED
- _CB_T_CUSTOMER_CONNECTED
- _CB_N_IS_SNOOZED
- _CB_T_NEXT_REDIAL_ATTEMPT
- _CB_N_CALLBACK_MEDIA_ATTEMPTS
- _CB_T_LAST_DIAL_ATTEMPT
- _CB_N_AGENT_ADDED_TO_I_XN

The keys that GMS sends depend on the scenario. To get a complete list of the keys that might be sent, refer to the [Callback KVPs](#) reference on this page.

Important

If the `_CB_T_CALLBACK_OFFERED` and `_CB_T_CALLBACK_ACCEPTED` KVPs must be added to the original session that initiated the callback request, the callback request must include the `_originating_interaction_id` option. In this scenario, in the callback request,

set the `_originating_interaction_id` value to the interaction ID of the inbound call that is managed by the ORS session.

Genesys Info Mart KVP Requirements

The following KVPs are mandatory. Genesys Info Mart will not create a record for the callback event if the KVP is missing from the UserEvent.

- `_CB_SERVICE_ID`
- `_CB_T_SERVICE_START`
- `_CB_D_CALLBACK_OFFER`
- `_CB_N_CALLBACK_OFFERED`
- `_CB_T_CALLBACK_OFFERED`

Important

If the `_CB_T_CALLBACK_OFFERED` and `_CB_T_CALLBACK_ACCEPTED` KVPs must be added to the original session that initiated the callback request, the callback request must include the `_originating_interaction_id` option. In this scenario, in the callback request, set the `_originating_interaction_id` value to the interaction ID of the inbound call that is managed by the ORS session.

The following four KVPs need to be sent in both UserEvents and as call-based attached data in TEvents. The duplicated KVPs enable Genesys Info Mart to associate the callback event with interaction data.

- `_CB_T_CALLBACK_ACCEPTED`
- `_CB_T_SERVICE_START`
- `_CB_SERVICE_ID`
- `_CB_T_CUSTOMER_CONNECTED`

For meaningful reporting, Genesys Info Mart requires several other KVPs, depending on the callback scenario. See the [Callback KVPs](#) reference, below, for the complete list.

Important

- The `_CB_SERVICE_ID` is returned by the GMS API in response to the callback request.
- For Inbound Calls, where the in-queue callback offer was presented and accepted, `_CB_T_CALLBACK_ACCEPTED`, `_CB_T_SERVICE_START`, and `_CB_SERVICE_ID` must be attached at the time at which the callback was accepted.
- For Virtual and Outbound Calls, `_CB_T_CUSTOMER_CONNECTED` must be attached at the time at which the customer was connected.

Virtual Queues

As a best practice, Genesys recommends creating virtual queues associated with the following interaction types:

- Virtual Queue for Inbound calls—This queue is where the regular inbound calls are going to be reported. Those calls are callbacks that were not offered or, offered and rejected.
- Virtual Queue for Virtual callbacks—This queue is where the virtual callbacks are going to be waiting for an agent.
- Virtual Queue for Outbound calls—This is where the callback application will place the real outbound call when it gets confirmation that the right person is connected. The call is removed from this queue after it is successfully delivered to an agent or is abandoned by the customer.

Important

Virtual queues (VQ) that are used for reporting will make metrics effective, but they are not used for routing in this context.

Related Resources for Historical Reporting

You may also be interested in reading:

- The [Genesys Info Mart Physical Data Model documentation](#) for your RDBMS.
- The [Reporting and Analytics Aggregates Physical Data Model documentation](#) for your RDBMS.

Configure Historical Reporting

Important

Genesys Info Mart and Genesys CX Insights (GCXI) support for callback offered

through GMS is provided out-of-box, with no additional configuration required. To see callback data in GCXI reports, however, you need to modify the configuration for other products as explained in this section.

Configure a Reporting DN

Open Genesys Administrator or Configuration Manager and create a new DN of type **Trunk Group DN**. The name of the DN is used inside SCXML scripts, so it should be meaningful and recognizable. For example: Sip_Switch > DN > REPORTING

Configure your Callback Service

Callback Delayed

Search Table Reporting + Add New Delete Advanced Parameters Refresh

Name	Value	Description
<input type="checkbox"/> _customer_number		Request Parameter - Customer's phone number. Can be used to match the call with service data when call direction is set to USERORIGINATED. Also used when call direction is USERTERMINATED to establish connection with the customer
<input type="checkbox"/> _service	<input type="checkbox"/> callback	
<input type="checkbox"/> _type	ors	
▼ Reporting (5)		
<input type="checkbox"/> _rep_userevent_dn	REPORTING	DN to which the reporting user event is sent. If _rep_userevent_enable is set to true, a value for this parameter is required.
<input checked="" type="checkbox"/> _rep_userevent_enable	true	If set to true, callback data is reported via user events to the switch and dn specified (_rep_userevent_switch and _rep_userevent_dn).
<input type="checkbox"/> _rep_userevent_switch	Sip_Switch	Switch to which the dn (_rep_userevent_dn) belongs. If _rep_userevent_enable is set to true, a value for this parameter is required.
<input type="checkbox"/> _reporting_aggregator_url		URL to which the reporting events will be sent when Reporting Aggregator use enabled.
<input type="checkbox"/> _use_reporting_aggregator		If enabled, reporting events will be sent to the configured Reporting Aggregator URL.

Edit your callback service in **Callback and Mobile Engagement > Configured Services**, expand the **Reporting** section:

- Set the `_rep_useevent_enable` option to true to enable reporting.
- Set the `_rep_useevent_dn` option to the Trunk Group DN that you created previously, used as destination DN of the reporting events.
- Set the `_rep_useevent_switch` option to the Switch name where you created this DN. This is Switch used to report the events.

Configure Orchestration Server

In the connections of your Orchestration Server application, add the T-Server used to define the reporting Switch and DN in the GMS service configuration. For example, `Sip_Switch`.

Configure Interaction Concentrator

To make Callback reporting work, you need to configure Interaction Concentrator (ICON) for Voice. See [here](#) for details.

Set the KVP list

- Configure ICON to store the KVP data provided in the UserData section of EventUserEvents. ICON will store this data in the `G_CUSTOM_DATA_S` table of the Interaction Database (IDB):
`ICON > Options > custom-states/store-event-data=all`

By default, `store-event-data` is set to none.

- Configure ICON to store required duplicate KVP data provided in the UserData attribute of TEvents. ICON will store this data in the `G_USERDATA_HISTORY` table. To enable this storage, modify your `ccon_adata_spec.xml` file to capture the four TEvents KVPs described in the Callback KVPs reference [below](#):
 - `_CB_T_CALLBACK_ACCEPTED`
 - `_CB_T_CUSTOMER_CONNECTED`
 - `_CB_T_SERVICE_START`
 - `_CB_SERVICE_ID`

Tip

See the `ccon_adata_spec_GIM_example.xml` file in the Genesys Info Mart installation package for an example of the required modification.

Check Interaction Concentrator Connections

Make sure that Interaction Concentrator is connected to the T-Server that is servicing the switch specified in the Callback Service.

For example: Sip_Switch.

Start Interaction Concentrator and use logs to verify that it registered on the REPORTING DN.

Important

Interaction Concentrator does not produce historical records for virtual interactions.

Configure Reporting and Analytics Aggregates

Edit the Genesys Info Mart application to enable the `agg-feature\enable-callback` option:
`agg-feature\enable-callback=yes`

Tip

See [here](#) for details about the configuration of your RAA application.

Configure Workspace

Important: In a Callback use case with preview, reporting user data is attached to the call that appears on Agent Desktop (WDE). Once the callback is finished, from a GMS Callback point of view, the agent is managing wrap-up operations for the call and sends a user request to the reporting server using the callback user data. The reporting server sees this data as an additional reporting operation.

To avoid sending this additional reporting data, the agent desktop application can configure the following option in the `interaction-workspace` section:

```
interaction.disposition.use-attached-data=false
```

Verify Reporting Data

1. Run your scenario by triggering Genesys Mobile Services and Orchestration Server (ORS) APIs directly.
2. Make sure user events are being delivered to Interaction Concentrator applications by checking T-Server logs. You should see something like this:

```

00:34:20.757 Int 04543 Interaction message "RequestDistributeUserEvent" received from
516 ("OrchestrationServer")
-- Absent ThisDN, REPORTING was used
@00:34:20.7570 [0] 8.1.000.62 send_to_client: message EventACK
AttributeEventSequenceNumber      0000000000000ef8
AttributeCustomerID                'Environment'
AttributeTimeinuSecs               757000
AttributeTimeinSecs                1348817660 (00:34:20)
AttributeReferenceID               431
AttributeThisDN                    'REPORTING'
AttributeUserEvent                  RequestDistributeUserEvent
00:34:20.757 Trc 04542 EventACK sent to [516] (00000003 OrchestrationServer 192.168.27.50:40727)
@00:34:20.7570 [0] 8.1.000.62 distribute_user_event: message EventUserEvent
AttributeEventSequenceNumber      0000000000000ef9
AttributeCustomerID                'Environment'
AttributeTimeinuSecs               757000
AttributeTimeinSecs                1348817660 (00:34:20)
AttributeUserEvent                  EventUserEvent
AttributeUserData                  [347] 00 0c 00 00..
'gms_AgentAvailable'              '1348817660755'
'gms_AgentConnected'              ''
'gms_IxnCompleted'                 ''
'gms_ServiceName'                  'inbound-delay'
'gms_ServiceStartAt'               '1348817660553'
'gms_ServiceStoppedAt'             ''
'gms_SessionEventSeq'              3
'gms_SessionId'                    '65UA6ISSJH76R340BNDQ2DG0DG000036'
'gms_UserConnected'                 ''
'gms_UserId'                        ''
'gms_WaitingForAgent'               '1348817660744'
'gms_externalId'                   ''
AttributeANI                        '777'
AttributeDNIS                       '333'
AttributeReferenceID                 431
AttributeThisDN                      'REPORTING'
00:34:20.758 Trc 04542 EventUserEvent sent to [508] (0000000c Icon_Voice 192.168.27.50:42678)
00:34:20.758 Trc 04542 EventUserEvent sent to [588] (00000004 Stat_Server 192.168.27.50:40728)
00:34:20.758 Trc 04542 EventUserEvent sent to [592] (00000005 Universal_Routing_Server
192.168.27.50:40744)

```

3. Check your Interaction Concentrator logs and the G_CUSTOM_DATA_S table in Interaction Database and make sure that data is recorded properly.

For example, you should see in Interaction Concentrator logs:

```
00:39:19.569 Int 04543 Interaction message "EventUserEvent" received from 65200 ("SIP_Server@REPORTING")
00:39:19.751 Int 04543 Interaction message "EventUserEvent" received from 65200 ("SIP_Server@REPORTING")
00:39:19.766 Int 04543 Interaction message "EventUserEvent" received from 65200 ("SIP_Server@REPORTING")
00:39:19.987 Trc 25016 Persistent Queue GUD: transaction 10929 is committed. 5 records written
into the queue
00:39:19.987 Trc 25003 Database queue [GUD]: persistent queue transaction 10929 is being processed.
00:39:20.001 Trc 25004 Database queue [GUD]: persistent queue transaction 10929 is processed, committed
and removed. 5 records are written.
```

4. Optionally, you can also check the content of the CALLBACK_FACT table in the Info Mart database to make sure that the transformation process is correctly executed as well. For example, you can try the following query:

```
SELECT * FROM dbo. CALLBACK_FACT
```

	ADDED_TS	DS_AUDIT_KEY	EVENT_SEQUENCE	CREATE_AUDIT_KEY	TENANT_KEY	SERVICE_ID	FINAL_RECORD	EWI_READY_TO_START_DSN	EWI_WHEN_OFFERED	POS_READY_TO_START_DSN	POS_WHEN_OFFERED	CALLBACK_I
1	1465324803	9864	3174	10002	1	445-17a6a647-0bc4-498a-adda-c586322eff1f	0	0	0	0	0	0
2	1465324821	9864	3232	10002	1	445-17a6a647-0bc4-498a-adda-c586322eff1f	1	0	0	0	0	0
3	1465327890	9864	3952	10008	1	445-9338c861-6f94-48fc-a3a3-81ca343c3cc2	0	0	0	0	0	0
4	1465327908	9864	4011	10008	1	445-9338c861-6f94-48fc-a3a3-81ca343c3cc2	1	0	0	0	0	0
5	1465331525	9864	4842	10008	1	445-16a902f1-4447-40f3-968c-4125c7aa493c	0	0	0	0	0	0
6	1465331543	9864	4901	10008	1	445-16a902f1-4447-40f3-968c-4125c7aa493c	1	0	0	0	0	0
7	1465395725	9953	3169	10018	1	445-aa8095bc-c556-42b6-b045-a17b53cd04641	0	0	0	0	0	0
8	1465395743	9953	3228	10018	1	445-aa8095bc-c556-42b6-b045-a17b53cd04641	1	0	0	0	0	0
9	1465396434	9953	3465	10018	1	445-1a6e76f8-20fc-4cb5-82c2-7a31e59646c6	0	0	0	0	0	0
10	1465396452	9953	3524	10018	1	445-1a6e76f8-20fc-4cb5-82c2-7a31e59646c6	1	0	0	0	0	0
11	1465398916	9953	4122	10020	1	445-f348ed5-110d-467d-a525-4e7af2c39e9	0	0	0	0	0	0
12	1465398934	9953	4180	10020	1	445-f348ed5-110d-467d-a525-4e7af2c39e9	1	0	0	0	0	0
13	1465400440	9953	4583	10020	1	445-ab1af229-bb0a-4770-b44d-1680e23cf581	0	0	0	0	0	0
14	1465400458	9953	4641	10020	1	445-ab1af229-bb0a-4770-b44d-1680e23cf581	1	0	0	0	0	0
15	1465408962	9953	6465	10021	1	445-7bc50c56-ab8e-4e7e-9aa7-f2e52ca44abf	0	0	0	0	0	0
16	1465408980	9953	6523	10021	1	445-7bc50c56-ab8e-4e7e-9aa7-f2e52ca44abf	1	0	0	0	0	0
17	1465411643	9978	3209	10021	1	445-d4ad696-d70d-4092-b95c-a7cd76e13287	0	0	0	0	0	0
18	1465411665	9978	3268	10021	1	445-d4ad696-d70d-4092-b95c-a7cd76e13287	1	0	0	0	0	0
19	1465411915	10023	3416	10095	1	445-e1345be8-81d0-43eb-800e-bee831237109	0	0	0	0	0	0
20	1465411933	10023	3475	10095	1	445-e1345be8-81d0-43eb-800e-bee831237109	1	0	0	0	0	0
21	1465481078	50016	17609	50098	1	445-659ab04d-97e5-41be-9c74-5c1244a06205	0	0	0	0	0	0
22	1465481096	50016	17668	50165	1	445-659ab04d-97e5-41be-9c74-5c1244a06205	1	0	0	0	0	0

How to Pass Reporting KVPs of the Inbound Call in the Callback Request

Some historical reporting KVP values are known only by the IVR or application that requests the callback service. Including these KVPs in the historical reporting is optional. If you want to include them, the values can be passed in the HTTP request that **starts** the Callback service. The following is the list of the KVP parameters that can be passed in the HTTP request. Each maps to the corresponding **_CB_X** KVP.

- **_cb_t_callback_offered**
- **_cb_d_callback_offer**
- **_cb_ewt_when_callback_was_offered**
- **_cb_pos_when_callback_was_offered**
- **_cb_t_callback_accepted**
- **_cb_dim_channel**
- **_cb_dim_callback_offer_type**
- **_cb_dim_offer_timing**

- `_cb_n_callback_offers_per_session`
- `_cb_d_last_callback_offer`

Important

If the agent submits the completed reason in the disposition result, the system will set the reporting key `_CB_DISPOSITION` to the provided `COMPLETED` reason.

Important

If the `_cb_t_callback_offered` and `_cb_t_callback_accepted` KVPs must be added to the original session that initiated the callback request, the callback request must include the `_originating_interaction_id` option. In this scenario, in the callback request, set the `_originating_interaction_id` value to the interaction ID of the inbound call that is managed by the ORS session.

Reference: Callback KVPs

The following table describes the KVPs that, if sent by GMS in UserEvents, Genesys Info Mart uses to enable Callback reporting.

The following four KVPs must also be sent as call-based attached data.

- `_CB_SERVICE_ID`
- `_CB_T_SERVICE_START`
- `_CB_T_CALLBACK_ACCEPTED`
- `_CB_T_CUSTOMER_CONNECTED`

Important

The sample attached-data specification file in the Genesys Info Mart IP includes these four KVPs by default.

KVP	Description	Info Mart Database Target
_CB_TENANT_DBID	The Tenant DBID.	CALLBACK_FACT.TENANT_KEY
_CB_DISPOSITION	<p>Callback state using the format <state>.<sub state> where:</p> <ul style="list-style-type: none"> <state> can be set to: SCHEDULED, QUEUED, ROUTING, PROCESSING, COMPLETED. <sub state> can be set: REDIAL_LIMIT_REACHED, CANCELLED, AGENT, ABANDONED_IN_QUEUE, REJECTED, PUSH_SEND, PUSH_DELIVERY_CONFIRMED, PUSH_SEND_ERROR, FAILED, CONNECTED, TRANSFERRED_TO_RP. 	CALLBACK_DIM_3.DISPOSITION (referenced through CALLBACK_FACT.CALLBACK_DIM_3_KEY)
_CB_SERVICE_ID*	The ID of the callback service request. Depending on the scenario, the value equals the ID of the GMS service instance or ID of the ORS session.	CALLBACK_FACT.SERVICE_ID
_CB_ORIGINATION_I_XN_ID Introduced: GMS 8.5.200.07	The ID of the inbound call where the callback was originally offered and accepted. You must pass the <code>_cb_origination_ixn_id</code> parameter in your Start Callback query when creating a callback request. If you do not pass the <code>_cb_origination_ixn_id</code> parameter, the value of <code>_CB_ORIGINATION_I_XN_ID</code> will be undefined. For chat scenarios, this ID should be the chat interaction ID.	CALLBACK_FACT.ORIGINATION_I_XN_ID
_CB_FIRST_OUT_I_XN_ID Introduced: GMS 8.5.200.07	The call ID of the first outbound call that the callback service created.	CALLBACK_FACT.FIRST_OUT_I_XN_ID
_CB_LAST_OUT_I_XN_ID Introduced: GMS 8.5.200.07	The call ID of the last outbound call that the callback service created.	CALLBACK_FACT.LAST_OUT_I_XN_ID

KVP	Description	Info Mart Database Target
<p>_CB_DIAL_1_RESULT</p> <p>Introduced: GMS 8.5.200.07</p>	<p>The result of the first callback dialing attempt. One of the following values:</p> <ul style="list-style-type: none"> • CREATE_CALL_ERROR • BUSY • NO_ANSWER • ANSWERING_MACHINE • ERROR_TONE • FAX • PERSON • CONNECTED • FAILED_TO_ESTABLISH_CUSTOMER_ORIGINATED_MEDIA • PUSH_DELIVERY_CONFIRMED • PUSH_SEND_ERROR • PUSH_DELIVERY_NOT_CONFIRMED • USERORIGINATED_CONNECTED <p>Notes: FAILED_TO_ESTABLISH_CUSTOMER_ORIGINATED_MEDIA is a result that must be reported by the user application; otherwise, there is no CTI data that will enable Genesys Callback to identify this result.</p>	<p>CALLBACK_DIAL_RESULTS.DIAL_1_RESULT (referenced through CALLBACK_FACT.CALLBACK_DIAL_RESULTS_KEY)</p>
<p>_CB_DIAL_2_RESULT</p> <p>Introduced: GMS 8.5.200.07</p>	<p>The result of the second callback dialing attempt. See _CB_DIAL_1_RESULT for possible values.</p>	<p>CALLBACK_DIAL_RESULTS.DIAL_2_RESULT (referenced through CALLBACK_FACT.CALLBACK_DIAL_RESULTS_KEY)</p>
<p>_CB_DIAL_3_RESULT</p>	<p>The result of the third callback dialing attempt. See _CB_DIAL_1_RESULT for possible values.</p>	<p>CALLBACK_DIAL_RESULTS.DIAL_3_RESULT (referenced through</p>

KVP	Description	Info Mart Database Target
Introduced: GMS 8.5.200.07		CALLBACK_FACT.CALLBACK_DIAL_RESULTS_KEY)
_CB_DIAL_4_RESULT Introduced: GMS 8.5.200.07	The result of the fourth callback dialing attempt. See _CB_DIAL_1_RESULT for possible values.	CALLBACK_DIAL_RESULTS.DIAL_4_RESULT (referenced through CALLBACK_FACT.CALLBACK_DIAL_RESULTS_KEY)
_CB_DIAL_5_RESULT Introduced: GMS 8.5.200.07	The result of the fifth callback dialing attempt. See _CB_DIAL_1_RESULT for possible values.	CALLBACK_DIAL_RESULTS.DIAL_5_RESULT (referenced through CALLBACK_FACT.CALLBACK_DIAL_RESULTS_KEY)
_CB_T_DIAL_1 Introduced: GMS 8.5.200.07	UTC Timestamp of the first dialing attempt.	CALLBACK_FACT.DIAL_1_TS
_CB_T_DIAL_2 Introduced: GMS 8.5.200.07	UTC Timestamp of the second dialing attempt.	CALLBACK_FACT.DIAL_2_TS
_CB_T_DIAL_3 Introduced: GMS 8.5.200.07	UTC Timestamp of the third dialing attempt.	CALLBACK_FACT.DIAL_3_TS
_CB_T_DIAL_4 Introduced: GMS 8.5.200.07	UTC Timestamp of the fourth dialing attempt.	CALLBACK_FACT.DIAL_4_TS
_CB_T_DIAL_5 Introduced: GMS 8.5.200.07	UTC Timestamp of the fifth dialing attempt.	CALLBACK_FACT.DIAL_5_TS
_CB_IXN_START_IGNOREING_AVAILABILITY	For premise callback, _CB_IXN_START_IGNOREING_AVAILABILITY will	CALLBACK_DIM_4.DIAL_IGNOREING_AVAILABILITY

KVP	Description	Info Mart Database Target
Introduced: GMS 8.5.200.07	always be 0.	
_CB_FINAL_RECORD	Indicates whether this is a final record about this callback service: 0 = No, 1 = Yes.	CALLBACK_FACT.FINAL_RECORD
_CB_EWT_WHEN_READY_TO_START_MEDIA_I_XN	The value of Expected Wait Time (EWT), in seconds, for the service request when the contact center was ready to start the first callback interaction, such as an outbound dialing attempt.	CALLBACK_FACT.EWT_READY_TO_START_I_XN
_CB_EWT_WHEN_CALLBACK_WAS_OFFERED	The value of EWT, in seconds, at the time the callback was offered.	CALLBACK_FACT.EWT_WHEN_OFFERED
_CB_POS_WHEN_READY_TO_START_MEDIA_I_XN	The customer position in the queue when the contact center was ready to start the first callback interaction, such as an outbound dialing attempt.	CALLBACK_FACT.POS_READY_TO_START_I_XN
_CB_POS_WHEN_CALLBACK_WAS_OFFERED	The customer position in the queue when callback was offered.	CALLBACK_FACT.POS_WHEN_OFFERED
_CB_D_CALLBACK_OFFER	The duration of the callback offer, in seconds.	CALLBACK_FACT.CALLBACK_OFFER_TIME
_CB_OFFER_EWT_INBOUND_VQ Introduced: GMS 8.5.111.04	Estimated Wait Time for the queue where rejected calls and not offered callbacks are being placed. This value is identical to _CB_EWT_WHEN_CALLBACK_WAS_OFFERED if the same Virtual Queue is used to place accepted callbacks.	CALLBACK_FACT.EWT_WHEN_REJECTED
_CB_N_ABANDONED_DURING_CALLBACK_OFFER Introduced: GMS 8.5.111.04	Indicates whether the caller dropped the call without explicitly accepting or rejecting the callback offer: 0 = No, 1 = Yes.	CALLBACK_DIM_4.ABANDONED_DURING_CB_OFFER (referenced through CALLBACK_FACT.CALLBACK_DIM_4_KEY)
_CB_CUSTOMER_ANI	ANI of the customer for in-queue scenarios. This value can match _CB_CUSTOMER_PHONE_NUMBER if the same	CALLBACK_FACT.CUSTOMER_ANI

KVP	Description	Info Mart Database Target
Introduced: GMS 8.5.111.04	number is confirmed or entered. Could also be empty if the ANI is not detected.	
_CB_T_SERVICE_END Introduced: GMS 8.5.111.04	UTC timestamp for when service was completed or terminated.	CALLBACK_FACT.SERVICE_END_TS
_CB_D_CUSTOMER_WAITED_BEFORE_OFFER Introduced: GMS 8.5.106.14	The amount of time, in seconds, the customer waited in the queue before a callback was offered.	CALLBACK_FACT.WAITED_BEFORE_OFFER_TIME
_CB_D_WAITING_FOR_AGENT_OFFLINE	The amount of time, in seconds, the customer was waiting offline for an agent to become available.	CALLBACK_FACT.WAIT_AGENT_OFFLINE_TIME
_CB_D_ESTABLISH_MEDIA_I_XN	The amount of time, in seconds, it took to establish the callback interaction, such as an outbound call.	CALLBACK_FACT.ESTABLISH_MEDIA_I_XN_TIME
_CB_D_CUSTOMER_CONNECTED_WAITING_FOR_AGENT	The amount of time, in seconds, the customer was waiting to be connected to the agent after the callback interaction was established.	CALLBACK_FACT.CONN_WAITING_AGENT_TIME
_CB_T_CALLBACK_ACCEPTED*	The UTC timestamp when the callback offer was accepted.	CALLBACK_FACT.CALLBACK_ACCEPTED_TS
_CB_T_CALLBACK_OFFERED	The UTC timestamp when the callback was offered.	CALLBACK_FACT.CALLBACK_OFFERED_TS
_CB_T_READY_TO_START_MEDIA_I_XN	The UTC timestamp when the contact center was ready to start the callback interaction. The value matches the time of either an outbound dialing attempt or a push notification prompting the customer to start a call or chat session. Note: Set this value only once, before the first dial attempt.	CALLBACK_FACT.READY_START_MEDIA_I_XN_TS
_CB_T_CUSTOMER_CONNECTED*	The UTC timestamp when the customer was reconnected to the contact center and started	CALLBACK_FACT.CUSTOMER_CONNECTED_TS

KVP	Description	Info Mart Database Target
	waiting for an agent to be connected.	
_CB_N_AGENT_ADDED_TO_I_XN	Indicates whether the agent was successfully added to the callback interaction: 0 = No, 1 = Yes.	CALLBACK_FACT.AGENT_ADDED_TO_I_XN
_CB_N_TRANSFER_TO_AGENT_FAILED	Number of times the callback interaction failed to transfer to the agent.	CALLBACK_FACT.XFER_TO_AGENT_FAILED
_CB_N_CUSTOMER_ABANDONED_WHILE_WAITING_FOR_AGENT	Indicates whether the customer abandoned the callback interaction while waiting to be connected to an agent: 0 = No, 1 = Yes.	CALLBACK_FACT.ABANDONED_WAITING
_CB_N_TIMEOUT_WHILE_WAITING_FOR_AGENT	Indicates whether the customer was disconnected because the timeout for waiting for an agent was reached: 0 = No, 1 = Yes.	CALLBACK_FACT.TIMEOUT_WAITING
_CB_N_I_XN_REQ_AGENT	Indicates whether the interaction required agent assistance: 0 = No, 1 = Yes.	CALLBACK_FACT.I_XN_REQ_AGENT
_CB_N_CALLBACK_OFFERED	Indicates whether callback was offered, at least once, during the session: 0 = No, 1 = Yes.	CALLBACK_FACT.CALLBACK_OFFERED
_CB_N_CALLBACK_ACCEPTED	Indicates whether a callback offer was accepted: 0 = No, 1 = Yes.	CALLBACK_FACT.CALLBACK_ACCEPTED
_CB_N_CALLBACK_MEDIA_ATTEMPTS	The total number of callback attempts or notifications, both successful and unsuccessful.	CALLBACK_FACT.CALLBACK_ATTEMPTS
_CB_T_SERVICE_START*	The UTC timestamp when the callback service started. This value represents either the time of the callback request or the time that the callback offer was played, depending on deployment.	CALLBACK_FACT.SERVICE_START_TS, CALLBACK_FACT.START_DATE_TIME_KEY
_CB_DIM_VQ_DBID	The DBID of the virtual queue used to find the target agent. Genesys Info Mart uses this value in combination to identify the RESOURCE_KEY to use.	CALLBACK_FACT.RESOURCE_KEY
VQ_CFG_TYPE_ID	The configuration type ID of the virtual queue used to find the target agent. Genesys Info Mart uses this value in combination to identify the	CALLBACK_FACT.RESOURCE_KEY

