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.

Deployment Guide

Genesys Web Engagement 8.1.1

12/30/2021

Table of Contents

Genesys Web Engagement Deployment Guide

Genesys Web Engagement provides the ability to monitor, identify, and proactively engage web visitors in conversations that match business objectives. Customers are identified using robust business rules that provide a simple and comprehensive means for identifying key customers based on their behavior on your website and their value to your business.

Key customers are then evaluated, leveraging the full power of Genesys Orchestration, and the best candidates are matched with the best agents allowing you to better achieve your business objectives, including new customer acquisition, product sales, or customer support.

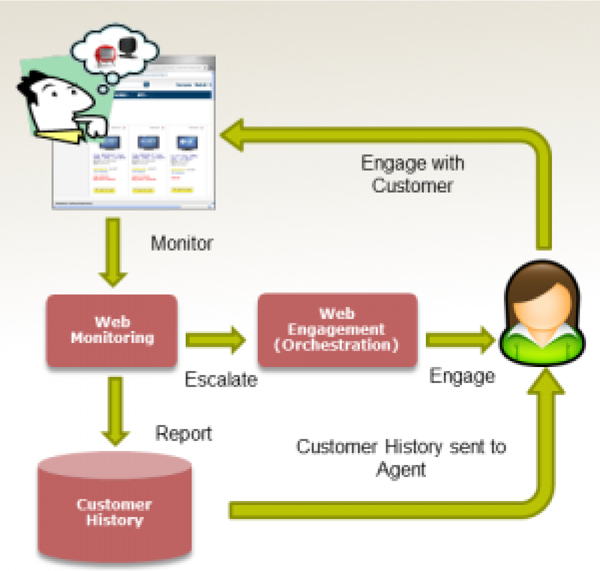
Product Overview

What is Genesys Web Engagement?

Genesys Web Engagement provides the ability to monitor, identify, and proactively engage web visitors in conversations that match business objectives. Customers are identified using robust business rules that provide a simple and comprehensive means for identifying key customers based on their behavior on your website and their value to your business. Key customers are then evaluated, leveraging the full power of Genesys Orchestration, and the best candidates are matched with the best agents, allowing you to better achieve your business objectives, including new customer acquisition, product sales, or customer support.

Genesys Web Engagement integrates the browsing activity of your web visitors into the overall Genesys customer service process. It records the customer web-browsing history, gathers accurate information, and converts it into Genesys interactions.

In addition to its monitoring features, Genesys Web Engagement enables you to engage the online customer by chat or voice callback.



From the customer standpoint, there is no visible change in the web experience.

Get to Know your Web Visitors

Genesys Web Engagement develops an understanding approach of the customer's interaction with your website. Its web monitoring features connects the web content as a self-service channel into the Context Services.

It translates the raw web activity into a form suitable to customer service.

- **Basic Usage Information.** Get to know if a customer has ever used the web channel, and if so, how recently (and/or how frequently).
- **Browsing History.** Each time that the customer browses your website, a session is created to store the visited pages in the customer history. This provides information on what the customer may have been looking for or may be interested in.
- **Activities and Outcomes.** Genesys Web Engagement allows you to tag web pages and define associations between URIs and outcomes to build a higher-level model of the customer browsing activity. This model is then usable to drive further interactions on other channels (chat and web callback, for now).

In addition, Genesys Web Engagement includes several scenarios for identified and unknown web visitors, detailed in Visitor Identification Scenario.

- If the user is not authenticated, the engagement decision should be taken by an agent or configured through custom Web Engagement rules.
- When engaging an unauthenticated user, Genesys Web Engagement asks for the user's registration.

Pro-active Follow-up

Genesys Web Engagement enables real-time or offline post processing of customers' web-browsing activity to identify potential for proactive follow-up. You can define service assistance to notify agents when some specific use cases should lead to a proactive follow-up. In addition, the flexibility of Genesys Web Engagement allows to submit this follow-up for validation to agents, in order to make sure that the follow-up is appropriate.

For example, some use cases could be:

- When a shopping cart is abandoned, it can be caused by a lack of information. A proactive follow-up by an agent - via any number of channels - can help to close the sale.
- If a customer bought a product several weeks ago but abandoned a transaction recently, an agent could call to ask about satisfaction with the earlier purchase and afterwards follow-up with a question about the abandonment.
- If a customer submits a bad rating or comments about one of your products, an agent could follow-up by e-mail with a survey about his dissatisfaction.

In addition, if the contact center decides to make a proactive offer, the Browser-Tier Component checks that the visitor is still present, then pops up a widget in the browser window ("Click here to chat"). If the visitor accepts the offer, the chat connection is made in the standard way using existing components.

Components

Genesys Web Engagement interfaces with the standard Genesys Call Center Solution and requires minimal changes to your website: to interface your website with this product, you simply add a JavaScript tracking code to your web pages. In addition, Standard Genesys interfaces, such as Composer, Genesys Rules Development Tool, and Rules Authoring Tool, enable you to develop and deploy custom rules and business attributes to fine-tune your web engagement scenarios. Genesys Web Engagement is composed of three components, detailed in the Architecture page:

- **Web Engagement Browser-Tier Agents** are loaded in the JavaScript tracking code, which submits system and custom events based on the customer's browsing activity.
- **Web Engagement Frontend Server** manages the event flow submitted by the browser-tier agents and is responsible for the complex event processing and submitting actionable events.
- **Web Engagement Backend Server** interfaces with the Web Engagement Frontend Server and the Genesys Contact Solution to store web engagement information and implement web engagement action.

In addition to rules templates, deliverables include the following plug-ins:

- **Web Engagement Plug-in for Administrator Extension**, implementing:
  - **Script Generator**, to generate the standard JavaScript tracking code that you must add to your web pages.
  - **Categories**, to create custom business data.
- **Web Engagement Plug-in for Interaction Workspace**, to get all the web-based contexts routed to agents. This plug-in is mandatory to enable chat and web callback engagement features in Interaction Workspace.

Browser Support

Genesys Web Engagement supports the following web browsers:

- Google Chrome
- Mozilla Firefox 9.0+
- Microsoft Internet Explorer 7, 8, 9, 10
- Apple Safari 5.0+

Genesys Web Engagement supports the following mobile browsers:

- iOS Safari
- Android Chrome

Features

Genesys Web Engagement includes the following features:

- Integrated Proactive Chat and Proactive Web Callback

  Optimization / Pacing of Proactive Engagement Invitations

  Included Web Engagement applications for proactive Genesys Chat and Genesys WebCallback

- Behavior Rules Authoring for simplified tooling of Web Pages
    - Categorization - Key word and regular expressions for out-of-the box web page identification
    - Out-of-the box business events for capturing searches and timeout as part of behavior rules
    - Out-of-the box rule templates and business rules interface for defining engagement rules based on customer behavior
    - Support for advanced business events to capture events not covered by categories
    - Business User friendly UIs for creating both Categories and Business rules
- Monitoring and data storage of customer web activity
    - Storage of web history for authenticated customers
    - RESTful API for full access to web history
- Integrated Agent Interface for Interaction Workspace
    - Out-of-the box Agent Desktop support
    - Live monitoring of customer during engagement
    - Integrated view of Web History
- Reporting – Historical and Real Time
    - Templates for out-of-the box real-time interaction reporting of Web Engagement
- Integrated with core Genesys product suite

Related Components

Genesys Web Engagement interacts with the following Genesys Products:

- Interaction Workspace—The Genesys Web Engagement Plug-in for Interaction Workspace is required to interface Genesys Interaction Workspace with Web Engagement. This plug-in enables you to get all the web-based contexts routed to agents. This plug-in is mandatory to enable chat and voice engagement.
- Context Services
- Orchestration Server
- Stat Server
- SIP Server
- Interaction Server
- Chat Server

Installing and Deploying Genesys Web Engagement

Objective	Related procedures and actions
1. Prepare your deployment.	Review the tasks on the Preparing your Deployment page.
2. Install Genesys Web Engagement.	Review the steps outlined in the task table on the Installation page. **Note:** Genesys recommends that you deploy Genesys Web Engagement in a lab environment. After successfully deploying Web Engagement, create and test a Web Engagement application before you switch to production.
3. Install the related plug-ins for Interaction Workspace and Genesys Administrator Extension.	Genesys Web Engagement has two available plug-ins: • Genesys Web Engagement Plug-in for Interaction Workspace to enable chat and web callback engagement features in Interaction Workspace. • Plug-in for Genesys Administrator Extension to use the Script Generator tool or create customized business information. See Installing Plug-ins for installation details.
4. Perform the setup tasks.	You will need to configure the related tools and components to work with Genesys Web Engagement after it has been installed. See Setup Tasks for details.
5. Create an application.	An application allows you to implement Web Engagement features in your Genesys Contact Center, and to add Web Engagement to your website. Before deploying Web Engagement in production, first create a test application in your lab environment. See the Genesys Web Engagement Developer's Guide for further details about creating your application.
6. Deploy to production.	Once Web Engagement is working correctly in your lab environment, you can deploy it to your production environment. See Deployment for details.
7. Implement load balancing.	Review the section on load balancing, and then complete the tasks on the load balancing tasks page. **Note:** Frontend and Backend load balancers are mandatory elements for a production deployment.

Objective	Related procedures and actions
	Interconnecting the Frontend and Backend servers directly is only appropriate for a development environment.

Preparing your Deployment

Purpose: To list the items to consider before you install and set up Genesys Web Engagement.

Complete the following tasks before installing Genesys Web Engagement:

- Refer to the Genesys Supported Operating Environment Reference Guide to ensure your operating environment meets the requirements to run Genesys applications.
- Review the Related Components and ensure your required components are installed.
- Review the Security and Multi-Tenancy and Load Balancing pages.

Related Components

	Purpose: List components required for installing Genesys Web Engagement and creating your web engagement application.

Server Dependencies

The following servers are mandatory connections for the Web Engagement Servers:

Server Name	Compliant Versions (and later)
Orchestration Server	8.1.300.25
Universal Routing Server	8.1.300.20
Stat Server	8.1.000.23
Interaction Server	8.1.200.27
Chat Server	8.0.100.08
Contact Server	8.1.000.01
Genesys Rules Authoring Server	8.1.200.17
Configuration Server (for UTF-8 support)	8.1.200.04

Components for your Application Development

The following components are mandatory for the creation of customized Web Engagement applications:

Component Name	Compliant versions	Details
Genesys Rules Authoring Tool	8.1.200.17 or later	You must create a user with Roles Privileges that enable the creation of Rules package in Genesys Rules Authoring. See Role-Based Access Control in the Genesys Rules System Deployment Guide.
Genesys Rules Development Tool	8.1.200.17 or later	This component must be deployed as Composer plug-in or independently in Eclipse; make sure that settings are correct for Configuration Server and Repository Server, as detailed in Installing the GRDT Component in the Genesys Rules System

Component Name	Compliant versions	Details
		Deployment Guide.
Composer	8.1.300.51 or later	Mandatory to publish the Web Engagement rules template. This component can also be used to update or deploy routing and engagement strategies.
Genesys Administrator Extension	8.1.400.43 or later	Mandatory for the simple engagement model.
Genesys Administrator	8.1.300.02 or later	Mandatory.

Components for Interaction Management

- Interaction Workspace, version 8.1.301.10 and later.

Additional Components (Optional)

- CCPulse+, version 8.0.000.36

Security and Multi-Tenancy

	Purpose: To provide background information on Security and Multi-Tenancy in Genesys Web Engagement.

Security

Genesys Web Engagement supports HTTPS/SSL/TLS to protect data over the web.

- All connections can be secured, including connections from the browser to the Frontend and Backend Servers.
- Applications defined in Configuration Server can have both HTTP and HTTPS connections.

Transport Layer Security (TLS) is supported above Java containers, Jetty and Apache Tomcat.

The user data submitted from the browser tier is always sent through secure connections.

Note: To support secure (TLS) connections to Configuration Server on Windows OS, if you use the JDK 6 64 bit, it is mandatory to do **one** of the following:

- Update JDK 6 64 bit with Java SE Development Kit 6, Update 38 (JDK 6u38) or older.
- Setup JDK 7.

Multi-Tenancy

To implement Genesys Web Engagement, you should create one application per tenant. This application supports multiple domains and all their associated subdomains. For instance, a `genesyslab` application supports the `genesyslab.com` domain and its associated subdomain, `docs.genesyslab.com`.

> ### Tip
> When a visitor crosses domain boundaries, Web Engagement creates a new visit and closes the previous visit.

As a result, your deployment must respect the following constraints:

- One tenant can contain multiple DSL files, but instrumentation will upload only one of them at a given time; a single DSL file is active for a given page during the customer's visit on your website.

- One Frontend Server node should only belong to one tenant.
- The Frontend load balancer (or Frontend Server) is the single entry point.

Load Balancing

Purpose: To describe how to implement Load Balancing for your web engagement servers.

Introduction

Genesys Web Engagement supports any third-party load balancer as long as the load balancing features include cookie support and URL encoding-based routing methods.

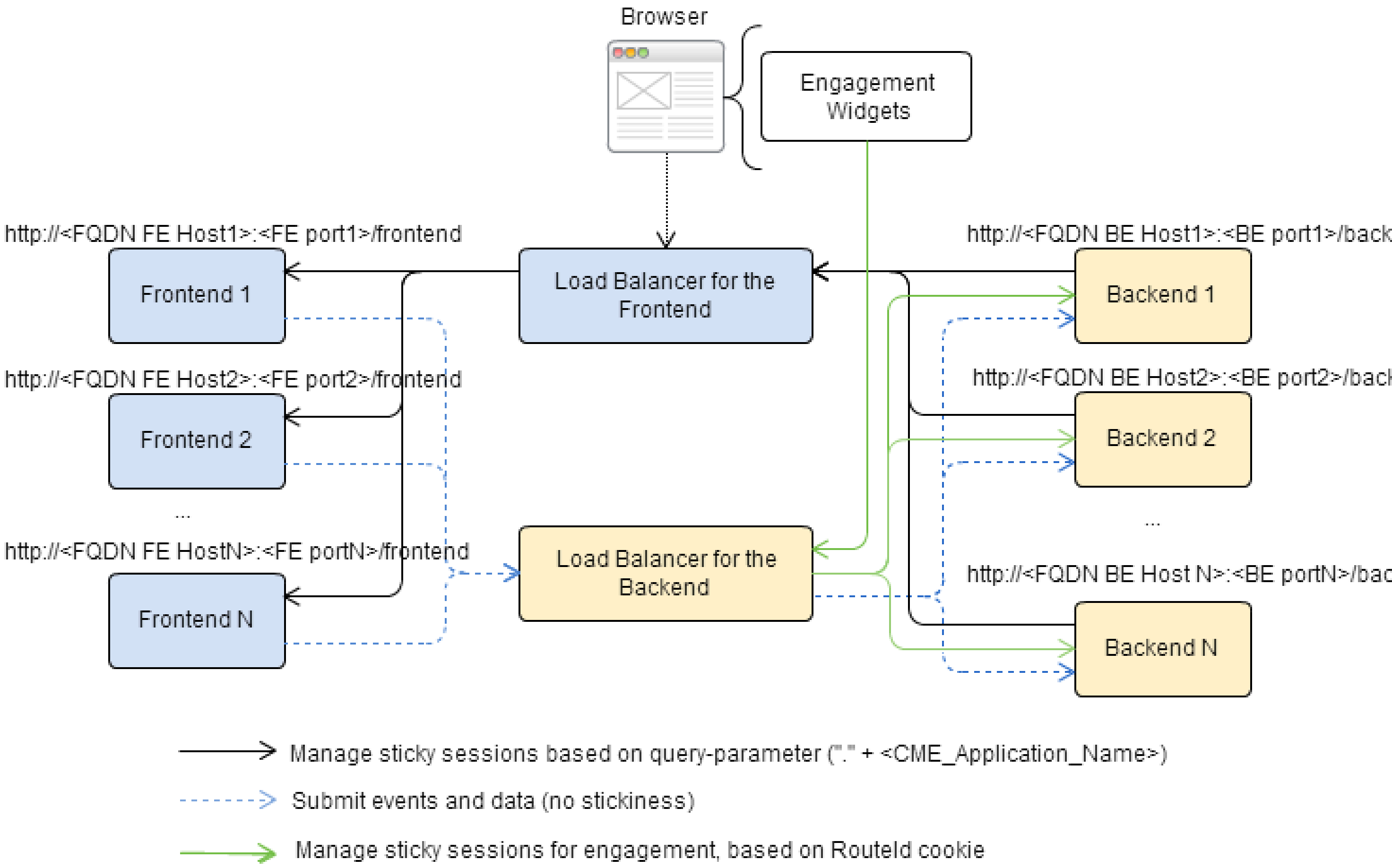
Load balancing deployment should occur at the latest stage of your deployment. Genesys recommends that you follow these steps:

1. Install Genesys Web Engagement, as described in the Installation chapter.
2. Develop and test a Web Engagement Application, as described in Application Development Workflow.
3. Follow the Load Balancing guidelines to install your servers.
4. Deploy your application on the Web Engagement servers, as detailed in Deploying to Production Environment

> ### Important
> Frontend and Backend load balancers are mandatory elements for a production deployment. Interconnecting the Frontend and Backend servers directly is only appropriate for a development environment.

Architecture

The following diagram shows how you can implement a load balancing configuration for your Web Engagement servers.


Browser
Engagement Widgets
http://<FQDN FE Host1>:<FE port1>/frontend
Frontend 1
Load Balancer for the Frontend
http://<FQDN BE Host1>:<BE port1>/back
Backend 1
http://<FQDN FE Host2>:<FE port2>/frontend
Frontend 2
http://<FQDN BE Host2>:<BE port2>/back
Backend 2
...
...
http://<FQDN FE HostN>:<FE portN>/frontend
Frontend N
Load Balancer for the Backend
http://<FQDN BE Host N>:<BE portN>/bac
Backend N
Manage sticky sessions based on query-parameter ("." + <CME_Application_Name>)
Submit events and data (no stickiness)
Manage sticky sessions for engagement, based on RouteId cookie

Sample of Deployment for Load Balancing

In the above example:

- The load balancer for the Frontend Server implements sticky sessions for the deployed Web Engagement application;
- The load balancer for the Backend Server implements sticky sessions to route IP addresses when customers are engaged.

Sticky sessions

Genesys Web Engagement uses sticky sessions as follows:

- The load balancer for the Frontend Server implements sticky sessions based on URL encoding static parameters for the deployed Web Engagement application. Web Engagement creates the parameter as follows:
  `"." + <CME_Application_Name>`.

  Note: If you are using Apache, you do not need to add `"."` to your static parameter; this will be done by the Web Engagement Server.
- The load balancer for the Backend Server implements sticky sessions based on cookies to route IP addresses when the customers are engaged. The load balancer for the Backend Server must support

the following features:

- Cookie-based stickiness to enable engagement.
- Storage of sticky parameters into cookies.

> **Important**
>
> All the cookies are created by the Load Balancing system, not by the Genesys Web Engagement servers.

Configurations

Apache is used by default as a sample in the configuration instructions, which are available on the following page:

- Implement Load Balancing

Installing Genesys Web Engagement

Purpose: List the Installation tasks for Genesys Web Engagement.

Note: If the installation is successful, you must create a Web Engagement application before your can run and test Web Engagement. See the Developer's Guide for further details.

Tasks	Procedures
1. Prepare Configuration Server.	Complete the following procedures: • Importing the Application Templates for the Web Engagement Servers • Creating the Configuration Application for the Backend Server • Creating the Configuration Application for the Frontend Server
2. Configure the Web Engagement servers.	Complete the following procedures: • Configuring the Backend Server Application • Configuring the Frontend Server Application
3. Install Genesys Web Engagement.	Complete one of the following procedures, as per your operating system: • Installing Web Engagement on the Windows operating system • Installing Web Engagement on the Linux operating system
4. Run the Provisioning Tool (optional).	The Provisioning Tool creates all the configuration information related to Genesys Web Engagement in Configuration Server. It is run automatically as part of the installation process, but the tool is also included in your Web Engagement installation and you can use it after installation to change your configuration. See Provisioning for details.
5. Install the Genesys Web Engagement plug-ins for Interaction Workspace and Genesys Administrator Extension.	See Installing Plug-ins for installation details.
6. Import the CCPulse+ sample reporting templates.	See Importing CCPulse+ Templates.

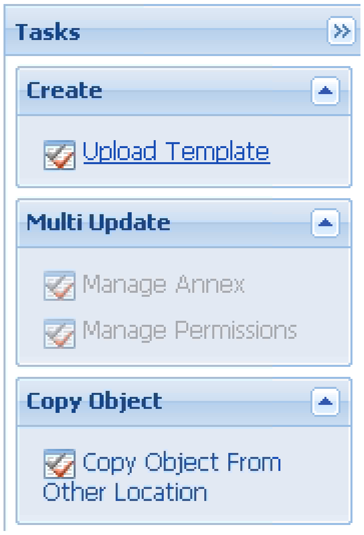
Prepare for Configuration Server

Importing the Application Templates for the Web Engagement Servers

Purpose: Import the templates you will use to create the configuration for the Web Engagement servers in Configuration Server.

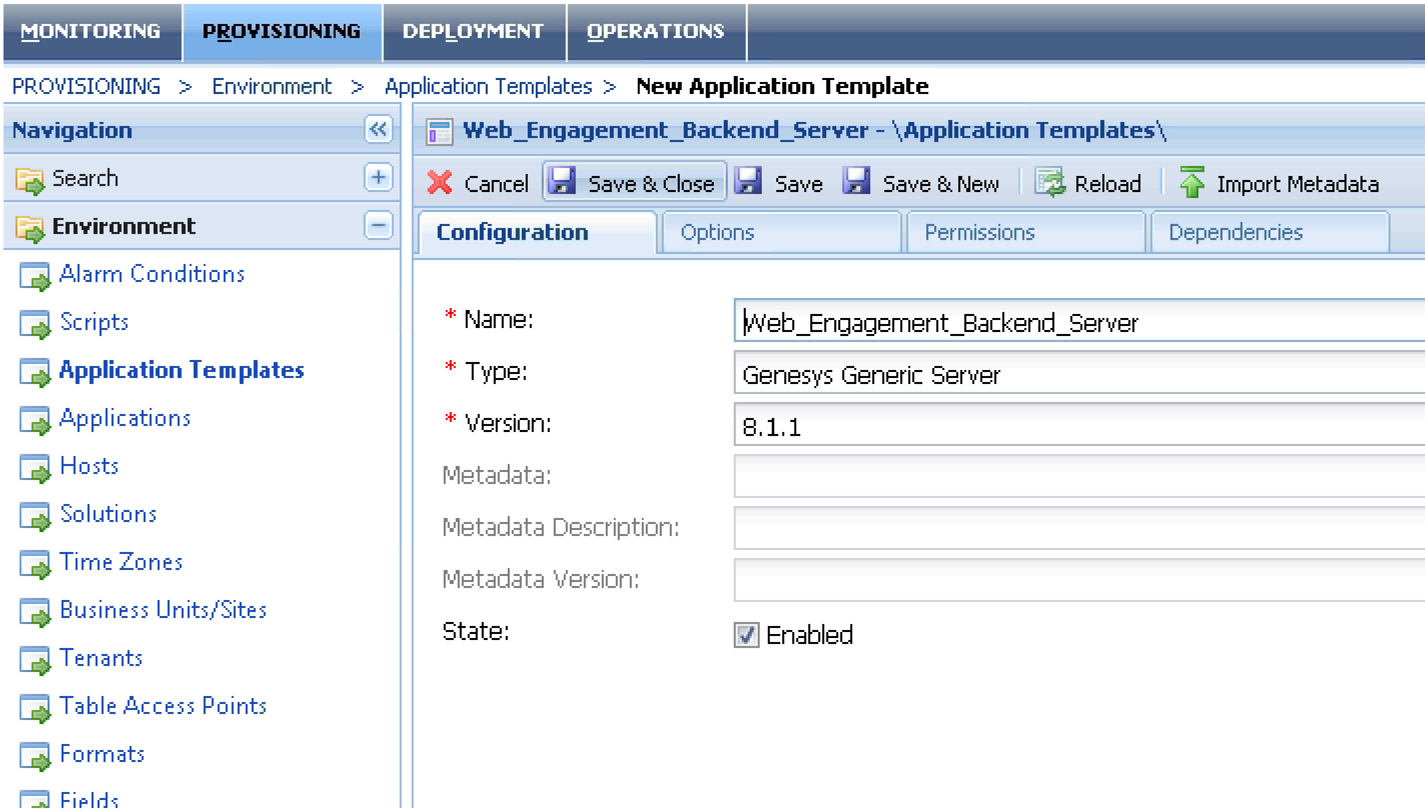
Start

1. Open Genesys Administrator and navigate to `PROVISIONING > Environment> Application Templates`.
2. In the `Create` menu of the `Tasks` panel, click the `Upload Template` link.



Upload Template link in the Tasks panel

3. Click the Add button of the `Click 'Add' and choose application template (APD) file to import` dialog box.
4. Browse to the `Web_Engagement_Backend_Generic_811.apd` file or, if your Configuration Server supports Web Engagement specific types, select `Web_Engagement_Backend_811.apd`, available in the `templates` directory of your installation CD. The `New Application Template` panel opens.



* Name: Web_Engagement_Backend_Server

* Type: Genesys Generic Server

* Version: 8.1.1

Metadata:

Metadata Description:

Metadata Version:

State: ☑ Enabled

New Application Template panel.

5. Click the `Save & Close` button.
6. Repeat steps 2 to 5 to import the `Web_Engagement_Frontend_Generic_811.apd` file, or, if your Configuration Server supports Web Engagement specific types, select `Web_Engagement_Frontend_811.apd file.`

End

Next Step

Creating the Configuration Application for the Backend Server

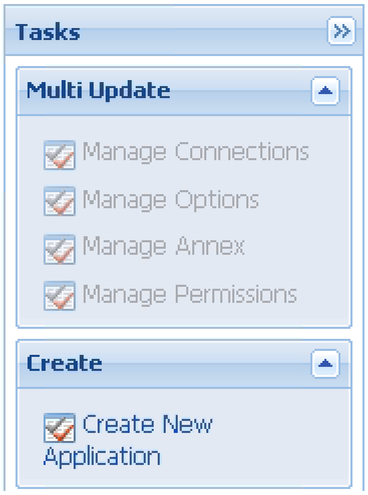
Creating the Configuration Application for the Backend Server

Purpose: Create the configuration application which contains all the configuring information related to the Backend Server.

Start

1. Open Genesys Administrator and navigate to `PROVISIONING > Environment > Application`.

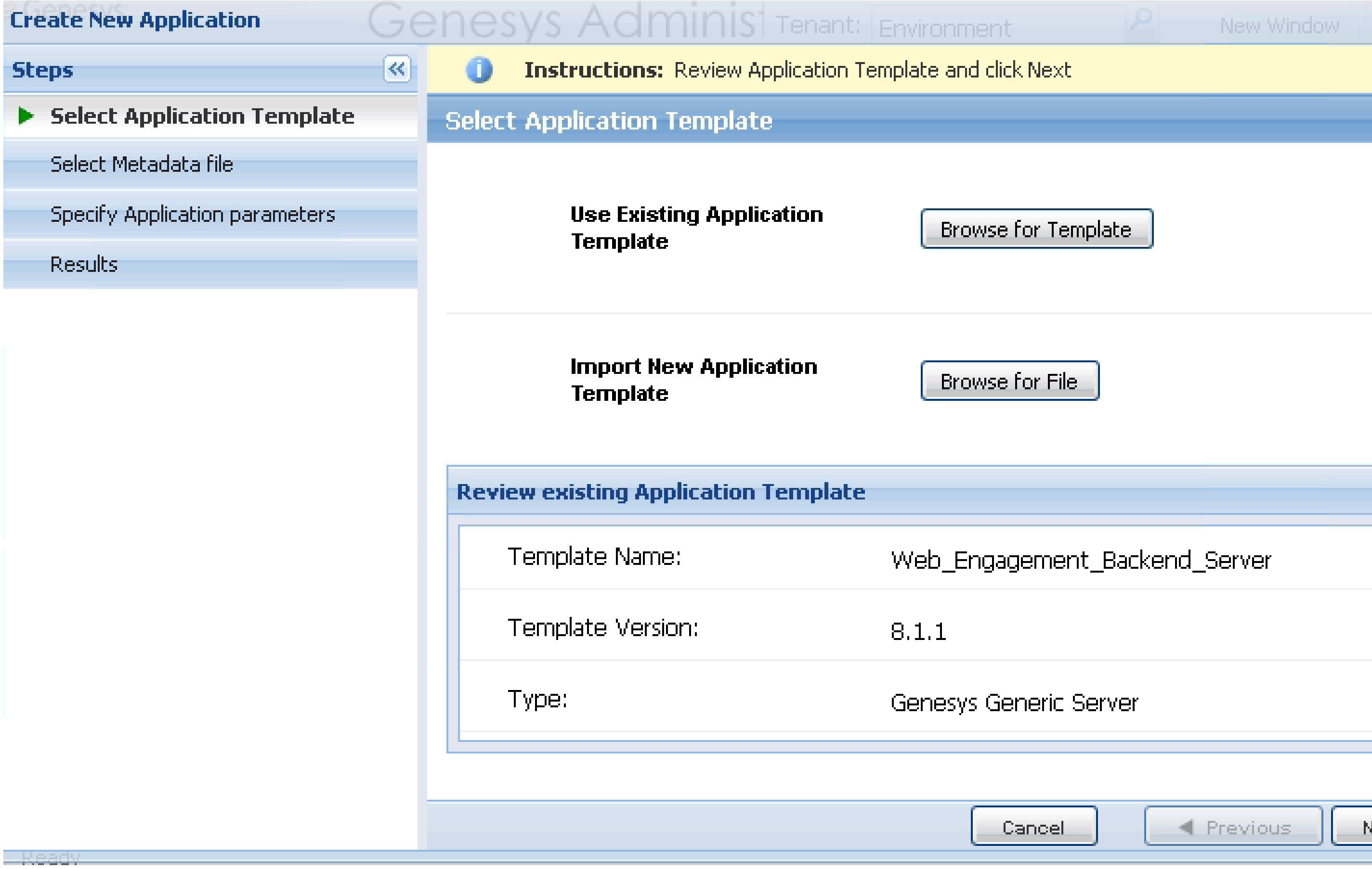
2. In the Create menu of the Tasks panel, click the Create New Application link.



Create New Application link.

3. In the Select Application Template panel, click Browse for Template and select the Backend Server template that you imported previously. Click OK.

4. The template is added to the Select Application Template panel. Click Next.

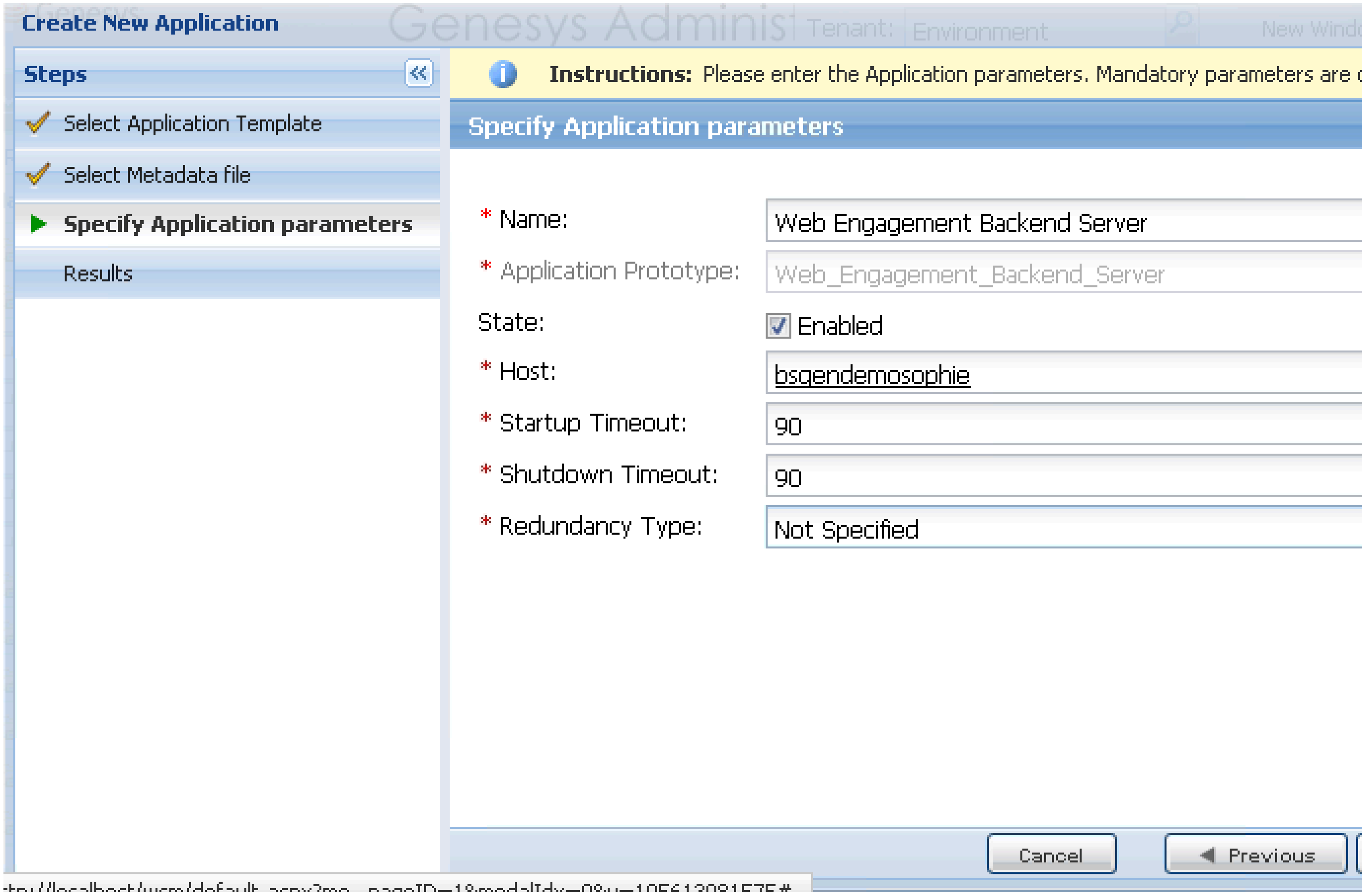

Create New Application
Genesys Administ
Tenant: Environment
New Window
Steps
Select Application Template
Select Metadata file
Specify Application parameters
Results
Select Application Template
Use Existing Application Template
Browse for Template
Import New Application Template
Browse for File
Review existing Application Template
Cancel
Previous
Ready

Instructions: Review Application Template and click Next

Template Name:	Web_Engagement_Backend_Server
Template Version:	8.1.1
Type:	Genesys Generic Server

Select Application Template and click Next.

5. In `Select Metadata file`, click `Next`;
6. In `Specify Application` parameters:
   - Enter a Name for your application—for instance, Web_Engagement_Backend_Server.
   - Enable the State.
   - Click the `Host` lookup button to select the application's host.
   - Click the `Create` button.



Instructions: Please enter the Application parameters. Mandatory parameters are c

Specify Application parameters

* Name: Web Engagement Backend Server

* Application Prototype: Web_Engagement_Backend_Server

State: ☑ Enabled

* Host: bsgendemosophie

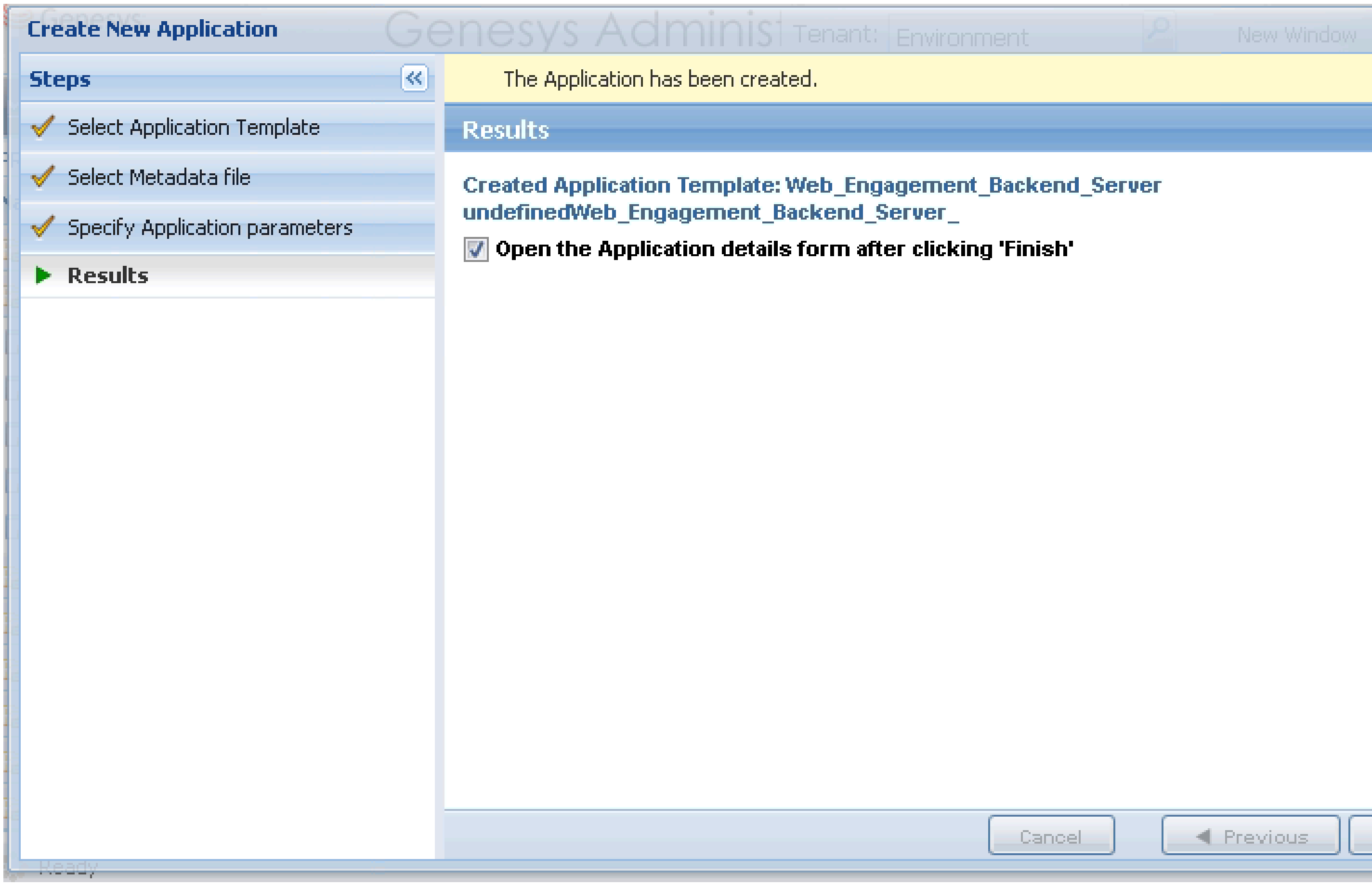
* Startup Timeout: 90

* Shutdown Timeout: 90

* Redundancy Type: Not Specified

Specify Application.

7. The Results panel opens. Click Finish.



Create New Application

The Application has been created.

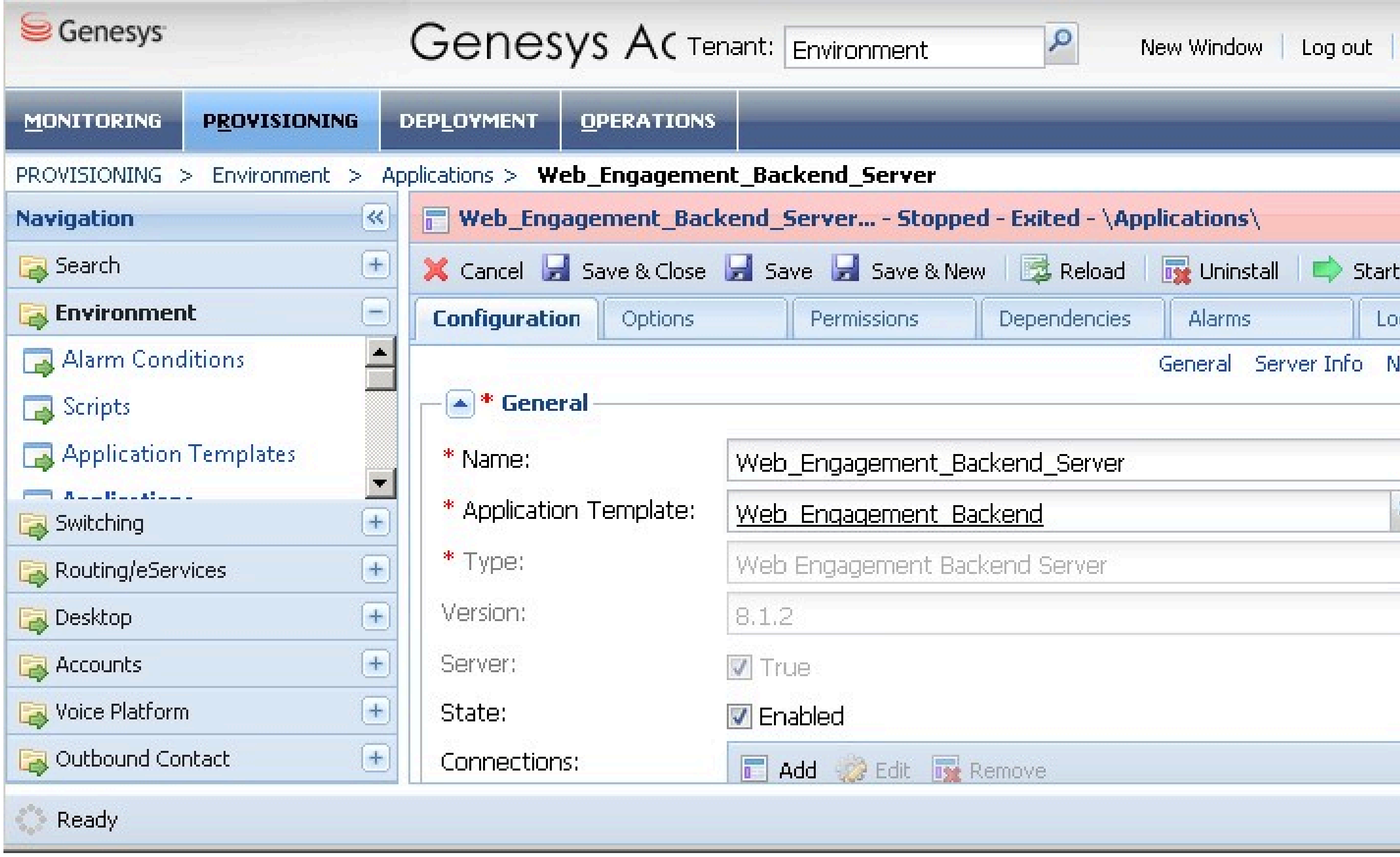
Results

Created Application Template: Web_Engagement_Backend_Server
undefinedWeb_Engagement_Backend_Server_

☑ Open the Application details form after clicking 'Finish'

The New application is created.

8. If you enabled `Opens the Application details form after clicking 'Finish'`, the Web Engagement Backend Server application form opens and you can start configuring the Backend Server application. See Configure the Backend Server



* Name: Web_Engagement_Backend_Server

* Application Template: Web_Engagement_Backend

* Type: Web Engagement Backend Server

Version: 8.1.2

Server: ☑ True

State: ☑ Enabled

Connections:

Application opened in Genesys Administrator.

End
Next Steps

- If the Configuration form is opened for the Backend Server, you can start the configuration as described in Configuring the Web Engagement Backend Server. **Note:** You will have to add additional connections after creating the configuration application for the Frontend Server.
- You can save and close to create the configuration application for the Frontend Server. In this case, you will be able to perform all the configurations steps later, at once.

Creating the Configuration Application for the Frontend Server

Purpose: Create the configuration application of the Web Engagement Frontend Server in Configuration Server.

Start

1. Open Genesys Administrator and navigate to PROVISIONING > Environment > Application.
2. In the Create menu of the Tasks panel, click the Create New Application link.
3. In the Select Application Template panel, click Browse for Template and select the Frontend Server template that you imported previously. Click OK.
4. The template is added to the Select Application Template panel. Click Next.
5. In Select Metadata file, click Next;
6. In Specify Application parameters:
   - Enter a name for your application—for instance, Web_Engagement_Frontend_Server.
   - Enable the State.
   - Click the Host lookup button to select the application's host.
   - Click the Create button.
7. The Results panel opens. Click Finish.
8. If you enabled Opens the Application details form after clicking 'Finish', the Web Engagement Frontend Server application form opens and you can start configuring the Frontend Server application. See Configure the Frontend Server

End

Next Steps

Configure the Frontend Server or Configure the Backend Server

Configure the Backend Server

Configuring the Backend Server Application

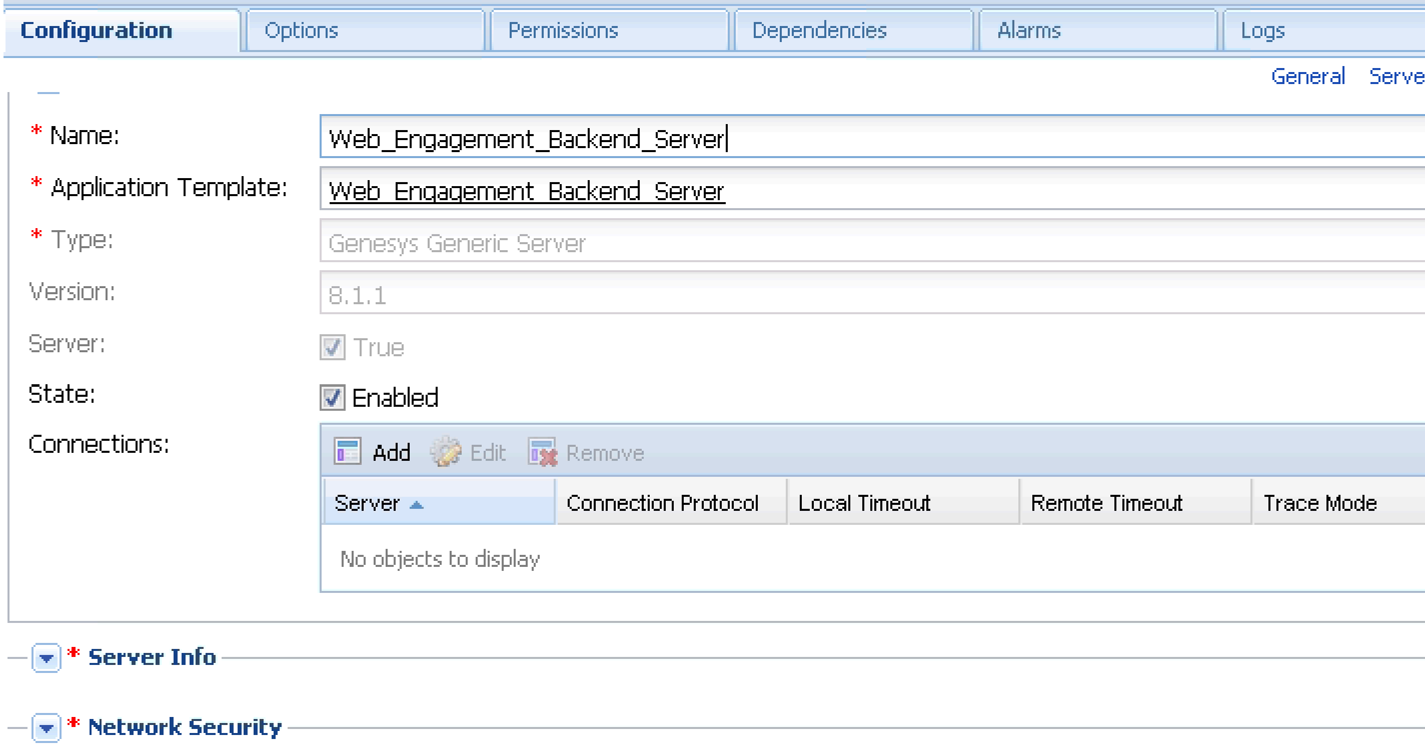
Purpose: Enter the configuring information for the Web Engagement Backend Server in Configuration Server.

Prerequisites

- You created the Web Engagement Backend Server application.

Start

1. Open Genesys Administrator and navigate to `PROVISIONING > Environment > Applications`. Select the application defined for the Web Engagement Backend Server and click `Edit...`



* Name: Web_Engagement_Backend_Server

* Application Template: Web_Engagement_Backend_Server

* Type: Genesys Generic Server

Version: 8.1.1

Server: ☑ True

State: ☑ Enabled

Connections:

Server ▲	Connection Protocol	Local Timeout	Remote Timeout	Trace Mode
No objects to display				

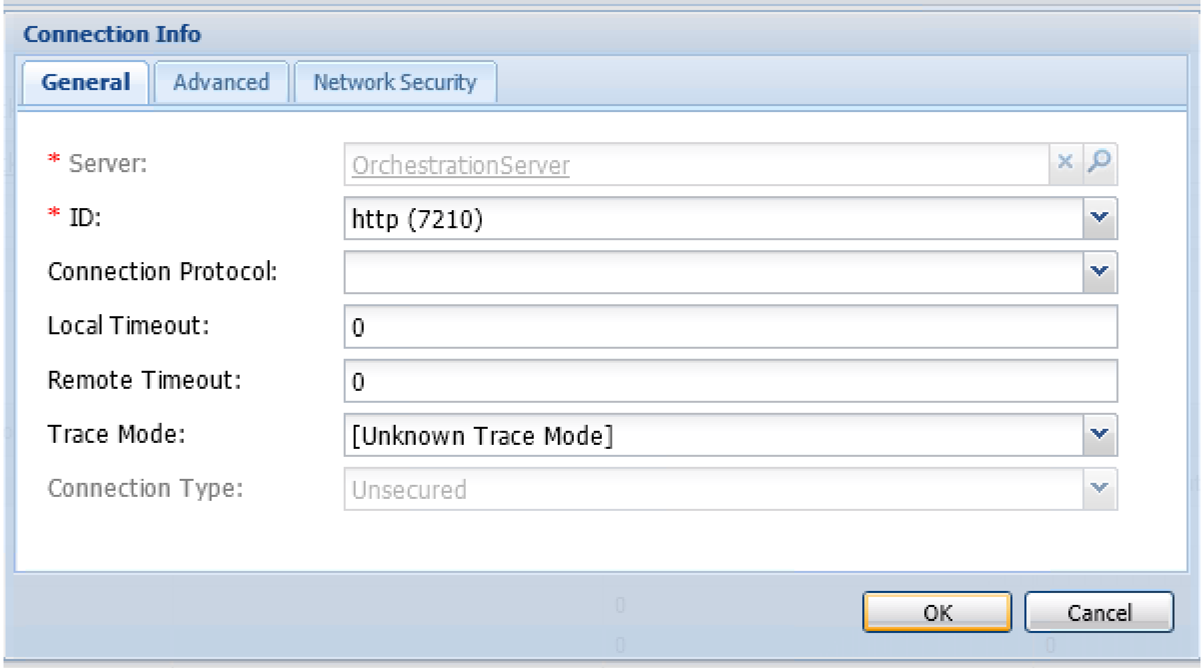
Configuration of the Web Engagement Backend Server.