KVP	Description	Info Mart Database Target
	RESOURCE_KEY to use.	
VQ_CFG_TYPE	The configuration type of the virtual queue used to find the target agent. Genesys Info Mart uses this value in combination to identify the RESOURCE_KEY to use.	CALLBACK_FACT.RESOURCE_KEY
_CB_DIM_VQ	The virtual queue used to find the target agent. Genesys Info Mart uses this value in combination to identify the RESOURCE_KEY to use.	CALLBACK_FACT.RESOURCE_KEY
_CB_DIM_CHANNEL	<p>The interaction channel from which the callback originated. One of the following values:</p> <ul style="list-style-type: none"> • IVR • WEB • MOBILE 	CALLBACK_DIM_1.CHANNEL (referenced through CALLBACK_FACT.CALLBACK_DIM_1_KEY)
_CB_DIM_CALLBACK_OFFER_TYPE	<p>The type of callback offer that was presented to the customer. For example, after business hours, SCHEDULED is the only available option; during business hours, business rules might allow only the WAIT_FOR_AGENT option or a combination of SCHEDULED and WAIT_FOR_AGENT. One of the following values:</p> <ul style="list-style-type: none"> • SCHEDULED • WAIT_FOR_AGENT • COMBINED_SCHEDULED_AND_WAIT_FOR_AGENT • IMMEDIATE 	CALLBACK_DIM_1.CALLBACK_OFFER_TYPE (referenced through CALLBACK_FACT.CALLBACK_DIM_1_KEY)
_CB_DIM_TYPE	The type of callback the customer requested. One of the following values:	CALLBACK_DIM_1.CALLBACK_TYPE (referenced through CALLBACK_FACT.CALLBACK_DIM_1_KEY)

KVP	Description	Info Mart Database Target
	<ul style="list-style-type: none"> • IMMEDIATE - The interaction is created right away while the customer is waiting for the agent (in an online chat session or waiting for a voice call). • WAIT_FOR_AGENT - The interaction is delayed until the agent is about to become available or actually becomes available (as in an agent first scenario). • SCHEDULED - The time for the callback interaction is negotiated with the customer. 	
_CB_DIM_CONNECT_ORDER	<p>The order in which the final callback interaction was connected. One of the following values:</p> <ul style="list-style-type: none"> • CUSTOMER_FIRST • AGENT_FIRST_PREVIEW • AGENT_FIRST_NO_PREVIEW 	CALLBACK_DIM_1.CONNECT_ORDER (referenced through CALLBACK_FACT.CALLBACK_DIM_1_KEY)
_CB_DIM_DIAL_DIALOG_RESULT	<p>The result of the final dialog for the callback. One of the following values:</p> <ul style="list-style-type: none"> • RIGHT_PERSON • RESCHEDULED • CANCELLED • TRANSFERRED_TO_RP • PERSON • CANCEL • ERROR_TONE 	CALLBACK_DIM_2.DIAL_DIALOG_RESULT (referenced through CALLBACK_FACT.CALLBACK_DIM_2_KEY)

KVP	Description	Info Mart Database Target
	<p>Important: If an error occurs during the callback outbound call, the value of <code>_CB_DIM_FINAL_DIAL_RESULT</code> might overlap with <code>_CB_DIM_DIAL_DIALOG_RESULT</code>.</p>	
<p><code>_CB_DIM_CALL_DIRECTION</code></p>	<p>The direction of the final callback interaction. One of the following values:</p> <ul style="list-style-type: none"> • <code>CUSTOMER_TERMINATED</code> - Outbound Callback scenarios in which the contact center is dialing out to the customer's number. • <code>CUSTOMER_ORIGINATED</code> - Inbound Callback scenarios in which the contact center notifies the customer-facing application that it is time for the callback interaction, after which the application creates the interaction (such as a call or chat), obtaining the phone number if necessary. In this scenario, a customer call comes into the contact center as a regular inbound call, but it is recognized as the callback interaction. 	<p><code>CALLBACK_DIM_2.CALL_DIRECTION</code> (referenced through <code>CALLBACK_FACT.CALLBACK_DIM_2_KEY</code>)</p>
<p><code>_CB_DIM_FINAL_DIAL_RESULT</code></p>	<p>The result of the final callback dialing attempt. One of the following values:</p> <ul style="list-style-type: none"> • <code>CREATE_CALL_ERROR</code> • <code>BUSY</code> • <code>NO_ANSWER</code> • <code>ANSWERING_MACHINE</code> • <code>ERROR_TONE</code> • <code>FAX</code> • <code>PERSON</code> 	<p><code>CALLBACK_DIM_2.FINAL_DIAL_RESULT</code> (referenced through <code>CALLBACK_FACT.CALLBACK_DIM_2_KEY</code>)</p>

KVP	Description	Info Mart Database Target
	<ul style="list-style-type: none"> • CANCEL • CONNECTED • FAILED_TO_ESTABLISH_CUSTOMER_ORIGINATED_MEDIA • PUSH_DELIVERY_CONFIRMED • PUSH_SEND_ERROR • PUSH_DELIVERY_NOT_CONFIRMED • USERORIGINATED_CONNECTED • REDIAL_LIMIT_REACHED • ABANDONED_IN_QUEUE • FAIL • UNKNOWN • RESCHEDULED • FAIL_FAX_REACHED <p>Notes:</p> <p>1. FAILED_TO_ESTABLISH_CUSTOMER_ORIGINATED_MEDIA is a result that must be reported by the user application; otherwise, there is no CTI data that will enable Genesys Callback to identify this result.</p> <p>2. CANCEL is set when the on_dial plugin returned action=CANCEL.</p>	
<p>_CB_DIM_OFFER_TIMING</p>	<p>Specifies whether the callback offer was made during operational (business) or non-operational hours. One of the following values:</p> <ul style="list-style-type: none"> • ON-HOURS 	<p>CALLBACK_DIM_2.OFFER_TIMING (referenced through CALLBACK_FACT.CALLBACK_DIM_2_KEY)</p>

KVP	Description	Info Mart Database Target
	<ul style="list-style-type: none"> • OFF-HOURS 	
_CB_DIM_FINAL_TARGET	The routing target that was used to find the agent.	CALLBACK_DIM_3.FINAL_TARGET (referenced through CALLBACK_FACT.CALLBACK_DIM_3_KEY)
_CB_OR_S_SESSION_ID Introduced: GMS 8.5.114.09	The Orchestration Server (ORS) session ID used to manage the callback. If multiple sessions were used (for example, because an ORS session terminated unexpectedly during the callback), the last session ID is reported.	CALLBACK_FACT.ORS_SESSION_ID
_CB_EWT_WHEN_READY_TO_START_LAST_MEDIA_I_XN Introduced: GMS 8.5.200.07	Estimated Wait Time in seconds when the last dial attempt was made or the last push notification sent.	CALLBACK_FACT.EWT_WHEN_LAST_DIAL
_CB_POS_WHEN_READY_TO_START_LAST_MEDIA_I_XN Introduced: GMS 8.5.200.07	Position in queue when the last dial attempt was made or the last push notification sent.	CALLBACK_FACT.POS_WHEN_LAST_DIAL
_CB_PRIORITY_WHEN_CALLBACK_ACCEPTED Introduced: GMS 8.5.200.07	Priority of the interaction (real or virtual) when the callback offer was accepted.	CALLBACK_FACT.PRIORITY_WHEN_CB_ACCEPTED
_CB_PRIORITY_WHEN_CUSTOMER_CONNECTED Introduced: GMS 8.5.200.07	Priority of the virtual interaction when the customer was connected.	CALLBACK_FACT.PRIORITY_WHEN_C_CONNECTED
_CB_PRIORITY_AT_THE_END_OF_ONLINE_WAIT Introduced: GMS 8.5.200.07	Priority of the virtual interaction when the customer was connected to the agent. If the customer abandoned while waiting in queue, then this value is the priority of the call when the customer disconnected.	CALLBACK_FACT.PRIORITY_WHEN_A_CONNECTED

KVP	Description	Info Mart Database Target
<code>_CB_EWT_THRESHOLD_WHEN_OFFERED</code> Introduced: GMS 8.5.200.07	Value of the EWT threshold used to decide whether the callback offer should be made or not. Pass this value as an argument of the application that is responsible for making the callback offer.	CALLBACK_FACT.EWT_THRESHOLD_WHEN_OFFERED

*This KVP must be sent twice -- as call-based attached data in a TEvent and as UserEvent-based user data.

Implementing ADDP

The Genesys Advanced Disconnect Detection Protocol (ADDP) protocol is a Keep-Alive protocol between servers. ADDP is embedded into the underlying connection to a server (a protocol in another protocol). The Keep-Alive protocol is shared: client-to-server, and server-to-client. ADDP traces can be activated at the client-side only, server-side only, both sides, or no traces. The protocol is responsible for sending and receiving Keep-Alive packets, setting and canceling timers waiting for the next packet, and sending an appropriate event to the application if the connection is lost.

Configuring ADDP

To enable ADDP between two applications, specify `addp` as the Connection Protocol when configuring the connection between applications; also, set values for the Local Timeout, Remote Timeout, and Trace Mode properties (off, client side, server side, both). For more information, refer to the [Framework Configuration Manager Help](#). For complete instructions on configuring ADDP between two applications using either Genesys Administrator or Configuration Manager, refer to *Appendix A* in the [Framework Deployment Guide](#).

Implementing IPv6

Overview

Internet Protocol version 6, commonly known as IPv6, is a network layer protocol for packet-switched inter-networks. It is designated as the successor of IPv4, the current version of the Internet Protocol, for general use on the Internet. Genesys Mobile Services (GMS) supports IPv6.

Note: Refer to the *Framework 8.1 Deployment Guide* for more information about IPv6. In particular, see the IPv6 Appendix.

Genesys Environment Variable

When to use an environment variable:

- If an IPv6 connection is to be established before an application is able to, or must, read information from Configuration Server.
- If you want all Genesys applications on the same host to support IPv6. You only have to configure the host once, rather than configure each application on that host individually. In addition, this host-level setting will override any application-level setting.

Set the environment variable `GCTI_CONN_IPV6_ON` to true, represented by any non-zero value, to enable IPv6. The default value of this environment variable is false (0), indicating that IPv6 support is disabled. This default value ensures backward compatibility.

Genesys Configuration Option

Do **not** use this procedure if:

- An IPv6 connection is to be established before an application is able to, or must, read information from Configuration Server.
- You want all Genesys applications on a specific host to support IPv6, and you want to set it only once.

Using either Genesys Administrator or Configuration Manager, set the following configuration option in the common section of the options of the component's Application object:

enable-ipv6

Valid Values:

0 - Off (default), IPv6 support is disabled.

1 - On, IPv6 support is enabled.

Mixed IPv4 and IPv6 Environments

You can configure IPv6 and IPv4 in the same environment. In this mixed environment, you can configure connections with servers that support IPv4, IPv6, and both.

To configure this choice, use the Transport parameter, `ip-version` on the Advanced tab of the Connection Info dialog box for the connection:

`ip-version`

Default Value: 4,6

Valid Values: 4,6 and 6,4

Specifies the order in which IPv4 (4) and IPv6 (6) are used for the connection with a server that has a mixed IPv4/IPv6 configuration. This parameter has no effect if the environment variable `GCTI_CONN_IPV6_ON` or the option `enable-ipv6` is set to 0. Management Framework components do not support this option. This option also has no affect on connections to Configuration Server that are established before the option value can be read from the Configuration Database.

Java Properties

You can use the following system properties:

- `java.net.preferIPv4Stack`
- `java.net.preferIPv6Addresses`

Limitations for Linux Systems

If you are using Centos or RedHat, you will need to disable IPv6 connections. You can also check official recommendations for [Centos 6](#) and [Centos 7](#).

To force GMS to use IPv4, edit the `launcher.xml` file and add the following lines:

```
<parameter name="jvm_option04" displayName="jvm_option04" mandatory="true"
hidden="true" readOnly="true">
  <description><![CDATA[Network disable IPv6]]></description>
  <valid-description><![CDATA[]]></valid-description>
  <effective-description/>
  <format type="string" default="-Djava.net.preferIPv4Stack=true" />
  <validation></validation>
</parameter>
```

GMS Alarms

The following alarms can be raised by Genesys Mobile Services based on system status/configuration:

Resources Configuration Alarm (EventId 2000)	This alarm is raised by GMS when the server detects a problem on resources configuration (Duplicated DN on same/different Groups)
No more resources Alarm (EventId 2001)	This alarm is raised by GMS when no more resources are available in GMS (LOCAL or CLUSTER strategy).
web_port option Alarm (EventId 2002)	This alarm is raised by GMS when at startup, a GMS is not available on web_port.
NO_JDK Alarm (EventId 2003) Introduced in 8.5.006.09	This alarm is raised when GMS is not started using a JDK (used to compile DFM files.)
Config Server: Handle Connection Failure Alarm (EventId 2004) Introduced in 8.5.114.09	GMS raises this alarm when it fails to connect the Configuration Server. In this scenario, the GMS alarm displays at which time GMS will try to connect again.
Config Server: Handle Registration Failure Alarm (EventId 2005) Introduced in 8.5.114.09	GMS raises this alarm when it fails to register on the Configuration Server. In this scenario, the GMS alarm displays at which time GMS will try to register again.

To create Alarms, refer to [Creating the SCS Alarm Conditions](#).

Configuring HTTP Caching

By default, HTTP Caching is disabled in GMS.

To configure caching, you must *first* start and stop GMS. **Tell me why**

The configuration file is embedded in the `gms.war` file; if you don't want to start and stop GMS, you must unzip the war file, edit the configuration file as described below, and zip the files in `gms.war`.

Then, you can configure caching in GMS by editing the following file, located in the WEB-INF directory of your GMS installation: `<GMS_HOME>/work/.../webapp/WEB-INF/servlet-context.xml`

You should set the `cacheSeconds` property value to 3600 for instance (default is 0).

```
<bean id="webContentInterceptor"
class="org.springframework.web.servlet.mvc.WebContentInterceptor">
<property name="cacheSeconds" value="3600" />
<property name="useExpiresHeader" value="true" />
<property name="useCacheControlHeader" value="true" />
<property name="useCacheControlNoStore" value="true" />
</bean>
```

Then, you should restart GMS.

Tip

You must set this option to an appropriate value in case of heavy traffic between ORS and GMS.

Enable Logging in UTF-8 Environment

To enable logging in UTF-8 environment for GMS, set the following options as JVM parameters in the <GMS_HOME>/launcher.xml file.

```
<parameter name="java_file_encoding" displayName="java_file_encoding" mandatory="true"
hidden="true" readOnly="true">
<description><![CDATA[Logging: enable UTF-8 general encoding]]></description>
<valid-description><![CDATA[]]></valid-description>
<effective-description/>
<format type="string" default="-Dfile.encoding=UTF-8" />
<validation></validation>
</parameter>
<parameter name="genesys_psd_k_multibytes" displayName="genesys_psd_k_multibytes"
mandatory="true" hidden="true" readOnly="true">
<description><![CDATA[Logging: disable multibytes support for PSDK]]></description>
<valid-description><![CDATA[]]></valid-description>
<effective-description/>
<format type="string" default="-Dtkv.multibytes=false" />
<validation></validation>
</parameter>
```

Admin UI Internationalization

This page describes how you can extend the languages supported in the Admin UI for [Internationalization and Localization](#).

Prerequisites

- GMS is installed
- GMS servers are stopped

Step 1. Create a locale.js File

1. Check the standardized nomenclature used to classify languages in the [list of ISO 639-1 codes](#) to find the four-letter ISO language code matching your language.
2. In the <GMS_HOME>/files directory, create a subdirectory using the nomenclature name, for example, br-FR.
3. Download the locale.js [file example](#) or copy one of the <GMS_HOME>/files/xx-XX/locale.js files, for example fr-CA/locale.js, to the new folder.
4. Edit this file to translate messages appropriately. For further information, check the [section reference](#) below.

Important

Do no edit keys. Modify values only.

```
Main_Login: {
  title: "Brezhoneg",
  APP_NAME_GMS_LOGIN: "GMS Management UI - Connexion",
  APP_NAME_GES_LOGIN: "GES Management UI - Connexion",
  loginFormTitle: {
    username:"arveriad",
    password:"Mot de passe",
    language:"Yezh",
    button: "Kennaskañ",
    forgotPassword:"Mot de passe oublié ?",
    page:"Degemer mad",
    copyright:"2021 Genesys Telecommunications",
    version:"1.0.0",
    termOfUse:"Conditions générales d'utilisation",
    privacyPolicy:"Politique de confidentialité"
```

```
    },
    errorMessages: {
      incorrectLogin: "Nom d'utilisateur et/ou mot de passe incorrect(s).",
      frameWorkConnectionError: "Perte de la connexion au serveur de
configuration.",
      emptyField: "Remplissez tous les champs obligatoires.",
      connectionError: "Erreur de connexion au serveur",
      maxSessions: "Cet utilisateur a atteint le nombre maximum de connexions
simultanées autorisées."
    }
  },
  ...
}
```

Step 2. Enable the new Language

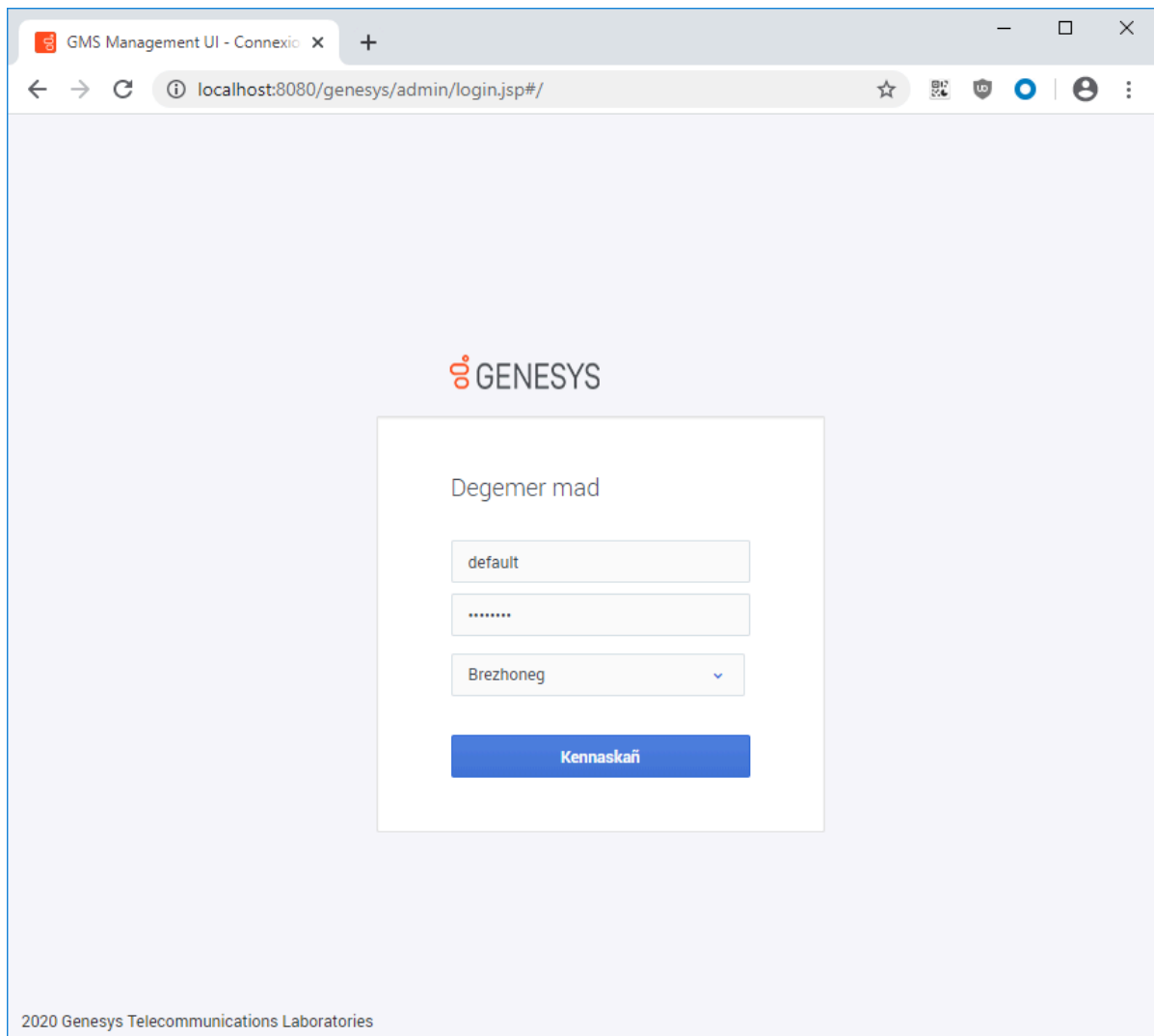
1. Edit the `files/locale.js` and add your language nomenclature to the list as shown in the below example for `br-FR`.

```
...
"ja-JP" : true,
"br-FR" : true
...
```

Important

To disable a language, set the language value to `false`.

2. Restart GMS.
3. To test the new language, change your web browser's language. For example, in Chrome, navigate through **Settings > Advanced > Languages > Language > Add Languages**. There, move the new language to the top of the list. For example, select `br[-FR]`.
4. Open the GMS Admin UI. The **Welcome** page is displayed by default with the new language. If not, select the language in the dropdown list.



Reference List of the locale.js Sections

UI Page	File Section
Welcome page	Main_Login
Home view, services hub page	Common, HOMEVIEW
MonitorView, services configuration page	Common, ModelManagement, TemplateChangeDialog, DownloadDfmTemplate, PatternGroupView, ICalenderView, ResourcesView, RemoveResourceGroup, NewResourceGroup, RemovePatternGroup, ReloadServiceTemplate, ServiceTemplateView, ServiceTemplatesView, StatisticsView, NewPatternGroupDialog,

UI Page	File Section
	SettingView, ServiceView, ServicesView, RemoveService, RemoveSetting, NewService, Provisioning, NavigationView, Error, Reporting, System, DATETIME_FORMATS, OfficeHourView, ConfigView, CapacityView, SampleView, MonitorView, ConfiguredServicesView, ServiceTemplatesViewNew, ErrorModal, NavBar
Callback view	Common, CallbackView_ANGULAR, DATETIME_FORMATS, ErrorModal, NavBar
Context Service view and Timeline web page	Common, ContextServices, HOMEVIEW, TimelineGeneral, FormControl, KPIInfo, RunQuery, DATETIME_FORMATS, ErrorModal, NavBar

Configuration Options Reference

Important

This page provides descriptions and explanations of Genesys Mobile Services-specific options. Refer to the [Genesys Mobile Services Options Reference](#) for a full list of options.

Default Configuration Sections

By default, the **Options** tab for your Genesys Mobile Services Application object contains several sections with configuration values.

Services Configuration Sections

Every service you want to provide using this instance of Genesys Mobile Services can have a custom entry created using this format. The default installation provides two examples:

- service.request-interaction
- service.query

The [Services and Tools UI](#) will create these options for you.

[chat] Section

This section configures additional chat parameters for the GMS application.

Required Chat Server Options (HA)

Section: endpoints:1	
Option Name	Option Value
default	Chat In
Section: settings	
Option Name	Option Value
session-restoration-mode	simple
transcript-auto-save	2

[config] Section

Added in 8.5.209.02

Options in the **config** section enable you to manage configuration objects' caching in GMS. By default, only Route Point objects are loaded at startup; other objects are not loaded unless you enable the corresponding option. See the [\[config\]](#) section in the *Genesys Mobile Engagement Configuration Options Reference* guide for a detailed list of options.

[log] Section

According to your application and your needs, you can activate the following additional logs for your GMS application. By default, all these log options are set to false.

For example:

```
[log]
ChatService=false
ClusterService=false
DataDepotService=false
DistributedJobExecutor=false
DistributedJobQueue=false
SharedService=false
[callback]
log-background-activity=false
```

[log-filter] and [log-filter-data] Section

These sections enable to hide selected attached data in Logs. See [Hiding Selected Data in Logs](#) for details and examples about filters.

- Changes take effect: Immediately.

Option	Default	Description
<KV-List-key> Optional	N/A	Describes the filter to hide parts or the totality of the associated string values in the logs. <ul style="list-style-type: none"> • copy • hide • hide-first,<n> • hide-last,<n>

Option	Default	Description
		<ul style="list-style-type: none"> • unhide-first,<n> • unhide-last,<n> • skip • tag (*) • tag() (*) • tag(,) (*) • tag(<custom_prefix>, <custom_postfix>); for instance tag(!!!,!!!) <p>(*) use default system prefix/postfix <# #></p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Important</p> <p>Passwords are automatically hidden in GMS logs.</p> </div>

[log-hidden-attributes] Section

Introduced: 8.5.200

Use this section to hide selected Genesys internal message attributes in Logs. See [Hiding Selected Data in Logs](#) for details.

Changes Take Effect: Immediately

Option	Description
<ProtocolName>.<MessageName>	Comma-separated list of message attributes.
<ProtocolName>.<Complex AttributeName>	Comma-separated list of complex attribute's attributes.

Here is a configuration sample, which hides the content of the chat text from the logs while GMS communicates with the Chat Server:

```
[log-hidden-attributes]
FlexChat.EventInfo=Text
FlexChat.MessageText=Text
FlexChat.NoticeText=Text
```

[notification] Section

unsubscribe-delay

Section: notification

Default Value: 0

Valid Values: Any positive integer

Changes Take Effect: Immediately

Introduced: 8.5.109.05

Time in seconds to wait for deleting notification subscriptions. In scenarios where the publish notification and the delete subscription requests are received concurrently, the subscription may be deleted before the notification gets published. If you set this option to a value greater than 0, you will force GMS to wait for the specified duration before deleting the subscription and this will allow the pending push notifications to be sent out.

[ors] Section

Modified in 8.5.107

_ors_lb_strategy

Section: ors

Default Value: circular

Valid Values: circular, linear

Changes Take Effect: After restart

Added in: 8.5.107

Strategy for ORS added to the **Connections** tab of the GMS application.

enable_ors_loadbalancer

Section: ors

Default Value: true since 8.5.107; false previously

Valid Value: true, false

Changes Take Effect: After restart

Enables GMS to send request to the /heartbeat URI of ORS to check availability.

max_ors_idle_connection_time

Section: ors

Default Value: 3600

Valid Values: Any integer

Changes Take Effect: After restart

Added in: 8.5.107

Maximum idle time (seconds) for an ORS connection before this connection will be deleted from the load-balancer cache.

ors_loadbalancer_refresh_rate

Section: ors

Default Value: 45000

Valid Values: Integer >= 30000

Changes Take Effect: After restart

Discontinued: 8.5.219.03

Refresh rate of the ORS Load balancer in milliseconds. This option value must be greater than or equal to 30,000 (30 seconds). By default, all ORS URL values are checked every 45 seconds.

[port_restrictions] Section

You can control port access to GMS APIs by adding a `port_restrictions` section in the **Options** tab of GMS configuration, at the node level or cluster level. This section is optional and not defined in the default template. The content of this section is a list of key/values, where key is an URI pattern (`/genesys/1/storage/*`, `/genesys/1/service/*`, `/genesys/1/service/request-interaction`, and so on), and the value is a list of ports or a port range.

Example [port_restrictions] section:

Option Name	Option Value	Description
<code>/genesys/1/storage*</code>	80-90	Storage API will be accessible from port 80 to port 90.
<code>/genesys/1/service/*</code>	92-98,100	Services API will be accessible from port 92 to port 98, plus the port 100.

Important

- There are no default values or default option names. You can define various URL patterns; such as `/genesys/1/resource*`, `/genesys/1/resource*`, `/genesys/1/service/*`, `/genesys/1/service/request-interaction`, and so on.
- If the request is sent on another port, an HTTP error 403 Forbidden occurs.
- The Admin UI and APIs not listed in the `port_restrictions` section will be available on all ports listed in the `port_restrictions` section.

See [Restricting Ports](#) for further information about these configuration options. Changes in this section require an update to the `jetty-http.xml` file on all GMS nodes, and then restarting GMS.

[push] Section

Changes take effect: After restart.

The push configuration includes three logical groups of options: general configuration, push provider configuration, and OS-specific message formatting. For more information about providers and OS-specific message formatting refer to [Genesys Mobile Services Push Notification Service](#).

It is possible for some mandatory options to be absent in this section. In this case, the corresponding push type will be disabled (even if enabled using the `push.pushEnabled` option) and a log entry will be created.

Important

For security reasons, if you wish to hide some private keys that are specific to the notification mechanism, you can define **sensitive** options in a dedicated section.

Common Notification Options

`customhttp.url`

Section: push

Default Value:

Valid Values: Any valid URL

Changes Take Effect: After restart

Mandatory URL where the notifications will be pushed. The subscriber must provide a URL that will be invoked. GMS posts the payload to this URL (using HTTP POST). The Payload is a JSON object that contains two properties: the `deviceId`, which is the custom id provided at subscription time by the subscriber, and the `message`, which is the notification message.

This option describes the provider configuration used for accessing the target (APPLE APNS service, HTTP address).

You can use the HTTPS scheme without adding the server certificate if you configure the `http.ssl_trust_all` option to true in the `[gms]` section.

If you do not use the `http.ssl_trust_all` option, add the server certificate to the Java cacerts.

For example, in a Linux platform:

```
keytool -import -alias genesys -keystore /etc/pki/java/cacerts -file /security/  
custom_https_server_certificate.crt -noprompt -storepass changeit
```

defaultSubscriptionExpiration

Section: push

Default Value:

Valid Values: Any integer (≥ 30)

Changes Take Effect: After restart

Default subscription expiration (in seconds). If the option is not set or if you assign an incorrect value, the default value (30) will be used.

filtering_chat_events

Section: push

Default Value: Notice.TypingStarted,Notice.TypingStopped

Valid Values:

Changes Take Effect: After restart

Comma-separated list of the following events:

- Notice.TypingStarted
- Notice.TypingStopped
- Notice.Joined
- Notice.Left

- Notice.PushUrl
- Notice.Custom
- Message.Text

A comma-delimited list that sets the default value for the `_filtering_chat_events` service parameter. By default, this list is set to "Notice.TypingStarted,Notice.TypingStopped".

pushEnabled

Section: push

Default Value: comet

Valid Values: android, gcm, ios, httpcb, orscb, customhttp, fcm,comet

Changes Take Effect: After restart.

Modified: 8.5.113.10, 8.5.112.05

A comma-delimited list of strings that describe the enabled push types. Currently, the following push types are supported:

- **android**
- **gcm**
- **ios**
- **httpcb**
- **orscb**
- **customhttp**
- **fcm** (starting in 8.5.112.05)
- **comet** (starting in 8.5.103.10)

Any other push type will be ignored. If an option value is not set, then it will be handled as empty string option value (that is, push will be disabled for all supported types and the push service will not work).

Note: Starting in 8.5.103.10, this option requires the default value (**comet**) even if you do not enable push notifications. If you enable push notifications, use one of the above valid values.

httpcb.connection_max_connections_per_route

Section: push

Default Value: 20
Valid Values: Any integer ≥ 2
Changes Take Effect: After restart.

The maximum allowed number of simultaneously opened connections for one route. Default value (used if option not set or incorrect) 20.

httpcb.connection_timeout

Section: push
Default Value: 5
Valid Values: Any positive integer
Changes Take Effect: After restart.

The http connection timeout in seconds. Default value is 5

httpcb.max_connections_total

Section: push
Default Value: 200
Valid Values: Any integer ≥ 5
Changes Take Effect: After restart.

The maximum allowed total number of simultaneously opened connections. Default value (used if option not set or incorrect) 200

localizationFileLocation

Section: push
Default Value:
Valid Values:
Changes Take Effect: After restart.

Location of the file containing the list of localized messages.

Apple Notification Options

Note: Please see the [relevant documentation](https://developer.apple.com) at developer.apple.com for information about OS-Specific message formatting options. Note that if no alert-related options are specified, the *alert* dictionary entry will not be included in the JSON sent to the Apple device.

apple.alert

Section: push

Default Value: No default value

Valid Values: Any string

Changes Take Effect: After restart

Enables an iOS standard alert and defines the text of this alert with two buttons: **Close** and **View**. If the user taps **View**, the application is launched. If this option is null, the **alert** property will not be added to the notification.

apple.alertMessage.action-loc-key

Section: push

Default Value:

Valid Values: Any string

Changes Take Effect: After restart

If set (not null), is used as an *action-loc-key* entry in the alert dictionary (iOS-specific).

apple.alertMessage.body

Section: push

Default Value:

Valid Values: any String, may be null(=absence of option)

Changes Take Effect: After restart.

If set, defines a body entry in the alert dictionary (iOS-specific).

apple.alertMessage.launch-image

Section: push

Default Value:**Valid Values:** Any string**Changes Take Effect:** After restart

If set, is used as the *badge* entry in the *aps* dictionary (iOS-specific).

apple.alertMessage.loc-argnames

Section: push**Default Value:****Valid Values:** Any string**Changes Take Effect:** After restart

If set (not null), used as a *loc-args* entry in the alert dictionary (iOS-specific).

apple.alertMessage.loc-key

Section: push**Default Value:****Valid Values:** Any string**Changes Take Effect:** After restart

If set (not null), used as *loc-key* entry in the alert dictionary (iOS-specific).

apple.title

Section: push**Default Value:** Empty string**Valid Values:** String**Changes Take Effect:** After restart.

Apple Notification title. If not specified, GMS sends a blank title.

apple.badge

Section: push

Default Value: 0

Valid Values: any, may be null (=not set)

Changes Take Effect: After restart.

If set, number used as *badge* entry in the *aps* dictionary (iOS-specific). If this property is absent, any badge number currently shown is removed. If not set, the *badge* entry will not be part of the push notification.

apple.content-available

Section: push

Default Value:

Valid Values: Any string

Changes Take Effect: After restart

Set this key with a value of 1 to indicate that new content is available and let the remote notification act as a silent notification. This is used to support Newsstand apps and background content downloads. Newsstand apps are guaranteed to be able to receive at least one push with this key per 24-hour window.

When a silent notification arrives, iOS wakes up your app in the background so that you can get new data from your server or do background information processing. Users aren't told about the new or changed information that results from a silent notification, but they can find out about it the next time they open your app.

apple.keystore

Section: push

Default Value:

Valid Values: Valid file path

Changes Take Effect: After restart.

keystore location (path to file) for iOS push notifications

apple.keystorePassword

Section: push

Default Value:

Valid Values: Not null (but may be empty string)

Changes Take Effect: After restart.

Password to access keystore. If the password is incorrect, the attempts to push messages will fail with the corresponding log entries.

apple.sound

Section: push

Default Value:

Valid Values: any String, may be null(=absence of option)

Changes Take Effect: After restart.

If set, used as *sound* entry in the *aps* dictionary (iOS-specific). Use the name of a sound file in the application bundle. The sound in this file is played as an alert. If the sound file doesn't exist or you set this value to **default**, the default alert sound is played. If not set, the corresponding entity will not be added to the notification.

debug.apple.keystore

Section: push

Default Value: No default value

Valid Values: Valid file path

Changes Take Effect: After restart

Keystore location (filepath) for iOS push notifications. This option applies to notifications whose debug value is set to true.

debug.apple.keystorePassword

Section: push

Default Value: No default value

Valid Values: Not null or empty string

Changes Take Effect: After restart

Password to access keystore. If the password is incorrect, the attempts to push messages will fail with the corresponding log entries. This option applies to notifications whose debug value is set to true.

Android Notification Options

android.collapseKey

Section: push

Default Value:

Valid Values: not empty

Changes Take Effect: After restart.

Discontinued: 8.5.114.09

An arbitrary string that is used to collapse a group of like messages when the device is offline, so that only the last message gets sent to the client. This is intended to avoid sending too many messages to the phone when it comes back online. Note that since there is no guarantee of the order in which messages get sent, the "last" message may not actually be the last message sent by the application server

android.delayWhileIdle

Section: push

Default Value: false

Valid Values: true, false

Changes Take Effect: After restart.

Discontinued: 8.5.114.09

If included and true, indicates that the message should not be sent immediately if the device is idle. The server will wait for the device to become active (only 1 last message will be delivered to device when it becomes active). Default (if not specified) - false;

android.gcm.apiKey

Section: push

Default Value:

Valid Values: Valid Google api Key. See Google GCM description.

Changes Take Effect: After restart.

Discontinued: 8.5.114.09

Valid Google API Key. See Google CDM description. Please see <https://developers.google.com/cloud-messaging/gcm>

android.gcm.retryNumber

Section: push

Default Value: 2

Valid Values: Any integer

Changes Take Effect: After restart.

Discontinued: 8.5.114.09

Retry attempts (in case the GCM servers are unavailable).

android.senderAccountType

Section: push

Default Value:

Valid Values: not null, may be empty

Changes Take Effect: After restart.

Discontinued: 8.5.114.09

Specified when initializing c2dm push service

android.senderEmail

Section: push

Default Value: @gmail.com

Valid Values: valid mail (sender account registered in Google service)

Changes Take Effect: After restart.

Discontinued: 8.5.114.09

Valid name of mail account. The notifications will be sent from behalf of this account. After signing up for C2DM, the sender account will be assigned the default quota, which currently corresponds to approximately 200,000 messages per day.

android.senderPassword

Section: push

Default Value:
Valid Values: valid password of registered account
Changes Take Effect: After restart.
Discontinued: 8.5.114.09

Password of account

android.source

Section: push
Default Value:
Valid Values: not empty
Changes Take Effect: After restart.
Discontinued: 8.5.114.09

Specifying when sending push notification service

android.ssl_trust_all

Section: push
Default Value: false
Valid Values: true, false
Changes Take Effect: After restart.
Discontinued: 8.5.114.09

If included and true, indicates that any SSL certificate provided during establishing https connection to <https://www.google.com/accounts/ClientLogin> and <https://android.apis.google.com/c2dm/send> addresses are considered valid, regardless of their presence in keystore/truststore used by environment. Default value - false. Please note that setting this option to true is highly unadvised. The most correct way is to configure the security system to permit the actually received certificates.

android.unavailability_retry_timeout

Section: push
Default Value: 120
Valid Values: Any positive integer
Changes Take Effect: After restart.
Discontinued: 8.5.114.09

This parameter specifies the default timeout (in seconds) to wait before Google C2DM service can

be accessed again if the request returned the 503 code (Service unavailable). Please note, that this value is ignored if the 503 response from Google contains valid Retry-After header. Default value for this parameter is 120 (used if value not set or incorrect).

Firebase Cloud Messaging

Introduced in: 8.5.112

Due to recent changes in Google Cloud Messaging, GMS now supports Firebase Cloud Messaging (FCM). To configure Native Push Notification through Firebase Cloud Messaging, you can either specify an apiKey or create a dedicated section to secure passwords (recommended for production environments).

Development

```
[push]
fcm.apiKey=<serverKey>
pushEnabled=fcm
```

Production

```
[push]
fcm=fcmsection
pushEnabled=fcm
```

```
[fcmsection]
password=***** (<serverKey>)
```

fcm.apiKey

Section: push

Default Value: No default value

Valid Values: String

Changes Take Effect: After restart

Introduced: 8.5.112.05

Valid Firebase Cloud Messaging API key. Refer to the [official documentation](#) for further details.

debug.fcm.apiKey

Section: push

Default Value: No default value

Valid Values: Any string

Changes Take Effect: After restart

Introduced: 8.5.114.09

Valid Firebase Cloud Messaging API key to use if debug=true.

You can also define title and body messaging for the event received at the provider event level detailed below [below](#).

fcmm.title

Section: push

Default Value: No default value

Valid Values: Any string

Changes Take Effect: After restart

Introduced: 8.5.114.09

Firebase Cloud Messaging title for an event defined at the provider level.

[more...](#)

fcmm.body

Section: push

Default Value: No default value

Valid Values: Any string

Changes Take Effect: After restart

Introduced: 8.5.114.09

Firebase Cloud Messaging body message for an event defined at the provider level.

[more...](#)

Windows Notification options

wns.clientSecret

Section: push

Default Value:

Valid Values:

Changes Take Effect: After restart

The secret key associated to the application. See [Microsoft Official documentation](#).

wns.notificationType

Section: push

Default Value:

Valid Values:

Changes Take Effect: After restart

Type of notification that GMS will send to the Windows application. This value must match the X-WNS-Type header. For example, you can specify a toast notification by setting this option to wns/toast.

wns.sid

Section: push

Default Value:

Valid Values:

Changes Take Effect:

Unique identifier for your Windows Store app. See [Microsoft Official documentation](#).

wns.xmlTemplate

Section: push

Default Value:

Valid Values:

Changes Take Effect: After restart

XML string that defines the notification. For example, to set up a toast notification, you can set this option to:

```
<toast><visual><binding template="ToastText01"><text id="1">bodyText</text></binding></visual></toast>
```

push.provider options

Each provider can contain 2 *channels* for message sending - **production** and **debug** for each target

type. The provider-affiliated options enlisted above describe the production channel. For each provider-related option **<option-name>** the sibling option can be provided with name **debug.<option-name>**. Such options will describe the provider-specific configuration of debug channel for corresponding target type. The debug channel will be enabled for enabled target type only if all mandatory options will be specified for debug channel. The OS-message formatting options do not have production-debug differentiation.

push.provider.providername Section

It is possible to create providers by adding **push.provider.providername** sections which contain the appropriate credential configuration options that are associated with a given provider. This allows you to control and isolate notifications and events between a given provider and the associated services/applications that are using it. This type of provider name section can only contain provider-related options (as listed in **push** section). All providers are isolated - if the option is not specified in provider's section, then it is not specified. If a mandatory option is missing then the corresponding target type will not be enabled, even if that type is present in the **pushEnabled** option.

Please note that we have the following restriction on **providername**: it may only contain alphanumeric characters, the underscore (`_`), and the minus sign (`-`).

push.provider.event Section

You can define the event definitions associated across providers by adding your **push.provider.event** section, and then setting the appropriate OS-specific attribute options within. This will allow you to add OS-specific attributes to a published event message that is going to any provider's push notification system. This section can contain OS formatting-related options. All other options will be ignored. For more information about providers and OS-specific message formatting refer to [Genesys Mobile Services Push Notification Service](#).

push.provider.event.eventname Section

You can define the events associated across providers by adding a custom push.provider.event.**eventname** section, and then setting the appropriate OS-specific attribute options within. This will allow you add OS-specific attributes to a published event message that is going to a specific channel for given group of events tags.

- This section can contain OS formatting-related options.
- All other options will be ignored.

Important

For more information about providers and OS-specific message formatting, see [Push Notification Service](#).

For instance, you can create a push.provider.event.chat section to define options for chat events. The following configuration samples show how to configure iOS chat push alert text for chat events.

Configuring iOS Chat Push Alert Text Without Localization

1. Edit your GMS application (with Configuration Manager for example) and select the **Options** tab.
2. Create a `push.provider.event.chat.newagentmessage` section.
3. Click **New** to set the `apple.alert` parameter to an accurate value; for instance, New message from Agent.

Configuring iOS Chat Push Alert Text Using Localization

1. Edit your GMS application (with Configuration Manager for example) and select the **Options** tab.
2. Create a `push.provider.event.chat.newagentmessage` section.
3. Click **New** to set the `apple.alertMessage.loc`-key parameter to `CHAT_NEW_AGENT_MESSAGE`.
4. Click **New** to set the `apple.alertMessage.action-loc`-key parameter to `ACTION_KEY`.

Important

`CHAT_NEW_AGENT_MESSAGE` and `ACTION_KEY` are keys to lookup the string values for the current language in the `Localizable.strings` resource of the iOS application. See also the official Apple documentation about [Localized Formatted Strings](#).

`push.provider.providername.event.eventname` Section

You can define the event definitions associated with given provider by adding your `push.provider.providername.event.eventname` section, and then setting the appropriate OS-specific attribute options within. This will allow you add OS-specific attributes to a published event message that is going to a specific provider and channel for given group of events tags. This section can contain OS formatting-related options. All other options will be ignored. For more information about providers and OS-specific message formatting refer to [Genesys Mobile Services Push Notification Service](#).

[resources] Section

`patterns_list_name`

Section: resources

Default Value: `GMS_Patterns`

Valid Values: Valid CME name for List object

Changes Take Effect: Immediately upon notification.

Name of the configuration object (with type List), which holds the configuration of patterns and pattern groups. For further details, see [Creating and configuring a pattern list](#).

resources_list_name

Section: resources
Default Value: GMS_Resources
Valid Values: Valid CME name for List object
Changes Take Effect: Immediately upon notification.

Name of the configuration object (with type List), which holds the configuration of resources and resource groups. For further details, see [Creating and configuring a resource list](#).

user_control

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

This option enables GMS to control resource access based on the gms_user header passed in the GMS request. This option is dynamic.

List Object Options

Each section in the Annex is a group that should have distinct list options specified. Changes take effect: Immediately.

Option	Default	Description
<p>_allocation_strategy RANDOM, LOCAL, CLUSTER Optional</p>	<p>RANDOM</p>	<p>Supported strategies:</p> <ul style="list-style-type: none"> • RANDOM—Allocate a randomly selected resource from the group. No reservations or locks are made, so the same resource can be selected by different users at the same time. • LOCAL—A resource is allocated from the group and reserved/locked, so that only one user can hold it at the time. For the resource to return to the group it should be released either by the corresponding API call or by a timeout.

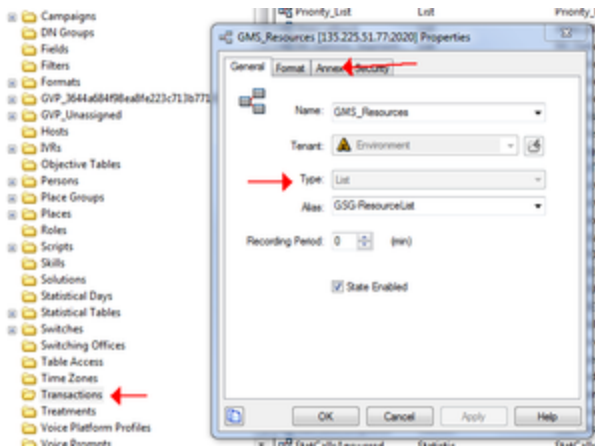
Option	Default	Description
		<ul style="list-style-type: none"> CLUSTER—A resource is allocated from the group and reserved/locked through the GSG cluster, so that only one user can hold it at the time. For the resource to return to the group it should be released either by the corresponding API call or by a timeout.
_booking_expiration_timeout Integer (s) Optional	30	Determines the maximum amount of time, in seconds, that a resource may be allocated. If the resource is not released before this time limit elapses, it is automatically returned to the pool of available resources. This option is used with the LOCAL and CLUSTER allocation strategies.
_backup_resource String Optional	String	The resource returned if there are no regular resources available. This option is used with the LOCAL and CLUSTER allocation strategies.

List Entries

Changes take effect: Immediately.

Option	Default	Description
All keys not starting with # or String Optional	N/A	The value is put into the pool of resources. The option name may be anything (since that value is not currently used).

The following screenshot shows an example of an application object configured in Configuration Manager.



Example

```
[Dnis_Pool]
_allocation_strategy = LOCAL
_booking_expiration_timeout = 20
dnis1 = 1-888-call-me1
dnis2 = 1-888-call-me2
dnis3 = 1-888-call-me3
```

Note: For testing purposes, Genesys recommends that you include at least three numbers in the pool. If only a single number is defined in the pool, when the API call is made, that number is allocated for 30 seconds (default). If another API call is made before the number is returned to the pool, an error will occur. Alternatively, if using a single number, use `_allocation_strategy = RANDOM`.

[server] Section

Modified in 8.5.107

`_ors`

Section: server
Default Value:
Valid Values:
Changes Take Effect: Immediately

Comma-separated list of ORS URLs.
`http://host1:port1,http://host2:port2`

This list will be used for all services sections and can be overridden in each service.

`_ors_lb_strategy`

Section: server

Default Value: circular

Valid Values: circular, linear

Changes Take Effect: Immediately

Strategy for the ORS load balancer in the server section and service sections; this value can be overridden in each service. Supported values are: circular or linear.

`access_code_prefix`

Section: server

Default Value:

Valid Values: Any integer

Changes Take Effect: Immediately

This value is a range of `access_code`; the value must be unique for each GMS node across the cluster. GMS will randomly choose within this range the `access_code_prefix` that it will associate as the prefix for `access_code`. If the option is not present, GMS will use the `nodeId` value instead. An example range is 455, 456-458 where the prefix can be 455, 456, 457, or 458.

`dateFormat`

Section: server

Default Value:

Valid Values:

Changes Take Effect: Immediately

The string used to format dates. The string syntax should match the expectations of the java class `java.text.SimpleDateFormat`. See [Simple Date Format](#) for details.

external_url_base

Section: server

Default Value:

Valid Values: http://<hostname>:<port>/ or https://<hostname>:<port>/

Changes Take Effect: Immediately

Specifies the external URL used by the Storage Service to allow the retrieval of a binary attachment. Configure this option in the case of a Load Balancer deployment.

The valid value is http://<hostname>:<port>/, where:

- <hostname> is used by the cluster service to identify a node
- <port> is used by the cluster service to identify a node.

The <port> value must be the same as the GMS port described in the jetty configuration file, otherwise, an alarm will be displayed in Solution Control Interface (SCI) and GMS will stop.

gsgadmin_redirect

Section: server

Default Value: default

Valid Values:

Changes Take Effect: Immediately

Configures the host and port to use in the redirection message that is sent by GSG Admin upon logout process. In a load balancing deployment, you should set up this option to make sure that GMS redirects to the Load Balancer address instead of the local GMS.

The possible values for this option are the following:

- default—Redirects to the local GMS instance (default behavior).
- external_url_base—Uses the value of external_url_base for the redirection.
- <host>:<port>—Specifies another URL to use for the redirection.

Limitation: Internet Explorer may not correctly depict the port redirection set in external_url_base.

max-sessions

Section: server

Default Value: 9999
Valid Values: Any integer
Changes Take Effect: Immediately

Maximum number of concurrent sessions for the Service Management UI.

node_id

Section: server
Default Value: 1
Valid Values:
Changes Take Effect:

Specifies a two digit number that should be unique in the GMS deployment. It is used in the generation of DTMF access tokens.

Cluster Service options

app_name

Section: server
Default Value:
Valid Values: Any valid URL
Changes Take Effect: Immediately

Web application "context" path.

web_host

Section: server
Default Value: Result of `InetAddress.getLocalHost()`
Valid Values: Valid host name
Changes Take Effect: Immediately

The default `InetAddress.getLocalHost()` value will be used in the most cases. Change this configuration value if you have issues obtaining the local name when your environment has

multiple network interfaces. In this scenario, to ensure GMS internode communication, set this option's value to the IP Address used by the Jetty interface (which is not configurable).

This option is required for internode communication.

web_port

Section: server

Default Value: 80

Valid Values: Valid TCP port; for HTTPS internode communication, 8443 or check either your jetty configuration or restriction port

Changes Take Effect: Immediately

Sets a port different from the port that GMS uses. Note: GMS uses port 8080, which can be changed in the jetty-http.xml file. This option can be used in the case of proxy role of the customer to forward requests.

At startup, GMS checks that a GMS is available on the port specified by web_port. If a GMS is not available, the web_port option alarm (EventId 2002) is thrown.

Required to ensure the GMS internode communication.

web_scheme

Section: server

Default Value: http

Valid Values: http or https

Changes Take Effect: Immediately

Scheme of the internal URL to https if GMS jetty is configured to support only SSL/TLS for one node or for a cluster of nodes.

Optional, required for GMS internode communication.

max-file-upload

Section: server

Default Value: 5000000

Valid Values: Long (bytes)
Changes Take Effect:

Allowed maximum size before uploads are refused.

Configuration Use Cases for web_host and Admin UI

In scenarios that involve internode communication, the value of the web_host option can determine the successful display of the GMS nodes status in the Admin UI. Check the examples below and edit this option accordingly.

Use case	Action	Comment
The Configuration Server hosts do not have IP addresses and no DHCP has been setup, or you cannot resolve the hostname for any other reason	Set the web_host option to the node's IP address for each GMS node	In the Admin UI, the node status will be true but the node_hostipaddress field in the IP address response will be: <not provided>
The Configuration Server hosts do not have IP addresses and the GMS host includes several network interfaces	Set the web_host option to the node's IP address for each GMS node	In the Admin UI, the node status will be true but the node_hostipaddress field in the IP address response will contain the IP address of another network interface of the host.
The Configuration Server hosts have IP addresses	N/A	In the Admin UI, the node status will be true and the node_hostipaddress field in the IP address response will contain the correct IP address.

[service.*servicename*] Section

You can create customized services by adding your service.*servicename* section, and then setting the appropriate options within. Additional options vary depending on the type of service being created. For more information, refer to documentation for the corresponding service in the [Genesys Mobile Services API Reference](#).

For a list of the available options, refer to the [Service option reference](#) page.

[stat.*statname*] Section

This section defines Stat Server statistics that can be opened at startup by listing them in the [\[reporting\] startup-statistics](#) configuration option. You can also subscribe to these statistics using the [Stat Service API](#). In either case, Genesys Mobile Services will initialize the statistics, start collecting the data from Stat Server, and place that data in the GMS cache.

Please note that:

- The cache is global and common to all GMS instances.
- The statistics that are not used are removed and closed by a scheduled function every 10 minutes.

filter

Section: stat.statname
Default Value: No default value
Valid Values: Any string
Changes Take Effect: After restart

The business attribute value to use to filter the results.

metric

Section: stat.statname
Default Value: No default value
Valid Values: Any string
Changes Take Effect: After restart

The name of the metric, for example, **TotalLoginTime**. This option defines a Stat Server statistic that can be opened at startup by listing it in the [\[reporting\] startup-statistics](#) configuration option.

notificationMode

Section: stat.statname
Default Value: No default value
Valid Values: NoNotification, Reset, or Immediate
Changes Take Effect: After restart

Notification mode. Mandatory.

objectId

Section: stat.statname

Default Value: No default value

Valid Values: Any string

Changes Take Effect: After restart

Statistic object ID.

objectType

Section: stat.statname

Default Value: No default value

Valid Values: Any string

Changes Take Effect: After restart

Statistic object type; for example, **Agent**.

tenant

Section: stat.statname

Default Value: No default value

Valid Values: Any string

Changes Take Effect: After restart

Tenant name; for example, **Environment**.

tenantPassword

Section: stat.statname

Default Value: No default value

Valid Values: Any string

Changes Take Effect: After restart

Tenant password.

Example

```
[reporting]
startup-statistics=stat1,stat2
```

```
[stat.stat1]
metric=TotalLoginTime
notificationMode=NoNotification
objectId=KSippola
objectType=Agent
tenant=Environment
filter=Bronze
```

```
[stat.stat2]
metric=ExpectedWaitTime
notificationMode=NoNotification
objectId=9002@SIP_Switch
objectType=Queue
tenant=Environment
```


Service Options

You can create customized GMS services by adding a service.<service_name> section to your GMS configuration or by adding a new service in the Service Management UI, and then setting the appropriate options within. Additional options vary depending on the type of service being created. For more information, refer to the documentation of the corresponding service in the [Genesys Mobile Services API Reference](#).

The following service options are available in the Genesys Mobile Services Options Reference Guide:

- [All GMS Basic services, including Office Hours.](#)
- [Callback Services](#)

Overwriting Configuration in Queries

Overwriting service configuration in queries is not possible for Basic GMS Services. If you pass one of the service parameters in a Create API query for a customized version of the service, the configuration option value will supersede the value passed in the query (that is, the passed value will be ignored).

For details about overwriting configuration in Callback queries, refer to the [Callback Services API Reference page](#).

Security

The following table lists additional security configurations that can be used with your GMS installation.

Page	Summary
Security and Access Control	An outline of key security concerns and information about providing access control.
Cassandra Security	Describes security configurations for Cassandra.
Restricting Ports	Provides instructions to configure and enable port control.
Basic Authentication	Describes configuration for basic authentication.
Transport Layer Security	Describes Transport Layer Security (TLS), which enables cryptographic and trusted communications between Genesys clients and servers.
Single Sign-On	Provides the settings needed to configure GMS to use your existing SSO infrastructure.
Hiding Selected Data in Logs	Describes configuration options to hide some data in GMS logs.

Secure GMS Access Control

This page covers the list of security points, mechanisms, and procedures that you can implement to secure your Genesys Mobile Environment.

Important

If you face any security issues, Genesys recommends that you upgrade to the latest 8.5.3 release.

Use HTTPS to avoid unsecured connections to GMS

HTTPS (Hypertext Transfer Protocol Secure) is an internet communication protocol that protects the integrity and the confidentiality of your users' data between the user's computer and the GMS. Implementing HTTPS connections with GMS prevents the following Transport Confidentiality attacks:

Attack	Description
MITM	Man In The Middle
BEAST	Browser Exploit Against SSL/TLS
CRIME	Compression Ratio Info-leak Made Easy

In 8.5.3, the Jetty server that GMS runs on was upgraded to release 12. As a result, GMS requires JVM 17 or later. In accordance with RFC 7231 recommendations, the Content-Type response header will provide charset values in lowercase: Content-Type → application/json; charset=utf-8

Add a Certificate to JETTY

1. Request a Certificate from a [Certificate Authority \(CA\)](#) to sign your key/certificate.
2. Once the CA has sent you a certificate, load it into a JSSE keystore. Run the JDK's `keytool` to load a certificate in PEM form directly into a keystore. The following command loads a PEM encoded certificate in the `jetty.crt` (JETTY) file into a JSSE keystore:

```
$ keytool -keystore mykeystore -import -alias jetty -file jetty.crt -trustcacerts
```

3. To add the certificate to JETTY, copy the `mykeystore` file to JETTY's `etc` folder, then edit `jetty-ssl-context.xml` to setup the certificate info.

Private key and cert files have to be in the keystore. You may also need to convert PEM-format keys to JKS format and PKCS12 to a JKS file. The PEM format is the most common format that Certificate Authorities issue certificates in.