2. Configure the Backend Server connections.ca
   - In the `Connections` section of the `Configuration` tab, click the Add button. The `Browse for applications` panel opens. Select the Genesys application defined for the Chat Server, then click

OK.

> ## Important
>
> Genesys Web Engagement also supports Chat Server clusters. To configure the Web Engagement Backend Server to work with a cluster of Chat Server objects, see Chat Server Cluster Configuration.

- Repeat the previous step to add the connections to the following servers:
  - Contact Server
  - Interaction Server
  - Stat Server
  - Orchestration Server. After you have added the connection, select the Orchestration Server row in the list of connections and click Edit. In the Connection Info window, select the default port for the ID and set the Connection protocol to http. Click OK.



Connection Info

* Server: OrchestrationServer

* ID: http (7210)

Connection Protocol:

Local Timeout: 0

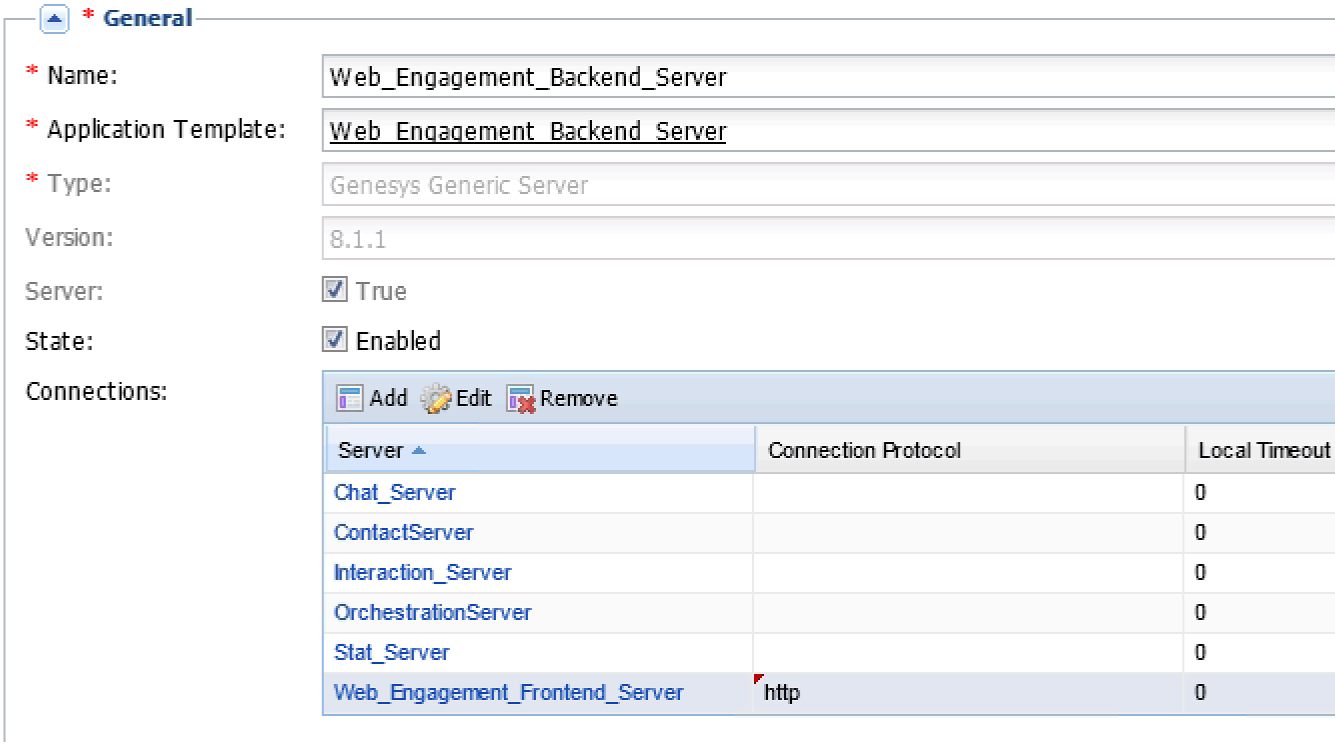
Remote Timeout: 0

Trace Mode: [Unknown Trace Mode]

Connection Type: Unsecured

ID must be set to http for the Orchestration Server

  - Web Engagement Frontend Server. If you already created the application for the Frontend Server, you can add a connection to it now. After you have added the connection, select the Frontend Server row in the list of connections and click Edit. In the Connection Info window, set Connection Protocol to http and click OK



* Name: Web_Engagement_Backend_Server

* Application Template: Web_Engagement_Backend_Server

* Type: Genesys Generic Server

Version: 8.1.1

Server: ☑ True

State: ☑ Enabled

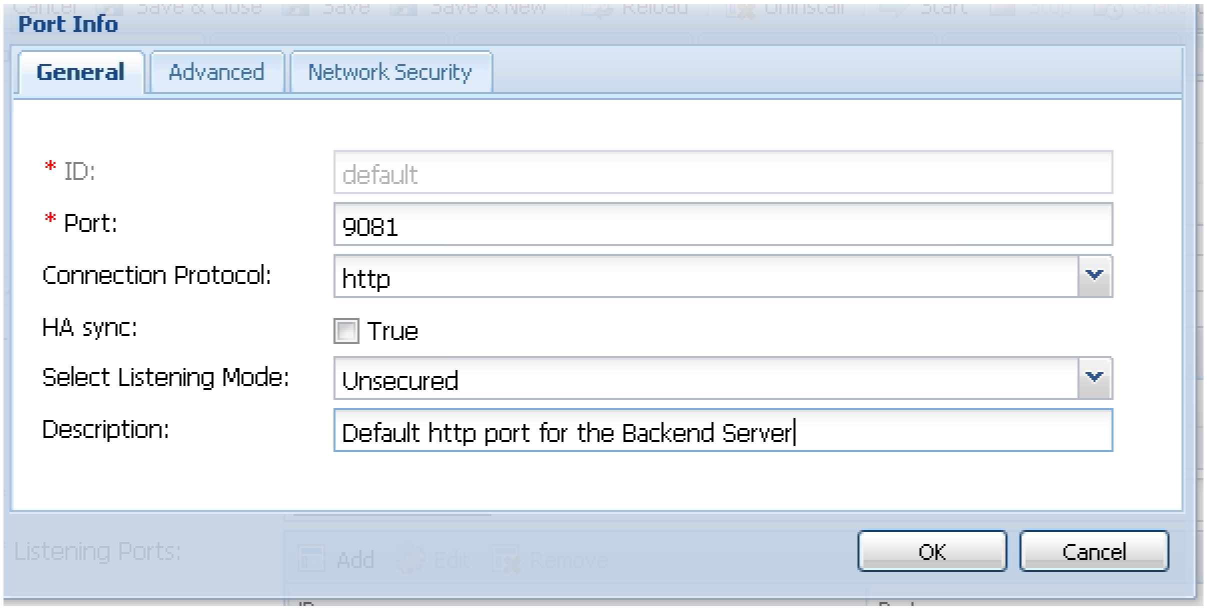
Connections:

Server	Connection Protocol	Local Timeout
Chat_Server		0
ContactServer		0
Interaction_Server		0
OrchestrationServer		0
Stat_Server		0
Web_Engagement_Frontend_Server	http	0

Backend Server Connections

If you haven't created the application for the Frontend Server, you can add a connection to it later.

3. Configure the `Server Info` section of the Configuration tab.
   - Click the `Add` button of the `Tenants` section.
   - Select your application tenant. For instance, `Environment`.
     Note: You should select only one tenant object.
   - Click `OK`. The tenant is added to the `Tenants` table.
4. If your `Host` is not defined, click the lookup icon to browse to the hostname of your application.
5. Configure the default http port:
   - Click `Add` in the `Listening ports` section. The `Port Info` dialog opens.
   - Enter the application's Port. For instance, `9081`.
   - **Mandatory:** Select `http` for the Connection Protocol field.
   -
   - (Optional) Enter a description.



* ID: default

* Port: 9081

Connection Protocol: http

HA sync: □ True

Select Listening Mode: Unsecured

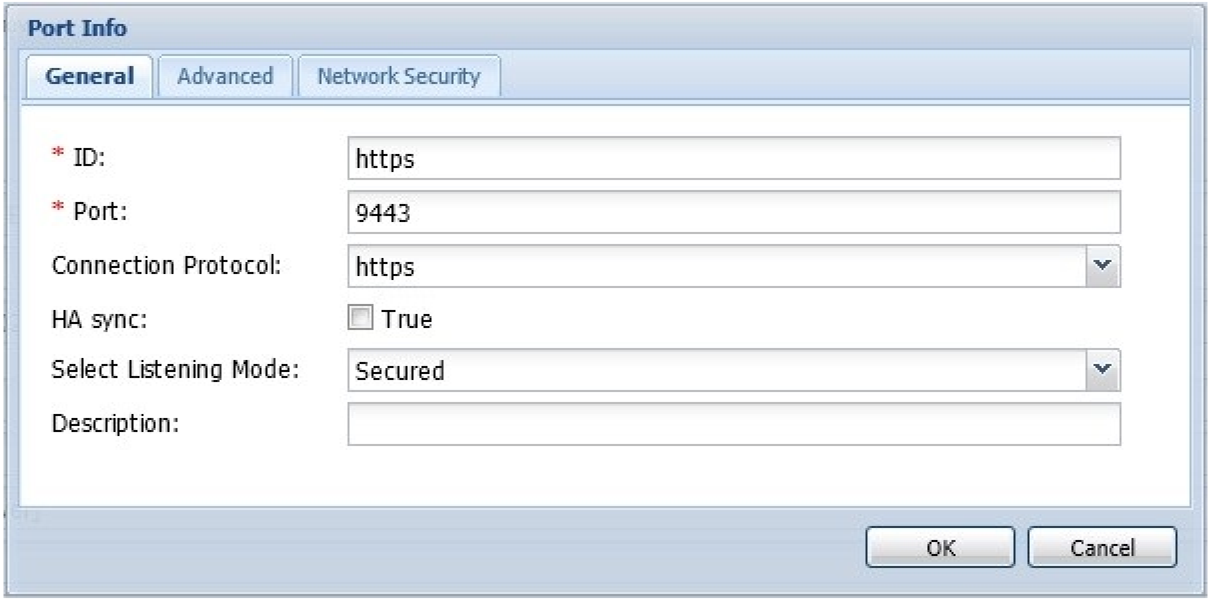
Description: Default http port for the Backend Server

Default http port

- Click OK. The HTTP port with the default identifier appears in the list of `Listening ports`.

6. Optionally, you can add a secure listening port for authenticated users, secured connections, and secure chat:

    - Click the Add button. The `Port Info` dialog opens.
    - Enter the ID. For instance, `https`.
    - Enter the Port. For instance, 9443.
    - Enter `https` for the Connection Protocol.
    - Choose `secured` for the Listening Mode.
    - Click `OK`.



* ID: https

* Port: 9443

Connection Protocol: https

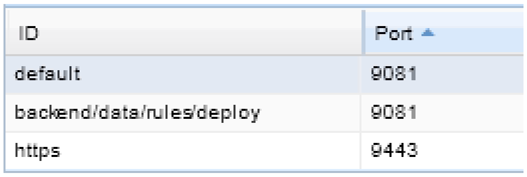
HA sync: □ True

Select Listening Mode: Secured

Description:

Secure listening port

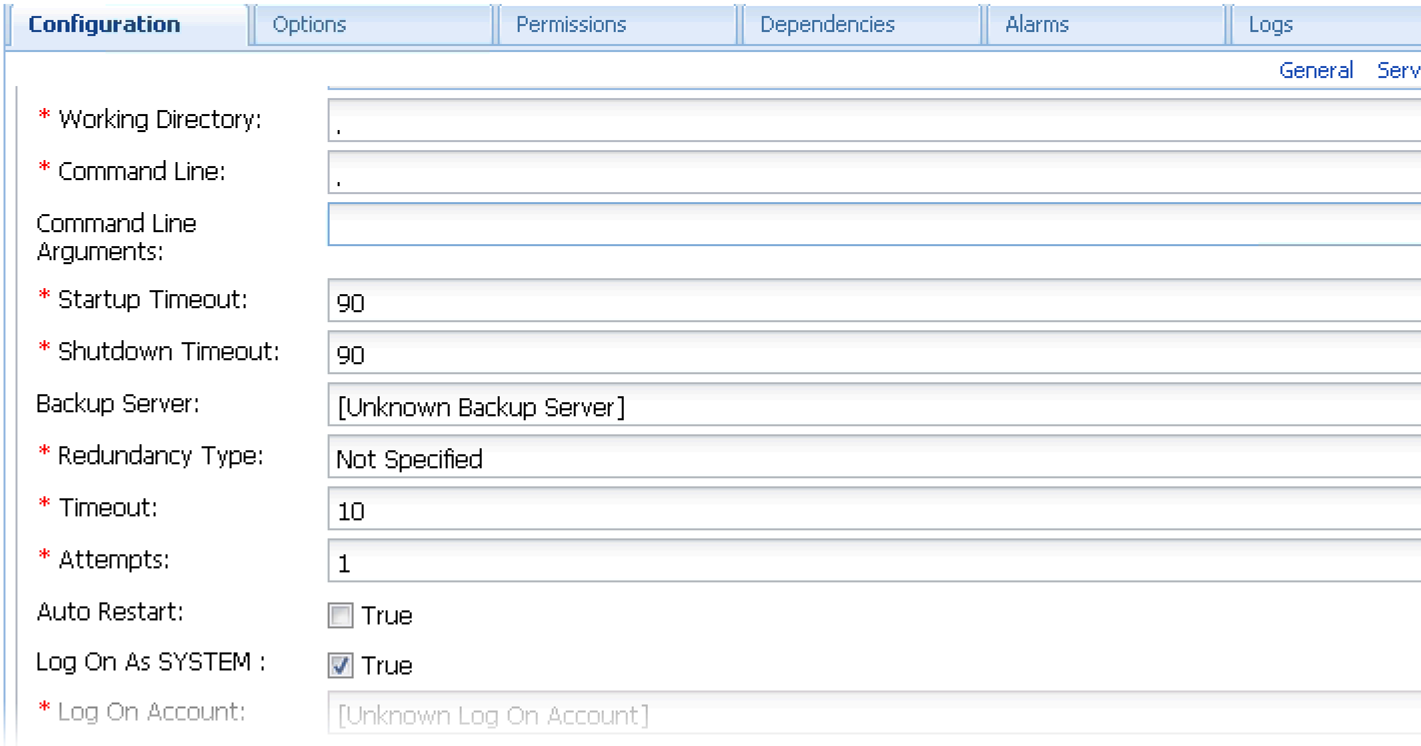
7. Create the port with the `backend/data/rules/deploy` identifier.
   - Click Add in the Listening ports section.
   - Enter `backend/data/rules/deploy` for the ID.
   - Enter the same port value as you did for the default port. For instance, 9081.
   - Select `http` for the `Connection Protocol` field.
   - (Optional) Enter a description.
   - Click `OK`. The HTTP port with `backend/data/rules/deploy` ID appears in the list of Listening ports.



ID	Port ▲
default	9081
backend/data/rules/deploy	9081
https	9443

Listening ports

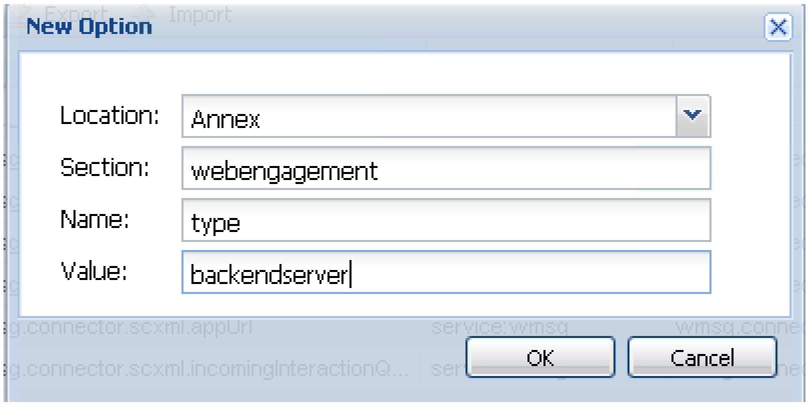
8. Enter a period (.) in Working Directory. The path will be filled in automatically when the new Application is installed and connects to Configuration Server;
9. Enter a period (.) in Command Line. The command line will be filled in automatically when the new Application is installed and connects to Configuration Server;
10. Leave Command Line Arguments blank. Any arguments will be filled in automatically when the new Application is installed and connects to Configuration Server;



* Working Directory:	.
* Command Line:	.
Command Line Arguments:	
* Startup Timeout:	90
* Shutdown Timeout:	90
Backup Server:	[Unknown Backup Server]
* Redundancy Type:	Not Specified
* Timeout:	10
* Attempts:	1
Auto Restart:	☐ True
Log On As SYSTEM :	☑ True
* Log On Account:	[Unknown Log On Account]

Commands

11. Click Save. The Confirm dialog displays the following message: The host and/or port(s) of the application will be changed. Do you want to continue?
    Click Yes.
12. Select the Options tab. By default, the all option is set to stdout and does not enable logging to a file. Enter a filename if you need logging to file. For example, enter the value "stdout, C:\Logs\WebEngagement\Backend_Server" to write logs both in the standard output console and in a file located in the c:\Logs\WebEngagement\ directory.
13. (Optional) You need to create a webengagementannex only if your Web Engagement Backend Server application's type is Genesys Generic Server. Click New on the Options menu.
    - Set Location to Annex.
    - Enter webengagement for the Section.
    - Enter type for the Name.
    - Enter backendserver for the Value.



Location: Annex
Section: webengagement
Name: type
Value: backendserver

Backend type.

- Click OK to create the new annex. You can see it by selecting Advanced View (Annex) in the View selector.
- If you added a connection to the Frontend Server, in the settings section, set event-mode to true.

14. Click the Save & Close button. If the Confirmation dialog opens, click Yes.

End

- Genesys recommends you set the Java heap size for the Backend Server to 3GB or more for a production site. See Generic Cassandra Configuration for details.
- The configuration application for the Backend Server is ready for provisioning. You may need to edit options later according to your application needs.
- If you did not add the Frontend Server to the Connections of the Backend Server, you must do it after the creation of the Frontend Server application.

Next Steps

- If you haven't configured the Frontend Server, go to Configure the Frontend Server.
- If you have configured the Frontend Server, you can install Genesys Web Engagement:
  - Install on Windows
  - Install on Linux

Chat Server Cluster Configuration

Genesys Web Engagement uses Web API Server as the entry point for a cluster of Chat Servers.

> **Important**
> A pair of primary/backup Chat Servers can also be used instead of a standalone Chat

> Server application.

To implement a Chat Server cluster for Web Engagement, you must first have a Web API Server application with connections to one or more Chat Server applications or one or more applications of the type Application Cluster. Your Web API Server must also have a connection to Solution Control Server.

For information about how to deploy and configure Web API Server, see the "Manual Deployment—Web API Server and UCS" chapter in the eServices 8.1 Deployment Guide.

When dealing with Web API Server, Web Engagement uses the eServices load balancer component to choose a Chat Server from the cluster for establishing a new chat session. See the "Load-Balancing Configuration for Web API Server" section in the eServices 8.1 User's Guide for more information about creating a Chat Server Application Cluster.

Once your Web API Server application is configured, you can Add a connection to Web API Server on the Backend Server application.

Add a connection to Web API Server on the Backend Server application

Prerequisites
For each Web Engagement Backend Server, you must have a dedicated Web API Server with connections to:

- One or more Chat Server applications, or one or more applications of the type Application Cluster.
- Solution Control Server.

> Important
>
> Web Engagement Backend Server uses Web API Server even if both Chat Server and Web API Server are specified in the Backend Server application's connections. In this case, the direct connection to Chat Server is ignored.

Start

1. Open Genesys Administrator and navigate to `PROVISIONING > Environment > Applications`. Select the application defined for the Web Engagement Backend Server and click `Edit...`
2. In the `Connections` section of the `Configuration` tab, click the `Add` button. The `Browse for applications` panel opens.
   - Select the Genesys application defined for the Web API Server, then click `OK`. Web API Server is added to the `Connections` list.
3. Click Save.

End

Configure the Frontend Server

Purpose: Describes the configuration tasks related to the Web Engagement Frontend Server.

Configuring the Frontend Server

Purpose: Configure the application previously created for the Frontend Server.

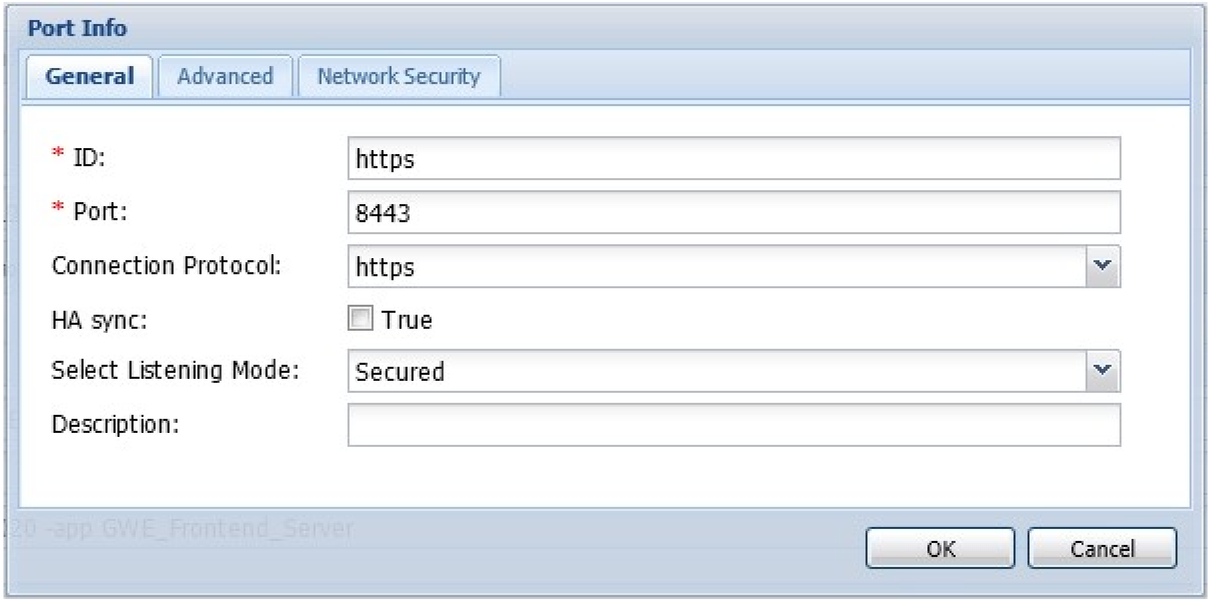
Prerequisites

- The Web Engagement Frontend Server application is created.

Start

1. In Genesys Administrator, navigate to `Provisioning > Environment > Applications`, select the Web Engagement Frontend Server application and click the `Edit...` button.
2. If the Backend Server has been configured, add a connection to it:
   - In the Connections section, click the `Add` button. The `Browse Applications` dialog box opens.
   - Select the Web Engagement Backend Server application and click `OK`. The Backend Server is added to the list of Connections.
   - Select the Backend Server from the list of connections and click `Edit`. In the `Connection Info` window, select the default port for the `ID` and set the `Connection` protocol to `http`. Click `OK`.
3. Expand the `Server Info` pane.
4. In the `Tenant` section, click the `Add` button and select your tenant. For instance, `Environment`. Click OK.
5. In the Listening Ports section, click the `Add` button. The `Port Info` dialog opens.
   - Enter the Port. For instance, 8081.
   - Choose `http` for the Connection Protocol.
   - Click `OK`.
6. Optionally, you can add a secure listening port for authenticated users, secured connections, and secure chat:
   - Click the `Add` button. The `Port Info` dialog opens.
   - Enter the ID. For instance, `https`.
   - Enter the Port. For instance, 8443.
   - Enter `https` for the Connection Protocol.
   - Choose `secured` for the Listening Mode.