PEM certificates usually have extensions such as `.pem`, `.crt`, `.cer`, and `.key`. They are Base64-encoded ASCII files and contain `"-----BEGIN CERTIFICATE-----"` and `"-----END CERTIFICATE-----"` statements.

Server certificates, intermediate certificates, and private keys can all be put into the PEM format. Several PEM certificates and even the private key can be included in one file, one below the other.

Here are some command examples to help you to manage .pfx certificate files and the Jetty's keystore:

Get the key only:

```
e:\openssl\openssl pkcs12 -in e:\certtest\upmc.pfx -nocerts -out e:\certtest\upmckey.pem
```

Get the cert only:

```
e:\openssl\openssl pkcs12 -in e:\certtest\upmc.pfx -clcerts -nokeys -out e:\certtest\upmccert.pem
```

Remove the passphrase from the key:

```
e:\openssl\openssl rsa -in e:\certtest\upmckey.pem -out e:\certtest\upmcserver.key
```

Create a combined .p12 file that you can import in the keystore:

```
e:\openssl\openssl pkcs12 -export -in E:\certtest\upmccert.pem -inkey e:\certtest\upmcserver.key -out e:\certtest\combined.p12 -name upmc_cert -CAfile e:\certtest\my_ca_bundle.crt -caname root
```

Add the combined key/cert to the keystore:

```
keytool -importkeystore -deststorepass **** -destkeypass **** -destkeystore mykeystore -srckeystore combined.p12 -srcstoretype PKCS12 -srcstorepass ****
```

Configure JETTY Secured Ports

1. Setup the secured port in the **jetty-ssl.xml** and **jetty.xml** files. The default HTTPS port is 8443.
2. Remove the content of the **jetty-http.xml** file (not relevant). Moreover, you will need to setup (operating system) firewall rules to control access to Jetty HTTP unsecured port (8080 by default).

```
<?xml version="1.0"?>
<!DOCTYPE Configure PUBLIC "-//Jetty//Configure//EN" "configure_10_0.dtd">

<!-- ===== -->
<!-- Configure the Jetty Server instance with an ID "Server" -->
<!-- by adding an HTTP connector. -->
<!-- This configuration must be used in conjunction with jetty.xml -->
<!-- ===== -->
<Configure id="Server" class="org.eclipse.jetty.server.Server">

  <!-- ===== -->
  <!-- Add an HTTP Connector. -->
  <!-- Configure an o.e.j.server.ServerConnector with a single -->
  <!-- HttpConnectionFactory instance using the common httpConfig -->
  <!-- instance defined in jetty.xml -->
  <!-- -->
  <!-- Consult the javadoc of o.e.j.server.ServerConnector and -->
  <!-- o.e.j.server.HttpConnectionFactory for all configuration -->
  <!-- that may be set here. -->
  <!-- ===== -->
  <!-- -->
```

```

    <Call name="addConnector">
      <Arg>
        <New id="httpConnector" class="org.eclipse.jetty.server.ServerConnector">
          <Arg name="server"><Ref refid="Server" /></Arg>
          <Arg name="acceptors" type="int"><Property name="jetty.http.acceptors"
default="1"/></Arg>
          <Arg name="selectors" type="int"><Property name="jetty.http.selectors"
default="-1"/></Arg>
          <Arg name="factories">
            <Array type="org.eclipse.jetty.server.ConnectionFactory">
              <Item>
                <New class="org.eclipse.jetty.server.HttpConnectionFactory">
                  <Arg name="config"><Ref refid="httpConfig" /></Arg>
                </New>
              </Item>
            </Array>
          </Arg>
          <Set name="host"><Property name="jetty.http.host" deprecated="jetty.host"
/></Set>
          <Set name="port"><Property name="jetty.http.port" deprecated="jetty.port"
default="8080" /></Set>
          <Set name="idleTimeout"><Property name="jetty.http.idleTimeout"
deprecated="http.timeout" default="30000"/></Set>
          <Set name="acceptorPriorityDelta"><Property
name="jetty.http.acceptorPriorityDelta" deprecated="http.acceptorPriorityDelta"
default="0"/></Set>
          <Set name="acceptQueueSize"><Property name="jetty.http.acceptQueueSize"
deprecated="http.acceptQueueSize" default="0"/></Set>
          <Get name="SelectorManager">
            <Set name="connectTimeout"><Property name="jetty.http.connectTimeout"
default="15000"/></Set>
          </Get>
          <Set name="reuseAddress"><Property name="jetty.http.reuseAddress"
default="true"/></Set>
          <Set name="reusePort"><Property name="jetty.http.reusePort"
default="false"/></Set>
          <Set name="acceptedTcpNoDelay"><Property name="jetty.http.acceptedTcpNoDelay"
default="true"/></Set>
          <Set name="acceptedReceiveBufferSize"
property="jetty.http.acceptedReceiveBufferSize" />
          <Set name="acceptedSendBufferSize" property="jetty.http.acceptedSendBufferSize"
/>
        </New>
      </Arg>
    </Call>
  -->
</Configure>

```

3. Update the start.ini file in <GMS_HOME> directory with the following text:

```

--module=server, ee8-jsp, jmx, ee8-websocket-
javax, resources, ext, ee8-plus, ee8-annotations, ee8-deploy, ee8-security, ee8-servlets, rewrite, rewrite-
compactpath, ssl, https
...
...
#=====
jetty.httpConfig.sendServerVersion=false
jetty.sslContext.keyStorePath=etc/keystore
jetty.sslContext.keyStorePassword=<keystorepassword>

```

```
jetty.sslContext.keyManagerPassword=<keystorepassword>
jetty.sslContext.trustStorePath=etc/ANTIMEAPPS01
jetty.sslContext.trustStorePassword=0BF:<encrypted keystore="" password="">
```

In your production environment, you should place your `mykeystore` file in a private directory with restricted access, not in a directory relative to the JETTY home directory. The `mykeystore` file in the example above is relative to the JETTY home directory.

The `jetty.sslContext.keyStorePassword` and `jetty.sslContext.keyManagerPassword` properties can also contain obfuscated passwords (OBF). Check the `jetty-ssl-context.xml` file (as an example). You can use the same process that is used for the `jetty.sslContext.trustStorePassword` OBF password generation.

Important

For production, choose a private directory with restricted access to keep your keystore in. Even though it has a password on it, the password may be configured into the runtime environment and is vulnerable to theft. You can now start Jetty the normal way (make sure that in your text `cert.jar`, `jnet.jar`, and `jsse.jar` are on your classpath) and SSL can be used with a URL like: `https://localhost:8443/`

You should encode the `the_password` value, for example by using the embedded password tool:

```
C:\8.5.3>java -cp
lib/jetty-http-12.0.8.jar;lib/jetty-util-12.0.8.jar
org.eclipse.jetty.util.security.Password username userpassword
Usage - java org.eclipse.jetty.security.Password [<user>] <password>
```

If the password is `?`, the user will be prompted for the password.

4. Force the use of the TLS protocol by adding a new `sslConnector` connector or by disabling the unsecured port. To modify the secured port, edit the `etc/jetty-ssl.xml` file:

```
<Call name="addConnector">
  <Arg>
    <New id="sslConnector" class="org.eclipse.jetty.server.ServerConnector">
      <Arg name="server"><Ref refid="Server" /></Arg>
      <Arg name="acceptors" type="int"><Property name="jetty.ssl.acceptors"
default="1"/></Arg>
      <Arg name="selectors" type="int"><Property name="jetty.ssl.selectors"
default="-1"/></Arg>
      <Arg name="factories">
        <Array type="org.eclipse.jetty.server.ConnectionFactory">
          <!-- uncomment to support proxy protocol
          <Item>
            <New class="org.eclipse.jetty.server.ProxyConnectionFactory"/>
          </Item>-->
        </Array>
      </Arg>
      <Set name="host"><Property name="jetty.ssl.host" deprecated="jetty.host" /></Set>
      <Set name="port"><Property name="jetty.ssl.port" deprecated="ssl.port"
default="8443" /></Set>
      <Set name="idleTimeout"><Property name="jetty.ssl.idleTimeout"
deprecated="ssl.timeout" default="30000"/></Set>
      <Set name="acceptorPriorityDelta"><Property
name="jetty.ssl.acceptorPriorityDelta" deprecated="ssl.acceptorPriorityDelta"
default="0"/></Set>
      <Set name="acceptQueueSize"><Property name="jetty.ssl.acceptQueueSize"
deprecated="ssl.acceptQueueSize" default="0"/></Set>
      <Get name="SelectorManager">
      <Set name="connectTimeout"><Property name="jetty.ssl.connectTimeout"
default="15000"/></Set>
      <!-- Set name="reservedThreads"><Property name="jetty.ssl.reservedThreads"
```

```

default="-2"/></Set -->
  </Get>
  <Set name="reuseAddress"><Property name="jetty.ssl.reuseAddress"
default="true"/></Set>
  <Set name="reusePort"><Property name="jetty.ssl.reusePort"
default="false"/></Set>
  <Set name="acceptedTcpNoDelay"><Property name="jetty.ssl.acceptedTcpNoDelay"
default="true"/></Set>
  <Set name="acceptedReceiveBufferSize"
property="jetty.ssl.acceptedReceiveBufferSize" />
  <Set name="acceptedSendBufferSize" property="jetty.ssl.acceptedSendBufferSize" />
</New>
</Arg>
</Call>

```

5. Edit your **ORS DFM files** to point to the new secured port of the TLS connector (8443 in our example).

```

<dfm name='gsgCallback'
namespace='http://www.genesyslab.com/modules/dfm/gsgCallback/v1'>
  <protocols>
  <protocol type='http'>
  <params>
  <param name='timeout' expr='20' />
  </params>
  <globalevents>
  <event name='done' mapping='200' />
  <event name='error' mapping='0,400-510' />
  </globalevents>
  </protocol>
  <protocol type='https'>
  <params>
  <param name='retries' expr='4' />
  <param name='timeout' expr='20' />
  </params>
  <globalevents>
  <event name='done' mapping='200' />
  <event name='error' mapping='400-510' />
  </globalevents>
  </protocol>
  </protocols>
  <connections>
  <connection name='HTTP_round_robin' type='https'
selectionmode='round-robin'>
  <servers>
  <server name='gsg1' host='1XX.1XX.10.01' port='8443' />
  <server name='gsg2' host='1XX.1XX.10.02' port='8443' />
  <server name='gsg3' host='1XX.1XX.10.03' port='8443' />
  <server name='gsg4' host='1XX.1XX.10.04' port='8443' />
  </servers>
  </connection>
  </connections>
  <transports>
  <transport name='HTTP_GET' conntype='HTTP_round_robin'>
  <param name='method' expr='GET' />
  <param name='enctype' expr='application/x-www-form-urlencoded' />
  </transport>
  <transport name='HTTP_POST' conntype='HTTP_round_robin'>
  <param name='method' expr='POST' />
  <param name='enctype' expr='application/x-www-form-urlencoded' />
  </transport>
  <transport name='HTTP_DELETE' conntype='HTTP_round_robin'>
  <param name='method' expr='DELETE' />
  <param name='enctype' expr='application/x-www-form-urlencoded' />

```

```

</transport>
<transport name='HTTP_PUT' conntype='HTTP_round_robin'>
<param name='method' expr='PUT' />
<param name='enctype' expr='application/json' />
</transport>
</transports>
<actions>
<action name='start-callback' transport='HTTP_POST' >
<overrides>
<param name='url' expr=
'/genesys/${apiVersion}/service/callback/${callbackexecutionname}' />
</overrides>
<attributes>
<attribute name='apiVersion' type='String' optional='false' />
<attribute name='callbackexecutionname' type='String'
optional='false' />
<attribute name='content' type='Expression' optional='false'
mapping='content' />
</attributes>
</action>
<action name='cancel-callback' transport='HTTP_DELETE' >
<overrides>
<param name='url'
expr='/genesys/${apiVersion}/service/callback/${callbackexecutionname}/${service_id}' />
</overrides>
<attributes>
<attribute name='apiVersion' type='String' optional='false'
/ >
<attribute name='callbackexecutionname' type='String' optional='false'
/ >
<attribute name='service_id' type='String' optional='false' />
</attributes>
</action>
<action name='reschedule-callback' transport='HTTP_PUT' >
<overrides>
<param name='url'
expr=
'/genesys/${apiVersion}/service/callback/${callbackexecutionname}/${service_id}'
/ >
</overrides>
<attributes>
<attribute name='apiVersion' type='String' optional='false' />
<attribute name='callbackexecutionname' type='String' optional='false' />
<attribute name='service_id' type='String' optional='false' />
<attribute name='content' type='Expression' optional='false'
mapping='content' />
</attributes>
</action>
<action name='query-callback' transport='HTTP_GET' >
<overrides>
<param name='url'
expr='/genesys/${apiVersion}/service/callback/${callbackexecutionname}
?customer_number=${customer_number_value}' />
</overrides>
<attributes>
<attribute name='apiVersion' type='String' optional='false' />
<attribute name='callbackexecutionname' type='String' optional='false'
/ >
<attribute name='customer_number_value' type='String' optional='false'
/ >
</attributes>
</action>
<action name='query-availability-timestamp' transport='HTTP_GET' >

```



```

<overrides>
<param name='url'
expr='/genesys/${apiVersion}/service/callback/${callbackexecutionname}
/availability?timestamp=${timestamp_value}'/>
</overrides>
<attributes>
<attribute name='apiVersion' type='String' optional='false' />
<attribute name='callbackexecutionname' type='String' optional='false'
/>
<attribute name='timestamp_value' type='String' optional='false' />
</attributes>
</action>
<action name='query-availability' transport='HTTP_GET' >
<overrides>
<param name='url'
expr='/genesys/${apiVersion}/service/callback/${callbackexecutionname}
/availability' />
</overrides>
<attributes>
<attribute name='apiVersion' type='String' optional='false' />
<attribute name='callbackexecutionname' type='String' optional='false'
/>
</attributes>
</action>
</actions>
</dfm>

```

6. Make the following changes to your Jetty server:

a. Download and copy the following files to <GMS_HOME>/etc directory:

- [keystore](#)
- [jetty-ssl-context.xml](#)

b. Update the start.ini file by adding "etc/jetty-ssl-context.xml" to the following list:

```

etc/jetty.xml
etc/jetty-ssl.xml
etc/jetty-ssl-context.xml
etc/jetty-deploy.xml
etc/jetty-http.xml
etc/jetty-https.xml
etc/jetty-rewrite.xml
#etc/jetty-overlay.xml
#etc/jetty-webapps.xml
#etc/jetty-contexts.xml

```

Redirect HTTP requests to HTTPS

If you want to keep an unsecured port (for instance port 8080) and redirect HTTP to HTTPS, the webapp should indicate that it needs CONFIDENTIAL or INTEGRAL connections from users. This is done in <GMS_HOME>/etc/webdefault-ee8.xml

```

<web-app>
<security-constraint>
<web-resource-collection>
<web-resource-name>Everything</web-resource-name>
<url-pattern>/*</url-pattern>
</web-resource-collection>
<user-data-constraint>
<transport-guarantee>CONFIDENTIAL</transport-guarantee>

```

```
</user-data-constraint>
</security-constraint>
```

GMS Options to Setup

Set the following options set in your GMS application at the server level, not the cluster level:

```
[server]
external_url_base=https://gmshost:8443/
web_port=8443
web_scheme=https
```

Configure HTTP Strict Transport Security

By default, HTTP Strict Transport Security (HSTS) is configured in GMS as follows:

Strict-Transport-Security max age: 31536000

Include subdomain: true

To update this configuration, edit the `etc/jetty-ssl.xml` file in the GMS installation directory as follows:

```
<New id="sslHttpConfig" class="org.eclipse.jetty.server.HttpConfiguration">
  <Arg><Ref refid="httpConfig"/></Arg>
  <Call name="addCustomizer">
    <Arg>
      <New class="org.eclipse.jetty.server.SecureRequestCustomizer">
        <Arg name="sniRequired" type="boolean"><Property name="jetty.ssl.sniRequired"
default="false"/></Arg>
        <Arg name="sniHostCheck" type="boolean"><Property name="jetty.ssl.sniHostCheck"
default="true"/></Arg>
        <Arg name="stsMaxAgeSeconds" type="int"><Property name="jetty.ssl.stsMaxAgeSeconds"
default="31536000"/></Arg>
        <Arg name="stsIncludeSubdomains" type="boolean"><Property
name="jetty.ssl.stsIncludeSubdomains" default="true"/></Arg>
      </New>
    </Arg>
  </Call>
</New>
```

Secure Access to Proxy Configuration

GMS can send outbound HTTP requests to Push Notification servers through the following notification services:

- APNS (Apple Push Notification Service)

- FCM (Firebase Cloud Messaging Service)
- Third Party Servers for custom notification

To secure access, GMS provides a Proxy Auto-Configuration file (PAC) mechanism. You can configure a PAC file in your GMS Application's settings, section gms:

Option	Type	Default value	Restriction on value	Description
http.proxy-auto-config-file	String		Optional Valid URL	Location of the proxy auto-config (PAC) file. For example: <ul style="list-style-type: none"> • file:///C:/GMS/proxy.pac : for a local file • http://127.0.0.1:8082/deploy/proxy.pac
http.proxy-cache-size	Integer	32	Valid integer (optional)	Size of the cache that stores URLs that were already processed. If the requested URL is in the cache, GMS will not process the PAC file.
http.proxy-ttl	Integer	5	Valid integer (Optional)	Interval in minutes before refreshing the PAC content.

You can use the following PAC file to secure GMS access for Apple and Google Servers:

```
function FindProxyForURL(url, host) {
if (dnsDomainIs(host, ".googleapis.com"))
    return "PROXY 127.0.0.1:808";

if (dnsDomainIs(host, ".wns.windows.com"))
    return "PROXY 127.0.0.1:808";

if (dnsDomainIs(host, ".apple.com"))
    return "SOCKS 127.0.0.1:1080";

    return "DIRECT";
}
```

Possible Values for Access-Control-Allow-Origin Header (CORS)

By default, the value for all the properties mentioned below is set to *, meaning *all origins*.

If you need to configure CORS for a specific domain, method, or header, then edit the **launcher.xml** file as described below; the following are the three types of properties used to configure CORS.

- com.genesyslab.common.cors.allowed-origins
- com.genesyslab.common.cors.allowed-methods
- com.genesyslab.common.cors.allowed-headers

Important

The CORS configuration options above apply for REST API /genesys URL. For the /cometd URL (cometd servlet), the CORS configuration options are the same and set up in the etc/webdefault-ee8.xml file.

Example:

```
<filter>
  <filter-name>cross-origin</filter-name>
  <filter-class>org.eclipse.jetty.servlets.CrossOriginFilter</filter-class>
  <async-supported>true</async-supported>
  <!-- default init-param !-->
  <init-param>
    <param-name>allowedOrigins</param-name>
    <param-value>*</param-value>
  </init-param>
  <init-param>
    <param-name>allowedMethods</param-name>
    <param-value>GET,POST,OPTIONS,DELETE,PUT,HEAD</param-value>
  </init-param>
  <init-param>
    <param-name>allowedHeaders</param-name>
    <param-value>origin, content-type, accept, authorization,Access-
Control-Request-Method, gms_user</param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>cross-origin</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

Allowed Origins

This is the list of allowed origins that can be specific origins. For example, <https://domain1.com>. A matched origin is listed in the 'Access-Control-Allow-Origin' response header of preflight actual CORS requests.

If the below property is not available, then, by default, all origins are allowed.

```
<parameter name="cors-allowed-origins" displayName="cors-allowed-origins" mandatory="false">
```

```

    <description><![CDATA[This should have the origins which we need to allow in CORS
policy, multiple origins can be
separated by coma]]></description>
    <valid-description><![CDATA[]]></valid-description>
    <effective-description/>
        <format type="string" default="-Dcom.genesyslab.common.cors.allowed-
origins=http://sample1.com,https://sample1
.com,http://sample2:8080,https://sample2:8080" />
    <validation/>
</parameter>

```

Important

- The value for this property should be a literal match for the string, that is, <http://sample1.com> is not equal to <https://sample1.com>.
- There are no line breaks in between domains or domain names.
- Blank spaces are not permitted in between domains or domain names.

Allowed Methods

Set the HTTP methods to allow specific methods, for example, **GET**, **POST**, and so on. By default all methods are allowed.

```

<parameter name="allowed-method" displayName="com.genesyslab.common.cors.allowed-methods"
mandatory="false">
    <description><![CDATA[This should have the methods which we need to allow in cors
policy, multiple methods can be separated by coma]]></description>
    <valid-description><![CDATA[]]></valid-description>
    <effective-description/>
        <format type="string" default="-Dcom.genesyslab.common.cors.allowed-
methods=GET,POST,PUT" />
    <validation></validation>
</parameter>

```

Important

- There are no line breaks in between method names.
- Blank spaces are not permitted in between method names.

Allowed Headers

Set the list of headers that a pre-flight request can list as allowed for use during an actual request. As per the CORS spec, the header name is not required to be listed if it is one of the below:

- Cache-Control

- Content-Language
- Expires
- Last-Modified
- Pragma

```
<parameter name="allowed-header" displayName="allowed-header" mandatory="false">
  <description><![CDATA[This should have the headers which we need to allow in cors
policy, multiple headers can be separated by coma]]></description>
  <valid-description><![CDATA[]]></valid-description>
  <effective-description/>
  <format type="string" default="-Dcom.genesyslab.common.cors.allowed-headers=X-
Requested-With,Origin" />
  <validation></validation>
</parameter>
```

Important

- There are no line breaks in between header names.
- Blank spaces are not permitted in between header names.

Add Custom Headers

To add custom headers to the GMS response (X-Frame-Options, X-Content-Type-Options, XSS-Protection, Content-Security-Policy, Access-Control-Allow-Headers, and so on), use Jetty to rewrite handles. This procedure is documented in the [Rewrite Handler](#) chapter of the official Jetty documentation. For instance, the following snippets added to the **jetty-rewrite.xml** file create the appropriate custom headers for the Storage API, based on the `/genesys/1/storage/*` pattern attribute:

```
<Call name="addRule">
  <Arg>
    <New class="org.eclipse.jetty.rewrite.handler.HeaderPatternRule">
      <Set name="pattern">/genesys/1/storage/*</Set>
      <Set name="headerName">Access-Control-Allow-Credentials</Set>
      <Set name="headerValue">>false</Set>
    </New>
  </Arg>
</Call>
<Call name="addRule">
  <Arg>
    <New class="org.eclipse.jetty.rewrite.handler.HeaderPatternRule">
      <Set name="pattern">/genesys/1/storage/*</Set>
      <Set name="headerName">X-Frame-Options</Set>
      <Set name="headerValue">SAMEORIGIN</Set>
    </New>
  </Arg>
</Call>
<Call name="addRule">
  <Arg>
```

```

        <New class="org.eclipse.jetty.rewrite.handler.HeaderPatternRule">
            <Set name="pattern">/genesys/1/storage/*</Set>
            <Set name="headerName">X-XSS-Protection</Set>
            <Set name="headerValue">1; mode=block</Set>
        </New>
    </Arg>
</Call>
<Call name="addRule">
    <Arg>
        <New class="org.eclipse.jetty.rewrite.handler.HeaderPatternRule">
            <Set name="pattern">/genesys/1/storage/*</Set>
            <Set name="headerName">Content-Security-Policy</Set>
            <Set name="headerValue">default-src 'self'</Set>
            <Set name="terminating">true</Set>
        </New>
    </Arg>
</Call>
<Call name="addRule">
    <Arg>
        <New class="org.eclipse.jetty.rewrite.handler.HeaderPatternRule">
            <Set name="pattern">/genesys/1/storage/*</Set>
            <Set name="headerName">Access-Control-Allow-Headers</Set>
            <Set name="headerValue">customerHeader1,customerHeader2</Set>
        </New>
    </Arg>
</Call>

```

As a result, you will get the following storage response:

```

HTTP/1.1 200 OK
Access-Control-Allow-Credentials: false
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
Content-Security-Policy: default-src 'self'
Content-Type: application/json;charset=utf-8
Transfer-Encoding: chunked

```

```
{"GMS_KEY": "GMS_VALUE"}
```

Set up DoS Filters to Limit Requests

An attacker can consume all of some required resources by generating enough traffic from your host to prevent legitimate users from using the system. See [Application Denial of Service](#)

The Denial of Service (DoS) filter limits exposure to request flooding, whether malicious or as a result of a misconfigured client. The DoS filter keeps track of the number of requests from a connection per second. If the number of requests exceeds the limit, JETTY rejects, delays, or throttles additional requests, and sends a warning message. The filter works on the assumption that the attacker might be written in a simple blocking style, so by suspending requests you are hopefully consuming the attacker's resources.

To configure the DoS filter in GMS, edit the `<GMS_HOME>/etc/webdefault-ee8.xml` file. Configure the options list below, save, and restart GMS to take the new settings into account.

Important

If you are using a load balancer in front of the GSM nodes, the load balancer must set the X-Forwarded-For HTTP header to allow the DoS filter to identify the client IP and control the request rates.

Parameter name	Default Value	Description
maxRequestsPerSec	25	Maximum number of requests per second from a connection. Additional requests are first delayed, then throttled.
delayMs	100	Delay in milliseconds that all requests over the rate limit must wait before they get processed: <ul style="list-style-type: none"> -1 = Reject request 0 = No delay
maxWaitMs	50	Maximum time to wait, in milliseconds, blocking for the throttle semaphore.
throttledRequests	5	Number of requests over the rate limit able to be considered at once.
throttleMs	30000L	Time, in milliseconds, to asynchronously wait for the throttle semaphore.
maxRequestMs	30000	Time, in milliseconds, to allow the request to run.
maxIdleTrackerMs	30000	Time, in milliseconds, to keep track of request rates for a connection, before deciding that the user has gone away, and discarding it.
insertHeaders	true	Keep true to insert the DoSFilter headers into the response; otherwise, false.
remotePort	false	Set to true to not use session tracking, then track by IP+port (effectively connection).
ipWhitelist		A comma-separated list of IP addresses that will not be rate-limited.
managedAttr		Set to true to set this servlet as a ServletContext attribute with the filter name as the attribute name. This allows a context external mechanism such as for

Parameter name	Default Value	Description
		example JMX via ContextHandler.MANAGED_ATTRIBUTES) to manage the configuration of the filter.

The following code is an example of DoS Filter file. In the **webdefault-ee8.xml** file the below snippet should be placed before the `<servlet>` tag.