- Click 0K.



* ID: https

* Port: 8443

Connection Protocol: https

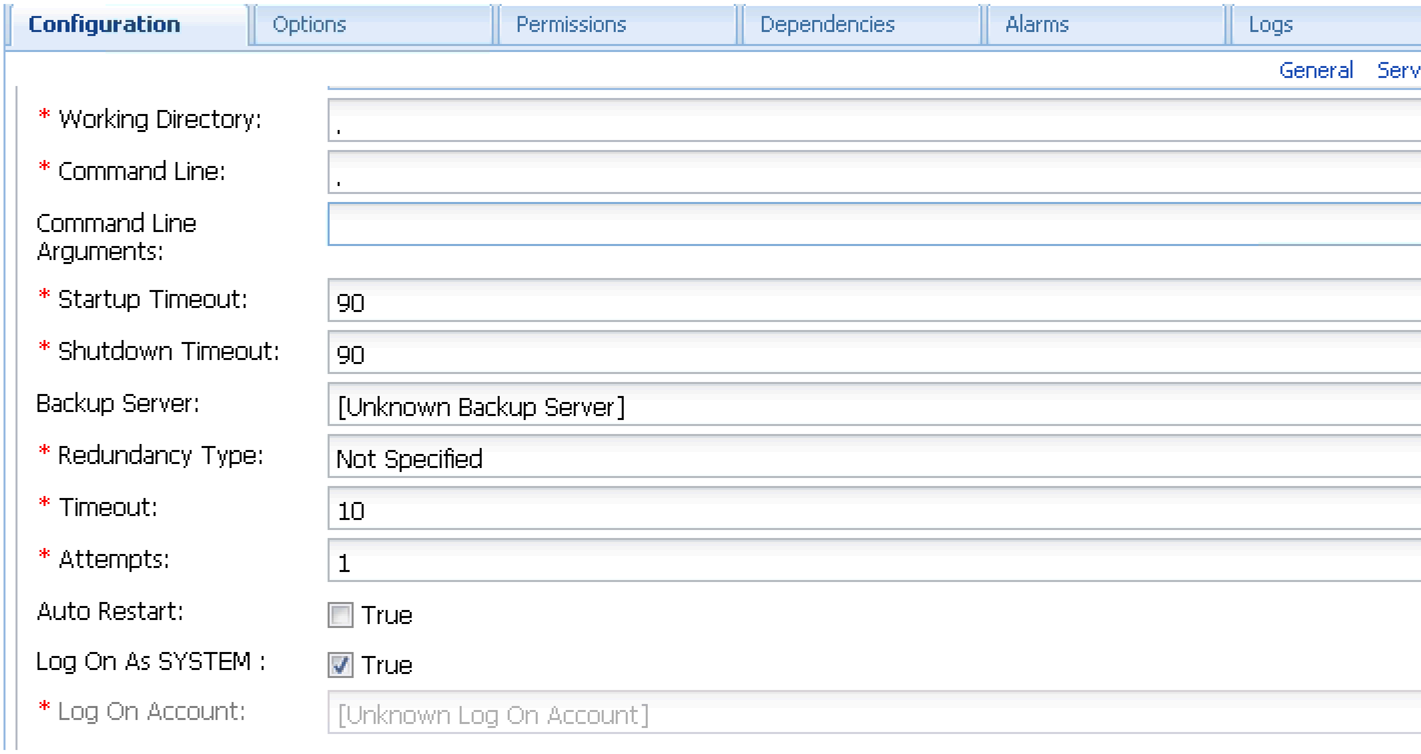
HA sync: □ True

Select Listening Mode: Secured

Description:

Secure listening port

7. Enter a period (.) in Working Directory. The path will be filled in automatically when the new Application is installed and connects to Configuration Server;
8. Enter a period (.) in Command Line. The command line will be filled in automatically when the new Application is installed and connects to Configuration Server;
9. Leave Command Line Arguments blank. Any arguments will be filled in automatically when the new Application is installed and connects to Configuration Server;



* Working Directory: .

* Command Line: .

Command Line Arguments:

* Startup Timeout: 90

* Shutdown Timeout: 90

Backup Server: [Unknown Backup Server]

* Redundancy Type: Not Specified

* Timeout: 10

* Attempts: 1

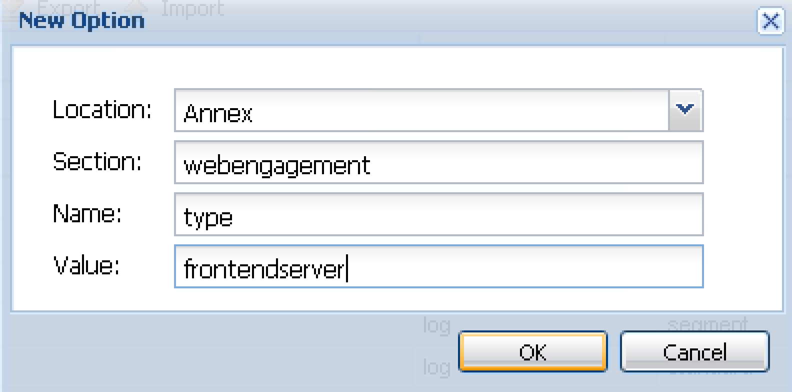
Auto Restart: □ True

Log On As SYSTEM : ☑ True

* Log On Account: [Unknown Log On Account]

Commands for the Frontend Server.

10. Click the Save button.
11. The Confirmation dialog for changing the application's port opens. Click Yes.
12. (Optional) Select the `Options` tab. In the `log` section, the `all` option is set to `stdout` by default. Enter a filename if you wish to enable logging to a file. For example, you can enter `stdout, c:\logs\WebEngagement\Frontend_Server` to force the system to write logs in the console and in a file.
13. (Optional) You need to create a `webengagement` annex only if your Web Engagement Frontend Server application's type is Genesys Generic Server. Click `New` in the `Options` tab. The `New Option` dialog opens.
    - Select Annex for Location.
    - Enter `webengagement` for Section.
    - Enter `type` for Name.
    - Enter `frontendserver` for Value.



Location: Annex

Section: webengagement

Name: type

Value: frontendserver

Annex for the Frontend.

- Click OK. You can see it by selecting `Advanced View (Annex)` in the `View` selector.

14. Click the `Save & Close` button. If the Confirmation dialog opens, click `Yes`.

End

Next Steps

- If you haven't configured the Backend Server, go to Configure the Backend Server.
- If you haven't added a connection to the Frontend Server on the Backend Server, go to (Optional) Add the Frontend Server Application to the Connections of the Backend Server Application.
- If you have configured the Backend Server, you can install Genesys Web Engagement:
    - Install on Windows
    - Install on Linux

(Optional) Add the Frontend Server Application to the Connections of the Backend Server Application

Purpose: Finalize the configuration of the Backend Server application.
Warning: You do not need to perform this procedure if you have already added the Frontend Server Application to the connections of the Backend Server Application. **Start**

1. In Genesys Administrator, navigate to `Provisioning > Environment > Applications`, select the Web Engagement Backend Server application, and click the `Edit...` button.
2. Configure the connection:
    - In the `Connections` section, click the `Add` button. The `Browse Applications` dialog opens.
    - Select the Web Engagement Frontend Server application and click OK. The Frontend Server is added to the list of Connections.

  - Click Save. The `Confirm` dialog displays the following message: `The host and/or port(s) of the application will be changed. Do you want to continue?` Click Yes.

3. Select the `Options` tab. In the `settings` section, set `event-mode` to `true`.
4. Click `Save & Close`.

End

Next Steps

- If you have configured the Backendend Server, you can install Genesys Web Engagement:
  - Install on Windows
  - Install on Linux

Install on Windows

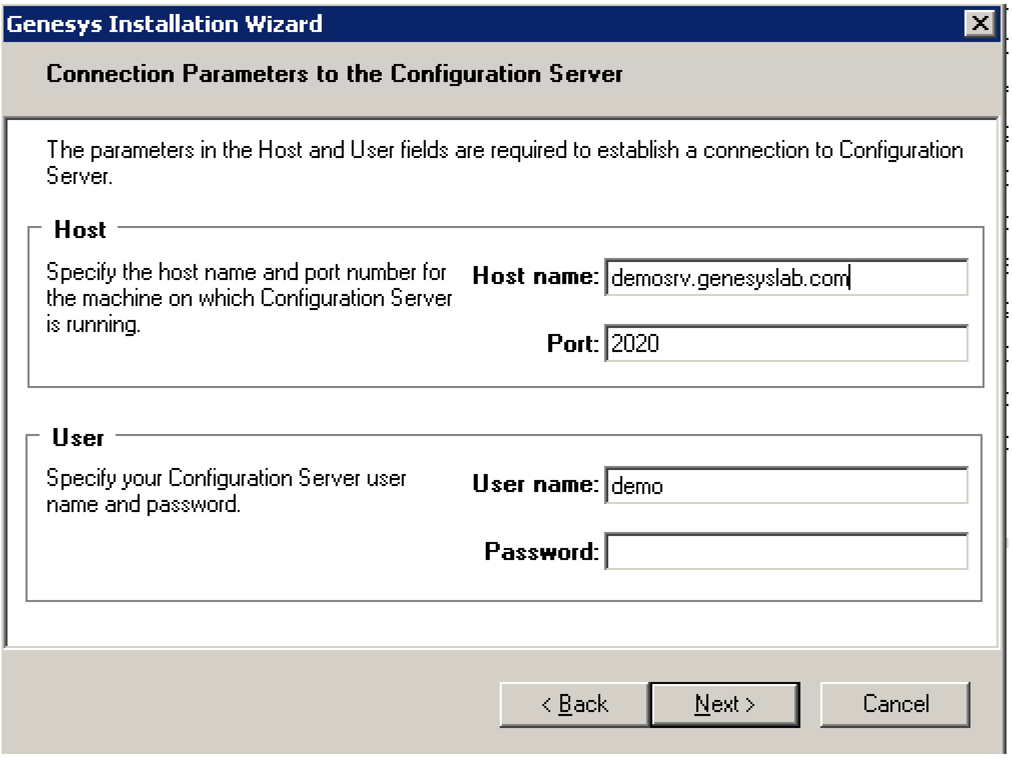
Installing Web Engagement Backend Server on the Windows operating system

Purpose
To install the deployment files for Genesys Web Engagement on the Windows web server.

	Note: After running one of the Windows installers, inspect the directory tree of your system to make sure that the files have been installed in the location that you intended.

Start

1. On your desktop, open the Genesys Web Engagement CD/DVD or the Genesys Web Engagement IP and double-click the `Setup.exe` file.
   You might be asked to reboot your system to delete or rename certain system files before the Installation Wizard runs.
   The Genesys Installation Wizard launches and the `Welcome` panel is displayed.
2. On the `Welcome` panel, do one of the following:
   - Click `Next` to begin the installation procedure.
   - Click `Cancel` to exit the Genesys Installation Wizard.
   - Click `About` to open the Genesys Web Engagement ReadMe file in your default browser.
     If you clicked `Next`, the Connection Parameters to the Configuration Server window is displayed.
3. Enter Connection parameters to Configuration Server:



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: demosrv.genesyslab.com

Port: 2020

User

Specify your Configuration Server user name and password.

User name: demo

Password:

You must enter connection values to your Configuration Server.

Click `Next`.

4. In the 'Select Application' dialog, select the Web Engagement Backend Server that you created previously in Configuration Server.
5. On the `Choose Destination Location` panel, specify the location on your web server in which Genesys Web Engagement is to be installed by doing one of the following:
   - Type a location in the `Destination Folder` text box.
   - Click `Default` to reset the location to the default location.
   - Click `Browse` to navigate to a destination folder.
     Note: This location must match the previous settings that you entered in Configuration Server.
6. With the destination folder specified, do one of the following:
   - Click `Next`.
   - Click Back to return to the `Select Options` panel.
   - Click `Cancel` to exit the Genesys Installation Wizard.
7. If you clicked Next, the `Ready to Install` panel is displayed. Do one of the following:
   - Click `Install` to install Genesys Web Engagement Backend Server.
   - Click Back to return to the `Choose Destination Location` panel.
   - Click `Cancel` to exit the Genesys Installation Wizard.
8. If the Installation Complete panel is displayed, click `Finish` to exit the Genesys Installation Wizard.

End

Next Steps

- Installing Web Engagement Frontend Server on the Windows operating system

Installing Web Engagement Frontend Server on the Windows operating system

Purpose

To install the deployment files for Genesys Web Engagement Frontend Server on the Windows web server.

	Note: After running one of the Windows installers, inspect the directory tree of your system to make sure that the files have been installed in the location that you intended.

Start

1. On your desktop, open the Genesys Web Engagement CD/DVD or the Genesys Web Engagement IP and double-click the `Setup.exe` file.
   You might be asked to reboot your system to delete or rename certain system files before the Installation Wizard runs.
   The Genesys Installation Wizard launches and the `Welcome` panel is displayed.
2. On the `Welcome` panel, do one of the following:
   - Click `Next` to begin the installation procedure.
   - Click `Cancel` to exit the Genesys Installation Wizard.
   - Click `About` to open the Genesys Web Engagement ReadMe file in your default browser.If you clicked Next, the Choose Destination Location panel is displayed.
3. If you previously installed a Backend Server on this host, in the `Select Installed Application` dialog you must select the Backend Server that you installed. Click `Next`.
4. Enter Connection parameters to the Configuration Server. Click `Next`.
5. In the `Select Application` dialog, select the Web Engagement Frontend Server that you created previously in Configuration Server.
6. On the `Choose Destination Location` panel, specify the location on your web server in which Genesys Web Engagement is to be installed by doing one of the following:
   - Type a location in the `Destination Folder` text box.
   - Click `Default` to reset the location to the default location.
   - Click `Browse` to navigate to a destination folder.
7. With the destination folder specified, do one of the following:
   - Click `Next`.

   - Click Back to return to the `Select Options` panel.
   - Click `Cancel` to exit the Genesys Installation Wizard.

8. If you clicked `Next`, the Ready to Install panel is displayed. Do one of the following:
   - Click `Install` to install Genesys Web Engagement Frontend Server.
   - Click Back to return to the `Choose Destination Location` panel.
   - Click `Cancel` to exit the Genesys Installation Wizard.
9. Click `Finish` to exit the Genesys Installation Wizard.

End

- Provisioning is done automatically.

Next Steps

- See Installing Plug-ins.

Install on Linux

Installing Web Engagement Backend Server on the Linux operating system

Purpose: To install the deployment files for the Genesys Web Engagement Backend Server on the Linux server.

	Note: After running one of the installers, inspect the directory tree of your system to make sure that the files have been installed in the location that you intended.

Start

1. Open a terminal in the Genesys Web Engagement CD/DVD or the Genesys Web Engagement IP, and run the `Install.sh` file.
   The Genesys Installation starts.
2. Enter the hostname of the host on which you are going to install.
3. Enter the connection information to log in to Configuration Server:
   - The hostname. For instance, demosrv.genesyslab.com;
   - The listening port. For instance, 2020.
   - The user name. For instance, demo.
   - The password.
   - If the connection settings are successful, a list of keys and Web Engagement applications is displayed.
4. Enter the key for the Web Engagement Backend Server application that you created previously in Configuration Server.
5. Enter the location where Genesys Web Engagement is to be installed on your web server.
   Note: This location must match the previous settings that you entered in Configuration Server.
6. If the installation is successful, the console displays the following message:

```
Installation of Genesys Web Engagement Backend, version 8.1.x has completed successfully.
```

End

Next Steps

- Installing Web Engagement Frontend Server on the Linux operating system

Installing Web Engagement Frontend Server on the Linux operating system

Purpose: To install the deployment files for the Genesys Web Engagement Frontend Server on the Linux server.

Note: After running one of the installers, inspect the directory tree of your system to make sure that the files have been installed in the location that you intended.

Start

1. Open a terminal in the Genesys Web Engagement CD/DVD or the Genesys Web Engagement IP, and run the `Setup.sh` file.
   The Genesys Installation starts.
2. Enter the hostname of the Configuration Server.
3. Enter the connection information to log in to the Configuration Server:
   - The hostname. For instance, demosrv.genesyslab.com.
   - The listening port; for instance, 2020.
   - The user name. For instance, demo.
   - The password.
   - If the connection settings are successful, a list of keys and Web Engagement applications is displayed.
4. Enter the key for the Web Engagement Frontend Server application that you created previously in Configuration Server.
5. Enter the location where Genesys Web Engagement is to be installed on your web server.
   Note: This location must match the previous settings that you entered in Configuration Server.
6. Enter 32 to install the 32-bit version of the product or 64 to install the 64-bit version of the product;
7. If the installation is successful, the console displays the following message:

```
Installation of Genesys Web Engagement Frontend, version 8.1.x has completed successfully.
```

End

- Provisioning is done automatically.

Next Steps

- See Installing Plug-ins.

Provisioning

Purpose: Lists the procedures related to Provisioning for the Genesys Web Engagement Servers.
Once provisioning is completed, you can create a Web Engagement application, as detailed in the Developer's Guide.

Provisioning for Web Engagement

Purpose Create all the configuration information related to Genesys Web Engagement in Configuration Server by running the Provisioning Tool available in the `tools\provisioning` directory. This tool connects to Configuration Server, reads the Web Engagement applications' configuration, creates the Genesys objects used by the Web Engagement Servers, and edits the XML configuration files required to launch the Web Engagement Servers.

Note: The Provisioning Tool is run automatically as part of the installation process, but you can also run the tool to modify your configuration information after Genesys Web Engagement is installed.

Prerequisites

- The configuration applications for the Web Engagement Servers are created in Configuration Server.
- The connections for the Backend Server application include the Frontend Server, Chat Server, Contact Server, Interaction Server, Orchestration Server, and the StatServer applications.
- The connections for the Frontend Server application include the Backend Server.
- You have a webengagement section created in the annex for the Backend and Frontend Server applications if you are using Genesys Generic Server templates. This section includes the `type` option set to:
  - `backendserver` for the Backend Server
  - `frontendserver` for the Frontend Server.

Start

1. Navigate to the Web Engagement installation directory and open the `tools\provisioning` folder.
2. To launch provisioning, open a Windows Command Prompt (cmd.exe) and type:

   ```
   webengagement_provisioning.bat -host <hostname> -port 2020 -user <user> -password
    <password>
    -app <Application name for Web Engagement Frontend Server>
   ```

   Note: User and password options may be optional, according to your Configuration Server's settings.

3. The provisioning script starts. If the provisioning is successful, the following message is displayed:

   ```
   Provisioning script successfully finished his work
   ```

End

- If this is not the first time that you run the provisioning tool, use the overwrite option. In overwrite mode, the provisioning tool replaces old objects with new objects.

  ```
  webengagement_provisioning.bat -host <hostname> -port 2020 -app <Application name for
  Web Engagement Frontend Server> -overwrite
  ```

- Now, you should be able to implement a Web Engagement application. See the Developer's Guide for further details.

Next Steps

- See Installing Genesys Web Engagement.

Installing the Plug-ins

	Purpose: To list the Installation tasks for the Genesys Web Engagement Plug-ins.

Tasks	Procedures
Install the Genesys Web Engagement Plug-in for Interaction Workspace.	The Genesys Web Engagement Plug-in for Interaction Workspace allows you to enable chat and web callback engagement features in Interaction Workspace (see Using the Plug-in for Interaction Workspace in the Developer’s Guide). To install this plug-in, complete the following procedures: • Install the Plug-in for Interaction Workspace
Install the Genesys Web Engagement Plug-in for Genesys Administrator Extension.	The Genesys Web Engagement Plug-in for Genesys Administrator Extension allows you to: • Use the Script Generator tool to create a JavaScript code snippet for your web pages (see Enable Web Engagement Monitoring in the Developer's Guide). • Create customized business information using the simple model (see Simple Engagement in the Developer's Guide). To install this plug-in, complete one of the following procedures: • Installation of the Genesys Web Engagement Plug-in for Administrator Extension on Windows • Installation of the Genesys Web Engagement Plug-in for Administrator Extension on Linux

Install the Plug-in for Interaction Workspace

Purpose: List the Installation tasks for the Genesys Web Engagement Plug-in for Interaction Workspace.

Install the Genesys Web Engagement Plug-in for Interaction Workspace on Windows

Prerequisites

- Interaction Workspace, version 8.1.301.10 and later, is installed.
- Installation must be driven from the host where you intend to install the plug-in.

Start

1. On your desktop, open the Genesys Web Engagement CD/DVD or the Genesys Web Engagement IP, navigate to the `web_engagement_iws_plug-in\windows` folder, and double-click the Setup.exe file.
2. On the Welcome panel, do one of the following:
   - Click `Next` to begin the installation procedure.
   - Click `Cancel` to exit the Genesys Installation Wizard.
   - Click `About` to open the Genesys Web Engagement ReadMe file in your default browser.If you clicked Next, a list of installed application is displayed.
3. Select the Interaction Workspace IP and click `Next`.
4. On the Ready to install panel, click the `Install` button to install the plug-in.
5. Complete the installation. As a result, the following files were copied to the Interaction Workspace installation directory:
   - InteractionWorkspace\Languages\Genesyslab.Desktop.Modules.WebEngagement.en-US.xml,
   - InteractionWorkspace\Genesyslab.Desktop.Modules.WebEngagement.dll
   - InteractionWorkspace\Genesyslab.Desktop.Modules.WebEngagement.module-config
   - InteractionWorkspace\Newtonsoft.Json.Net35.dll

End

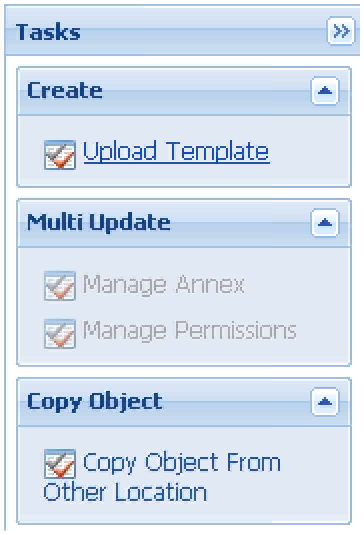
Next Steps

Import the Interaction Workspace Plug-in template

Import the Interaction Workspace Plug-in template

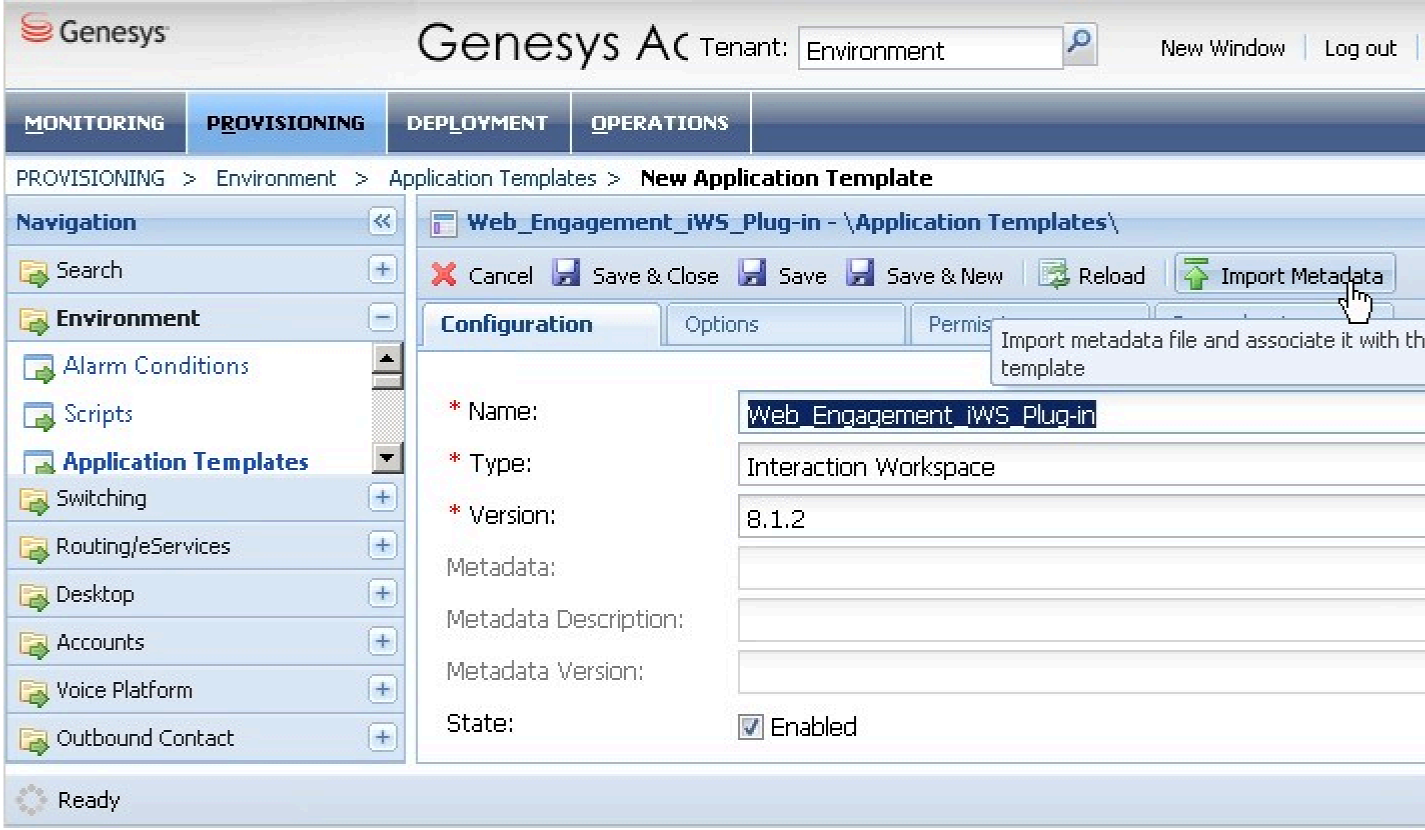
Purpose: Import the template which enables you to create the configuration of the Web Engagement servers in the Configuration Server.
Start

1. Open Genesys Administrator and navigate to `PROVISIONING > Environment > Application Templates`.
2. In the `Create` menu of the `Tasks` panel, click on the `Upload Template` link.