```
<filter>
  <filter-name>DoSFilter</filter-name>
  <filter-class>org.eclipse.jetty.ee8.servlets.DoSFilter</filter-class>
  <async-supported>true</async-supported>
  <init-param>
    <param-name>maxRequestsPerSec</param-name>
    <param-value>200</param-value>
  </init-param>
  <init-param>
    <param-name>ipWhitelist</param-name>
    <param-value>127.0.0.1</param-value>
  </init-param>
</filter>
```

Secure your GMS Sessions

Attack	Description
Session Fixation	Session fixation attacks represent a potential risk where a malicious attacker tries to create a session by accessing a site, then persuades other users to log in with the same session by sending them a link containing the session identifier as a parameter, for example.

To prevent attacks on your sessions, you should configure the following mechanisms.

Mechanism	Description
HttpOnly flag is true by default	HttpOnly is an additional flag included in a Set-Cookie HTTP response header. Using the HttpOnly flag when generating a cookie helps mitigate the risk of client side script accessing the protected cookie (if the browser supports it).
Enable the secure flag in HTTPS environment	The secure flag is an option that you can set in the application server when sending a new cookie to the user within an HTTP Response. The purpose of the secure flag is to prevent cookies from being observed by unauthorized parties due to the transmission of a cookie in clear text. To enable it, navigate to the <code><GMS_HOME>/etc</code> folder and edit the <code>webdefault-ee8.xml</code> file. Secure flag setting

Mechanism	Description
	<pre data-bbox="824 310 1339 598"> <session-config> <session-timeout>20</session-timeout> <cookie-config> <secure>>false</secure> <!-- true if HTTPS is used--> <path>/genesys</path> <http-only>>true</http-only> <max-age>86400</max-age> </cookie-config> </session-config> </pre>
<p data-bbox="175 1008 646 1039">Session timeout (default is 20 minutes)</p>	<p data-bbox="824 634 1445 714">To change this timeout, navigate to the <GMS_HOME>/etc/ folder, and the edit webdefault-ee8.xml file.</p> <p data-bbox="824 735 1079 760">Session timeout setting</p> <pre data-bbox="824 787 1315 913"> <session-config> <session-timeout>20</session-timeout> <!-- Value is in minute--> ... </session-config> </pre> <p data-bbox="824 934 1429 1003">The session timeout defines the action window time for a user, and thus, the period while an attacker can try to steal and use an existing user session.</p> <ul data-bbox="836 1029 1445 1396" style="list-style-type: none"> • Set session-timeout to a minimal value, according to the application context. • Avoid "infinite" session timeout. • Prefer a declarative definition of the session timeout to apply a global timeout for all application sessions. • Trace session creation/destruction to analyze the creation trend and detect an abnormal number of session creations, that would indicate some application profiling phase during an attack.

Configure Access Rules for Security Gateway in Front of Genesys Mobile Services

The following access rules should be used as a model for your environment, allowing a list of services provided by your Genesys Mobile Services deployment to be accessed while restricting the use of internal-only services.

Important

Only allow access to the limited set of services listed by name.

Allow access from the end-user application or proxy - mobile, web, etc:

```
-- async notifications over HTTP API:
If client is using cometd transport (typically for chat):
{base url:port}/genesys/cometd
-- service API:
{base url:port}/genesys/{version}/service/{service name}
{base url:port}/genesys/{version}/service/{id}/storage
{base url:port}/genesys/{version}/service/{id}
{base url:port}/genesys/{version}/service/{id}/{request name}
-- chat media API:
{base url:port}/genesys/{version}/service/{id}/ixn/chat
{base url:port}/genesys/{version}/service/{id}/ixn/chat/refresh
{base url:port}/genesys/{version}/service/{id}/ixn/chat/disconnect
{base url:port}/genesys/{version}/service/{id}/ixn/chat/startTyping
{base url:port}/genesys/{version}/service/{id}/ixn/chat/stopTyping
{base url:port}/genesys/{version}/service/{id}/ixn/chat/*
```

Only when client need direct access allow (in most cases only

ORS/scxml need it):

```
-- storage API
{base url:port}/genesys/{version}/storage/{ttl}
{base url:port}/genesys/{version}/storage/{data id}/{ttl}
{base url:port}/genesys/{version}/storage/{data id}
{base url:port}/genesys/{version}/storage/{data id}/{key}

-- callback (management) API
{base url:port}/genesys/{version}/service/callback

/{callback-execution-name}/{service_id}

{base url:port}/genesys/{version}/service/callback

/{callback-execution-name}/{service_id}

{base url:port}/genesys/{version}/service/callback

/{callback-execution-name}/availability?{timestamp=value}

{base url:port}/genesys/{version}/admin/callback

/queues?target={target_name}&end_time={iso_end_time}

Allow access from load balancer for node health check:
{base url:port}/genesys/{version}/node
```

Allow access from the intranet:

```
-- admin UI
{base url:port}/genesys/admin/*
{base url:port}/genesys/{version}/admin/*
{base url:port}/genesys/{version}/reports/*
{base url:port}/genesys/{version}/statistic/*
-- datapot (typically accessed from ORS/scxml):
{base url:port}/genesys/{version}/datapot/{tenantId}
```

/activityName/dates

```
{base url:port}/genesys/{version}/datapot/{tenantId}/activities
{base url:port}/genesys/{version}/datapot/{tenantId}
```

/activityName/aggregates

```
{base url:port}/genesys/{version}/datapot/{tenantId}
```

/activityName/aggregates/unique

```
-- all built-in services (except chat)
{base url:port}/genesys/{version}/service/request-access
{base url:port}/genesys/{version}/service/match-interaction
-- notification API (typically accessed from ORS/scxml):
{base url:port}/genesys/{version}/notification/subscription
{base url:port}/genesys/{version}/notification/subscription/{id}
{base url:port}/genesys/{version}/notification/publish
```

Mobile Native Push Notification Configuration Details

Three major mobile platforms are currently supported: Android, Apple, and Windows. Details about Genesys Mobile Services configuration could be found on the [Push Notification Service](#) page of the API Reference.

You can set sensitive options in different sections of your application configuration. In the following example, the configuration structure allows you to set the password in another section.

```
[push]
apple=apple-desc;
debug-apple=debug-apple-desc
...
[apple-desc]
password=*****
apple.keystore=xxxxxxxxxxx
[debug-apple-desc]
password=*****
apple.keystore=xxxxxxxxxxx
```

In case of:

- IOS (APNS): The password option is used instead of "apple.keystorePassword".

- Android (FCM): The password option is used instead of "fcm.apiKey".
- Microsoft (WNS): The password option is used instead of "wns.clientSecret".

Set all the mandatory options and keep optional options in the push section. See the [options' detail](#) for more information.

If GMS is deployed behind a firewall, edit your rules to allow the server fcm.googleapis.com and range 5228-5230 for ports, as detailed in the [FCM ports and your firewall](#) section of the Firebase Cloud Messaging documentation.

Hide Sensitive Data in Logs

Genesys recommends using log data filtering to hide sensitive configuration data in log files. The given product will then use that filter to determine what content should be filtered or masked when creating log files. The only real interfaces to Genesys Mobile Services are APIs and having to configure filter criteria every time you add a new parameter or API is cumbersome and complicated. Therefore, Genesys Mobile Services supports filtering and masking data in log files for any API parameter that matches the following naming convention:

- Filter: Any parameter name that is prefixed with XXX will be filtered out of the log files entirely. Example: *XXX_FilterParam*
- Mask: Any parameter name that is prefixed with MMM will be masked in the log files, showing in the log with the letters MMM. Example: *MMM_MaskParam*

Use of the Storage API

For security reasons, if you plan using the [GMS Storage API](#), you can define a list of supported mime types in the allowed-mime-types option and configure allow-file-control = true to enable file control.

Secure Connections to URS or ORS

Introduced in 8.5.211

By default, GMS connects to ORS and URS using HTTP. You can use HTTPS for these connections by following the steps below.

HTTPS Connection to URS

1. Retrieve the URS certificate.
2. Import this certificate in the Java keystore.
The following command line imports the certificate in the default JDK keystore from JDK, that is, using the default java keystore password 'changeit'.

```
$ keytool -import -alias urscertificate -keystore /etc/pki/java/cacerts -file
/security/urscertificate.crt -noprompt -storepass changeit
```

Note that changeit is the default password provided during at Java installation.

3. Configure the URS URL in GMS to point to the secured endpoint.
The example below shows how to provision a GMS service for a given URS Agent statistic.

```
[service.agent-stat-proxy-https]
_caching_policy=5
_service=urs-stat
_type=builtin
_urs_stat_url_parameters=tenant=Environment◇=KSippola.A&json&ext
_urs_url=https://hosturs:443/urs/stat/targetstate
```

Testing the URS Connection

To test the URS connection, enter the following commands:

```
$ curl -k https://hosturs:443/urs/version
8.1.400.41
$ curl -k "https://hosturs:443/urs/stat/targetstate?tenant=Environment◇=KSippola.A&json&ext"
{
"status":0,"agent":"KSippola"
}
```

Testing your GMS connection to URS

To test the GMS connection to URS, enter the following commands:

```
$ curl "http://hostgms:8080/genesys/1/service/agent-stat-proxy-https"
{"agent":"KSippola","status":0}
```

In case of any issue while connecting to URS, the response would look like below:

```
$ curl "http://hostgms:8080/genesys/1/service/agent-stat-proxy-https"  
{ "exception": "java.io.IOException", "message": "URSStatisticService: cannot submit to URS for  
any URL in 'service.agent-stat-proxy-https' section" }
```

HTTPS Connection to ORS

1. Retrieve the ORS certificate.
2. Import this certificate in the Java keystore.
The following command line imports the certificate in the default JDK keystore from JDK, that is, using the default java keystore password 'changeit'.

```
$ keytool -import -alias orscertificate -keystore /etc/pki/java/cacerts -file  
/security/orscertificate.crt -noprompt -storepass changeit
```

3. Configure the ORS URL in GMS to point to the secured endpoint.

```
[server]  
_ors=https://<hostors>:8443
```

Testing your ORS connection

To test the ORS connection, enter the following commands:

```
$ curl -k https://hostors:8443/heartbeat  
...  
< HTTP/1.1 200 OK  
< Server: GTS_CORE 2.0  
< Content-Type: text/html;  
< Cache-control: private  
< Etag: 1560434146  
< Content-length: 33  
<  
Node is operating in primary mode
```

Testing your GMS connection to ORS

To do so, create a callback through the Callback Service API or using the [Callback UI](#).

Cassandra Security

This page discusses security configurations for Cassandra.

Protecting Data Stored in the Cassandra Database

The `<Genesys Mobile Services installation directory>/etc/cassandra.yaml` file enables or disables encryption of Cassandra inter-node communication using `TLS_RSA_WITH_AES_128_CBC_SHA` as the cipher suite for authentication, key exchange, and encryption of the actual data transfers. To encrypt all inter-node communications, set to `all`. You must also generate keys, and provide the appropriate key and trust store locations and passwords. Details about Cassandra options are available from:

- [internode_encryption](#)
- [authenticator](#)

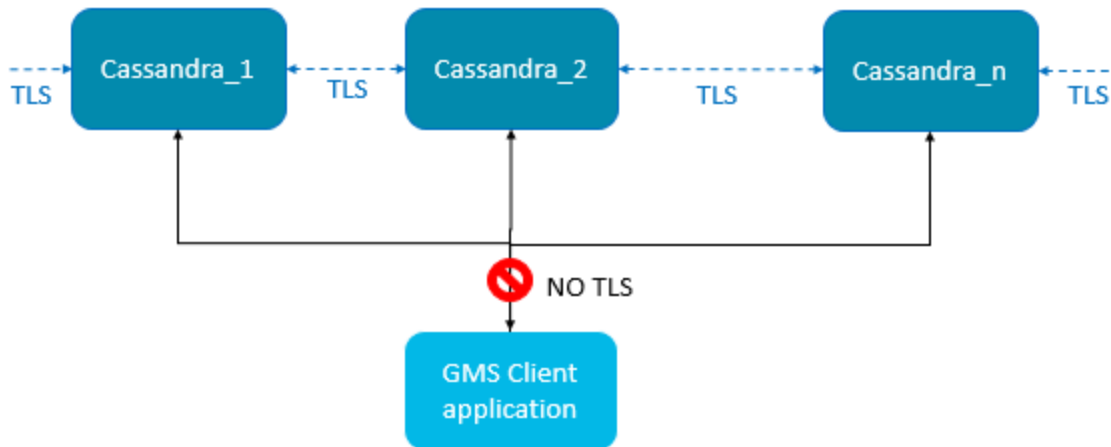
All transient service session-related data is stored in a Cassandra database that uses memory and the file system. See the `<Genesys Mobile Services installation directory>/data` folder. Files located here should be protected from unauthorized access.

Cassandra Authentication

The Cassandra API for custom authentication and authorization has been deprecated in 2.x. Although the legacy classes for authentication and authorization are still implemented in GMS for backward compatibility, Genesys recommends that you use the [External Cassandra](#) configuration for both authentication and authorization.

Cassandra TLS Support

Modified in: 8.5.204.00



TLS

support for external Cassandra includes:

- Inter-node communication (Gossip)
- JMX connection (Cassandra tools for monitoring)

Cassandra TLS Configuration

Create and Import your Certificate

To enable Cassandra TLS feature, use the JDK `keytool` command to create a certificate per node. For example, the commands below create the `keystore.node1` file, then export it as `node1.cer`, and create the `truststore.node1` file.

```

<GMS client side> $ keytool -genkey -keyalg RSA -alias node1 -keystore keystore.node1
-storepass cassandra -keypass cassandra -validity 36500 -dname "CN=192.168.2.1, OU=None,
O=None, L=None, C=None"
<GMS client side> $ keytool -export -alias node1 -file node1.cer -keystore keystore #
password: cassandra
<GMS client side> $ keytool -import -v -trustcacerts -alias node1 -file node1.cer -keystore
truststore.node1 # new password: cassandra + yes
  
```

Prepare the keystore file used for cassandra configuration file (`cassandra.yaml`) by copying the keystore file to in the `<cassandra home>/conf` directory:

```

$ cp keystore.node1 .keystore    ## for Cassandra server side (client_encryption config)
# only for your eyes, restrict access to this file, Keep the crackers from your door
$ chmod 600 .keystore
  
```

Edit `cassandra.yaml` to include this file in the Cassandra configuration.

```

native_transport_port_ssl: 9142 # uncomment this line to have both ports (secured on 9142 and
default unsecured on 9042)
# client_encryption_options:
  enabled: true
  keystore: conf/.keystore # keystore file location
  
```

```
keystore_password: cassandra # password of keystore
```

Edit launcher.xml

Find the launcher.xml file that is part of the GMS Installation directory. Add these parameters to enable TLS support:

```
<parameter name="cassandranodes_truststore" displayName="cassandratruststore" mandatory="true"
hidden="true" readOnly="true">
  <description><![CDATA[Certificates trustStore for Cassandra nodes]]></description>
  <valid-description><![CDATA[]]></valid-description>
  <effective-description/>
  <format type="string" default="-Djavax.net.ssl.trustStore=client.truststore" />
  <validation></validation>
</parameter>
<parameter name="cassandranodes_trustStorePassword" displayName="cassandratrustStorePassword"
mandatory="true" hidden="true" readOnly="true">
  <description><![CDATA[Certificates trustStore password for Cassandra
nodes]]></description>
  <valid-description><![CDATA[]]></valid-description>
  <effective-description/>
  <format type="string" default="-Djavax.net.ssl.trustStorePassword=cassandra" />
  <validation></validation>
</parameter>
<parameter name="debugtls" displayName="Debug TLS" mandatory="true" hidden="true"
readOnly="true">
  <description><![CDATA[Add debug mode for SSL]]></description>
  <valid-description><![CDATA[]]></valid-description>
  <effective-description/>
  <format type="string" default="-Djavax.net.debug=ssl" />
  <validation></validation>
</parameter>
```

For Windows, Global Certificate File

For Windows, add the Certificates trustStore for Cassandra nodes to the main JRE lib/security/cacerts file:

```
$ keytool -importcert -file cassandranodes.cer -alias cassandranodes -keystore $JAVA_HOME/jre/
lib/security/cacerts -storepass changeit
```

Important

changeit is the default password at Java installation time.

Edit the GMS Node Configuration

Here are the cassandra options for TLS cassandra (native-port and secured options) in GMS application options:

```
[cassandra]
native-port=9142
nodes=<cassandra nodes comma separated>
secured=true
strategy-class=SimpleStrategy
```

```
strategy-option=replication_factor:2
```

For further details about these options, refer to the [cassandra](#) section of the Options' reference guide.

Cassandra Gossip TLS

In Cassandra version 1.1.x (the Cassandra version in GMS), the internode (gossip) encryption is set up in the *cassandra.yaml* file. Locate the following lines in the *cassandra.yaml* file:

```
encryption_options:  
  internode_encryption: none  
  keystore: conf/.keystore  
  keystore_password: cassandra  
  truststore: conf/.truststore  
  truststore_password: cassandra
```

Replace `internode_encryption: none` with `internode_encryption: all`, as shown in the following example:

```
encryption_options:  
  internode_encryption: all  
  keystore: conf/.keystore  
  keystore_password: cassandra  
  truststore: conf/.truststore  
  truststore_password: cassandra
```

For managing keystore and truststore (and password), see the Oracle documentation [keytool-Key and Certificate Management Tool](#) or the [Oracle security guide](#).

Cassandra JMX TLS

Warning

The instructions detailed in this section apply only if Cassandra is embedded in GMS (before GMS version 8.5.203).

Cassandra monitoring and management can be done using a Java Management Extensions (JMX) tool. The JMX access must be protected in order to avoid any remote managing on the GMS embedded Cassandra. To protect JMX access, edit the `launcher.xml` file that contains the following lines (by default):

```
<parameter name="jmxport" displayName="jmxport" mandatory="true"  
hidden="true" readOnly="true">  
  <description><![CDATA[JMX related]]></description>  
  <valid-description><![CDATA[]]></valid-description>  
  <effective-description/>  
  <format type="string"  
default="-Dcom.sun.management.jmxremote.port=9192" />  
  <validation></validation>
```

```
</parameter>
<parameter name="jmxssl" displayName="jmxssl" mandatory="true"
hidden="true" readOnly="true">
  <description><![CDATA[virtual machine related]]></description>
  <valid-description><![CDATA[]]></valid-description>
  <effective-description/>
  <format type="string" default="-Dcom.sun.management.jmxremote.ssl=false" />
  <validation></validation>
</parameter>
<parameter name="jmxauthenticate" displayName="jmxauthenticate"
mandatory="true" hidden="true" readOnly="true">
  <description><![CDATA[virtual machine related]]></description>
  <valid-description><![CDATA[]]></valid-description>
  <effective-description/>
  <format type="string"
default="-Dcom.sun.management.jmxremote.authenticate=false" />
  <validation></validation>
</parameter>
```

By default, the TLS and authentication parameters are disabled:

```
com.sun.management.jmxremote.ssl=false
com.sun.management.jmxremote.authenticate=false
```

For information about enabling these parameters and managing JMX and TLS, see the *Monitoring and Management Using JMX Technology* chapter in the [Oracle Java SE Monitoring and Management Guide](#).

Restricting Ports

You can control access to GMS APIs by configuring your firewall to allow or block other hosts (such as public internet, intranet, specific IP addresses, and so on) from accessing TCP/IP ports on the host where GMS is running.

You can configure and enable port control through the following process:

1. Set configuration options.
2. Paste code snippet into the jetty-http.xml file.
3. Restart GMS.

Configuration

Configuration Options

You can control port access to GMS APIs by adding a `port_restrictions` section in the GMS configuration, at the node level or cluster level. This section is optional and not defined in the default template. The content of this section is a list of key/values. Where key is an URI pattern (`/genesys/1/storage/*`, `/genesys/1/service/*`, `/genesys/1/service/request-interaction`, and so on), and the value is a list of ports or a port range.

1. In Configuration Manager, select *Environment > Applications*.
2. Locate and open the Application object for GMS.
3. Select the *Options* tab.
4. Add the `port_restrictions` section, and then set the options and values with the URL and ports you wish to control.
5. Save your changes.

Example `port_restrictions` section:

Option Name	Option Value	Description
<code>/genesys/1/storage*</code>	80-90	Storage API will be accessible from port 80 to port 90.
<code>/genesys/1/service/*</code>	92-98,100	Services API will be accessible from port 92 to port 98, plus the port 100.

Important

- There are no default values or default option names. You can define various URL patterns; such as /genesys/1/resource*, /genesys/1/resource*, /genesys/1/service/*, /genesys/1/service/request-interaction, and so on.
- If the request is sent on another port, an HTTP error 403 Forbidden occurs.
- The Admin UI and APIs not listed in the port_restrictions section will be available on all ports listed in the port_restrictions section.

jetty-http.xml File

1. Go to the <GMS_HOME>/etc/jetty-http.xml file, and add the code snippet after the GMS default HTTP connector (used to open default port 8080).

Example Set connectors section:

```
<!-- ===== -->
<!-- Set connectors -->
<!-- ===== -->
<Call name="addConnector">
  <Arg>
    <New class="org.eclipse.jetty.server.ServerConnector">
      <Arg name="server"><Ref refid="Server" /></Arg>
      <Arg name="acceptors" type="int"><Property name="jetty.http.acceptors"
default="1"/></Arg>
      <Arg name="selectors" type="int"><Property name="jetty.http.selectors"
default="-1"/></Arg>
      <Arg name="factories">
        <Array type="org.eclipse.jetty.server.ConnectionFactory">
          <Item>
            <New class="org.eclipse.jetty.server.HttpConnectionFactory">
              <Arg name="config"><Ref refid="httpConfig" /></Arg>
            </New>
          </Item>
        </Array>
      </Arg>
      <Set name="host"><Property name="jetty.http.host" deprecated="jetty.host" /></Set>
      <Set name="port"><Property name="jetty.http.port" deprecated="jetty.port"
default="8025" /></Set>
      <Set name="idleTimeout"><Property name="jetty.http.idleTimeout"
deprecated="http.timeout" default="30000"/></Set>
      <Set name="acceptorPriorityDelta"><Property
name="jetty.http.acceptorPriorityDelta" deprecated="http.acceptorPriorityDelta"
default="0"/></Set>
      <Set name="acceptQueueSize"><Property name="jetty.http.acceptQueueSize"
deprecated="http.acceptQueueSize" default="0"/></Set>
      <Get name="SelectorManager">
        <Set name="connectTimeout"><Property name="jetty.http.connectTimeout"
default="15000"/></Set>
      </Get>
      <Set name="reuseAddress"><Property name="jetty.http.reuseAddress"
default="true"/></Set>
      <Set name="reusePort"><Property name="jetty.http.reusePort" default="false"/></Set>
    </New>
  </Call>

```

```

        <Set name="acceptedTcpNoDelay"><Property name="jetty.http.acceptedTcpNoDelay"
default="true"/></Set>
        <Set name="acceptedReceiveBufferSize" property="jetty.http.acceptedReceiveBufferSize"
/>
        <Set name="acceptedSendBufferSize" property="jetty.http.acceptedSendBufferSize" />
    </New>
</Arg>
</Call>

```

Important

- Replace the port number in the `jetty.http.port` parameters.
- For multiple ports, copy the entire `<Call name="addConnector">` property and modify it to add additional port numbers.
- To specify a range of port number, for example 8092-8095, you have to include the entire snippet in the **jetty-http.xml** file for each port: 8092, 8092, 8093, 8094, and 8095.

2. Restart GMS.

Disabling Port Restrictions

1. In Configuration Manager, select *Environment > Applications*.
2. Locate and open the Application object for GMS.
3. Select the *Options* tab.
4. Select the `port_restrictions` section.
5. Right-click, and enter a hash tag (#) in front of `port_restrictions` so it appears like this: `#port_restrictions`. The port restrictions are now disabled, and the **Service Management User Interface > Lab > Config** tab will display: port restrictions has not been enabled.

Setting Default GMS Port

By default, the GMS Port is set to 8080. You can modify this value by editing the `<GMS_HOME>/etc/jetty-http.xml` file and changing this value in the `addConnector` section. When you are finished, restart GMS.

```

<Call name="addConnector">
  <Arg>
    <New class="org.eclipse.jetty.server.ServerConnector">
      <Arg name="server"><Ref refid="Server" /></Arg>
      <Arg name="acceptors" type="int">
<Property name="http.acceptors" default="-1"/></Arg>
      <Arg name="selectors" type="int">

```