Upload Template link in the Tasks panel

3. Click the Add button of the `Click 'Add' and choose application template (APD) file to import` dialog box.
4. Browse the `Web_Engagement_iWS_Plug-in_811.apd` file available in the `templates` directory of your installation CD. The Configuration tab for the new template opens.
5. Click the `Import Metadata` button.



* Name: Web_Engagement_iWS_Plug-in

* Type: Interaction Workspace

* Version: 8.1.2

Metadata:

Metadata Description:

Metadata Version:

State: ☑ Enabled

Tenant: Environment

Click the Import Metadata button.

6. Select `Web_Engagement_iWS_Plug-in_811.xml` metadata file and click `Open`. The metadata fields in the Configuration tab are now filled.
7. Click `Save & Close`.

End

Next Steps

Add a connection to the Backend Server

Add a connection to the Backend Server

Purpose: Add a connection to the Backend Server to Interaction Workspace.

Start

1. In Genesys Administrator, navigate to `Provisioning > Environment > Applications`, select the Interaction Workspace application and click the `Edit...` button.

2. Configure the connection:
   - In the Connections section, click the Add button. The Browse Applications dialog opens.
   - Select the Web Engagement Backend Server application and click OK. The Backend Server is added to the list of Connections.

End

Next Steps

- Configure Authentication (optional)

Configure Authentication (optional)

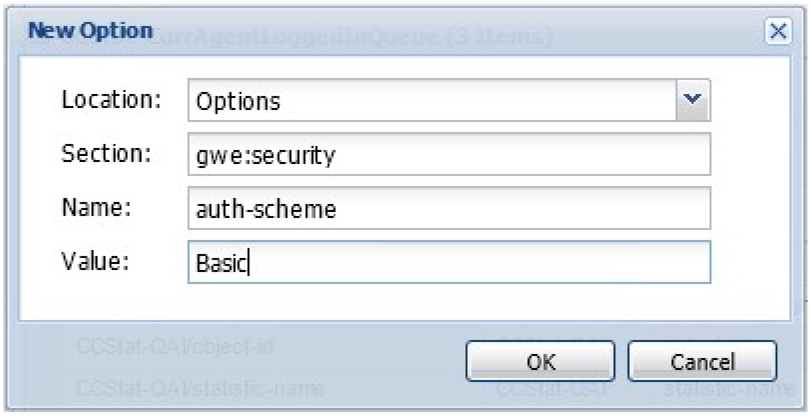
Purpose: To configure the security options to support authentication for the REST API.

Prerequisites

- Your Web Engagement Backend Server supports authentication. See Authentication for details.

Start

1. In Genesys Administrator, navigate to Provisioning > Environment > Applications, select the Interaction Workspace application and click the Edit... button.
   Note: Before configuring the authentication options, be sure to read about each option to help determine the correct values for your deployment:
   - auth-scheme
   - user-id
   - password
2. Click the options tab and then click New.
3. In the New Option window, configure the following:
   a. Set Section to gwe:security
   b. Set Name to auth-scheme
   c. Set Value to your authentication scheme. For example, Basic.



New Option

Location: Options

Section: gwe:security

Name: auth-scheme

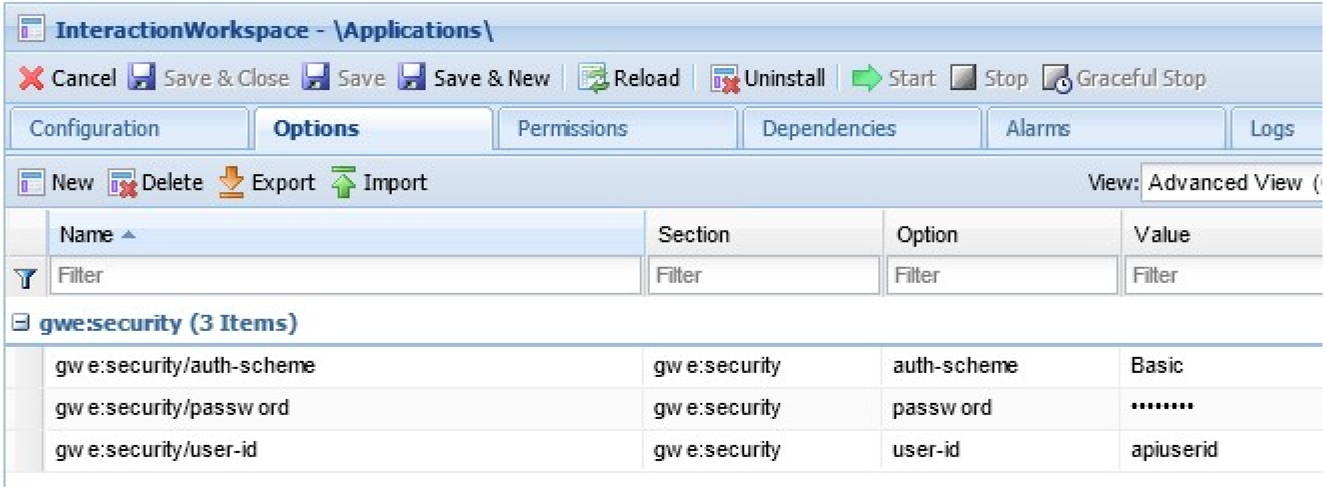
Value: Basic

d. Click OK

4. Complete steps a-d to configure the remaining security options:

Section	Name	Value
gwe:security	user-id	Your user ID.
gwe:security	password	Your user password.

Your configuration options for Interaction Workspace should now have a new section for the Genesys Web Engagement security options:



InteractionWorkspace - \Applications\

gwe:security (3 Items)

Name	Section	Option	Value
gwe:security/auth-scheme	gwe:security	auth-scheme	Basic
gwe:security/password	gwe:security	password	••••••••
gwe:security/user-id	gwe:security	user-id	apiuserid

New security options for Genesys Web Engagement.

5. Click Save & Close.

End

Next Steps

- Configure Role-Based Access Control for Web Engagement

Configure Role-Based Access Control for Web Engagement

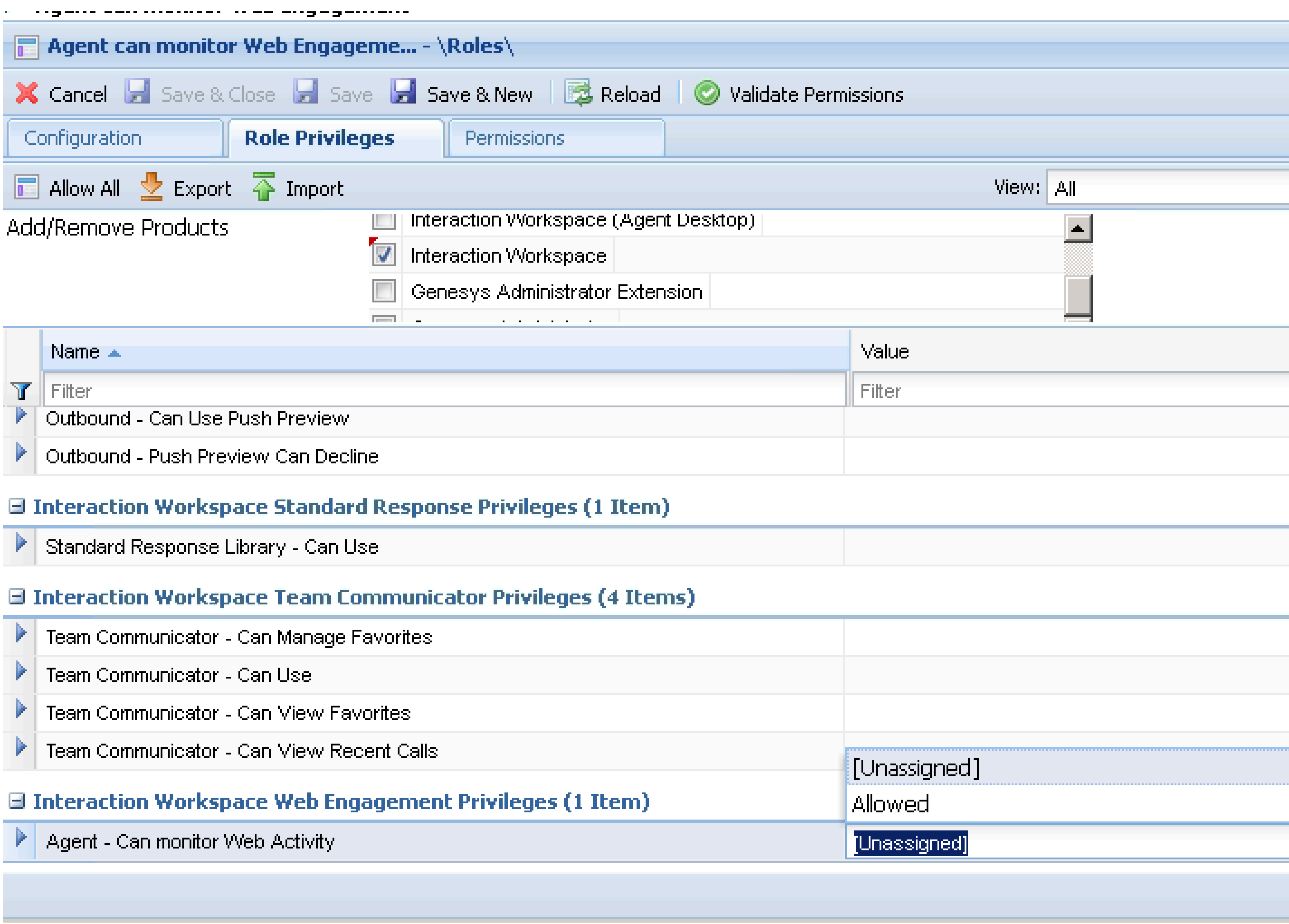
Purpose: To accomplish some higher-level business or configuration goal.

Prerequisites

- You already imported the Plug-in template for Interaction Workspace

Start

1. Open Genesys Administrator and navigate to `PROVISIONING > Accounts > Roles`
2. Edit or create a Role responsible for managing Web Engagement in Interaction Workspace; for instance, create the `Agent can Monitor Web Engagement` role by clicking the New button.
3. Select the `Role Privileges` tab.
4. In the `Add/Remove Products` top panel, enable Interaction Workspace and expand the Interaction Workspace Web Engagement Privileges section.
5. Set the `Allowed` value for the `Agent - Can Monitor Web Activity` option



- ☐ Interaction Workspace (Agent Desktop)
- ☑ Interaction Workspace
- ☐ Genesys Administrator Extension

Name	Value
Outbound - Can Use Push Preview	
Outbound - Push Preview Can Decline	
Interaction Workspace Standard Response Privileges (1 Item)	
Standard Response Library - Can Use	
Interaction Workspace Team Communicator Privileges (4 Items)	
Team Communicator - Can Manage Favorites	
Team Communicator - Can Use	
Team Communicator - Can View Favorites	
Team Communicator - Can View Recent Calls	
Interaction Workspace Web Engagement Privileges (1 Item)	
Agent - Can monitor Web Activity	[Unassigned]

Select Allowed

.

6. In the Members section of the Configuration tab, add users or groups who should get this role.
7. Click `Save and close`.

End

Next Steps

Back to Task Table

Install the Plug-in for GAX

Purpose: To describe deployment of the Genesys Web Engagement Plug-in for Genesys Administrator Extension.
The Genesys Web Engagement Plug-in for Genesys Administrator Extension adds the following menu items to the Genesys Administration Extension Tool:

- Categories interface – tool for creating and manage web engagement categories;
- Script Generator – tracking code generation tool.

Installation of the Genesys Web Engagement Plug-in for Genesys Administrator Extension on Windows

Prerequisites

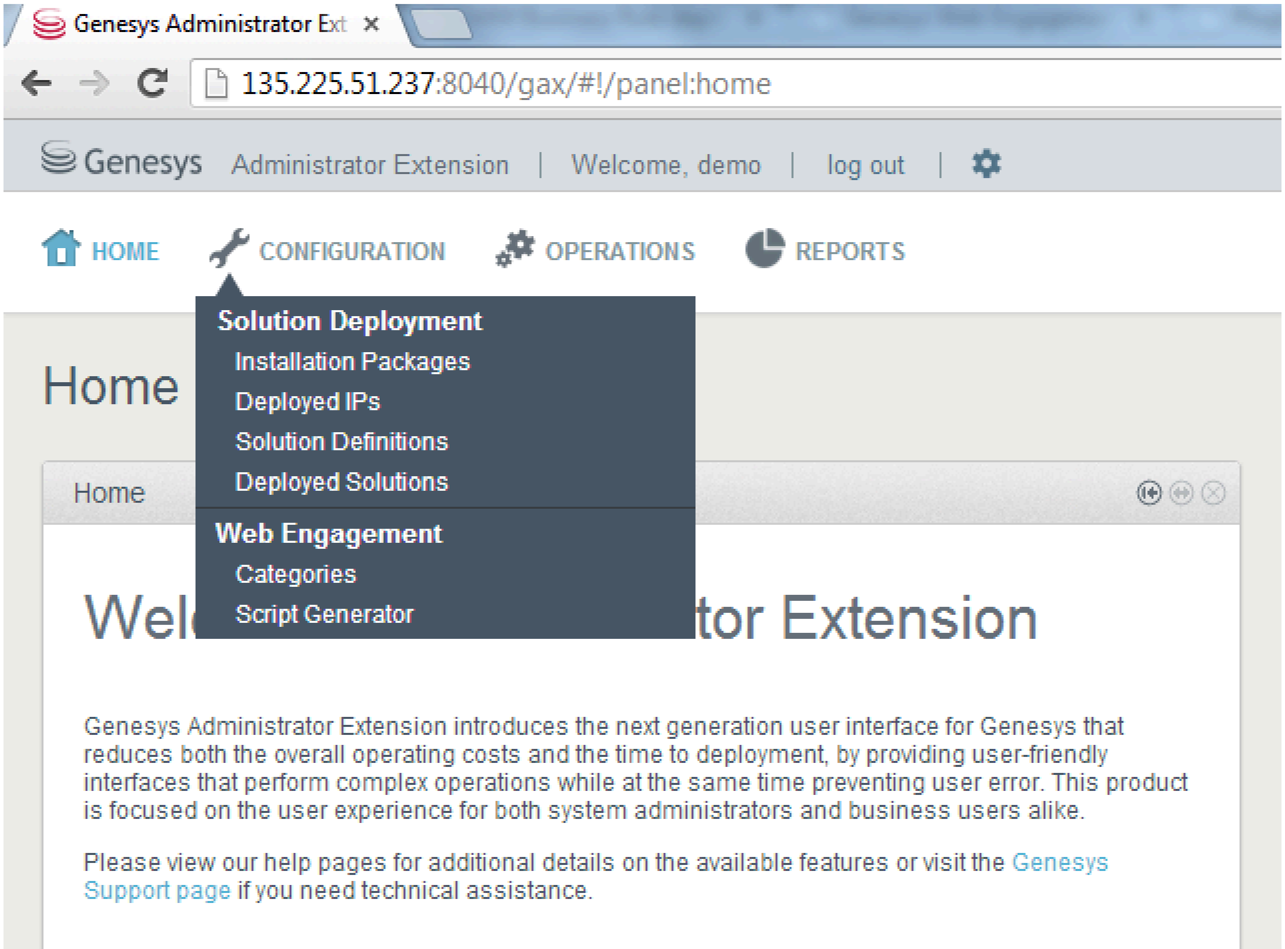
- Genesys Administrator Extension, version 8.1.301.02, or later, must be installed. For further information about installing the Genesys Administrator Extension, refer to the Genesys Administrator Extension Deployment Guide.
- Installation must be driven from the host where you intend to install the plug-in.

Start

1. Open Genesys Administrator and navigate to `Environment > Applications`. Select the Genesys Administrator Extension application and stop it.
2. On your desktop, open the Genesys Web Engagement CD/DVD or the Genesys Web Engagement IP, navigate to the web_engagement_gax_plug-in\windows folder, and double-click the Setup.exe file.
3. On the Welcome panel, do one of the following:
    - Click `Next` to begin the installation procedure.
    - Click `Cancel` to exit the Genesys Installation Wizard.
    - Click `About` to open the Genesys Web Engagement ReadMe file in your default browser.If you clicked Next, the Destination folder dialog is displayed.
4. Browse the installation folder or use the default location and click Next.
5. On the `Ready to install` panel, click the Install button to install the plug-in. As a result, the `gax-webme-plugin-[version].jar` and `wmcategory-[version].jar` files were copied to the `[Genesys Administrator Extension Tomcat server]/webapps/gax/WEB-INF/lib/` folder.

6. Restart Genesys Administrator Extension Server. Open Genesys Administrator and navigate to `Environment > Applications`. Select the Genesys Administrator Extension application and start it.
7. Open Genesys Administration Extension. The `CONFIGURATION` menu of Genesys Administrator Extension now includes `Web Engagement` items, `Category Manager`, and `Script Generator`.

End
For further information about the tools, see Using the Plug-in for Genesys Administrator Extension.



Genesys Administrator Extension introduces the next generation user interface for Genesys that reduces both the overall operating costs and the time to deployment, by providing user-friendly interfaces that perform complex operations while at the same time preventing user error. This product is focused on the user experience for both system administrators and business users alike.

Please view our help pages for additional details on the available features or visit the Genesys Support page if you need technical assistance.

Genesys Administrator Extension CONFIGURATION menu including the Web Engagement plug-in.

Next Steps

Back to Task Table

Installation of the Genesys Web Engagement Plug-in for Genesys Administrator Extension on Linux

Prerequisites

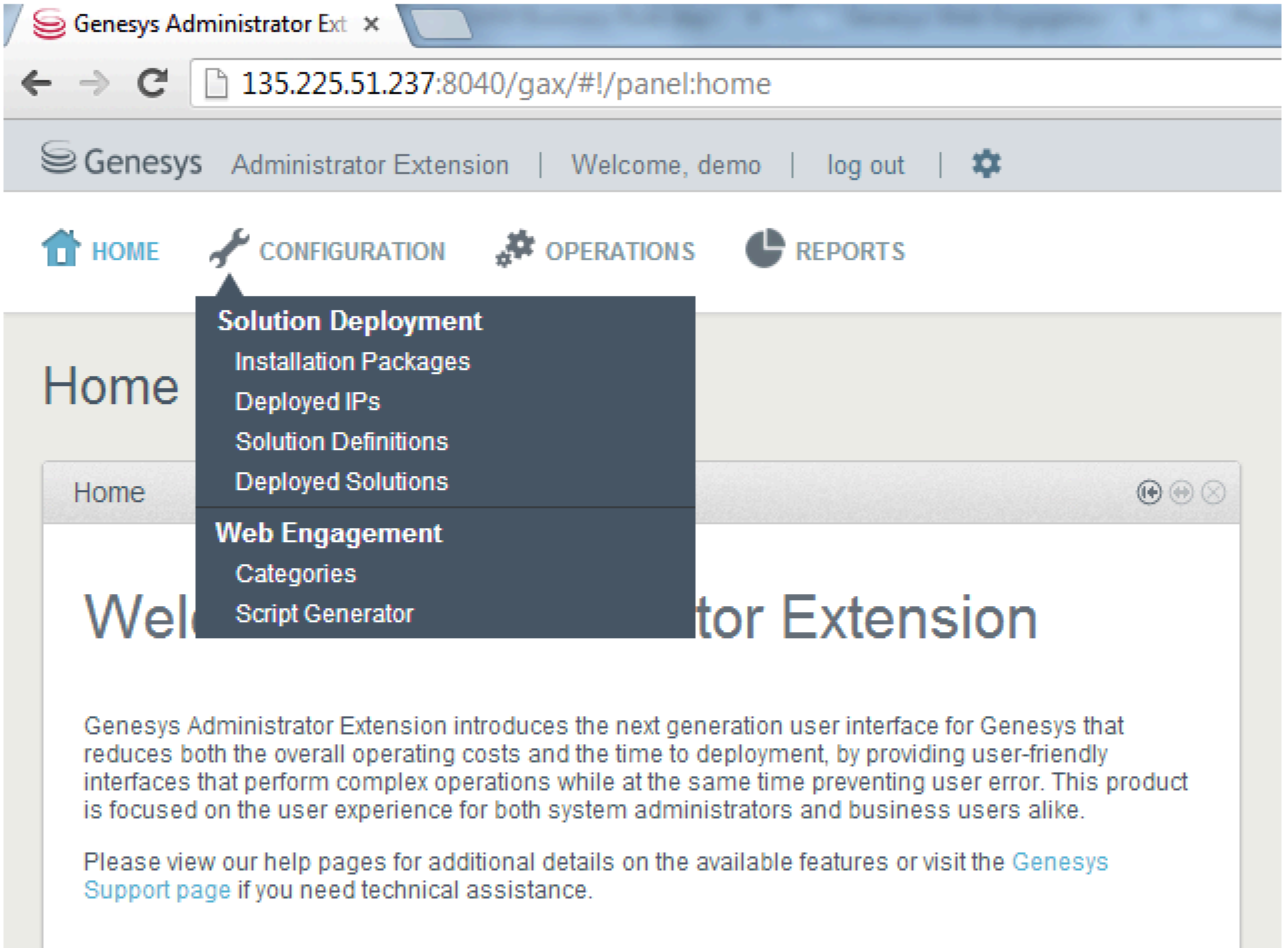
- Genesys Administrator Extension, version 8.1.301.02, or later, must be installed. For further information about installing the Genesys Administrator Extension, refer to the Genesys Administrator Extension Deployment Guide.
- Installation must be driven from the host where you intend to install the plug-in.

Start

1. Open Genesys Administrator and navigate to `Environment > Applications`. Select the Genesys Administrator Extension application and stop it.
2. On your console, open the Genesys Web Engagement CD/DVD or the Genesys Web Engagement IP, navigate to the web_engagement_gax_plug-in\linux folder, and run the Setup.sh script. The installation starts.
3. Enter the path to an installation folder.
4. The installation starts. As a result, the `gax-webme-plugin-[version].jar` and `wmcategory-[version].jar` files were copied to the `[Genesys Administrator Extension Tomcat server]/webapps/gax/WEB-INF/lib/` folder.
5. Restart Genesys Administrator Extension Server. Open Genesys Administrator and navigate to `Environment > Applications`. Select the Genesys Administrator Extension application and start it.
6. Open Genesys Administration Extension. The `CONFIGURATION` menu of Genesys Administrator Extension now includes `Web Engagement` items, `Category Manager`, and `Script Generator`.

Stop

For further information about the tools, see Using the Plug-in for Genesys Administrator Extension.



Genesys Administrator Extension introduces the next generation user interface for Genesys that reduces both the overall operating costs and the time to deployment, by providing user-friendly interfaces that perform complex operations while at the same time preventing user error. This product is focused on the user experience for both system administrators and business users alike.

Please view our help pages for additional details on the available features or visit the Genesys Support page if you need technical assistance.

Genesys Administrator Extension CONFIGURATION menu including the Web Engagement plug-in.

Next Steps

Back to Task Table

Setup Tasks

	Purpose: To list the Setup tasks to perform after the installation of Genesys Web Engagement.

Tasks	Procedures
1. Configure the Genesys Rules Authoring Tool to work with Genesys Web Engagement Servers.	Configuring the Genesys Rules Authoring Tool
2. Configure the Genesys Rules Development Tool to work with Web Engagement.	Configure the Genesys Rules Development Tool
3. Configure Genesys Roles for the Genesys Rules Authoring Tool (optional).	Configure Roles Settings for Rules Management **Note:** This step is optional if you already created a user with Roles Privileges that enable the creation of Rules package in Genesys Rules Authoring. See Role-Based Access Control in the Genesys Rules System Deployment Guide.
4. Configure the Pacing Algorithm to improve workload distribution (optional).	Complete the following procedures: • Configuring your Optimization Model • Configuring Agent Groups
5. Configure a Chat Channel (optional).	Configuring a Chat Channel
6. Import the CCPulse+ sample reporting templates.	Importing CCPulse+ Templates
7. Load the certificate and private keys into the Java and Jetty keystores.	Loading Certificate for SSL
8. Configure the Frontend Server to support UTF-8 in Configuration Server.	This procedure enables Frontend Server to support multi-language categories and should only be completed if your version of Configuration Server (8.1.200.04+) supports UTF-8. Configure the Frontend Server for UTF-8
9. Configure Cassandra for the Backend Server.	Generic Cassandra Configuration

Configure Genesys Rules Authoring Tool

Purpose: To enable the Genesys Rules Authoring Tool to work with Genesys Web Engagement servers.

Configuring Genesys Rules Authoring Tool

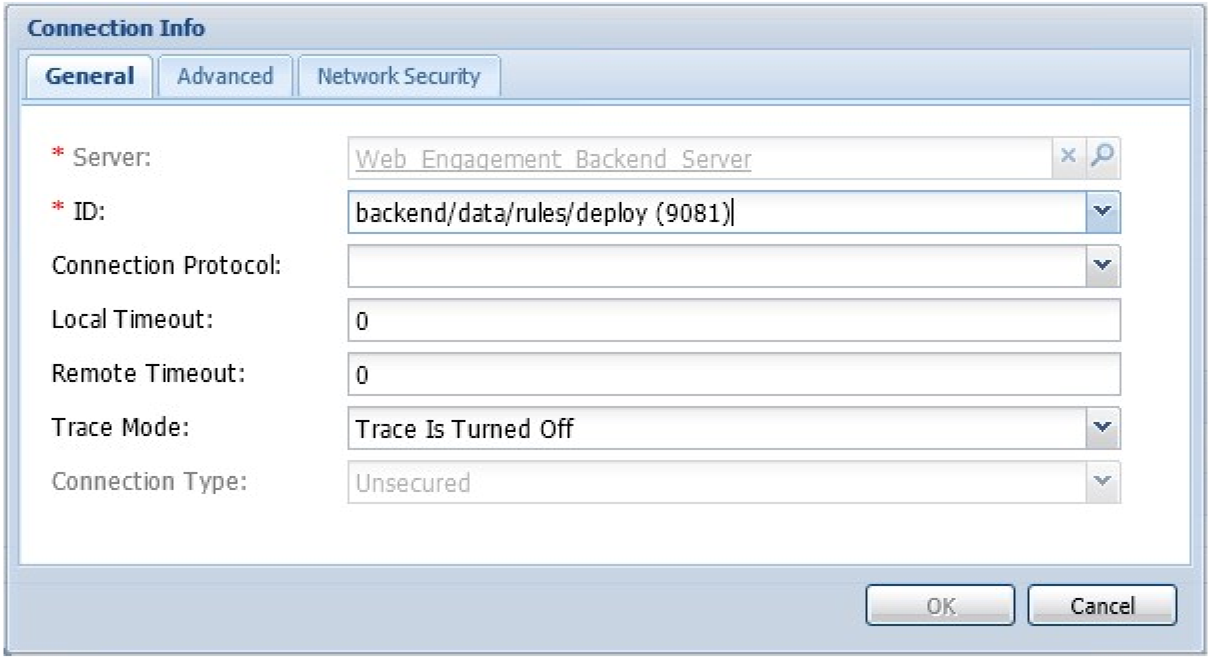
Purpose: To enable the Genesys Rules Authoring Tool to work with Genesys Web Engagement servers.

Prerequisites

- If your version of Genesys Rules Authoring Tool is older than 8.1.300.13, you must first Replace the GRAT JAR files.

Start

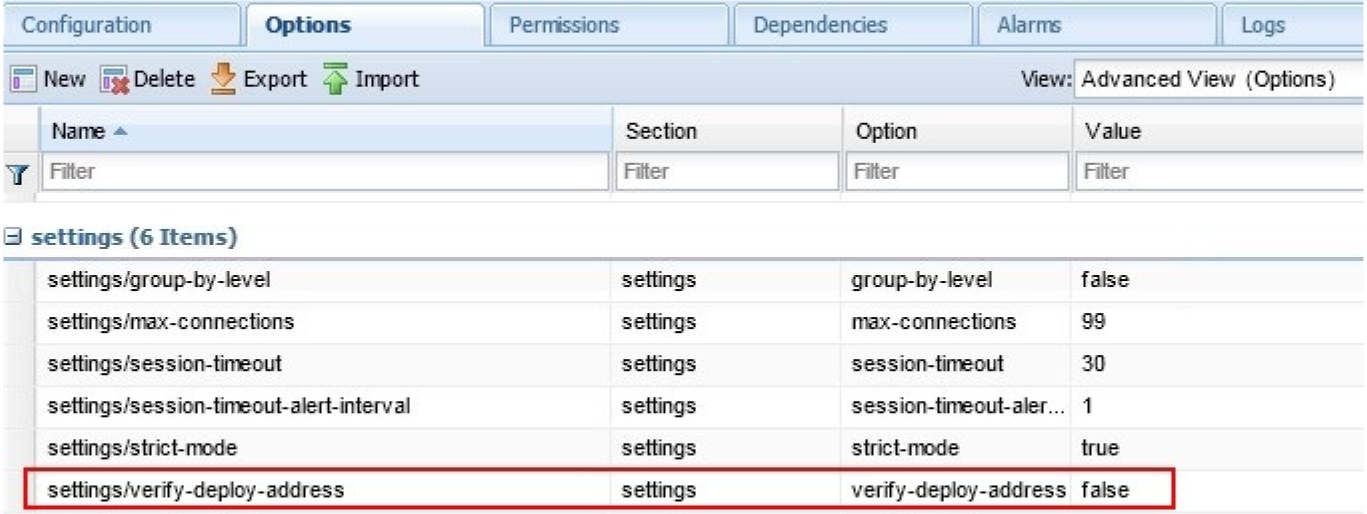
1. Add the Genesys Web Engagement Backend Server to the Connections of the Genesys Rules Authoring Tool.
   - In Genesys Administrator, navigate to `PROVISIONING > Environment > Applications`. Select the application of the Genesys Rules Authoring Tool and click `Edit...`.
   - In the `Connections` section, click `Add...`. Select the Genesys Web Engagement Backend Server, and click `OK`.
   - Select the Genesys Web Engagement Backend Server connection that was just added and click `Edit`. In the `Connection Info` window, select `backend/data/rules/deploy` for the ID field. Click `OK`.



* Server:	Web_Engagement_Backend_Server
* ID:	backend/data/rules/deploy (9081)
Connection Protocol:	
Local Timeout:	0
Remote Timeout:	0
Trace Mode:	Trace Is Turned Off
Connection Type:	Unsecured

Connection Info window

2. Configure Genesys Rules Authoring Tool. Select the Options tab and edit the `Settings` section:
    - In the `Settings` section, set `verify-deploy-address` to `false`.



settings (6 Items)

Name	Section	Option	Value
settings/group-by-level	settings	group-by-level	false
settings/max-connections	settings	max-connections	99
settings/session-timeout	settings	session-timeout	30
settings/session-timeout-alert-interval	settings	session-timeout-aler...	1
settings/strict-mode	settings	strict-mode	true
settings/verify-deploy-address	settings	verify-deploy-address	false

Options settings section

3. In the `Security` tab, set a user who has "Read", "Create", "Change" rights for the Scripts folder in `Log On As`. This user should also have: "Read" access to all tenants which are supposed to be used; "Role" with sufficient permissions (as detailed in Genesys Rules System Deployment Guide), "Read" access to Business Structure folder and associated nodes that are supposed to be used; "Read" access to Scripts folder and Scripts objects (which are representations of the rule templates).

End

Next Steps

Back to Task Table

Replace the GRAT JAR files

Purpose: Replace the cepspi-<version>.jar and droolsspi-<version>.jar files in the Genesys Rules Authoring Tool installation with similar JAR files from the Genesys Web Engagement installation.

Start

1. Prepare genesys-rules-authoring.war.
   - Extract genesys-rules-authoring.war to the C:\temp\genesys-rules-authoring folder.
2. Delete cepspi-8.1.100.xx.jar and droolsspi-8.1.100.xx.jar in the C:\temp\genesys-rules-authoring\WEB-INF\lib folder.
3. Copy C:\<GWE installation directory>\tools\maven\repository\com\genesyslab\wme\cepspi\<version>\cepspi-<version>.jar to C:\temp\genesys-rules-authoring\WEB-INF\lib.
4. Copy C:\<GWE installation directory>\tools\maven\repository\com\genesyslab\wme\droolsspi\<version>\droolsspi-<version>.jar to C:\temp\genesys-rules-authoring\WEB-INF\lib.
   Note: The JAR files you copy over in steps 3 and 4 should be version 8.1.100.27 or later.
5. Select the files in the C:\temp\genesys-rules-authoring folder and pack them as a zip archive called genesys-rules-authoring.war.
6. Deploy the genesys-rules-authoring.war file to the application server where GRAT will run.

End

Next Steps

Configuring Genesys Rules Authoring Tool

Configure GRDT

	Purpose: To configure Genesys Rules Development Tool.

Configure the Genesys Rules Development Tool

Purpose: To configure the Genesys Rules Development Tool for Genesys Web Engagement.

Prerequisites

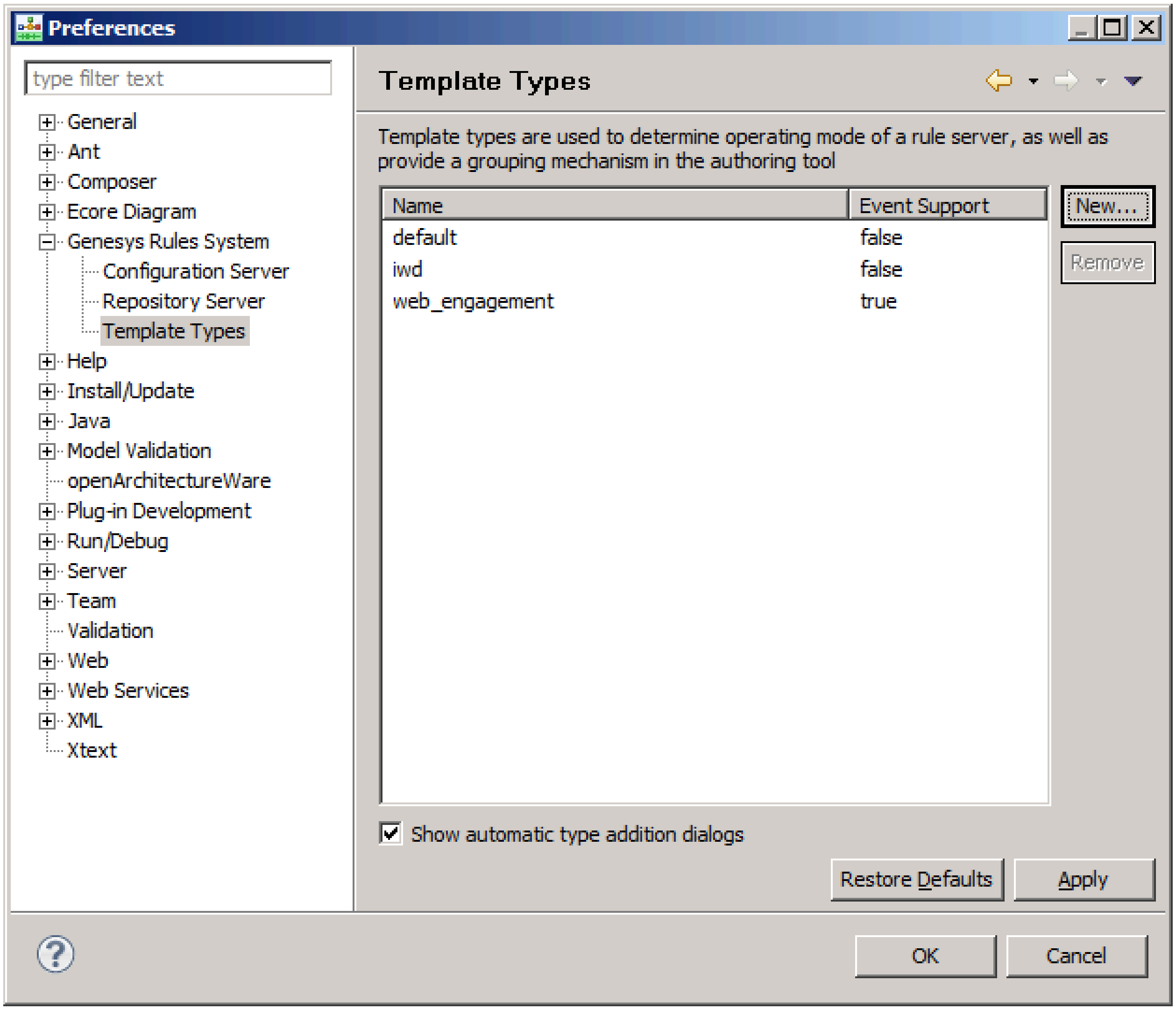
- Genesys Rules Development Tool (deployed as Composer plug-in or independently in Eclipse), version 8.1.200.17 and higher, is installed.
- You enabled the Galileo update site in GRDT, as described in Installing the GRDT Component in the the Genesys Rules System Deployment Guide.

Start

1. Open Genesys Rules Development Tool by starting Composer or Eclipse.
2. Navigate to `Window > Preferences`. The Preferences dialog box opens.
3. Navigate to `Genesys Rules System > Configuration Server`. Edit the Configuration Server settings.
    - Enter the Configuration Server hostname. For instance, localhost;
    - Enter the Configuration Server port: 2020;
    - Enter the application name for the Rules Authoring Client application; for instance, RulesAuthoringClient.
    - In the Authentication section, enter the name and password for a user who can connect to the Configuration Server.
4. Click on the `Apply` button.
5. Navigate to `Genesys Rules System > Repository Server`. Edit the settings.
    - Enter the Repository Server hostname; for instance, localhost;
    - Enter the Repository Server port; for instance, 8020;
    - Enter the Servlet path; for instance, genesys-rules-authoring;
    - In the Authentication section, enter a name and password for a user who:
        - has Read and Execute permissions for the Genesys Rules Authoring client application (set up in the Configuration Server); this user must have explicit Read and Execute permissions or must belong to an access group with those permissions;
        - Belongs to a Role with the following privileges: Template - Create, Template - Modify, Template -

Delete;

6. Click on the `Apply` button.
7. Navigate to `Genesys Rules System > Template types`. Check the following template types:
   - `default` without enabling the event supporting option.
   - `web_engagement` with enabling of the event supporting option.



Template Types

Template types are used to determine operating mode of a rule server, as well as provide a grouping mechanism in the authoring tool

Name	Event Support
default	false
iwd	false
web_engagement	true

☑ Show automatic type addition dialogs

Template types in Composer

End

Next Steps

Back to Task Table

Configure Roles for GRAT

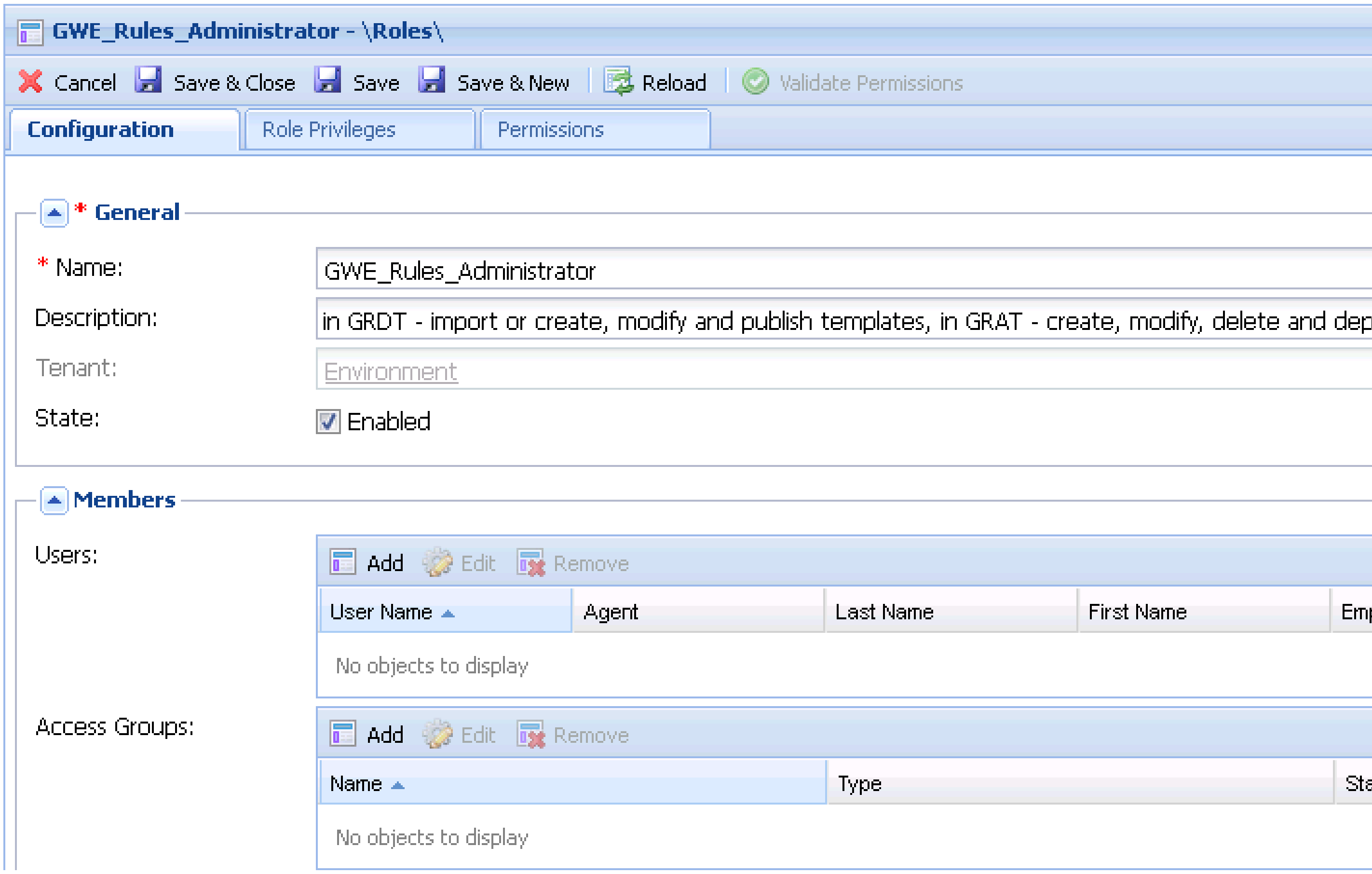
	Purpose: Configure Roles for Genesys Rules Authoring Tool.

(Optional) Configure Roles Settings for Rules Management

Purpose: To import permissions which enable a user to create rules for Genesys Web Engagement. Once roles are imported, you can assign them to the user involved in publishing the rules template and creating rules for Web Engagement.
Note: This step is optional if you already created a user with Roles Privileges that enable creating Rules packages in Genesys Rules Authoring. See Role-Based Access Control in the Genesys Rules System Deployment Guide.

Start

1. Start Genesys Administrator and navigate to `PROVISIONING > Accounts > Roles`. Click `New...`
2. Enter a name for this role. For instance, `GWE Rules Administrator`.
3. Expand `Members`. You can now specify who will have this role privilege. Click on the `Add` buttons to add an access group or a user. For instance, `demo`.



GWE_Rules_Administrator - \Roles\

General

* Name: GWE_Rules_Administrator

Description: in GRDT - import or create, modify and publish templates, in GRAT - create, modify, delete and dep

Tenant: Environment

State: ☑ Enabled

Members

Users:

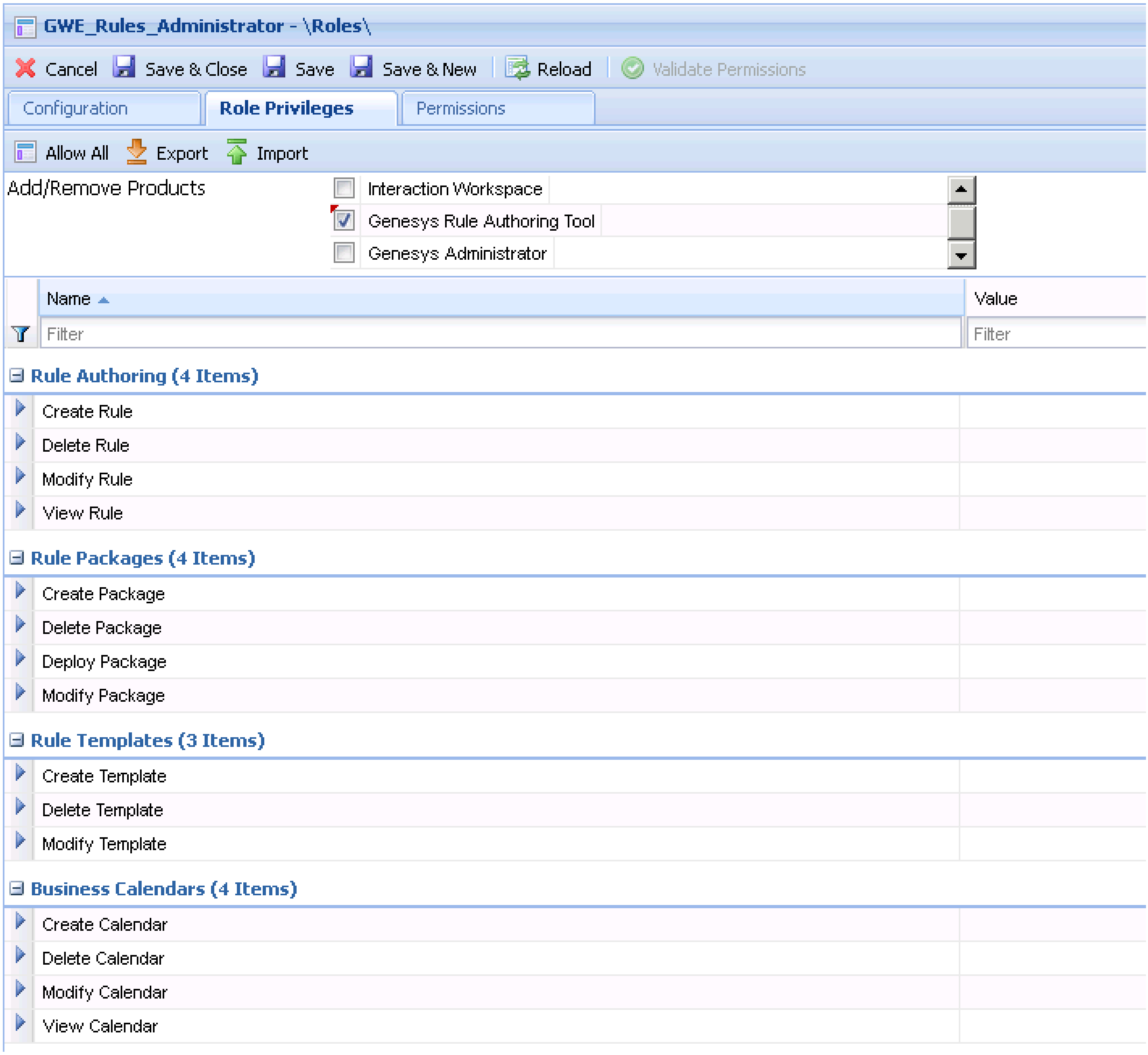
User Name	Agent	Last Name	First Name	Em
No objects to display				

Access Groups:

Name	Type	Sta
No objects to display		

New Role

4. Select the `Role Privileges` tab and select `Genesys Rules Authoring Tool` in the Add/Remove Products section. The list of privileges appears.



Add/Remove Products

- □ Interaction Workspace
- ☑ Genesys Rule Authoring Tool
- □ Genesys Administrator

Name	Value
Filter	Filter
Rule Authoring (4 Items)	
Create Rule	
Delete Rule	
Modify Rule	
View Rule	
Rule Packages (4 Items)	
Create Package	
Delete Package	
Deploy Package	
Modify Package	
Rule Templates (3 Items)	
Create Template	
Delete Template	
Modify Template	
Business Calendars (4 Items)	
Create Calendar	
Delete Calendar	
Modify Calendar	
View Calendar	

Imported Role Privileges

5. Click `Save & Close`.

End

- Members of this role can do the following:
  - Create and publish CEP rule templates
  - Create, modify, and deploy rule packages

- Create, modify, and delete rules

Next Steps

➡ Back to Task Table

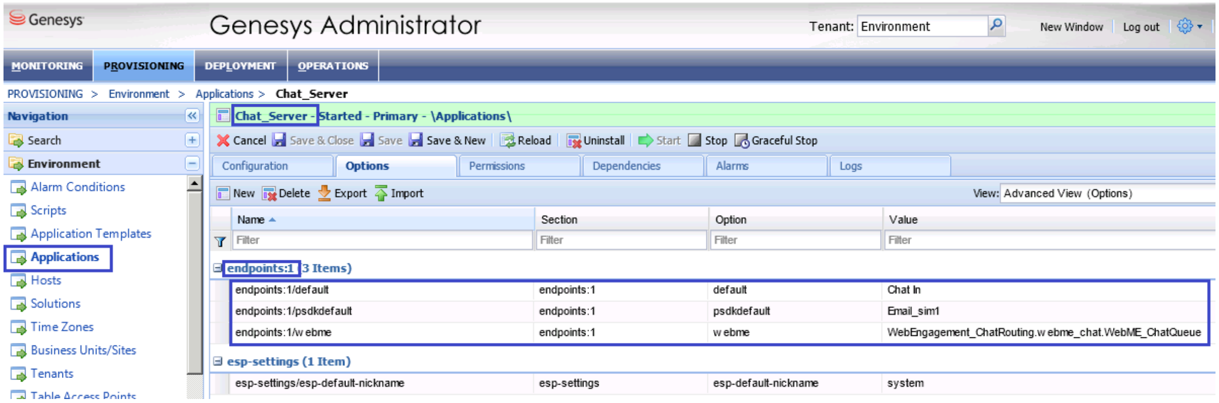
Configure a Chat Channel

	Purpose: To change the way the interaction is distributed to the contact center resource.