```
<Property name="http.selectors" default="-1"/></Arg>
  <Arg name="factories">
    <Array type="org.eclipse.jetty.server.ConnectionFactory">
      <Item>
        <New class="org.eclipse.jetty.server.HttpConnectionFactory">
          <Arg name="config"><Ref refid="httpConfig" /></Arg>
        </New>
      </Item>
    </Array>
  </Arg>
  <Set name="host"><Property name="jetty.host" /></Set>
  <Set name="port"><Property name="jetty.port" default="8080" /></Set>
  <Set name="idleTimeout">
<Property name="http.timeout" default="30000"/></Set>
  <Set name="soLingerTime">
<Property name="http.soLingerTime" default="-1"/></Set>
  <Set name="acceptorPriorityDelta">
<Property name="http.acceptorPriorityDelta" default="0"/></Set>
  <Set name="selectorPriorityDelta">
<Property name="http.selectorPriorityDelta" default="0"/></Set>
  <Set name="acceptQueueSize">
<Property name="http.acceptQueueSize" default="0"/></Set>
  </New>
</Arg>
</Call>
```

Important

Make sure that the values of the `external_url_base` option set for the load-balancer URL base and the `server/web_port` option used for CometD node to node communication are consistent with each other.

Enabling Basic Authentication

Modified in 8.5.108

HTTP Basic Authentication is a method for an HTTP client to provide a user name and a user password for each HTTP request where a resource needs access control.

GMS supports Basic Authentication on the following services:

- Storage service
- Notification service
- Callback service
- and all services that have a section (with the prefix **service.**) in the GMS application.

Important

Best practice is to use Basic Authentication over HTTPS.

Configuration

By default, the Basic Authentication feature is turned off. The following sections and options must be set in the GMS application in order to turn on this feature.

Important

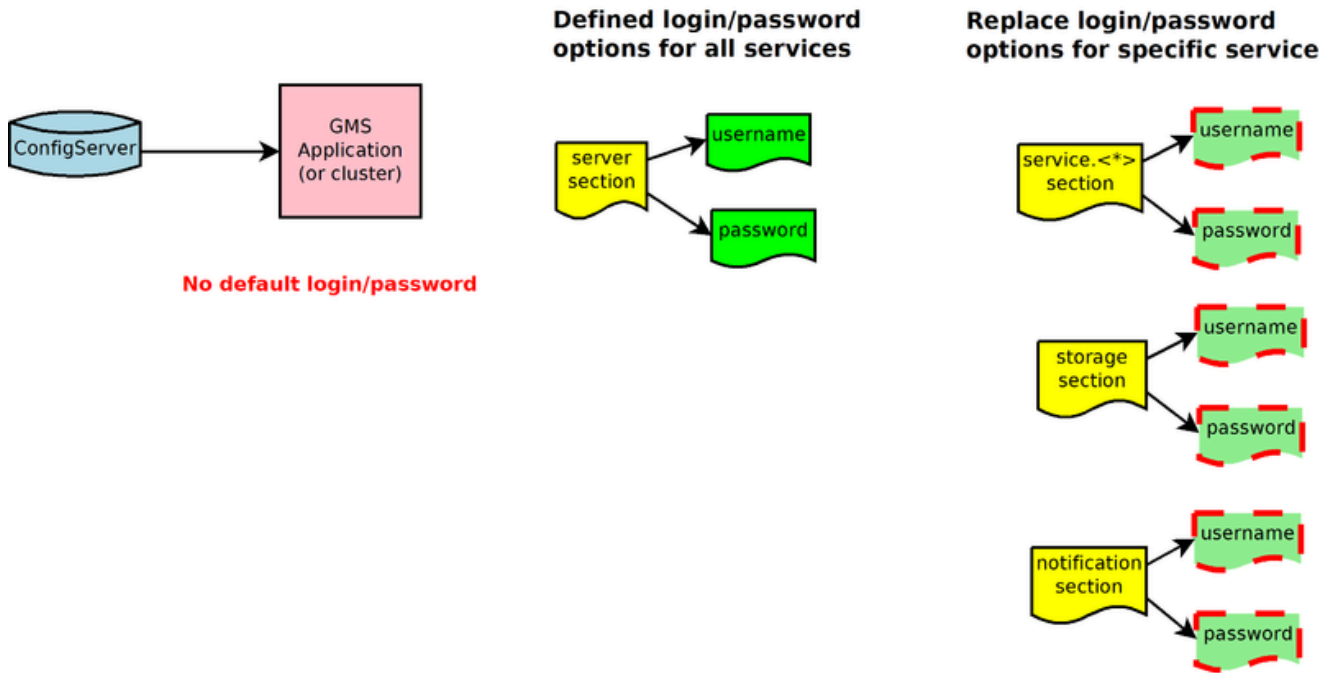
Basic Authentication options are taken into account dynamically.

Section	Option	Supersedes	Description
server	realm		Defines the authentication scheme. The default value is Genesys Application Configuration Needed.
server	username		Defines a global username for all services. Note: Without the password option, no authentication is effective.

Section	Option	Supersedes	Description
server	password		Defines a global password for all services. With this option, Basic Authentication is turned on for all services.
service.*	username	server/username	Defines a specific username for one service. Note: Without the password option in the same service section, the server section password is used. If there is no server section password, no authentication is applied.
service.*	password	server/password	Defines a specific password for one service.
callback	username	server/username	Defines a specific username for the callback service. This option is not taken into account for other callback API URLs which still use service.* configuration (if defined). Note: Without the password option in the same service section, the server section password is used. If there is no server section password, no authentication is applied.
callback	password	server/password	Defines a specific password for the callback service.
storage	username	server/username	Defines a specific username for the storage service. Note: Without the password option in the same service section, the server section password is used. If there is no server section password, no authentication is applied.
storage	password	server/password	Defines a specific

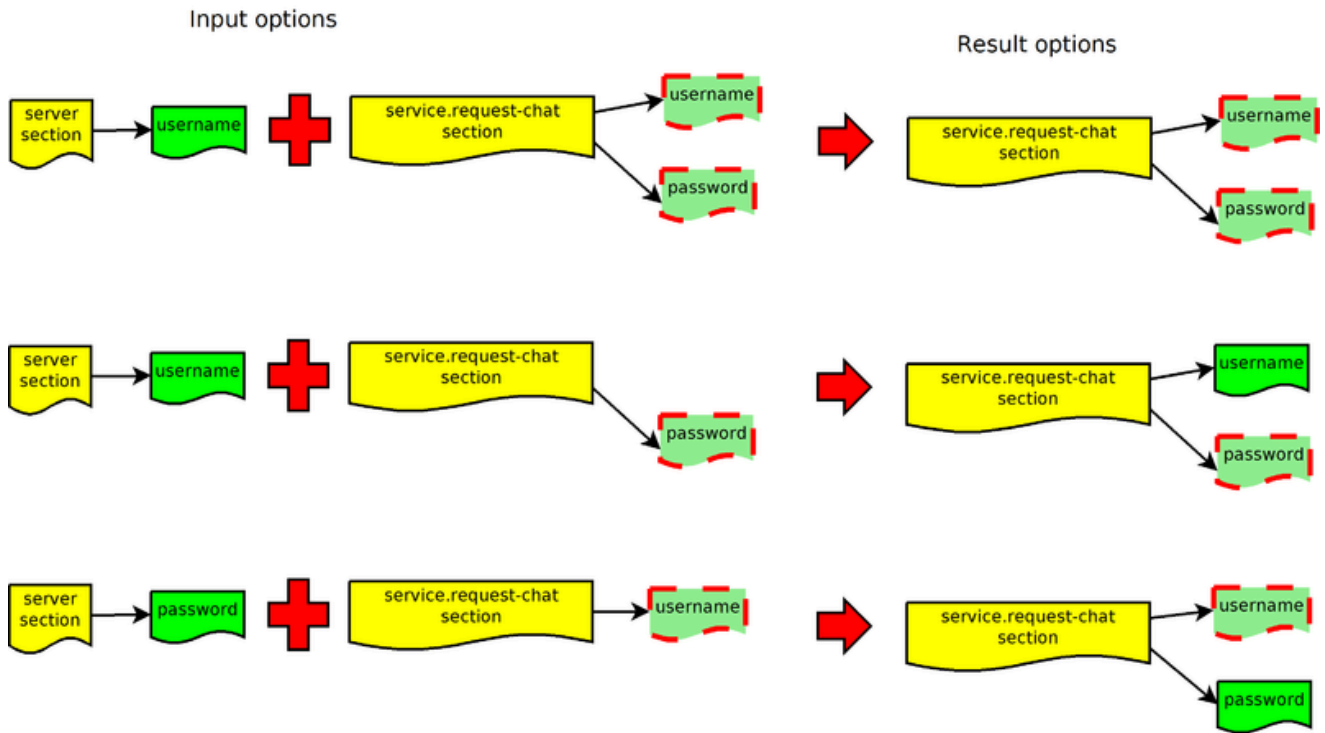
Section	Option	Supersedes	Description
			password for the storage service.
notification	username	server/username	Defines a specific username for the notification service. Note: Without password option in the same service section, the server section password is used. If there is no server section password, no authentication is applied.
notification	password	server/password	Defines a specific password for notification service.

GMS Options for HTTP Basic Authentication



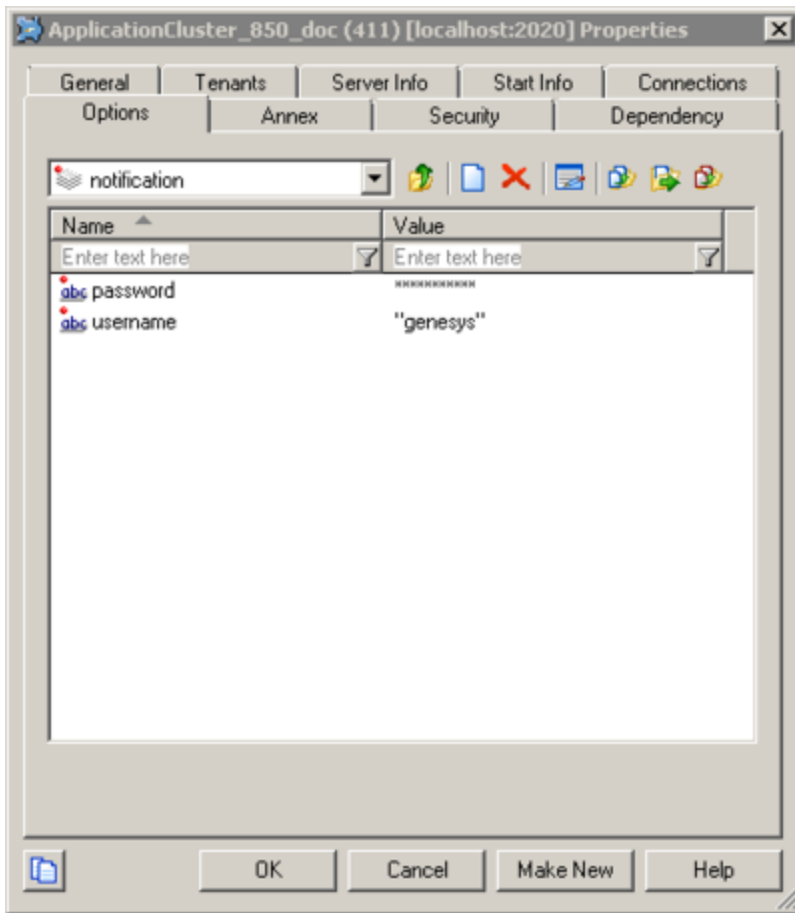
Precedence Example

Example

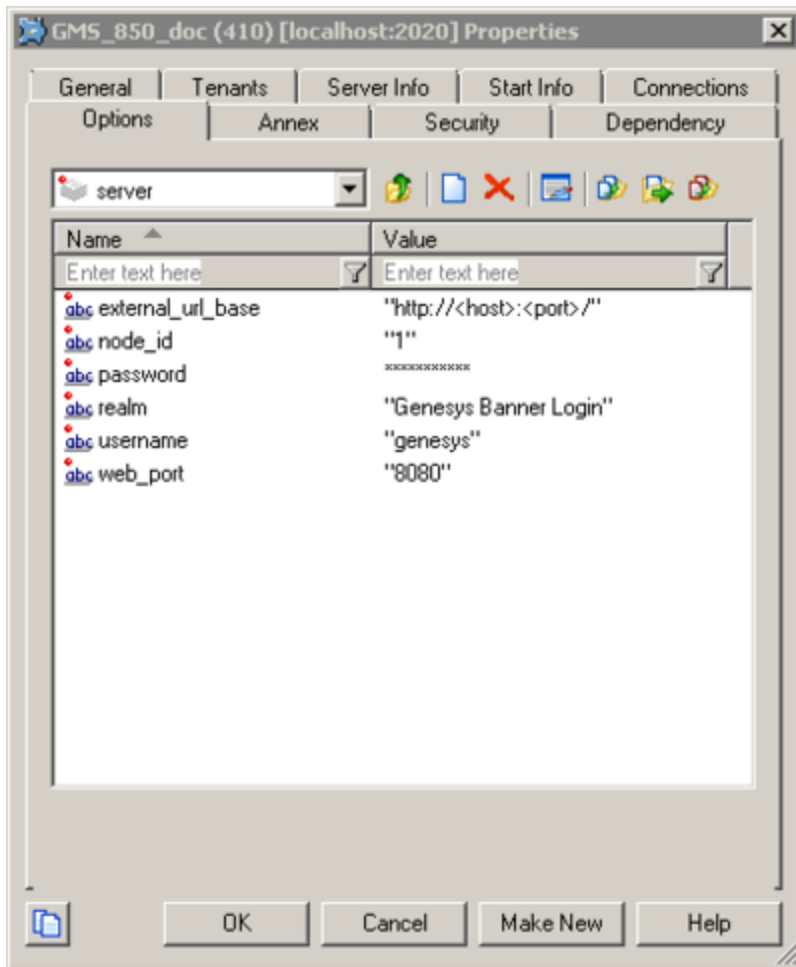


Configuration Option Examples

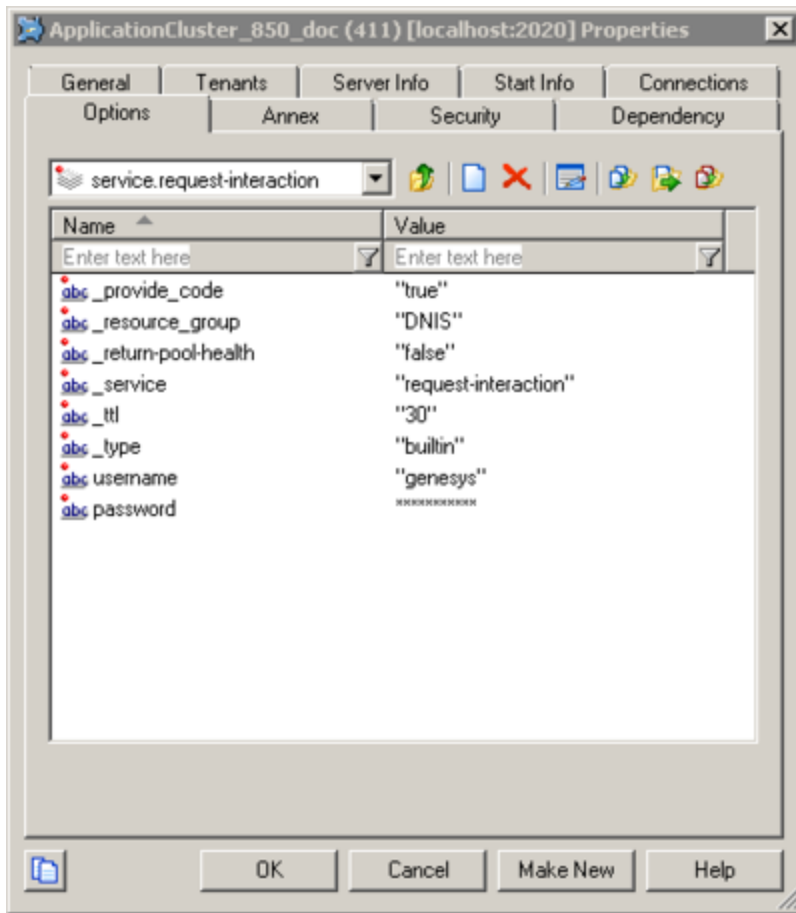
Basic Authentication Username and Password for Notification API:



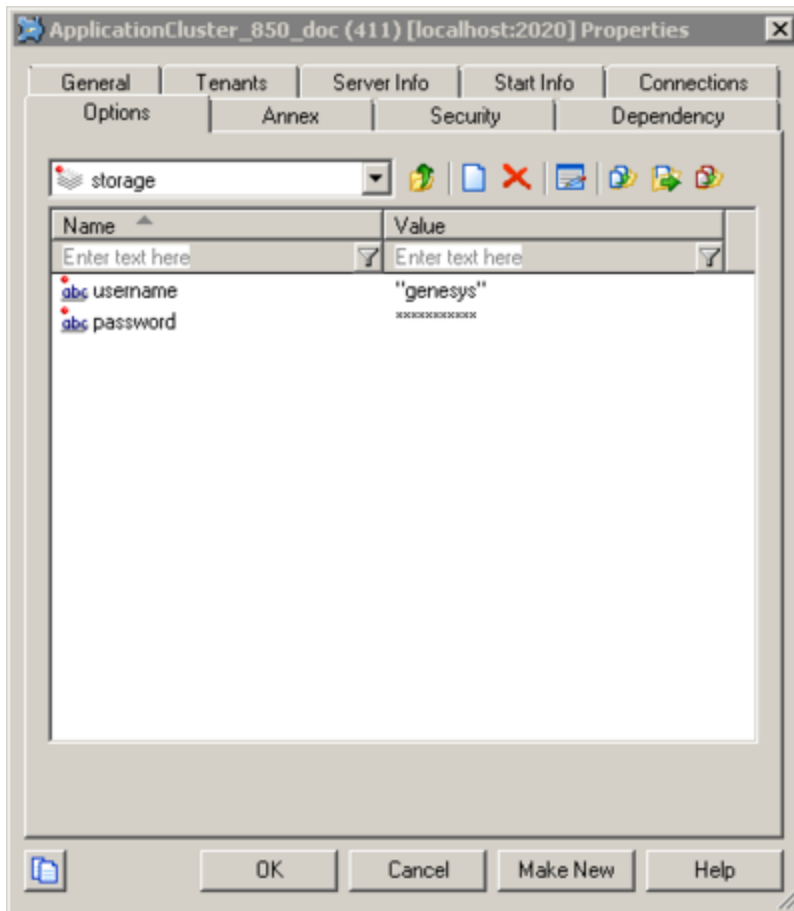
Basic Authentication Username and Password for GMS Node API:



Basic Authentication Username and Password for Service API:



Basic Authentication Username Password for Storage API:



Digital Channels API

Starting in 8.5.108, Basic Authentication is supported for the Digital Channels API. You can configure Basic Authentication for your GMS application using the options described above and these settings will apply to all of your services; or, you can turn Basic Authentication on and off on the Digital Channels channel or service level independently from GMS.

For compatibility reasons, Basic Authentication for Digital Channels API is turned off by default. To enable this feature, create the `disable_authentication` option in the appropriate media section (chat or email) and set this option to `false`.

```
disable_authentication=false
```

By default, `disable_authentication=true` and this disables Basic Authentication for Digital Channels API even if you configure Basic Authentication in your GMS configuration.

You can also define and override the username and password options either in server, media, or media service sections. Refer to Digital Channels API [configuration](#) for additional details. .

Client Side

When Basic Authentication is turned on from the GMS server-side, the client must manage the HTTP Authentication header to set the username and password in the Authorization header. Only HTTP Basic Authentication is supported. The header request for authentication should look like the following (credential is a string with a specific format [username:password], and base64 encoded):

```
Request
> GET /genesys/1/storage/id HTTP/1.1
> Host: localhost:8080
> Accept: */*
> Authorization: Basic ZGVmYXVsdDpwYXNzd29yZA==
```

```
Response
< HTTP/1.1 200 OK
< Date: Thu, 13 Feb 2014 14:44:14 GMT
```

If the authentication fails, the response looks like this:

```
Request
> GET /genesys/1/storage/id HTTP/1.1
> Host: localhost:8080
> Accept: */*
> Authorization: Basic ZGVmYXVsdDpwYXNzd29yZA==

Response
< HTTP/1.1 401 Unauthorized
< Date: Thu, 13 Feb 2014 14:47:55 GMT
< WWW-Authenticate: Basic realm="Genesys Application Configuration Needed"
< Content-Length: 0
```

Example:

Configuration Manager is configured for the `service.request-interaction` section using the following basic authentication parameters:

- username = genesys
- password = genesys

Request without credential:

```
POST /genesys/1/service/request-interaction HTTP/1.1
Host: localhost:8080
Accept: */*
Content-Length: 40
Content-Type: application/x-www-form-urlencoded
```

Response with the authentication error:

```
HTTP/1.1 401 Unauthorized
Date: Thu, 13 Mar 2014 07:55:38 GMT
WWW-Authenticate: Basic realm="Genesys Application Configuration Needed"
```

The same request with credential:

```
POST /genesys/1/service/request-interaction HTTP/1.1
```

Stat Service API

Two access interfaces are available for the Stat Service API.

One interface is for external access with Basic Authentication.

Example:

```
curl --request POST \  
  --url http://demosrv-gme.genesyslab.com/genesys/1/statistic \  
  --header 'authorization: Basic ZGVmYXVsdDpwYXNzd29yZA==' \  
  --header 'content-type: application/x-www-form-urlencoded' \  
  --data objectId=KSippola \  
  --data objectType=Agent \  
  --data tenant=Environment \  
  --data metric=TotalLoginTime
```

And the other interface is for internal usage (that is, internal access without authentication, for instance, between an ORS strategy and GMS).

```
curl --request POST \  
  --url http://demosrv-gme.genesyslab.com/genesys/1/internal_statistic \  
  --header 'content-type: application/x-www-form-urlencoded' \  
  --data objectId=KSippola \  
  --data objectType=Agent \  
  --data tenant=Environment \  
  --data metric=TotalLoginTime
```

In the external access interface for the Stat Service API, Basic Authentication is achieved using credentials from the Config Server. Configure a user in CME/GAX, in the **Member Of** tab add Administrator, and in the **Annex** tab specify the following key/value:

```
[gms]  
roles=Administrator,Supervisor
```

For the other GMS services/APIs, authentication is set up in the GMS application options:

- For all API authentication setup, the option can be set in the server section using the following options:
username and password
- For individual selection of authentication setup per service, the username and password options can be set in the following sections:
 - callback section for all callback services (Callback API)
 - service.<callback name> section for a callback service (Callback API)
 - storage section for the Storage API
 - notification section for the Subscribe/Publish API (Notification API)

Transport Layer Security for Third-Party Servers

Genesys Mobile Services (GMS) supports Transport Layer Security (TLS), which enables cryptographic and trusted communications between Genesys clients and servers.

TLS features include:

- Upgrade mode for Configuration Server
- No mutual TLS mode where server and client exchange their certificate (only server certificate is checked)

See the [Genesys Security Deployment Guide](#) for additional information about TLS.

Supported TLS versions

The list of supported TLS versions depends on the latest third-party Jetty library embedded in GMS. As a best practice, Genesys recommends using the latest GMS version available as GMS third-party libraries are updated regularly to fix issues and vulnerabilities. For further details about TLS usage in Jetty, read the *Configuring SSL/TLS* section in [Jetty Official Documentation](#).

Important

- TLSv1.0 and TLSv1.1 are no longer supported in the latest Jetty versions and therefore should not be used with GMS.

For further details about TLS versions, you can also check the [Transport Layer Security](#) Wikipedia page.

TLS Interconnections in GMS Cluster

To use SSL/TLS for all incoming GMS connections for one node or for a cluster of nodes, you must set up your nodes to use the SSL/TLS port, by using the following options:

- `server/web_scheme = https` (to change from the default http protocol)
- `server/web_port = 443` (or 8443, instead of using 80 or 8080)

Instead of using SSL/TLS certificates, you can also make GMS trust everything with the following

option: gms/http.ssl_trust_all=true

GMS now supports secure connections towards eServices, Chat Server, E-mail Server Java, and Universal Contact Server. To implement TLS to Chat Server, you must set up the trust server mode described above.

TLS Customization

To disable SSL 2.0 and SSL 3.0, edit the JDK security configuration file:

- Open the `jdk-17.x/conf/security/java.security` file and update the following list:
- `jdk.tls.disabledAlgorithms=SSLv2,SSLv3, TLSv1, TLSv1.1, RC4, DES, MD5withRSA,`

Starting in GMS 8.5.227+, to modify the list of supported TLS protocols, open the `etc/jetty-ssl-context.xml` file and edit the properties as follows:

```
<Set name="SniRequired">
  <Property name="jetty.sslContext.sniRequired" default="false"/>
</Set>
<Set name="IncludeProtocols">
  <Array type="String">
    <Item>TLSv1.3</Item>
    <Item>TLSv1.2</Item>
  </Array>
</Set>
<Set name="ExcludeProtocols">
  <Array type="String">
    <Item>TLSv1</Item>
    <Item>TLSv1.1</Item>
    <Item>SSLv3</Item>
  </Array>
</Set>
```

Important

Do not try to authorize protocols that the Jetty third-party library do not support. Genesys recommends using the above configuration as a best practice.

GMS TLS Connections with other Genesys Servers

The following table summarizes the GMS TLS connection support for Genesys servers.

GMS connection to	TLS support	Comment
Configuration Server	Yes	Upgrade mode only.
Message Server	Yes	TLS server port must be enabled.

GMS connection to	TLS support	Comment
Statistics Server	Yes	<p>No special procedures. Statistics Server is configured to listen to a TLS port using certificates.</p> <p>In the GMS Connections tab, add StartServer TLS connection (one-way or two-way "mutual" authentication) and copy server certificates on GMS hosts if needed.</p>
Chat Server	Yes	TLS between GSG/GMS and Chat Server in trust server mode (do not check the certificate).
Universal Contact Server	Yes	TLS between GSG/GMS and Universal Contact Server in trust server mode (do not check the certificate).
E-mail Server Java	Yes	TLS between GSG/GMS and E-mail Server Java in trust server mode (do not check the certificate).
Orchestration Server	Yes	You can set up an HTTPS connection. Not configured at startup (that is, not in the GMS Connection tab). Note: GMS uses HTTPClientFactory, and a TLS option can be set (section gms, option http.ssl_trust_all, value=false, true).
Web API Server	Yes	You can set up an HTTPS connection. Not configured at startup (that is, not in the GMS Connection tab). Note: GMS uses HTTPClientFactory, and a TLS option can be set (section gms, option http.ssl_trust_all, value=false, true).
Universal Routing Server	Yes	You can set up an HTTPS connection. Not configured at startup (that is, not in the GMS Connection tab). Note: GMS uses HTTPClientFactory, and a TLS option can be set (section gms, option http.ssl_trust_all, value=false, true).

Single Sign-On (SSO) (Deprecated)

Important

Deprecation notice: Beginning with GMS version 8.5.240 and later, the Single Sign-On (SSO) functionality has been deprecated.

Important

- This feature requires specific configuration/updates on [Genesys Management Framework components](#) (LDAP, IdP, Config Server, and so on).
- Single Sign-On is available only if GMS is deployed with JDK 8.

GMS version 8.5.219.03 and later enables you to use Single Sign-on (SSO) and SSO Logout (SLO) with the GMS Service Management UI. This page describes the settings needed to configure GMS to use your existing SSO infrastructure.

Login

Initiates Security Assertion Markup Language (SAML) login procedure.

`http://<gmshost>:<gmsportport>/genesys/admin/`

All authenticated Genesys users defined in Configuration Manager can access the GMS Service Management UI. GMS requires a valid user defined in Configuration Manager in order to allow administration tasks. Genesys Config Server must be configured to use external authentication functionality; Config Server users must be defined to use external authentication, pointing to the authentication system (LDAP, and so on).

Logout

Close the browser or remove browser cookies.

Deployment

SSO deployment requires the following steps:

Start

1. Uncomment the SAML parameter in the launcher.xml file.
2. Create keystore.
3. Create server-settings.yaml file and configure the following settings:
 - adminUrl
 - caCertificate
 - jksPassword
 - encryptionKeyName
 - signingKeyName
 - identityProviderMetadata: idp-metadata.xml
4. Start GMS.
 - Generate GMS metadata.
 - Update Identity Provider (IdP) information with GMS metadata.

End

Launcher.xml

Uncomment the following parameter in launcher.xml:

```
<parameter name="saml-settings" displayName="saml-settings" mandatory="false">
  <description><![CDATA[GMS Server SAML init]]></description>
  <valid-description><![CDATA[]]></valid-description>
  <effective-description/>
  <format type="string" default="server-settings.yaml" />
  <validation></validation>
</parameter>
```

Generating Security Keys

To generate a keystore, you can use the keytool utility that is included with Java SDK. To generate a JKS keystore, use the following command:

```
keytool -genkey -keystore keystore.jks -alias <encryptionKeyName> -keypass <signingKeyName>
-storepass <jksPassword> -dname <distinguished_name>
```

For example:

```
keytool -keystore /security/keystore.jks -alias genesys -keypass genesys -storepass genesys
-dname "CN=gms.genesys.local, OU=R&D, O=Genesys, L=France, S=Finistere, C=FR"
```


server-settings.yaml

Security Keys

In order to enable SAML, you must specify the following mandatory properties in the general section into server-settings.yaml:

- **adminUrl** (mandatory) - The URL will be used as a unique entity ID in SP metadata.
- **caCertificate** (mandatory) - A path to a key storage in JKS format containing all necessary keys.
- **jksPassword** (mandatory) - A password for the key storage specified above.

Important

Identity Provider (IdP) and Service Provider (SP) must use the same HTTP scheme. For example, if **adminUrl** is HTTPS in the above file, then the IDP configuration must also provide an HTTPS endpoint.

SAML Settings Section

In order to enable SAML, you must specify the following mandatory properties in the samlSettings section into server-settings.yaml:

- encryptionKeyName
- signingKeyName
- identityProviderMetadata

Settings

Name	Mandatory?	Description
encryptionKeyName	Yes	SAML encryption key name. This key must be present in the JKS key storage specified above. This key is used to encrypt the SAML message sent to IdP.
signingKeyName	Yes	SAML signing key name. This key must be present in the JKS key storage specified above.
responseSkewTime	No	Sets maximum difference between local time and time of the assertion creation, which still allows messages to be processed. Determines the maximum difference between clocks of the IdP and SP servers.

Name	Mandatory?	Description
		Defaults to 60 seconds.

Note: You can use the same key for signing and encryption.

Identity Provider

Name	Mandatory?	Description
identityProviderMetadata	Yes	Identity Provider XML metadata file path or URL. If the IdP metadata file is exposed by the remote server over HTTP, it is possible to also specify the URL (default request timeout of 5 seconds will be applied). Check the metadata URL of your IdP server.

Example

```
adminUrl: http://<gmshost>:<gmsportport>/genesys/admin
caCertificate: c:/GMS//keystore.jks
jksPassword: password
samlSettings:
  encryptionKeyName: client
  signingKeyName: client
  identityProviderMetadata: idp-metadata.xml
```

Generating GMS Metadata

GMS metadata (SP metadata) are available at the following URL:

```
http://<gmshost>:<gmsportport>/genesys/saml/metadata
```

Use this file to update your IdP server.

Hiding Selected Data in Logs

This feature implements a Genesys Standard detailed in the [Genesys Security Deployment Guide](#). It enables you to hide selected KV pairs in the User Data, Extensions, and Reasons attributes of log messages generated by Genesys Mobile Services.

- You can choose to hide just the value itself by replacing it with a series of asterisks (*), or you can remove the whole KV pair from the log output.
- You can also hide only part of the value in a particular KV pair.
- The data can be masked completely or partially, or identified by specified characters (called tags).

This provides the intended security, but with enough data to use for tracking field values, if necessary.

Important

Passwords are automatically hidden in GMS logs.

Configuring [log-filter] and [log-filter-data] sections

This feature is implemented by defining the following configuration options in the GMS Application object:

- **default-filter-type** in the **[log-filter]** section defines the treatment for all KV pairs in the User Data, Extensions, and Reasons attributes.
 - This setting will be applied to the attributes of all KVList pairs in the attribute except those that are explicitly defined in the **[log-filter-data]** section.
- One or more **<key-name>** options in the **[log-filter-data]** section define the treatment for specific keys in the log, overriding the default treatment specified by **default-filter-type**.
 - If no value is specified for this option, no additional processing of this data element is performed.

Important

The default settings of the options enable all data to be visible in the log.

You can get additional implementation samples in the [Genesys Security Deployment Guide](#). For detailed descriptions of the configuration options used to configure this feature, refer to the [Framework Configuration Options Reference Manual](#).

Supported Filters

Important

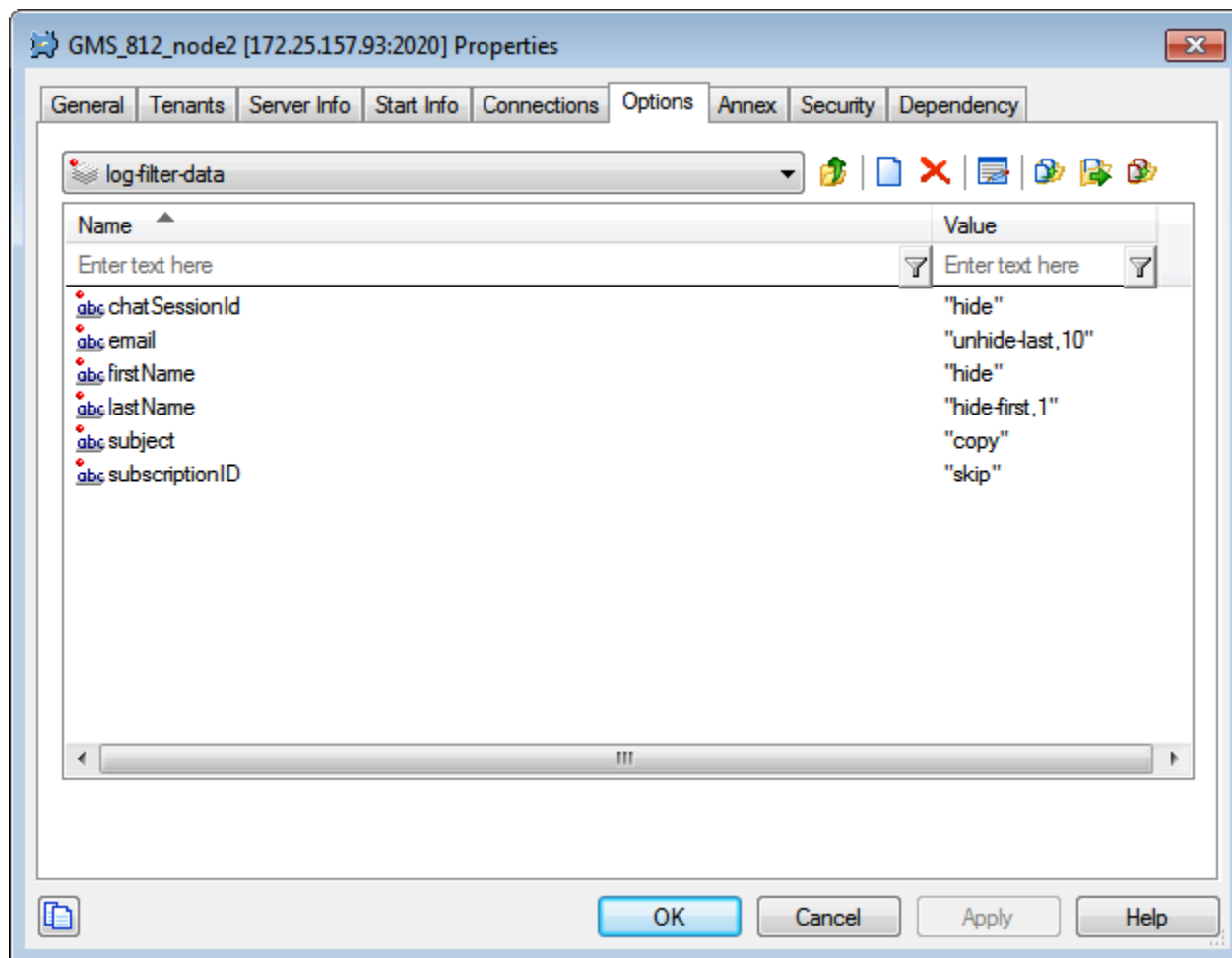
custom-filter options are not supported for now.

Filter Name	Description
copy	The keys and values of the KVList pairs are copied to the log.
hide	The keys of the KVList pairs are copied to the log; the values are replaced with strings of asterisks.
hide-first,<n>	The keys of the KVList pairs are copied to the log; the first <n> characters of the value are replaced with asterisks. If <n> exceeds the number of characters in the value, the number of asterisks will be equal to the number of characters in the value.
hide-last,<n>	The keys of the KVList pairs are copied to the log; the last <n> characters of the value are replaced with asterisks. If <n> exceeds the number of characters in the value, the number of asterisks will be equal to the number of characters in the value.
unhide-first,<n>	The keys of the KVList pairs are copied to the log; all but the first <n> characters of the value are replaced with asterisks. If <n> exceeds the number of characters in the value, the value of the key appears, with no asterisks.
unhide-last,<n>	The keys of the KVList pairs are copied to the log; all but the last <n> characters of the value are replaced with asterisks. If <n> exceeds the number of characters in the key, the value of the key appears, with no asterisks.
skip	The KVList pairs are not copied to the log.

Examples

Here is a configuration sample, which hides from the logs the chat session ID and replaces some characters with 'x' in specified fields.