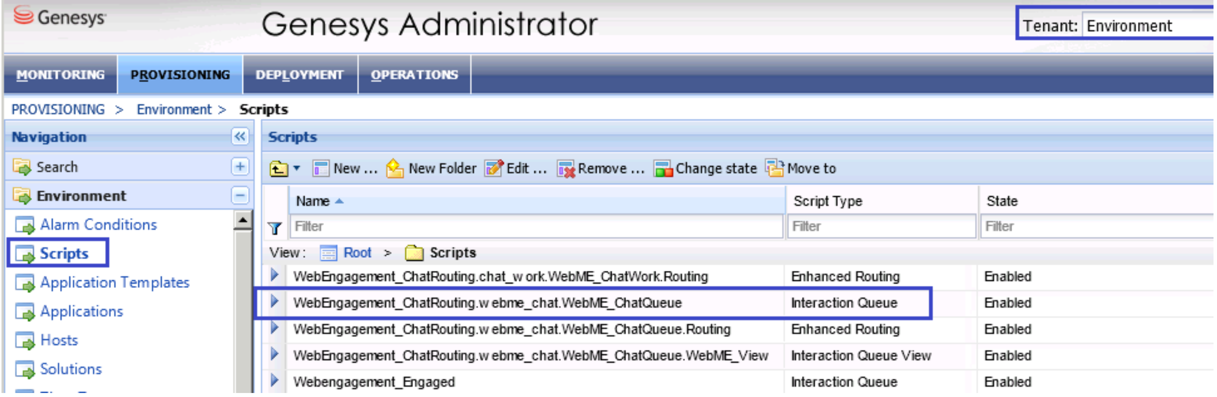
Procedure: Specifying a Chat Strategy for Routing Chat Interactions

Purpose: To configure a chat strategy for routing chat interactions.

1. Start Genesys Administrator and navigate to `PROVISIONING > Environment > Applications`.
2. Open the application for Chat Server.
3. Configure endpoints on the Chat Server application.
   - Select the `Options` tab and find the endpoints section for your tenant: `endpoints:<tenant ID>`. For example, if Chat Server works with the tenant Environment, there should be a section called `endpoints:1`.
   - Set the endpoint value for the `endpoints:<tenant ID>/webme` option to the name of the Interaction Queue configuration object where the chat interaction should be placed.
     Note: Each Interaction Queue can be related to one routing strategy, either Orchestration Server (ORS) or Universal Routing Server (URS).

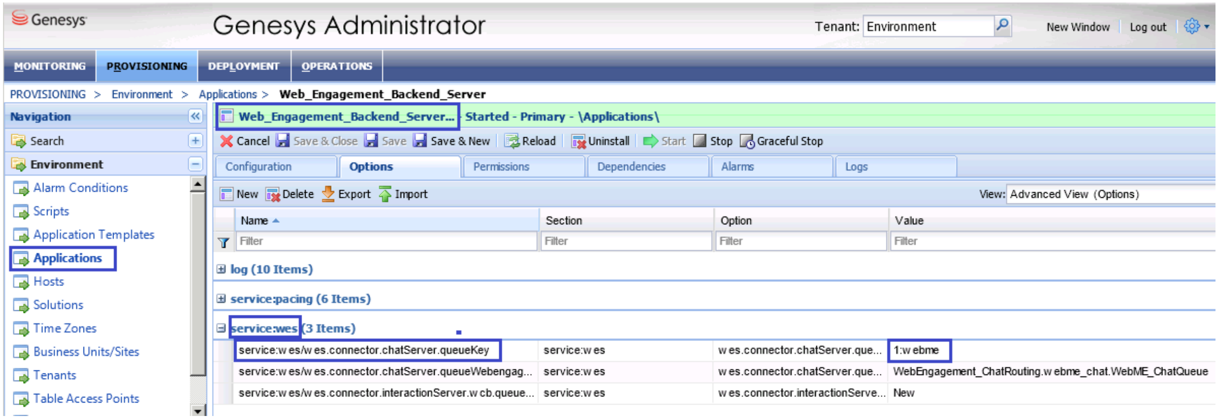


Name	Section	Option	Value
endpoints:1 (3 Items)			
endpoints:1/default	endpoints:1	default	Chat In
endpoints:1/psdkdefault	endpoints:1	psdkdefault	Email_sim1
endpoints:1/webme	endpoints:1	webme	WebEngagement_ChatRouting.webme_chat.WebME_ChatQueue
esp-settings (1 Item)			
esp-settings/esp-default-nickname	esp-settings	esp-default-nickname	system



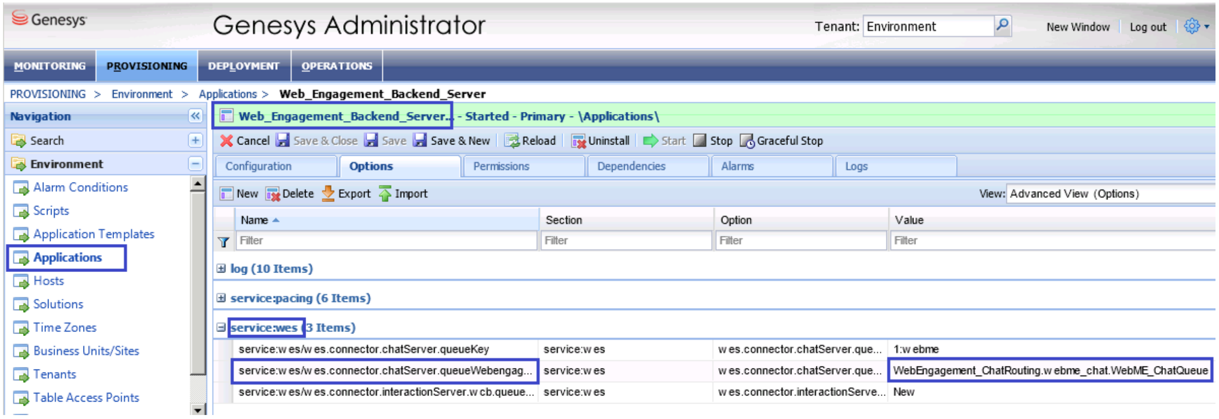
Name	Script Type	State
WebEngagement_ChatRouting.chat_work.WebME_ChatWork.Routing	Enhanced Routing	Enabled
WebEngagement_ChatRouting.webme_chat.WebME_ChatQueue	Interaction Queue	Enabled
WebEngagement_ChatRouting.webme_chat.WebME_ChatQueue.Routing	Enhanced Routing	Enabled
WebEngagement_ChatRouting.webme_chat.WebME_ChatQueue.WebME_View	Interaction Queue View	Enabled
Webengagement_Engaged	Interaction Queue	Enabled

4. Click `Save & Close`. If the Confirmation dialog opens, click `Yes`.
5. Configure related options for the Web Engagement Backend Server application.
   - Open the application for Web Engagement Backend Server.
   - Select the `Configuration` tab and make sure the Chat Server application is listed in the Connections.
   - Select the `Options` tab.
   - In the `service:wes` section, set the value of the wes.connector.chatServer.queueKey option to the name of the endpoint specified in the Chat Server application options. The format is `<tenant ID>:<Endpoint name>`. The figure below shows the endpoint `webme` in the Environment tenant, which works with the Interaction Queue `WebEngagement_ChatRouting.webme_chat.WebME_ChatQueue`.



Name	Section	Option	Value
log (10 Items)			
service:pacing (6 Items)			
service:wes (3 Items)			
service:wes/wes.connector.chatServer.queueKey	service:wes	wes.connector.chatServer.que...	1:webme
service:wes/wes.connector.chatServer.queueWebengag...	service:wes	wes.connector.chatServer.que...	WebEngagement_ChatRouting.webme_chat.WebME_ChatQueue
service:wes/wes.connector.interactionServer.wcb.queue...	service:wes	wes.connector.interactionServe...	New

6. Specify the Interaction Queue used in the pacing algorithm in order to detect the count of interactions in queue.
   - In the `service:wes` section, set the value of the wes.connector.chatServer.queueWebengagement option to the Interaction Queue.



Name	Section	Option	Value
log (10 Items)			
service:pacing (6 Items)			
service:wes (3 Items)			
service:wes/wes.connector.chatServer.queueKey	service:wes	wes.connector.chatServer.que...	1:webme
service:wes/wes.connector.chatServer.queueWebengag...	service:wes	wes.connector.chatServer.que...	WebEngagement_ChatRouting.webme_chat.WebME_ChatQueue
service:wes/wes.connector.interactionServer.wcb.queue...	service:wes	wes.connector.interactionServe...	New

7. Click `Save & Close`. If the Confirmation dialog opens, click Yes.

End

Procedure: Configuring Chat as the Default Channel of Engagement

Purpose: To configure chat as the default channel of engagement using the wmsg.connector.defaultEngagementChannel option.

> ## Important
> The `wmsg.connector.defaultEngagementChannel` option is intended for development purposes only and should not be used in a production environment.

Specifying chat as the default channel will turn off the pacing algorithm. As a result, the engagement attempt will always be activated on the chat channel and the count of ready agents will be ignored. If the `wmsg.connector.defaultEngagementChannel` option is not specified or specified with empty value, the pacing algorithms will be used.

Start

1. Start Genesys Administrator and navigate to `PROVISIONING > Environment > Applications`.
2. Open the application for the Web Engagement Backend Server.
3. Set the chat channel as the default channel of engagement.
   - In the `service:wmsg` section, set the value of the `wmsg.connector.defaultEngagementChannel` option to `proactiveChat`.

End

Configure Pacing Algorithm

	Purpose: Configuring the Pacing Algorithm to improve workload distribution.

Configuring your Optimization Model

Purpose: To set up your Optimization Model.

Start

1. Start Genesys Administrator and navigate to `PROVISIONING > Environment > Applications`.
2. Open the application for the Web Engagement Backend Server.
3. Set up the `wmsg.connector.pacing.algorithm` option. The supported value is:
   - `PROGRESSIVE`—Recommended for small agent groups (1-30 agents).
4. Set the optimization options as follows:

Value for optimizationTarget	Value for optimizationGoal
ABANDONEMENT_RATE	from 3 to 5
BUSY_FACTOR	from 70 to 85

End
Now that the pacing algorithm is based on the agent groups, you must configure the groups.

Next Steps
Configuring Agent Groups

Configuring Agent Groups

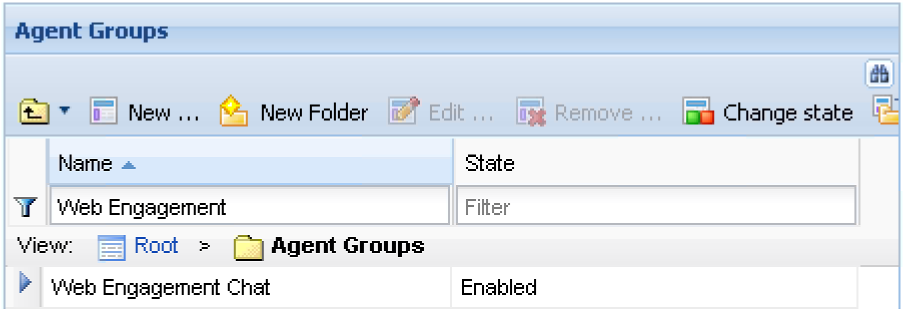
Purpose: To create agent groups required by the pacing algorithm. **Note:** Two agent groups are created by default when you install Genesys Web Engagement:

- Web Engagement Chat
- Web Engagement Voice

These default groups can be changed by setting new group names for the `wmsg.connector.pacing.chatGroup` and `wmsg.connector.pacing.voiceGroup` options of the pacing section of your Web Engagement Backend Application.

Start

1. Open Genesys Administrator and navigate to `PROVISIONING > Accounts > Agent Groups`. Make sure that the Web Engagement Chat and Web Engagement Voice groups are created.



Name	State
Web Engagement	Filter
Web Engagement Chat	Enabled

Use the filter to display Web Engagement Agent groups.

2. Navigate to `PROVISIONING > Accounts > Users.`
3. Select an agent that should manage Web Engagement interactions and click on the `Edit...` button.
4. Select the `Agent Group` tab and click on the `Add` button. The Browse dialog opens. Select one of the Web Engagement group and click `OK`.
   GWE-AgentAddedToWEChat.png|frame|center|Agent named Spencer now belongs to the Web Engagement Chat group.
5. Repeat step 4 as many time as needed. **Note:** An agent should belong to exactly one of these groups. If you add one agent to both groups, the pacing algorithm will produce incorrect results.

End

Next Steps

Back to Task Table

Importing CCPulse Templates

Procedure: Importing Genesys Web Engagement Sample Reporting Templates in CCPulse+

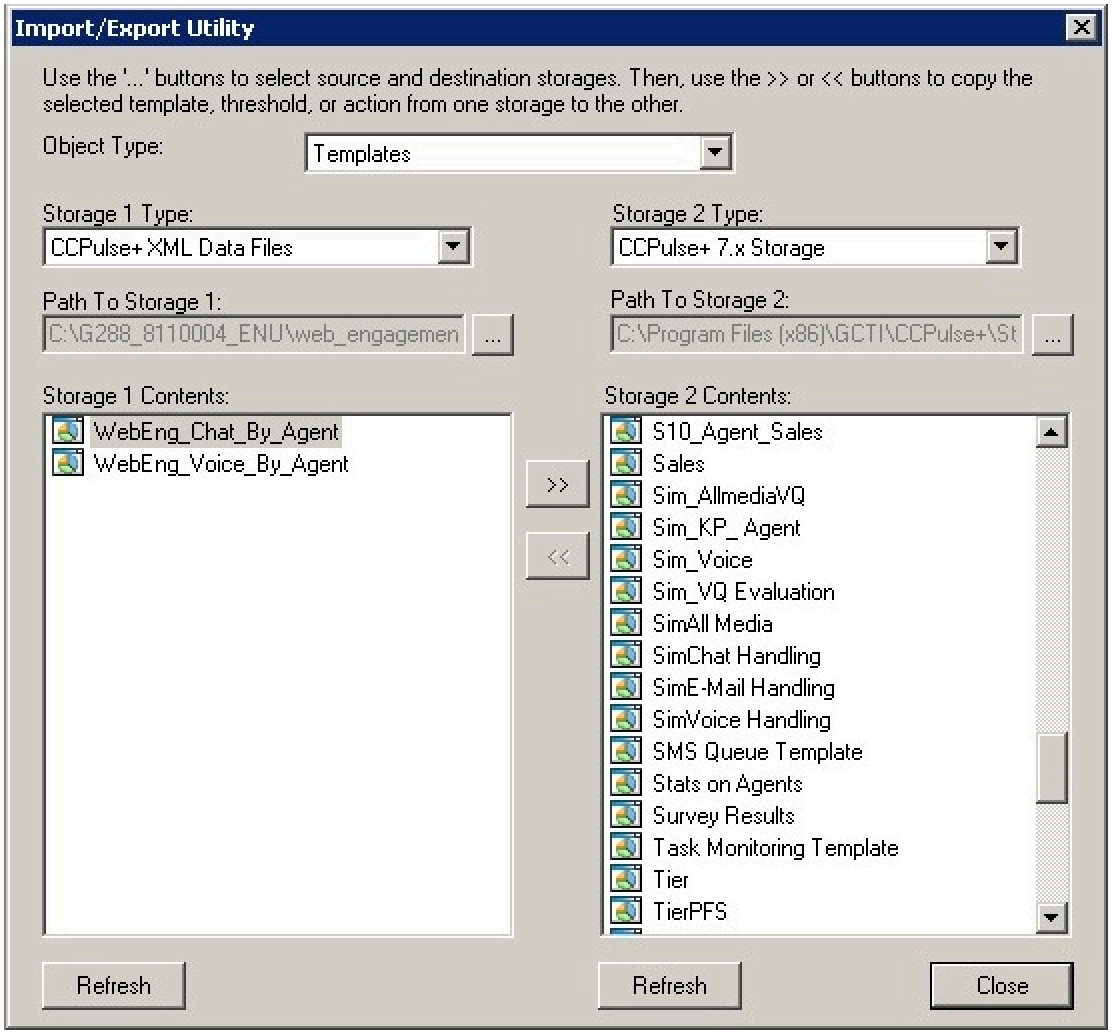
Purpose: To import the Genesys Web Engagement Sample Reporting Templates in CCPulse+.
Note: For information about using CCPulse+, click Help in the CCPulse+ user interface or see the *CCPulse+ Administrator's Guide*. The Web Engagement templates for CCPulse+ have an extension of .xtpl, which is an XML format. The documentation for CCPulse+ does not distinguish files by extension and refers to .xtpl files as XML files.

Prerequisites

- You have installed CCPulse+ version 8.0.000.36 (or later).
- You are a CCPulse+ administrator.

Start

1. Open CCPulse+ and navigate to Tools > Import/Export. The Import/Export Utility opens.
2. Select Templates for the Object Type.
3. Choose the import source.
   - For Storage 1 Type, select CCPulse+ XML Data Files.
   - For Path To Storage 1, click ... and select the path to the reporting XML template(s) you wish to import. The Genesys Web Engagement Sample Reporting Templates are located in the web_engagement_reporting_template directory on your installation CD.
   - Click Open. The reporting templates are displayed in the Storage 1 Contents box below. For example, if you select the Web Engagement Media Based folder for Path To Storage 1, the reporting templates WebEng_Chat_By_Agent.xtpl and WebEng_Voice_By_Agent.xtpl are displayed Storage 1 Contents.



Import/Export Utility

Use the '...' buttons to select source and destination storages. Then, use the >> or << buttons to copy the selected template, threshold, or action from one storage to the other.

Object Type: Templates

Storage 1 Type: CCPulse+ XML Data Files

Storage 2 Type: CCPulse+ 7.x Storage

Path To Storage 1: C:\G288_8110004_ENU\web_engagemen

Path To Storage 2: C:\Program Files (x86)\GCTI\CCPulse+\St

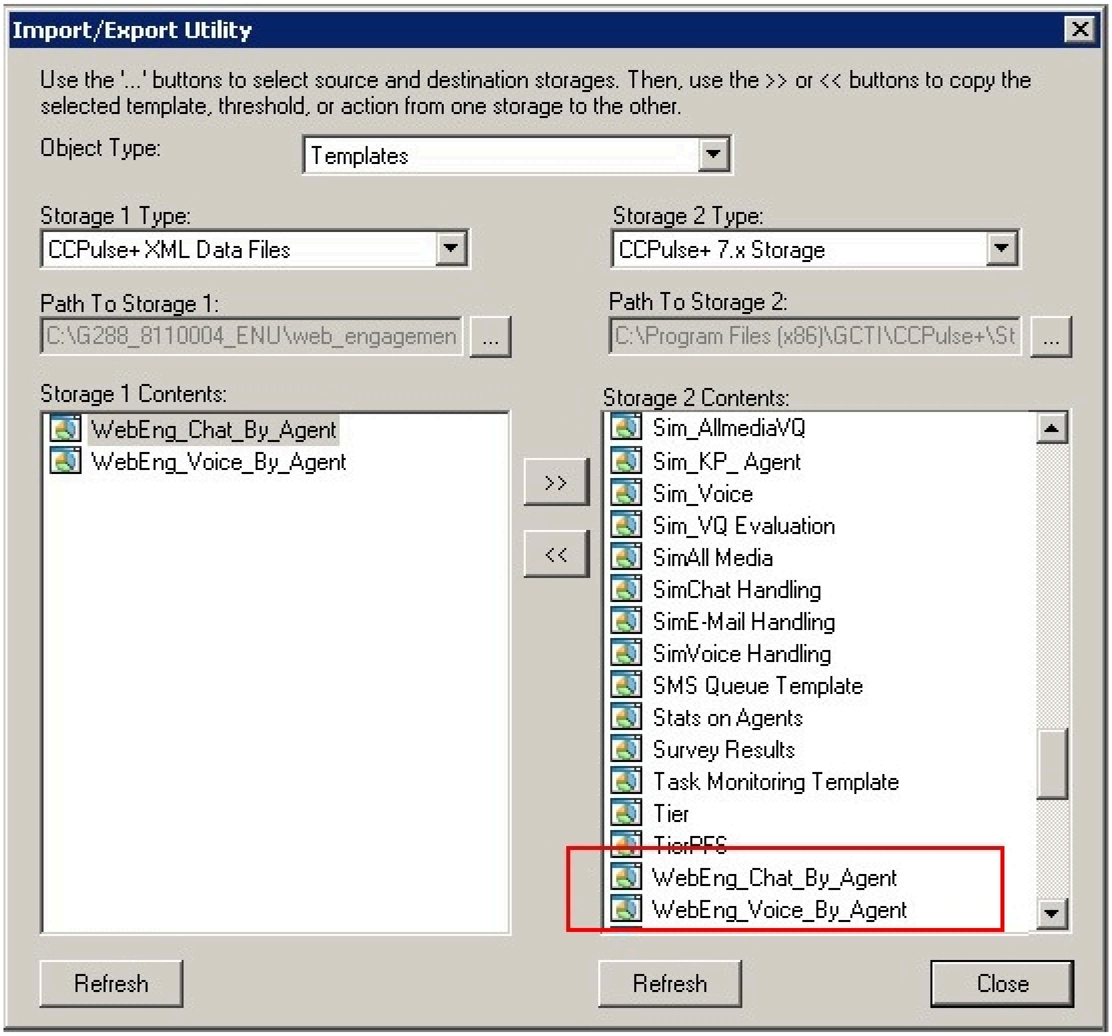
Storage 1 Contents:

- WebEng_Chat_By_Agent
- WebEng_Voice_By_Agent

Storage 2 Contents:

- S10_Agent_Sales
- Sales
- Sim_AllmediaVQ
- Sim_KP_ Agent
- Sim_Voice
- Sim_VQ Evaluation
- SimAll Media
- SimChat Handling
- SimE-Mail Handling
- SimVoice Handling
- SMS Queue Template
- Stats on Agents
- Survey Results
- Task Monitoring Template
- Tier
- TierPFS

4. Choose the import destination.
   - Select `CCPulse+ 7.x Storage` for the `Storage 2 Type`.
   - Select a template in `Storage 1 Contents` and click the >> button. The template is copied to the `Storage 2 Contents` box. Repeat for each template you wish to copy from `Storage 1 Contents`.



Use the '...' buttons to select source and destination storages. Then, use the >> or << buttons to copy the selected template, threshold, or action from one storage to the other.

Object Type: Templates

Storage 1 Type: CCPulse+ XML Data Files

Storage 2 Type: CCPulse+ 7.x Storage

Path To Storage 1: C:\G288_8110004_ENU\web_engagemen

Path To Storage 2: C:\Program Files (x86)\GCTI\CCPulse+\St

Storage 1 Contents:

WebEng_Chat_By_Agent
WebEng_Voice_By_Agent

Storage 2 Contents:

Sim_AllmediaVQ
Sim_KP_ Agent
Sim_Voice
Sim_VQ Evaluation
SimAll Media
SimChat Handling
SimE-Mail Handling
SimVoice Handling
SMS Queue Template
Stats on Agents
Survey Results
Task Monitoring Template
Tier
TierPFS
WebEng_Chat_By_Agent
WebEng_Voice_By_Agent

5. Repeat steps 3 and 4 for each template you would like to import into CCPulse+.
6. Click `Close`. The Import/Export Utility closes.

End

Next Steps

Back to Task Table

Loading Certificate for SSL

The procedures on this page provide examples of ways to load SSL certificates and configure Jetty. These examples may vary depending on your environment.

Note: You must use the Java Development Kit version 1.6.0_29 or higher to support the JSSE keystore.

Load an SSL Certificate and Private Key into a JSSE keystore

Note: In a development environment, you can use self-signed certificates, but in a production environment you should use a certificate issued by a third-party Certificate Authority, such as VeriSign.

Prerequisites

- An SSL certificate, either generated by you or issued by a third-party Certificate Authority. For more information on generating a certificate, see http://docs.codehaus.org/display/JETTY/How+to+configure+SSL.

Start

- Depending on your certificate format, do **one** of the following:
  - If your certificate is in PEM form, you can load it to a JSSE keystore with the keytool using the following command:
    ```
    keytool -keystore <keystore> -import -alias <alias> -file <certificate_file>
    -trustcacerts
    ```

    Where:

    `<keystore>` is the name of your JSSE keystore.