```
[log-filter-data]
chatSessionId=hide
email=unhide-last,10
firstName=hide
lastName=hide-first,1
subject=copy
subscriptionID=skip
```



This set of filters will generate the following output in the logs:

```
14:56:07.422 Dbg 09900 [qtp1928680974-298] (POST) Client IP Address: 127.0.0.1,
URI:http://127.0.0.1:8080/genesys/1/service/8ele3406-8e48-4846-83f1-c7be1818acf7/ixn/chat
14:56:07.431 Dbg 09900 [qtp1928680974-298] Params: KVLList:
'lastName' [str] = "*oe"
'firstName' [output suppressed]
'email' [str] = "*****@gmail.com"
'subject' [str] = "Question about your product"
```

Configuring [log-hidden-attributes] Section to Hide Selected

Internal Message Attributes

Introduced in 8.5.200

This feature enables you to hide selected attributes in the log messages generated by Genesys Mobile Services when communicating with other Genesys components. When configured, the selected attribute will be logged as [output suppressed] instead of the attribute value. This feature provides the intended level of security with enough data to ensure troubleshooting if necessary.

Here is a configuration sample which hides the content of the chat or email text from the logs while the GMS node communicates with the Chat Server or E-mail Server Java:

```
[log-hidden-attributes]
FlexChat.EventInfo=Text
FlexChat.MessageText=Text
FlexChat.NoticeText=Text
EspEmail.RequestCreateWebEmailIn=Text
```

Important

Options and values are case-sensitive.



If you need additional assistance with configuring a specific key name for the attribute that you want to hide, please contact Genesys Customer Care Team:

1. Provide a sample of the logs that cover your test scenario.
2. Indicate which attribute you would like to hide.

Starting and Stopping GMS

Overview

You can start and stop Genesys Mobile Services in the following ways:

Objective	Related procedures and actions
Using Solution Control Interface (SCI)	Complete the following procedure:  Starting and Stopping GMS Using Solution Control Interface
Using Genesys Administrator	Complete the following procedure:  Starting and Stopping GMS Using Genesys Administrator

Starting and Stopping GMS Applications Using Solution Control Interface

Prerequisites

- Genesys Mobile Services is installed.

Start

1. From the Applications view in SCI, select Genesys Mobile Services Application object on the list pane.
2. Click the appropriate button (Start, Stop, or Stop Gracefully) on the toolbar, or select that command from either the Action menu or the shortcut menu. (Right-clicking your Application object displays the shortcut menu.)
3. Click Yes in the confirmation box that appears. Your application obeys the command that you selected.

End

For information about how to use SCI, refer to [Framework Solution Control Interface Help](#).

Starting and Stopping GMS Applications Using Genesys Administrator

Prerequisites

- Genesys Mobile Services is installed.

Start

1. Log in to Genesys Administrator.
2. On the Provisioning tab, select Environment > Applications.
3. Select the GMS Application.
4. Right-click the application, and then select the appropriate command from the drop-down menu. These three choices apply:
 - Start applications
 - Stop applications
 - Stop applications gracefully

End

Configuring a Delay for Graceful Shutdown

When you select a graceful shutdown, the Solution Control Server (SCS) sends a suspend message to GMS, so that GMS sets itself to OFFLINE. Load Balancer will no longer send requests to this GMS. After a configurable delay, SCS sends a STOP signal, and the GMS will stop. To configure this delay:

Start

1. In Configuration Manager, select the GMS Application.
2. On the *Annex* tab, *sml* section, set the suspending-wait-timeout to the desired value:
 - Default Value: 10
 - Valid Values: 5-600

End

Troubleshooting

How to display the correct ANI on the agent and customer's end in a GMS Agent-First Scenario?

If you have a pool of external numbers and one of them is used for an outbound customer call, if you want the customer to get the correct ANI displayed, follow the instructions below.

For each external number of the pool, create a **DN** of type trunk in your configuration.

- Each trunk DN must have a unique prefix option, an empty replace-prefix option, and a unique cpn option.
- Other options can be identical for all of the trunks.

In this scenario, the strategy adds the matching trunk to the dialing number prefix with the proper cpn and prefix options. Then, after finding the matching trunk, the SIP Server removes the prefix option by applying the replace-prefix empty option. As a result, the SIP Server uses the cpn value as an invite for the username.

If no Cometd handshake/connect is done, which mode does GMS use by default? Can GMS initiate a chat interaction in polling mode only, without CometD?

When the chat interaction is initiated, GMS uses both modes, CometD and polling.

- If you already use a CometD connection for chat, you can poll the chat session at any time.
- If you start a chat session using polling, you cannot use CometD to receive chat messages for the same session later.

In a GMS-clustered architecture, do all the GMS nodes communicate with a particular client across the _genesys CometD channel?

No. Only one GMS node of the cluster handles the CometD connection with a given client application at a time. For example, if the client application requests a CometD/handshake request, it will receive only one response from one GMS node; it will not receive multiple responses from each GMS node of

the cluster.

Error when starting cqlsh

If you start `bin/cqlsh` and receive the following error:

Traceback (most recent call last):

...

LookupError: unknown encoding:

Change the line in `bin/cqlsh` from:

```
self.output_codec = codecs.lookup(encoding)
```

to:

```
self.output_codec = codecs.lookup("UTF-8")
```

Endpoint Collision: External Cassandra (on start message) A node with address <IP address> already exists, cancelling join

See the Cassandra documentation for [replacing a dead seed node](#).

Error in GMS logs: WARNING: Failed to check connection:

`java.net.SocketException: Too many open files`

On RHEL Systems, change the system limits from 1024 to 10240 in `/etc/security/limits.d/90-nproc.conf`, and then start a new shell for these changes to take effect:

```
soft nproc 10240
```

See the Cassandra documentation for [recommended settings](#).

GMS does not install / start on Linux 64

If you install GMS IPs on a 64-bit Linux host, install the compatibility packages on the Linux host. The compatibility packages are available with the OS distribution media.

When several GMS nodes are in a cluster, the following logs are on all GMSs:

New node has joined the ring: http://135.39.40.56:8080/genesys

Node has been removed from the ring: http://135.39.40.56:8080/genesys

New node has joined the ring: http://135.39.40.56:8080/genesys

Node has been removed from the ring: http://135.39.40.56:8080/genesys

New node has joined the ring: http://135.39.40.56:8080/genesys

Node has been removed from the ring: http://135.39.40.56:8080/genesys

New node has joined the ring: http://135.39.40.56:8080/genesys

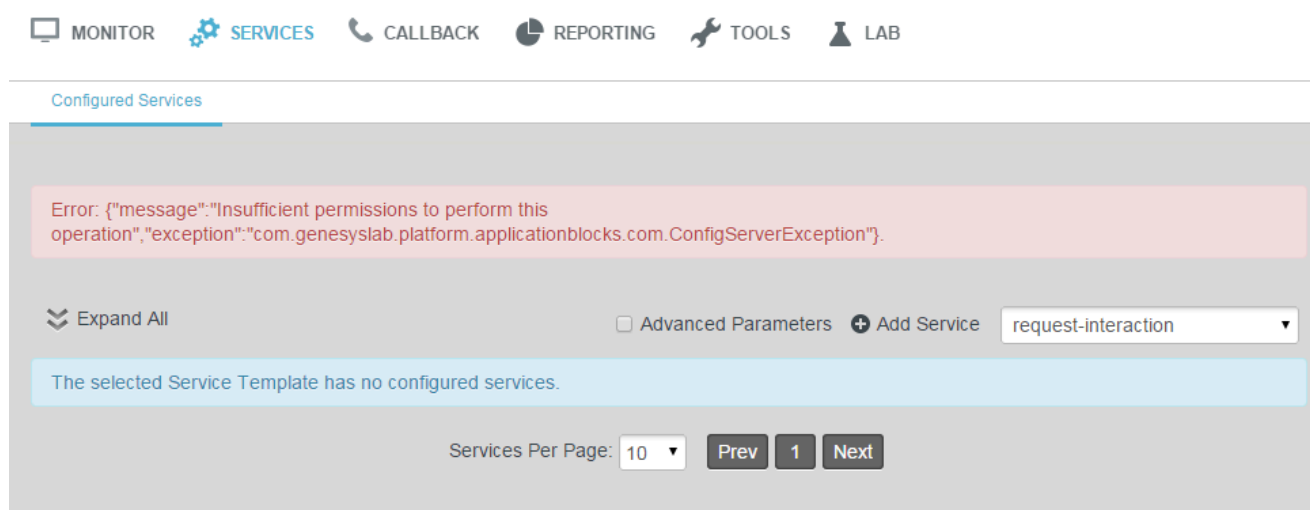
Node has been removed from the ring: http://135.39.40.56:8080/genesys

Synchronize the clock of all GMS nodes. On Windows, use following command:

```
net time \\<ComputerName> /set /yes
```

Where \\<ComputerName> specifies the name of a server you want to check or with which you want to synchronize.

I cannot update GMS configuration through the GMS Service Management UI, and the following error appears in the GMS logs:



```
Insufficient permissions to perform this operation
com.genesyslab.platform.applicationblocks.com.ConfigServerException: Insufficient
permissions to perform this operation (ErrorType=CFGNoPermission,
ObjectType=CFGApplication, ObjectProperty=CFGDBID)
```

Make sure that all GMS applications have the user assigned in the *Security* tab, in Configuration Manager. Or, in Genesys Administrator, browse the User for Log On Account in the **Server Information** section of your application.

Most Common System Errors

This page is a non-exhaustive list of common errors that GMS can throw in the console and in the log files of GMS.

At Start-up

Transport error 202: bind failed: Address already in use

Issue: The provided jetty port is already in use.
Action: Check if another application uses this port.

Cannot connect to ConfigServer 'ConfigServer' at host '[host]', port [port], reason '[host:port] Error opening connection'

Issue: GMS cannot connect to the Configuration Server.
Action: Check your connection.

Cannot connect to ConfigServer '[ConfigServer]' at host '[host]', port [port], reason '[host:port] Error registering. Code: 5835015; Server with this name is already running'

Issue: Another GMS instance already uses this application name.
Action: Change the name of your application.

Cannot start Pelops Client for Cassandra [...]: connect timed out

Issue: GMS cannot connect to your external Cassandra cluster.
Action: Check your configuration (Cassandra section, nodes option) and/or your Cassandra cluster's status.

Service option [option name] is not configured for service [service name]

Issue: The service is not configured and the associated GMS feature will not work.
Action: Check your service configuration.

production [provider] provider can not be set for provider [push section] reason: Mandatory option [option name] not specified for provider section [push section]

Issue: You did not properly set your GMS configuration for push notification.
Action: Check the [push notification](#) section and options.

Cannot connect to LocalControlAgent 'LCA'

Issue: LCA is not installed or started on the host.

Action: Start LCA to fix the issue.

Wrong value [value] for option [option name]

Issue: You set an incorrect value for the given option. GMS will use the default value instead.

Action: Update this option's value in your configuration.

Errors in GMS Core Functionality

Failure contacting master Response is null

Issue: GMS cannot request the resource on master node, a new request is sent to the GMS master node.

Error configuring HttpClient with specified configuration [exception]

Issue: GMS cannot create the HTTP client and retrieve resources from the master node.

Content in file with .json extension is not JSON formatted. Assuming this is not a template definition file [exception]

Issue: GMS cannot parse JSON content of file with .json extension.

Action: Check the content of the JSON file in your service template file.

JSON file ([file name]) is not a template file. Template name specified doesn't match the id in the template file or template type is not specified.

Issue: The id parameter of the JSON file in the template file does not match the name of the file, or the id parameter is not set.

Action: Check the service template name and the JSON id parameter of the JSON file which is part of the service template zip file.

Cannot find jettyConfig!

Issue: GMS cannot find the <GMS_HOME>/etc/jetty-http.xml file.

Action: Check your [GMS installation](#).

Cannot parse file [jetty file]: [exception]

Issue: GMS cannot parse the content of jetty-http.xml file.

Action: Check the XML format of this file.

Error retrieving connection in jetty-http.xml [exception]

Issue: GMS cannot find the following properties in the jetty-http.xml file:

```
<Set name="port"><Property name="jetty.port" default="8080"/></Set>
```

Action: Check these properties for each server connector (HTTP, HTTPS...).

Error with Jetty Snippet! [exception]

Issue: GMS cannot find or process jetty port restriction for delivering a jetty template snippet.

Action: Check that the jettyconfig.ftl file is available in the gsg-core jar file.

Error with Jetty Snippet template! [exception]

Issue: GMS cannot process jetty port restriction in jetty template file for delivering a jetty snippet.

Action: Check that the jettyconfig.ftl file available in gsg-core jar file is valid.

Callback Service Errors

Unable to process Callback with ServiceId=[Uuid]: [exception]

Issue: GMS cannot process the callback interaction.

Action: Check ORS status and your Cassandra storage.

Unable to process URS statistic [exception]

Issue: GMS cannot process URS queries.

Action: Check the URS status and configuration (url, host, port); GMS returns the default value ('0') for this query.

_chat_endpoint is empty for this service, although parameter _media_type=chat

Issue: The _chat_endpoint is empty although the media type is chat.

Action: Check the service option and the default_chat_endpoint option in the chat section.

Debug error: _chat_endpoint not found for this service although parameter _media_type=chat

Issue: The _chat_endpoint option is incorrect.

Action: Check the default_chat_endpoint option in the chat section.

Debug error: `_client_timeout` is empty for this service, although parameter `_media_type=chat`

Issue: The `_client_timeout` option is empty although it is required for the chat media.

Action: Check the `_client_timeout` option in the chat section.

Debug error: `_client_timeout` not found for this service although parameter `_media_type=chat`

Issue: The `_client_timeout` option is not set and GMS will use the default option value (900 seconds).

Action: Check the `_client_timeout` option in the chat section.

Debug error: `_client_timeout_notification` is not defined in chat service section

Issue: The `_client_timeout_notification` option is not set and GMS will use the default option value (90 seconds).

Action: Check the `_client_timeout_notification` option in the chat section.

Debug error: `_client_timeout_notification` is not defined in chat section

Issue: The `_client_timeout_notification` option is not set and GMS will use the default option value (90 seconds).

Action: Check the `_client_timeout_notification` option in the chat section.

Debug error: `_client_timeout_notification` is not an Integer, option set to default value: [defaultChatClientNotificationTimeout]

Issue: The `_client_timeout_notification` option is incorrect and GMS will use the default option value (90 seconds).

Action: Set an integer value for the `_client_timeout_notification` option in the chat section.

Session doesn't exist yet or was not started by GMS : [Interaction ID] = [session ID]

Issue: The session is not available in storage.

Unable to submit event to Context Services: [exception]

Issue: GMS cannot use Context Service to process the callback event.

Debug error: Discarding error from ORS. Skip based on response: [message]

Issue: GMS cannot cancel the callback request in ORS.

Action: Check that ORS is up and ready.

Callback filtering criteria issue: unable to parse callback desired time [exception]

Issue: GMS cannot parse the callback desired time parameter.

Action: Check the callback desired time parameter in your configuration.

Callback filtering criteria issue: unable to configure time operation [exception]

Issue: GMS cannot register the callback time operation.

Action: Check the springframework SpEL feature.

Unable to peek EWT statistic: [exception]

Issue: GMS cannot retrieve statistics about Estimated Waiting Time from Stat Server.

Action: Check the statistic request and the Stat Server configuration and status.

Unable to peek EWT statistic: parameter `_vq` is not defined.

Issue: GMS cannot get statistics about Estimated Waiting Time because the virtual queue parameter is missing from the query or service definition.

Action: Check your request or service. By default, GMS returns the default value ('0').

No ewt value in fetched URS Statistic

Issue: The URS response does not contain the 'ewt' key.

Action: Check URS status and configuration; the default value is returned ('0').

Unable to process Callback with ServiceId=[Uuid] - service deleted?

Issue: GMS cannot find the callback interaction in Cassandra storage. The interaction may have been deleted.

Unable to process Callback with ServiceId=[Uuid]: request probably re-scheduled later...

Issue: The desired date is not defined or not included in the slot of EWT time; the callback must be re-scheduled by the customer.

Completed Callback request ServiceId=[Uuid] : [exception]

Issue: GMS submitted a callback request to ORS and it failed.

Action: Consider updating the callback in the storage with the new callback state: completed and the reason: submit failed.

Unable to process queue metadata [exception]

Issue: GMS cannot parse or create JSON object from the callback interaction before storing it in the queue.

Action: Check your Cassandra storage.

cannot store user data for callback service: [uuid]

Issue: GMS cannot access and store user data in Cassandra.

Action: Check your Cassandra status.

Chat Service

Chat Load Balancer request to [request] failed: [HTTP status]

Issue: The request sent to the WebAPI Server failed, resulting in the given HTTP status (not 200 OK).
Action: GMS automatically submits the request to another Web API Server.

Chat request failed: [request] [exception]

Issue: The request sent to the WebAPI Server failed, resulting in IO issue with the Web API Server.
Action: GMS automatically submits the request to another Web API Server.

Chat removedIdleChatJob failed. [Exception]

Issue: GMS cannot process the *remove idle job* for chat because of a storage issue.
Action: You should check your Cassandra status.

Chat Jobs processing error: [exception]

Issue: GMS cannot schedule a new chat task in the job engine and refresh the transcript.
Action: You should check the GMS cluster status, the Cassandra status (up), and if there is a promoted master.

ChatJobWorker: Cannot perform chat refresh. Service [Service Id] not found

Issue: The service Id cannot be found. The service has been deleted or its TTL (Time Tracking Loop) has expired.
Action: Because the service expired, you should request a new session.

ChatJobWorker: Unable to retrieve data from StorageService for Service [Service Id]

Issue: The user data for the given service does not exist in storage. The user data has been deleted or its TTL (Time Tracking Loop) has expired.
Action: Because the service expired, you should request a new session.

ChatJobWorker: Error refreshing chat text for Service [Service Id]

Issue: GMS is unable to refresh the chat transcript for this chat session.
Action: Check if the chat session is already terminated or check your chat server connection.

ChatJobWorker: Unable to send push notification to client for service [service Id]

Issue: GMS is unable to push notifications to the client application.

Action: Check your push provider configuration and check the push section in GMS application options.

Job Engine Errors

Cannot execute job: [exception]

Issue: GMS cannot start the job.

Action: Check the GMS configuration, check the Cassandra storage.

Cannot delete job: [exception]

Issue: GMS cannot delete the job.

Action: Check the GMS configuration and your Cassandra storage.

Cannot find master server: [exception]

Issue: GMS cannot get the master server.

Action: Check the GMS configuration and your Cassandra storage.

Cannot getLastScheduledTime: [exception]

Issue: GMS cannot get the last scheduled time.

Action: Check the GMS configuration and your Cassandra storage.

Cannot trackLastScheduledTime: [exception]

Issue: GMS cannot get the last scheduled time.

Action: Check the GMS configuration and your Cassandra storage.

Cannot accept job: [exception]

Issue: GMS cannot accept job.

Action: Check the GMS configuration and your Cassandra storage.

Cannot reject job: [exception]

Issue: GMS cannot reject the job.

Action: Check the GMS configuration and your Cassandra storage.

Cannot completed job: [exception]

Issue: GMS cannot complete the job.

Action: Check the GMS configuration and your Cassandra storage.

Cannot failed job: [exception]

Issue: GMS cannot process in the task manager the 'failed' request from a job worker.

masterUrl is null or empty. Cannot send response to master

Issue: GMS cannot send request to the master node.

Action: Check the GMS configuration and your Cassandra storage.

cannot create HTTP client for URL=[URL]

Issue: GMS cannot create the HTTP client for sending request to the master node.

Action: Check the GMS configuration.

Failed to submit request, httpStatus=[HTTP code status]

Issue: GMS cannot send the HTTP request to the master node.

Action: Check the GMS configuration and master node status.

ORS Service

Service type is not configured for service: [service name]

Issue: This service does not exist in GMS application configuration.

Action: Check your service application.

Error with parameter [key]; exception: [exception]

Issue: The value set for the given key is incorrect.

Action: Modify the input value for this key.

handleCalendarService: parameter nDays is out of Range [1, 366], actual value is: [days]. It will be reset to 1.

Issue: The number-of-days input parameter of this service is out of range and is reset to 1.

Action: You should set a proper value for this parameter.

Unable to release resource: _access_number

Issue: GMS cannot release the resource with the given access number.

_client_timeout option is not defined in chat service section

Issue: The _client_timeout option is missing in the chat service section of your GMS configuration.

Action: You must set this option.

`_client_timeout` option is not an Integer, option set to default value: 900

Issue: The `_client_timeout` option (in the chat service section of your GMS configuration) is not set to an integer value and its default value (900 seconds) was used instead.

Action: Check this option's value in your configuration.

Statistics Service

Unable to read configuration: [exception]

Issue: GMS cannot read the tenant data in the Configuration Server.

Error updating statistic value for stat: [statistic]

Issue: GMS cannot add the statistic value to the cache.

Action: Check your Cassandra storage.

Unable to peek statistic for: [statistic], error: [exception]

Issue: GMS cannot retrieve the statistic value from Stat Server.

Action: Check the Stat Server status, then, check that your statistic request and parameters are correct.

URSStatisticService: Remote Request failed ([HTTP status code])

Issue: The URS request failed.

Action: Read the URS response for additional details.

StatisticService: objectType not valid.

Issue: The object type of the statistic request is not valid.

Action: Check the objectType parameter.

peekStatistic/getElement/EventCurrentTargetStateSnapshot: key=[reference id], value is null, take value from local cache

Issue: GMS cannot find the statistic value of this reference in the cache. GMS will use the value from the local variable instead.

StatisticService: HandleConnectionFailure: [statistic server context]

Issue: A connection failure occurred. WarmStandby is trying to setup a new connection using the configuration information for primary/backup statistic servers defined in GMS connections.

Cannot convert to String value [state value]

Issue: GMS cannot transform the statistic object into a JSON string value.

StatisticService: ChannelError [cause]

Issue: The Stat Server raised a channel error.

Unable to retrieve Statistic for String conversion

Issue: GMS cannot convert the statistic object into a JSON string because the retrieved object is null.

Cassandra Storage

Value for key ([key]) is undefined

Issue: This input key parameter has no value and will not be stored.

Error in getCometClientState

Issue: GMS cannot get data for comet client state and will use default values.
Action: Check comet client status and your Cassandra storage.

Troubles getting binary storage item [exception]

Issue: GMS cannot get the binary storage for a key.
Action: Check key value and your Cassandra storage.

Troubles getting storage data for storage: [uuid] key: [key]

Issue: GMS cannot get the data storage for a key for an uuid and returns a null value.
Action: Check the uuid value, the key value, and your Cassandra storage.

Troubles getting storage data for storage: [uuid] key:

Issue: GMS cannot get the last refresh time for the chat session.

Troubles getting file for storage: [file path] key

Issue: GMS cannot get the file content of this file path.

Found [services list size]. Retrying in 500ms.

Issue: Found several services that match. There is a Cassandra NTP synchronization issue in this cluster. The deletion process is in progress and was not taken into account instantaneously.

InterruptedException while waiting 500ms [exception]

Issue: Sleep interrupted. There is a Cassandra NTP synchronization issue in this cluster. The deletion process is in progress and was not taken into account instantaneously.

Failure updating user resource allocation [exception]

Issue: GMS cannot update the resource for a user.

Failure deleting user resource allocation ([user],[resource]) [exception]

Issue: GMS cannot delete the resource for a user.

Failure deleting failed master column [exception]

Issue: GMS cannot delete the master node of the cluster. Action: Check your node status and your Cassandra storage.

Failure creating allocation [exception]

Issue: GMS cannot store resource allocation.
Action: Check the allocation object and your Cassandra storage status.

Failure deleting allocation [exception]

Issue: GMS cannot delete the resource allocation.
Action: Check the resource name and your Cassandra storage status.

Failure reading allocations [exception]

Issue: GMS cannot read the resource allocations.
Action: Check the list of resources list and your Cassandra storage status.

Unable to fetch default document for service: [service name] [exception]

Issue: GMS cannot read the document for a service template.
Action: Check the service and file name parameters, the service template directory, and your Cassandra storage.

Error retrieving file properties from storage. ([file name])

Issue: GMS cannot read the document file properties.
Action: Check the service template directory and your Cassandra storage.

Error retrieving documents list from storage. Possibly, templates have not been uploaded. [exception]

Issue: GMS cannot read the service template properties.
Action: Check the service template directory and your Cassandra storage.

No document in storage for service ([service name])

Issue: No document exists for this service name.
Action: Check the service name parameter, the service template directory, and your Cassandra storage.

Record ([currRow]) not taken into account..

Issue: GMS cannot process the calculation of the given slice for the report.

Subscription/Notification Service

The push service reported device, specified in subscription [subscription] to be unreachable, subscription will be removed

Issue: GMS cannot push notification on the given device.

Action: Check notification device detail, or check push notification provider.

Error parsing value [target] in 'pushEnabled' list in section [section] , ignored

Issue: GMS cannot parse the comma-separated value list in the 'pushEnabled' option.

Action: Check the 'pushEnabled' option in the push provider section of your configuration.

NotificationProviderFactory: error creating the sender for [provider]
corresponding push will be disabled, reason: [message]

Issue: GMS cannot create a provider for the push section.

Action: Check the options set in the push section of this provider.

Cannot send notification, GCM returns following error code: [error code]

Issue: GMS cannot send notifications to the GCM provider.

Action: Check options in this section for this provider, check GCM provider.

Error initializing C2DM, PUSH to android will be disabled : code: [error code]

Issue: GMS cannot send notification to a C2DM provider.

Action: Check options in this section for this provider, check C2DM provider.

Error initializing C2DM, PUSH to android will be disabled : message: [error message]

Issue: GMS cannot send notification to a C2DM provider.

Action: Check options in this section for this provider, check C2DM provider.

Error initializing C2DM, PUSH to android will be disabled [exception]

Issue: GMS cannot send notification to a C2DM provider.

Action: Check C2DM provider,

Important

C2DM is to be shut down completely as of July 30, 2015. Existing C2DM developers are encouraged to migrate to Google Cloud Messaging for Android (GCM).

Error sending push to Apple service: [exception]

Issue: GMS cannot send notification to a Apple Push Notification Service provider.

Action: Check options in this section for this provider, check APNS provider.

Error sending message through Comet: [exception]

Issue: GMS cannot send notification to a Comet provider.

Action: Check options in this section for this provider, check GMS Comet provider.

url [URL] is reported to be invalid -> target unreachable

Issue: GMS cannot submit a POST request to a custom HTTP provider with the given URL,.

Action: Check custom HTTP option in GMS application push section

Error sending the message to [URL]

Issue: GMS cannot send a notification to custom HTTP/HTTP Callback provider with this URL.

Action: Check custom HTTP/HTTP Callback option in GMS application push section, check custom HTTP/HTTP Callback server

Error after sending message to [URL]

Issue: GMS cannot send a notification to HTTP Callback provider with this URL.

Action: Check HTTP Callback option in GMS application push section, check HTTP Callback server.

url , passed as deviceId ([URL]) is reported to be invalid -> target unreachable

Issue: GMS cannot create a POST request for a HTTP Callback provider with this URL.

Action: Check HTTP Callback option in GMS application push section.

Error reaching java target [target]; [exception]

Issue: GMS cannot send notifications to the Java Callback provider.

Action: Check Java Callback option in GMS application push section.

Testing Your Deployment

This chapter shows how to perform a basic deployment of your GMS application.

Important

You should not use this deployment as-is for production applications. For example, many of the settings in these procedures use default option values that may not apply to your environment.

Testing your deployment consists of the following tasks:

1. [Configuring the GMS Builtin Services](#)
2. [Testing the GMS Builtin Services](#)
3. [Configuring the ORS-based Services](#)
4. [Testing the ORS-based Services](#)

Configuring the GMS Built-in Services

Now that all of the dependencies are set up, you can start to configure the service and use the samples to test it. Three samples are available for download, and any one of these samples can be used to validate the deployment. The JavaScript sample comes pre-installed with the GMS Installation Package and is ready to use, so the instructions outlined here will focus on the JavaScript sample. To access the sample:

1. To enable the sample in the Service Management UI, configure `enable-sample = true` in your GMS configuration.
2. In your browser, open the Service Management UI by accessing either of the following URLs:
 - `<GMS Local Host>:8080/genesys/home/login.jsp#/`
 - `<GMS Local Host>:8080/genesys/home/index.jsp#/`
3. Click **Tools > Samples**.
4. Click the Scenario drop-down to display the list of scenarios supported. The first two scenarios are of type `builtin`, which means that GMS handles the service features by itself, although the actual interaction must be handled by an ORS workflow or URS strategy. The other scenarios depend on ORS-based Callback services and require advanced configuration. These instructions on this page will show you how to configure and test the `builtin` scenarios.

Prerequisite

You must have [configured the dependencies](#).

Step 1: Resource Group—Add Access Number

Why:

GMS provides this access number to the user, and the user dials into this access number.

How:

GMS Service Management UI

Procedure:

1. Go to the GMS Service Management UI > Tools > Resources.
2. Add the access number to the DNIS group.

Step 2: GMS Service—Create Service request-interaction

Why:

This service is responsible for receiving the GMS request and providing an access number to the user.

How:

GMS Service Management UI

Procedure:

1. Go to the GMS Service Management UI > Services > Configured Services.
2. Click Add Service.
3. Set Configure Service = request-interaction.
4. Set Service Name = request-interaction.
5. Click Save.

Step 3: GMS Service—Create Service match-interaction

Why:

This service helps to match a voice call with an existing GMS service responsible for providing the access number.

How:

GMS Service Management UI

Procedure:

1. Go to the GMS Service Management UI > Services > Configured Services.
2. Click Add Service.
3. Set Configure Service = match-interaction.
4. Set Service Name = match-interaction.
5. Click Save.

Step 4: GMS Service—Create Service request-access

Why:

This service lets you:

- Create a new access to a service
- Allocate a new DN in the resource group

How:

GMS Service Management UI

Procedure:

1. Go to the **GMS Service Management UI > Services > Configured Services**.
2. Click Add Service.
3. Set Configure Service = request-access.
4. Set Service Name = request-access.
5. Click Save.

REMOVE for GMS-9451? Step 5: GMS Service—Create Service request-chat

Why:

This service is responsible for receiving the GMS request and providing a URL to start the chat interaction.

How:

GMS Service Management UI

Procedure:

1. Go to the GMS Service Management UI > Services > Configured Services.
 2. Click Add Service.
 3. Set Configure Service = request-chat.
 4. Set Service Name = request-chat.
 5. Click Save.
 6. Set the service property `_chat_endpoint = Environment:gms_builtin` (Note: For single tenant:
-

Resources:gms_builtin.)

Step 6: Inbound SCXML Service—Voice

Why:

The inbound service matches the voice call with an existing GMS service. If a matching service is found, the GMS user data is attached to the interaction, and the call is routed to the agent.

How:

- Configuration Manager > Switches > SIP_Switch
- Configuration Manager > Scripts

Procedure:

1. Create a route point associated with the access number configured in the procedure [Resource Group Add Access Number](#).
2. Set Annex > Orchestration section > application = script:GMSInbound.Voice.GMSMatchBuiltin.
3. Create an enhanced routing script GMSInbound.Voice.GMSMatchBuiltin.
4. Set Annex > Application section > url = http://<gmshost:gmsport>/genesys/1/document/service_template/callback/src-gen/IPD_Voice_GMSMatch.scxml.
5. In Annex > ApplicationParms, set:
app_find_agent_timeout = 30
app_match_gms_builtin = true.
app_match_target = <target> (Example: Customer_Service@stat_server.GA).
app_no_match_target = <target> (Example: All_Standard_Agents@stat_server.GA).
app_require_access_code = false.
app_require_ani = true.
app_treatment_waiting_for_agent = <blank> (A blank value will force the service to use a packaged music file.).
6. Make sure that MSML capabilities are configured and working to play treatments. This step is required because this service includes play treatments, and has a dependency on Media Server.

REMOVE for GMS-9451? Step 7: Inbound SCXML Service—Chat

Why:

This inbound service attaches the GMS user data to the interaction and routes the interaction to the agent.

How:

- Configuration Manager > Chat Server
- Configuration Manager > Scripts

Procedure:

1. Go to Configuration Manager > Chat Server.
2. Create an end point that was specified in procedure [GMS Service Create Service request chat](#) (sub-step 6):
 - `gms_builtin = GMSInbound.Chat.QueueBuiltin`
3. Go to Configuration Manager > Scripts.
4. Create an interaction queue that you just specified, above.
 - Name: `GMSInbound.Chat.QueueBuiltin`
 - Annex > Orchestration/application = `script:GMSInbound.Chat.QueueBuiltin.Routing`
5. Create an interaction queue view.
 - Name: `GMSInbound.Chat.QueueBuiltin.View 1`
 - Annex > View/Queue = `GMSInbound.Chat.QueueBuiltin`
6. Create an Enhanced Routing Object that you just specified, above.
 - Name: `GMSInbound.Chat.QueueBuiltin.Routing`
 - Annex > Application/url = `http://<gms_host>:<gms_port>/genesys/1/document/service_template/callback/src-gen/IPD_Chat_QueueBuiltin.scxml`
 - Annex > ApplicationParms/app_find_agent_timeout = 30
 - Annex > ApplicationParms/app_match_gms_builtin = true
 - Annex > ApplicationParms/app_match_target = <target> (Example: `Customer_Service@Stat_Server.GA`)
 - Annex > ApplicationParms/app_no_match_target = <target> (Example: `All_Standard_Agents@Stat_Server.GA`)

Step 8: Interaction Workspace—Display GMS Attached Data

Why:

GMS attaches data to the call prior to routing it to the agent. This attached data is displayed to the agent when the call arrives at the agent desktop (Interaction Workspace), and helps the agent to understand the source of the

call, as well as to understand the additional information sent from the customer's device when creating the Callback.

How:

Configuration Manager > Business Attributes

1. Create a new business GMSCaseData attribute of type Interaction Operational Attribute.
2. Create new attribute values:
 - first_name
 - last_name
 - location_lat
 - location_long
 - GMS_Call_Direction
 - GMS_MatchMethod_AccessNumber
 - GMS_MatchMethod_ANI
 - GMS_MatchResult
 - GMS_MatchReason
 - GMS_ServiceName
 - GMS_UserData
3. Set the following Application > InteractionWorkspace options:
 - interaction-workspace > interaction.case-data.format-business-attribute = GMSCaseData
 - interaction-workspace > toast.case-data.format-business-attribute = GMSCaseData

Next Steps

[Test the GMS Built-in Services](#)

Testing the GMS Built-in Services

Now that you have configured the built-in services, it's time to test them.

Prerequisites

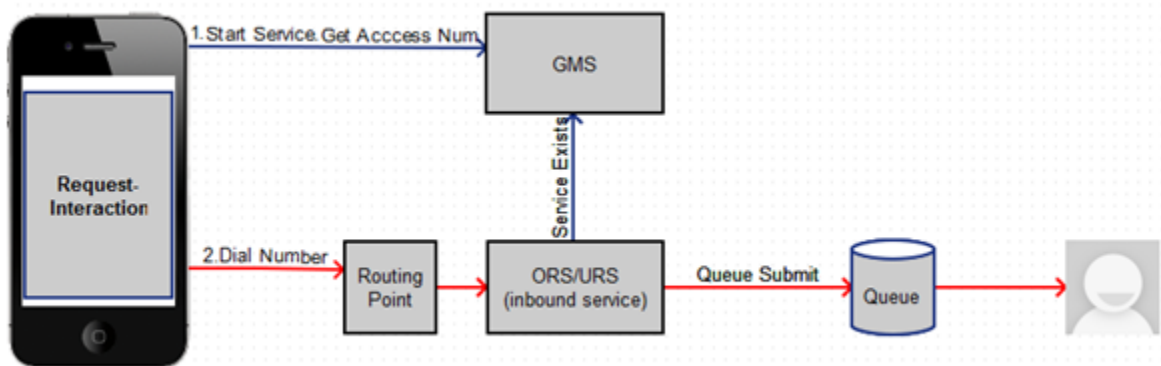
You must have completed the following:

1. [Configured the dependencies.](#)
2. [Configured the built-in services.](#)

Important

In the following scenarios, if the GMS Match fails, there will be no user data attached to the interaction.

Scenario request-interaction Test Procedure



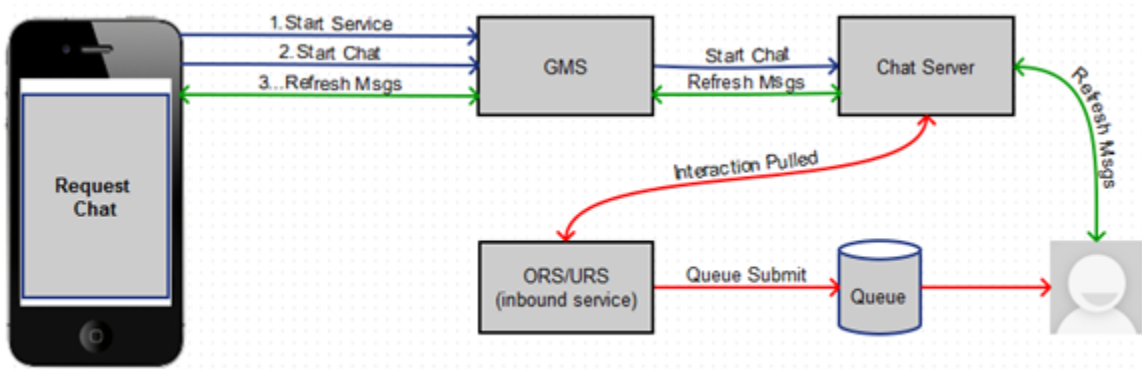
1. On the Agent Desktop:
 - Log in agent.
 - Make voice ready.
2. Using the Javascript sample: Service Management UI > Lab > Sample:
 - Log in agent and make voice ready.

- Set Contact# = <customer phone from which call will be dialed>
- Set Scenario = REQUEST-INTERACTION
- Click Connect.
- Dial displayed Number to Call.

3. Expected result:

- Treatment is played.
- Call is routed to agent.
- Toast is displayed with attached data.
- Call is connected to agent.
- For a successful GMS call, GMS_MatchResult = SUCCESS is displayed in the agent desktop as attached data.

REMOVE for GMS-9451? Scenario request-chat Test Procedure



1. Agent Desktop

- Log in agent.
- Make chat ready.

2. Using the Javascript sample: Service Management UI > Lab > Sample:

- Set Scenario = REQUEST-CHAT
- Click Connect.

3. Expected result:

- GMS app displays chat tab.
- Chat interaction is routed to agent.
- Toast is displayed with attached data.

- Chat is connected to agent.
- GMS app shows agent has joined chat.
- Agent desktop shows customer has joined chat.
- On a successful GMS call `GMS_MatchResult = SUCCESS`
- Customer and agent can now exchange messages.

Next Steps

[Configure the ORS-based Services](#)

Configuring the ORS-based Services

Now that the basic scenarios are working, let's get started with the ORS-based advanced scenarios.

Prerequisites

You must have completed the following:

1. [Configured the dependencies.](#)
2. [Configured the Builtin services.](#)
3. [Tested the Builtin services.](#)

Step 1: GMS Service – Samples

Why:

This service is responsible for receiving the GMS request from the sample application.

How:

GMS Service Management UI > Services > Configured Services

Procedure:

1. Click Add Service
2. Set Configure Service = callback
3. Set Service Name = samples.
4. Set Common Default Configuration = samples
5. Click Save.
6. Set `_target` = `<router>` Example: `Customer_Service@Stat_Server.GA`
7. Set `_urs_virtual_queue` = `GMS_VQ_SIP_Switch`
8. Set `_route_point` = `8999`
9. Set service property `_chat_endpoint` = `Environment:gms_ors` (or `Resources:gms_ors` if single tenant).

Step 2: Inbound SCXML Service - Voice

Why:

This inbound service matches the voice call with an existing GMS service. If a matching service is found, it moves the interaction to the GMS service (ORS session), which attaches the GMS User Data, and routes the call to the agent.

How:

- Configuration Manager > Switches > SIP_Switch
- Configuration Manager > Scripts

Procedure:

1. Create a route point associated with the access number configured in the procedure [Resource Group Add Access Number](#).
2. Set Annex > Orchestration section > application = script:GMSInbound.Voice.GMSMatchORS.
3. Create an enhanced routing script GMSInbound.Voice.GMSMatchORS.
4. Set Annex > Application section > url = http://<gmshost:gmSPORT>/genesys/1/document/service_template/callback/src-gen/IPD_Voice_GMSMatch.scxml
5. Set Annex > ApplicationParms/app_find_agent_timeout = 30
6. Set Annex > ApplicationParms/app_match_gms_builtin = false
7. Set Annex > ApplicationParms/app_no_match_target = <target> (Example: All_Standard_Agents@Stat_Server.GA)
8. Set Annex > ApplicationParms/app_require_access_code = false
9. Set Annex > ApplicationParms/app_require_ani = true
10. Set Annex > ApplicationParms/app_treatment_waiting_for_agent = <blank> (A blank value will force the service to use a packaged music file.)

Step 3: Inbound SCXML Service - Chat

Why:

This inbound service attaches the GMS user data to the interaction, and routes the interaction to the agent.

How:

- Configuration Manager > Chat Server
- Configuration Manager > Scripts

Procedure:

1. Go to Configuration Manager > Chat Server.
2. Create an end point that was specified in the procedure [GMS Service Create Service request chat](#) (sub-step 6):
 - `gms_ors = GMSInbound.Chat.QueueORS`
3. Go to Configuration Manager > Scripts.
4. Create interaction queue that you just specified, above.
 - Name: `GMSInbound.Chat.QueueORS`
 - Set Annex > Orchestration/application = `script:GMSInbound.Chat.QueueORS.Routing`
5. Create an interaction queue view.
 - Name: `GMSInbound.Chat.QueueORS.View 1`
 - Set Annex > View/Queue = `GMSInbound.Chat.QueueORS`
6. Create an Enhanced Routing Object that you just specified, above.
 - Name: `GMSInbound.Chat.QueueORS.Routing`
 - Set Annex > Application/url = `http://<gms_host>:<gms_port>/genesys/1/document/service_template/callback/src-gen/IPD_Chat_QueueORS.scxml`
 - Set Annex > ApplicationParms/app_find_agent_timeout = `30`
 - Set Annex > ApplicationParms/app_match_gms_builtin = `false`
 - Set Annex > ApplicationParms/app_no_match_target = `<target>` (Example: `All_Standard_Agents@Stat_Server.GA`)

Next Steps

[Test the ORS-based Services](#)

Testing the ORS-based Services

Now that you have configured the ORS-based services, it's time to test them.

Prerequisites

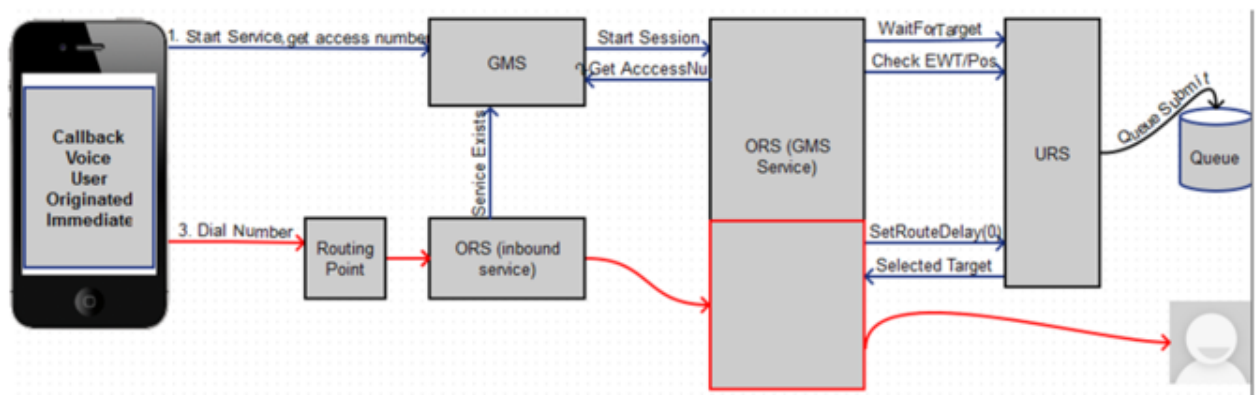
You must have completed the following:

1. Configured the dependencies.
2. Configured the built-in services.
3. Tested the built-in services.
4. Configured the ORS-based services.

Important

In the following scenarios, if the GMS Match fails, there will be no user data attached to the interaction.

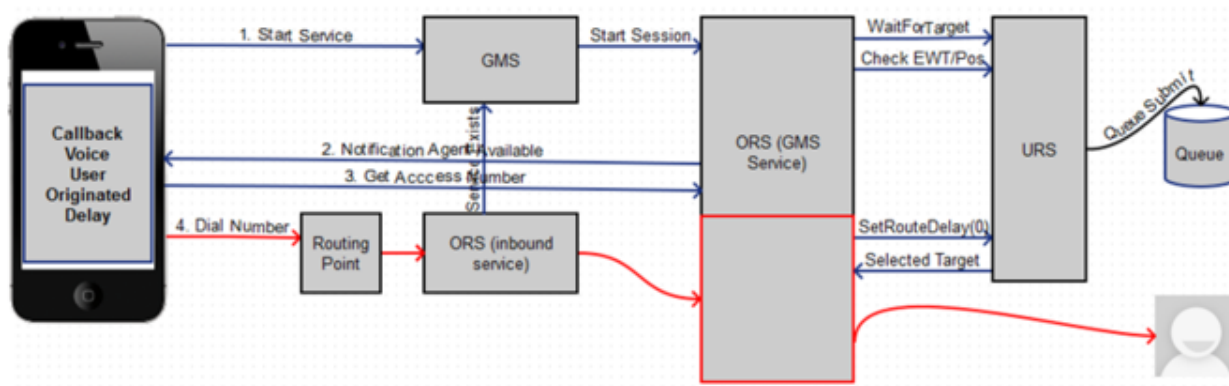
Scenario VOICE-NOW-USERORIG Test Procedure



1. Agent Desktop:
 - Log in Agent.

- Make voice ready.
- Using JavaScript sample: GMS Service Management UI > Lab > Sample:
 - Set Contact# = <customer phone from which call will be dialed>.
 - Set Scenario = VOICE-NOW-USERORIG.
 - Click Connect.
 - Dial displayed Number to Call.
 - Expected result:
 - Treatment is played.
 - Call is routed to Agent.
 - Toast is displayed with attached data.
 - Call is connected to Agent .
 - On a successful GMS call GMS_MatchResult = SUCCESS.

Scenario VOICE-WAIT-USERORIG Test Procedure



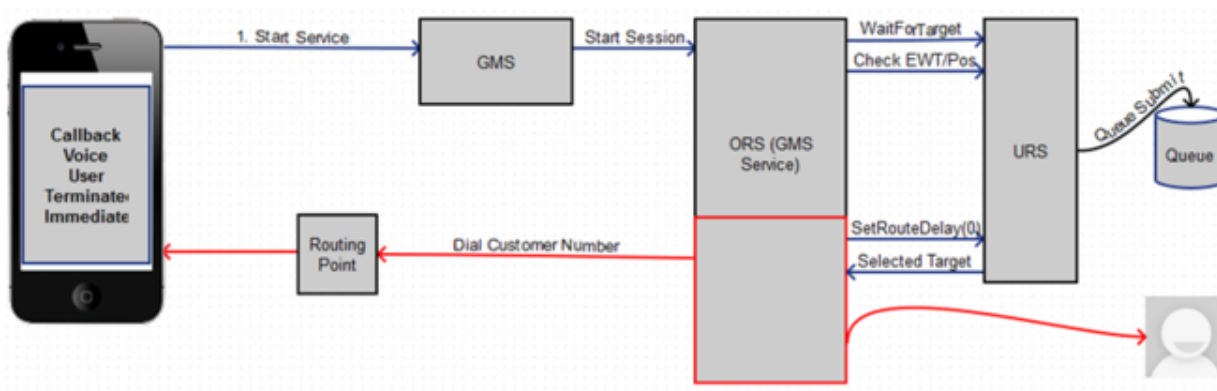
- Agent Desktop:
 - Log in Agent.
 - Make voice ready.
- Using JavaScript sample: GMS Service Management UI > Lab > Sample:
 - Set Contact# = <customer phone from which call will be dialed>.
 - Set Scenario = VOICE-WAIT-USERORIG.
 - Click Connect.
 - Click OK on the message.

- Wait for Agent Available message.
- Select Yes, I am ready to talk.
- Dial displayed Number to Call.

3. Expected result:

- Treatment is played.
- Call is routed to Agent.
- Toast is displayed with attached data.
- Call is connected to Agent.
- On a successful GMS call `GMS_MatchResult = SUCCESS`.

Scenario VOICE-NOW-USERTERM Test Procedure



1. Agent Desktop:

- Log in Agent.
- Make voice ready.

2. Using Javascript sample: GMS Service Management UI > Lab > Sample:

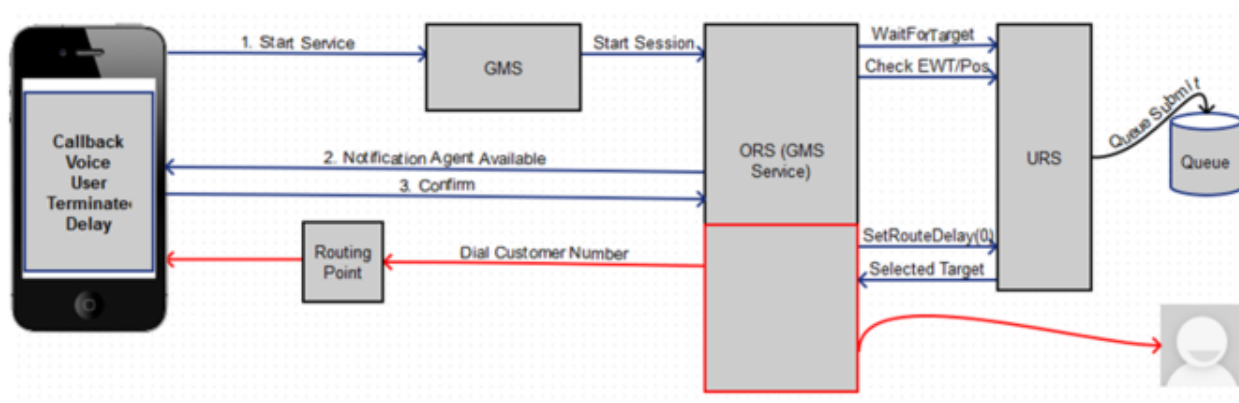
- Set Contact# = <customer phone to which call will be dialed>.
- Set Scenario = VOICE-NOW-USERTERM.
- Click Connect.
- Message displays: You will receive a call shortly.
- Click OK.

3. Expected result:

- Call is received.

- Treatment is played.
- Call is routed to Agent.
- Toast is displayed with attached data.
- Call is connected to Agent.
- On a successful GMS call `GMS_MatchResult = SUCCESS`.

Scenario VOICE-WAIT-USERTERM Test Procedure



1. Agent Desktop:

- Log in Agent.
- Make voice ready.

2. Using Javascript sample: GMS Service Management UI > Lab > Sample:

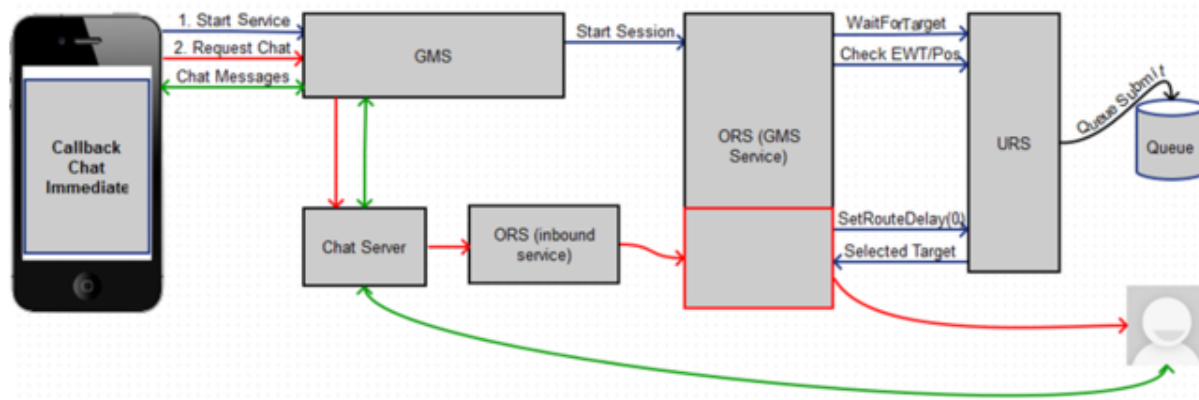
- Set Contact# = <customer phone to which call will be dialed>.
- Set Scenario = VOICE-WAIT-USERTERM.
- Click Connect.
- Click OK on the message.
- Wait for Agent available message.
- Select Yes, I am ready to talk.
- Message displays: You will receive a call shortly.

3. Expected result:

- Call is received.
- Treatment is played.
- Call is routed to the Agent.

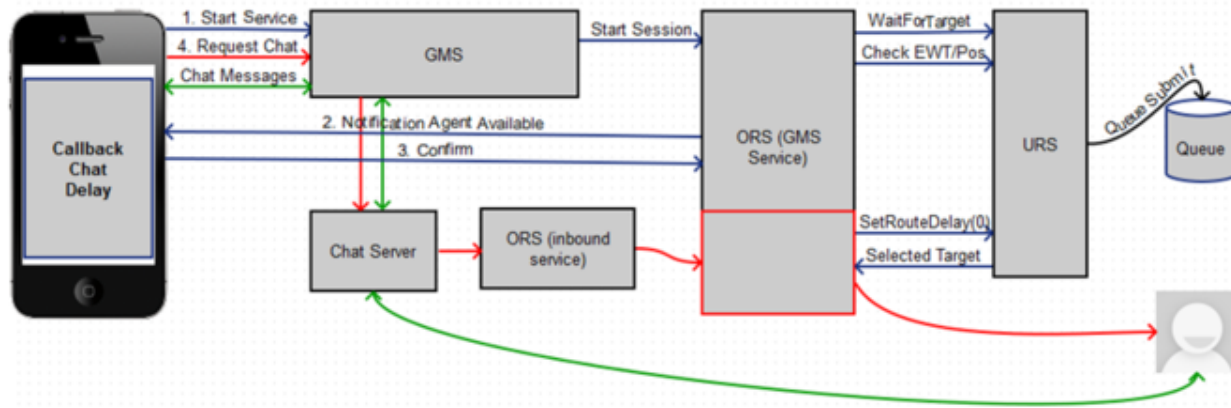
- Toast is displayed with attached data.
- Call is connected to the Agent.
- On a successful GMS call `GMS_MatchResult = SUCCESS`.

Scenario CHAT-NOW Test Procedure



1. Agent Desktop:
 - Log in Agent.
 - Make chat ready.
2. Using Javascript sample: GMS Service Management UI > Lab > Sample:
 - Set Scenario = CHAT-NOW.
 - Click Connect.
3. Expected result:
 - GMS app displays chat tab.
 - Chat interaction is routed to the Agent.
 - Toast is displayed with attached data.
 - Chat is connected to the Agent.
 - GMS app shows agent has joined chat.
 - Agent Desktop shows Customer has joined chat.
 - On a successful GMS call `GMS_MatchResult = SUCCESS`.
 - Customer and Agent can now exchange messages.

Scenario CHAT-WAIT Test Procedure



1. Agent Desktop:

- Log in Agent.
- Make chat ready.

2. Using Javascript sample: GMS Service Management UI > Lab > Sample:

- Set Scenario = CHAT-WAIT.
- Click Connect.
- Click OK on the message.
- Wait for Agent Available message.
- Select Yes, I am ready to chat.

3. Expected result:

- GMS app displays chat tab.
- Chat interaction is routed to Agent.
- Toast is displayed with attached data.
- Chat is connected to Agent.
- GMS app shows agent has joined chat.
- Agent Desktop shows customer has joined chat.
- On a successful GMS call `GMS_MatchResult = SUCCESS`.
- Customer and Agent can now exchange messages.

What's Next?

Congratulations - you have successfully tested your GMS deployment! You can now go ahead and configure additional Callback services as needed. Note that you can quickly configure a Callback service to one of the above scenarios by selecting the appropriate default configuration after you add a Callback service.