    `<alias>` is the unique alias for your certificate in the JSSE keystore.

    `<certificate_file>` is the name of your certificate file. For example, `jetty.crt`.
  - If your certificate and key are in separate files, you must combine them into a PKCS12 file before loading it to a keystore.
    1. Use the following command in openssl to combine the files:
       ```
       openssl pkcs12 -inkey <private_key> -in <certificate> -export -out
       <pkcs12_file>
       ```

       Where:

       `<private_key>` is the name of your private key file. For example, `jetty.key`.

       `<certificate>` is the name of your certificate file. For example, `jetty.crt`.

       `<pkcs12_file>` is the name of the PKCS12 file that will be created. For example, `jetty.pkcs12`.

2. Load the PKCS12 file into a JSSE keystore using keytool with the following command:

    `keytool -importkeystore -srckeystore <pkcs12_file> -srcstoretype <store_type> -destkeystore <keystore>`

    Where:

    `<pkcs12_file>` is the name of your PKCS12 file. For example, `jetty.pkcs12`.

    `<store_type>` is the file type you are importing into the keystore. In this case, the type is PKCS12.

    `<keystore>` is the name of your JSSE keystore.

    Note: You will need to set two passwords during this process: keystore and truststore. Make note of these passwords because you will need to add them to your Jetty SSL configuration file.

End

Next Steps

- Configure Jetty

Configure Jetty

Prerequisites

- You have completed Load an SSL Certificate and Private Key into a JSSE keystore

Start

1. Open the Jetty SSL configuration file in a text editor: `<jetty_installation>/etc/jetty-ssl.xml`.
2. Add a new "addConnector" section to the file, with the following information:

```
<Call name="addConnector">
    <Arg>
      <New class="org.mortbay.jetty.security.SslSocketConnector">
        <Set name="Port">8443</Set>
        <Set name="maxIdleTime">30000</Set>
        <Set name="keystore">
                   <SystemProperty name="jetty.home" default="." />
                   /etc/keystore
                </Set>
        <Set name="password">OBF:<obfuscated_truststore_password></Set>
        <Set name="keyPassword">OBF:<obfuscated_keystore_password></Set>
        <Set name="truststore">
                   <SystemProperty name="jetty.home" default="." />
                   /etc/keystore
                </Set>
        <Set name="trustPassword">OBF:<obfuscated_truststore_password></Set>
      </New>
        </Arg>
</Call>
```

Note: You can run Jetty's password utility to obfuscate your passwords. See http://docs.codehaus.org/display/JETTY/Securing+Passwords.

3. Save your changes.

End

Choosing a Directory for the Keystore

The keystore file in the example above is given relative to the Jetty home directory. For production, you should keep your keystore in a private directory with restricted access. 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 jcert.jar, jnet.jar and jsse.jar are on your classpath) and SSL can be used with a URL, such as https://localhost:8443/

Setting the Port for HTTPS

Remember that the default port for HTTPS is 443 not 80, so change 8443 to 443 if you want to be able to use URLs without explicit port numbers. For a production site, it normally makes sense to have an HttpListener on port 80 and a SunJsseListener on port 443. Because these are privileged ports, you might want to use a redirection mechanism to map port 80 to 8080 and 443 to 8443, for example. The most common mistake at this point is to try to access port 8443 with HTTP rather than HTTPS.

Redirecting HTTP requests to HTTPS

To redirect HTTP to HTTPS, the web application should indicate it needs CONFIDENTIAL or INTEGRAL connections from users. You need to tell the plain HTTP connector that if users try to access that web application with plain HTTP, they should be redirected to the port of your SSL connector (the "confidential port"):

```
<Call name="addConnector">
  <Arg>
     <New class="org.mortbay.jetty.nio.SelectChannelConnector">
        <Set name="port">8080</Set>
        <Set name="maxIdleTime">30000</Set>
        <Set name="Acceptors">2</Set>
        <Set name="confidentialPort">443</Set>
     </New>
  </Arg>
</Call>
```

Next Steps

- Configure Java

Configure Java

Prerequisites

- You have completed Load an SSL Certificate and Private Key into a JSSE keystore

Start

1. Navigate to your installation directory for the Backend Server and open the launcher.xml file with a text editor.
2. Add the following parameters:

   ```
   <parameter name="javax.net.ssl.trustStore" displayName="javax.net.ssl.trustStore"
   mandatory="false">
      <description><![CDATA[]]></description>
      <valid-description />
      <effective-description />
      <format type="string" default="<cacerts_filepath>" />
      <validation />
   </parameter>
   <parameter name="javax.net.ssl.trustStorePassword"
   displayName="javax.net.ssl.trustStorePassword" mandatory="false">
      <description><![CDATA[]]></description>
      <valid-description />
      <effective-description />
      <format type="string" default="<truststore_password>" />
      <validation />
   </parameter>
   <parameter name="javax.net.ssl.keyStore" displayName="javax.net.ssl.keyStore"
   mandatory="false">
      <description><![CDATA[]]></description>
      <valid-description />
      <effective-description />
      <format type="string" default="<keystore_filepath>" />
      <validation />
   </parameter>
   <parameter name="javax.net.ssl.keyStorePassword"
   displayName="javax.net.ssl.keyStorePassword" mandatory="false">
      <description><![CDATA[]]></description>
      <valid-description />
      <effective-description />
      <format type="string" default="<keystore_password>" />
      <validation />
   </parameter>
   <parameter name="javax.net.ssl.keyAlias" displayName="javax.net.ssl.keyAlias"
   mandatory="false">
      <description><![CDATA[]]></description>
      <valid-description />
      <effective-description />
      <format type="string" default="<alias>" />
      <validation />
   </parameter>
   ```

3. Save your changes.
4. Repeat steps 1-3 for the Frontend Server.
5. Save your changes.

End

Configure the Frontend Server for UTF-8

Purpose: To configure your Frontend Server to support UTF-8 in Configuration Server. You must complete this procedure to enable Frontend Server to support multi-language categories.

Prerequisites

- Your version of Configuration Server supports UTF-8 (version 8.1.200.04+).

Start

1. Navigate to your installation directory for the Frontend Server and open the `launcher.xml` file with a text editor. For example: `C:\GCTI\Genesys Web Engagement\servers\frontend\launcher.xml`.
2. Find the parameter named `useUTF8CfgSrvConnect`.
3. In the `format` element, change the `default` value to `true`. For example:

   ```
   <parameter name="useUTF8CfgSrvConnect" displayName="useUTF8CfgSrvConnect"
    mandatory="true">
        ...
        <format type="boolean" default="true" />
        <validation />
    </parameter>
   ```

4. Save your changes.

End

Generic Cassandra Configuration

Configure the Java heap size

Purpose: To configure the size of the Java heap for Cassandra.

Start

1. Navigate to your installation directory for the Backend Server and open the `launcher.xml` file with a text editor.
2. The default values for the Java heap size are stored in the `jvm_option2` and `jvm_option3` parameters:

```
        <parameter name="jvm_option2" displayName="jvm_option1" mandatory="true"
hidden="true" readOnly="true">
            <format type="string" default="-Xms512m"/>
...
        </parameter>
        <parameter name="jvm_option3" displayName="jvm_option2" mandatory="true"
hidden="true" readOnly="true">
            <format type="string" default="-Xmx1024m"/>
            <validation></validation>
...
        </parameter>
...
```

Modify the `default` value in the `format` element of these parameters according to your hardware configuration. As a rule, the value should be a maximum of 8GB or half your total RAM, whichever is lower; however; Genesys recommends **3GB** or more for a production site. Cassandra's default configuration opens the JVM with a heap size that is based on the total amount of system memory:

System Memory	Heap Size
Less than 2GB	1/2 of system memory
2GB to 4GB	1GB
Greater than 4GB	**1/2 system memory, but not more than 4GB**

Genesys recommends you set the `default` value in the `format` element for both parameters to the same value. For example:

```
        <parameter name="jvm_option2" displayName="jvm_option1" mandatory="true"
hidden="true" readOnly="true">
            <format type="string" default="-Xms3072m"/>
...
        </parameter>
        <parameter name="jvm_option3" displayName="jvm_option2" mandatory="true"
hidden="true" readOnly="true">
            <format type="string" default="-Xmx3072m"/>
            <validation></validation>
...
        </parameter>
...
```

Consult the Cassandra documentation at http://www.datastax.com/docs/1.0/index for more information about Cassandra clusters and memory.

End

Next Steps

Configure Local Properties

Configure Local Properties

Purpose: To configure Cassandra for the Backend Server.

Prerequisites

- Read the documentation on Cassandra configuration parameters, available here: Apache Cassandra 1.0 Documentation

Start

1. To configure Cassandra, navigate to the Web Engagement installation directory and edit the `\apps\<application name>\environment\environment.local.properties` file.

   - The values specified in `environment.local.properties` are inserted into the `cassandra.yaml` and `wmbackend.properties` files during the application build step. All configuration parameters for Cassandra in `environment.local.properties` start with `wmdb.cassandra.cluster`. The parameters and their default values:

   ```
   # WMDB-Cassandra
   # Tokens in cassandra.yaml and wmbackend.properties files will be substituted by
   these parameter values.
   wmdb.cassandra.cluster.name=Cluster
   wmdb.cassandra.cluster.keyspaceName=WebMonitoring
   wmdb.cassandra.cluster.defaultStrategy=SimpleStrategy
   wmdb.cassandra.cluster.defaultReplicationFactor=1
   wmdb.cassandra.cluster.listenAddress=localhost
   wmdb.cassandra.cluster.rpcAddress=localhost
   wmdb.cassandra.cluster.rpcPort=19160
   wmdb.cassandra.cluster.sslStoragePort=17001
   wmdb.cassandra.cluster.storagePort=17000
   wmdb.cassandra.cluster.seedNodes=127.0.0.1
   wmdb.cassandra.cluster.dataDirectory=DATA_DIR_PLACEHOLDER
   wmdb.cassandra.cluster.commitLogDirectory=COMMITLOG_DIR_PLACEHOLDER
   wmdb.cassandra.cluster.savedCachesDirectory=SAVED_CACHES_DIR_PLACEHOLDER
   ```

   - Read Node and Cluster Configuration (cassandra.yaml) for information on all Cassandra configuration parameters.
     Note: Making changes in the `environment.local.properties` file will keep the client and server connection parameters synchronized.

2. Update the Cassandra configuration parameters with values appropriate for your deployment. You will need to modify the following parameters:

   - `wmdb.cassandra.cluster.defaultReplicationFactor` — The keyspace replication factor. When choosing a replication factor value, take into account that Cassandra uses consistency level

QUORUM for both writes and reads. The recommended formula for the replication factor is <number_of_nodes> / 2 + 1. Consider the following examples when determining your replication factor:

Number of nodes in the cluster	Consistency level	Replication factor	Result
4	QUORUM	1	1 node will contain unique information.
4	QUORUM	3	3 nodes will contain the duplicated information; the information is written to all nodes directly.
4	QUORUM	4	3 nodes will contain the duplicated information; the information is written to all nodes directly. One node will contain information replicas in the background.

- `wmdb.cassandra.cluster.defaultStrategy` — The default value is `SimpleStrategy`, but if you plan to deploy the cluster across multiple data centers, consider using the `NetworkTopologyStrategy`.
- `wmdb.cassandra.cluster.seedNodes` — Choose your seed nodes, keeping in mind that if you plan to deploy the Cassandra cluster across multiple data centers, you should have at least one seed node from each data center. Set the value of `wmdb.cassandra.cluster.seedNodes` to a comma-separated list of the IPs for your seed nodes. For example, `135.225.54.236, 135.225.54.245`.
- `listen_address` and `rpc_address` — These parameters are likely to be different for each Backend Server / Cassandra node in the cluster and must be set directly in the `\servers\backend\etc\cassandra.yaml` file for each Backend Server. See Configure Cassandra Cluster for details on setting these parameters.
-

3. Save your changes.

End

Next Steps

Enable data compression on the Cassandra cluster

Enable data compression on the Cassandra cluster

Purpose: To enable data compression on the Cassandra cluster. After you apply the script in this procedure, Cassandra will use the compression chunk length of 64kb.

> **Important**
>
> Although Cassandra is embedded in Genesys Web Engagement, you must follow the steps in this procedure to download and install a separate instance of Cassandra in order to get the tools needed to enable data compression.

Start

1. Download Cassandra.
   a. Go to http://cassandra.apache.org/download/, find the correct version of Cassandra, and download the archive. Genesys Web Engagement 8.1.1 uses Cassandra 1.0.6, so you should download `apache-cassandra-1.0.6-bin.tar.gz`.
   b. Extract the files to a directory; for example, `C:\cassandra`. This directory will be referred to as `%CASSANDRA_HOME%` in the steps below.
2. Save the following script in a text file; for example, `compression_script.txt`.

```
// enable compression
use WebMonitoring;
update column family visits with
compression_options={sstable_compression:DeflateCompressor,chunk_length_kb:64};
update column family userAgents with
compression_options={sstable_compression:DeflateCompressor,chunk_length_kb:64};
update column family sessions with
compression_options={sstable_compression:DeflateCompressor,chunk_length_kb:64};
update column family pages with
compression_options={sstable_compression:DeflateCompressor,chunk_length_kb:64};
update column family ixnProfiles with
compression_options={sstable_compression:DeflateCompressor,chunk_length_kb:64};
update column family indexesByString with
compression_options={sstable_compression:DeflateCompressor,chunk_length_kb:64};
update column family identities with
compression_options={sstable_compression:DeflateCompressor,chunk_length_kb:64};
update column family events with
compression_options={sstable_compression:DeflateCompressor,chunk_length_kb:64};
update column family engagementAttempts with
compression_options={sstable_compression:DeflateCompressor,chunk_length_kb:64};
```

3. Apply the script to the `seed_provider_node` by running the following command inside the `%CASSANDRA_HOME%/bin` directory:

```
>cassandra-cli.bat -h <seed_provider_node_ip_address> -p 19160 -f <path_to_script_file>
```

   For example:

```
>cassandra-cli.bat -h 135.225.54.236 -p 19160 -f C:\compression_script.txt
```

4. The existing SSTables are compressed when the normal Cassandra compaction process occurs. You can force Cassandra to rewrite and compress the existing SSTables with the following:

```
nodetool upgradesstables
```

End

You can disable compression with the following script:

```
// disable compression
use WebMonitoring;
update column family visits with compression_options=null;
update column family userAgents with compression_options=null;
update column family sessions with compression_options=null;
update column family pages with compression_options=null;
update column family ixnProfiles with compression_options=null;
update column family indexesByString with compression_options=null;
update column family identities with compression_options=null;
update column family events with compression_options=null;
update column family engagementAttempts with compression_options=null;
```

Next Steps

➩ Back to Task Table

Configure Cassandra Cluster

Configure Cassandra for the Backend Server "seed" node

Start

1. For each Backend Server, open the the `backend/etc/cassandra.yaml` file with a text editor;
2. Edit the specified strings:

```
listen_address: <backend-node_ip_address>
rpc_address: 0.0.0.0
initial_token: 0
seeds_provider:
- seeds: "<backend-node_ip_address>"
```

.

End

Next Steps

- Configure Cassandra for the Backend Server "non-seed" nodes

Configure Cassandra for the Backend Server "non-seed" nodes

Start

1. For each Backend Server, open the `backend/etc/cassandra.yaml` file with a text editor.
2. Edit the specified strings:

   ```
   listen_address: <node_ip_address>
    rpc_address: 0.0.0.0
    initial_token: <initial token for this node>
    seeds_provider:
                    - seeds: "<seed-backend-ipaddress>"
   ```

   Note:

   - If you want to add more than two servers, you must recalculate all initial tokens. See http://www.datastax.com/docs/0.8/install/cluster_init#token-gen-cassandra.
   - When adding a new node to the Cassandra cluster, you must reconfigure all of the nodes in the cluster. Genesys recommends that you calculate the capacity of the cluster. The Cassandra cluster works with N / 2 +1 operating units and more, where N is the number of nodes of the Cassandra cluster.

- Confirm that the following parameters point to the correct paths:

- data_file_directories — For example, `C:\WebME Cluster\Node1\backend\storage\data`
- commitlog_directory — For example, `C:\WebME Cluster\Node1\backend\storage\commitlog`
- saved_caches_directory — For example, `C:\WebME Cluster\Node1\backend\storage\saved_caches`

End

The following shows a sample configuration with two servers:

- Backend Server Seed Node

```
listen_address: 135.225.51.148
rpc_address: 0.0.0.0
initial_token: 0
seeds_provider:
                - seeds: "135.225.51.148"
```

- Backend Server Non-seed Node

```
listen_address: 192.168.3.103
rpc_address: 0.0.0.0
initial_token: 85070591730234615865843651857942052864
seeds_provider:
                - seeds: "135.225.51.148"
```

Next Steps

Configure Rules Deployment for the Cluster

Deployment

You should install Genesys Web Engagement in a lab environment, and then create and test an application before deploying to production. See Genesys Web Engagement Developer's Guide for details about creating your application. If you have successfully created and tested an application, see Deploying to Production Environment.

Deploying to Production Environment

> ### Important
>
> Genesys recommends that you deploy Genesys Web Engagement in a lab environment to create and test a Web Engagement application, before you switch to production. See the Genesys Web Engagement Developer's Guide for further details about creating your application.
>
> If your tests provide successful results, you can deploy to a production environment where the Frontend and Backend servers are available on multiple hosts (according to your network configuration and security needs). See Load Balancing for details.

Prerequisites
The following steps must be completed before beginning any of the procedures on this page:

1. You installed the Web Engagement Frontend and Backend servers on the same host in a test environment.
2. You have successfully developed and tested an application for serving your site, **including definitions of categories and rules deployment**.
3. In this procedure, it is assumed that your test and production environments share the same Configuration Management Environment (CME). If this is not the case, you should clone the Frontend Server and Backend Server application objects into your production CME before you begin.
4. You have stopped your Frontend Server and Backend Server.

Update the Backend Server application

Start

1. Open Genesys Administrator and navigate to `PROVISIONING > Environment > Applications`. Select the application defined for the Web Engagement Backend Server and click `Edit...`
2. Change the host of the application to the host planned for your Backend load balancer. See Load Balancing for more information.
   Note: You may need to create this host object. For details about creating host objects in Genesys Administrator, see the *Configuring Hosts* section in the Management Framework 8.1 Deployment Guide.
3. Change *all* ports for the application to the port for the Backend load balancer.
4. Run the provisioning tool using the `-overwrite` parameter. This will prepare all the related objects in your Configuration Management Environment (scripts, transactions, and so on) to work with load balancing. See Provisioning for more information about using the tool.

End

Next Steps
Configure the Frontend Server nodes

Configure the Frontend Server nodes

> **Important**
> Complete the steps below for each planned node in your Frontend Server cluster.

Start

1. In Genesys Administrator, navigate to `PROVISIONING > Environment > Applications`. Select your Frontend Server application and click `Edit...`
2. Clone the Frontend Server application by clicking `Save & New`. Update the following fields:
   - Enter an application name.
   - Remove the connection to the Backend Server.
   - Specify the host and ports (including secure port, if needed) where the Frontend Server node will run.
   - Select the `Options` tab and configure the following:
     - In the `log` section, update the `all` option and provide a distinct file name for this application. For example, `c:\logs\WebEngagement\Web_Engagement_Frontend_Node_1`. This will allow you to distinguish between the logs produced by the different nodes in the cluster.
     - In the `settings` section, set the value of the `loadbalancer` option to `<schema>://<BackendLoadBalancerHost>:<BackendLoadBalancerPort>/backend`
       Where:
       - <schema> — http or https
       - <BackendLoadBalancerHost> — The FQDN or IP of the host for the Backend load balancer. This should be the value set in step 2 of the Update the Backend Server application procedure.
       - <BackendLoadBalancerPort> — The port of the Backend load balancer.
3. Copy the `<GWE_Installation_Home>\servers\frontend\` directory to the host planned for this node.
4. In the `<GWE_Installation_Home>\servers\frontend\` directory, change the following values in the `launcher.xml`, `launcher_32.xml`, and `launcher_64.xml` files:
   - In the `format` tag of the `webs_host` parameter, replace the `default` attribute value with the host (FQDN or IP). For example, `<format type="string" default="123.34.56.78" />` or `<format type="string" default="my.site.com" />`.
   - In the `format` tag of the `http_port` parameter, replace the `default` attribute value with the port for this node. For example, `<format type="numeric" default="11081" />`.
   - In the `format` tag of the `https_port` parameter, replace the `default` attribute value with the secure port for this node, if applicable. For example, `<format type="numeric" default="11483" />`.

- In the format tag of the app parameter, replace the default attribute value with the name of the application. For example, <format type="string" default="Web_Engagement_Frontend_Node_1" />.

End

Next Steps

- Configure the Backend Server nodes

Configure the Backend Server nodes

> **Important**
> Complete the steps below for each planned node in your Backend Server cluster.

Start

1. In Genesys Administrator, navigate to PROVISIONING > Environment > Applications. Select your Backend Server application and click Edit...
2. Clone the Backend Server application by clicking Save & New. Update the following fields:
   - Enter an application name.
   - Remove the connection to the Frontend Server.
   - Specify the host and ports (including secure port, if needed) where the Backend Server node will run.
   - Select the Options tab and configure the following:
     - In the log section, update the all option and provide a distinct file name for this application. For example, c:\logs\WebEngagement\Web_Engagement_Backend_Node_1. This will allow you to distinguish between the logs produced by the different nodes in the cluster.
     - In the settings section, set the value of the loadbalancer option to <schema>://<FrontendLoadBalancerHost>:<FrontendLoadBalancerPort>/frontend Where:
       - <schema> — http or https
       - <FrontendLoadBalancerHost> — The FQDN or IP of the host for the Frontend load balancer.
       - <FrontendLoadBalancerPort> — The port of the Frontend load balancer.
3. Copy the <GWE_Installation_Home>\servers\backend\ directory to the host for this node.
4. In the copied \servers\backend\ directory, change the following values in the launcher.xml, launcher_32.xml, and launcher_64.xml files:
   - In the format tag of the webs_host parameter, replace the default attribute value with the host (FQDN or IP) for this node. For example, <format type="string" default="123.34.56.78" /> or

`<format type="string" default="my.site.com" />`.

- In the `format` tag of the `http_port` parameter, replace the `default` attribute value with the port for this node. For example, `<format type="numeric" default="11081" />`.
- In the `format` tag of the `https_port` parameter, replace the `default` attribute value with the secure port for this node, if applicable. For example, `<format type="numeric" default="11483" />`.
- In the `format` tag of the app parameter, replace the `default` attribute value with the name of the application . For example, `<format type="string" default="Web_Engagement_Backend_Node_1" />`

End

Next Steps

- Implement Load Balancing

Start the server clusters

Prerequisites

- You have implemented load balancing.

Start

1. Start the Frontend load balancer and the Backend load balancer.
2. Start the Backend cluster:
    - Start the Backend Server on the node(s) that is the cluster's seed and wait until it is running.
    - Start the Backend Server on the other nodes of the cluster and wait until they are running.
3. Start the Frontend cluster:
    - Start the Frontend Server on the nodes of the cluster and wait until they are running.

End

> Tip
>
> When installing the cluster, there may be problems with Fully Qualified Domain Names (related to re-defining the FQDN to the desired IP). You must check that there are no issues each time you deploy Web Engagement into the cluster.
>
> Categories are updated by Web Engagement servers "on the fly" and no additional configuration is necessary.

Load Balancing

Objective	Related procedures and actions
1. Review the load balancing overview.	See the load balancing page.
2. Implement load balancing.	The following procedures provide a sample load balancing configuration for Apache: • Install the Load Balancing Frontend and Backend Servers • Check the Configuration for the Load Balancing Frontend and Backend Servers • Create Server Nodes • Configure Apache for the Load Balancing Frontend Server • Configure Apache for the Load Balancing Backend Server • Configure Cassandra for the Backend Server "seed" node • Configure Cassandra for the Backend Server "non-seed" nodes • Configure Rules Deployment for the Cluster
3. Configure Cassandra.	• Configure the Java heap size • Configure Local Properties • Enable data compression on the Cassandra cluster

Implement Load Balancing

Purpose: Deploying for load balancing; this page defines the creation of nodes enabling load balancing. Read Load Balancing before you start your configuration.

Install the Load Balancing Frontend and Backend Servers

Purpose: Create the Load Balancing server instances.

Prequisites

- You already imported templates for the Frontend and Backend Servers application;
- For this configuration example, you installed and configured Apache, version 2.2;

Start

1. Create and configure the Load Balancing Frontend Server, including provisioning, as detailed in the previous steps of the Installation chapter. See Configuring the Web Engagement Frontend Server for its configuration.
2. Create and install the Load Balancing Backend Server, including provisioning, as detailed in the previous steps of the Installation chapter. See Configuring the Web Engagement Backend Server for its configuration.
3. Create, build, deploy and test a web engagement application as detailed in the Developer's Guide; see Develop your Web Engagement Application.

End

Next Steps

- Check the Configuration for the Load Balancing Frontend and Backend Servers

Check the Configuration for the Load Balancing Frontend and Backend Servers

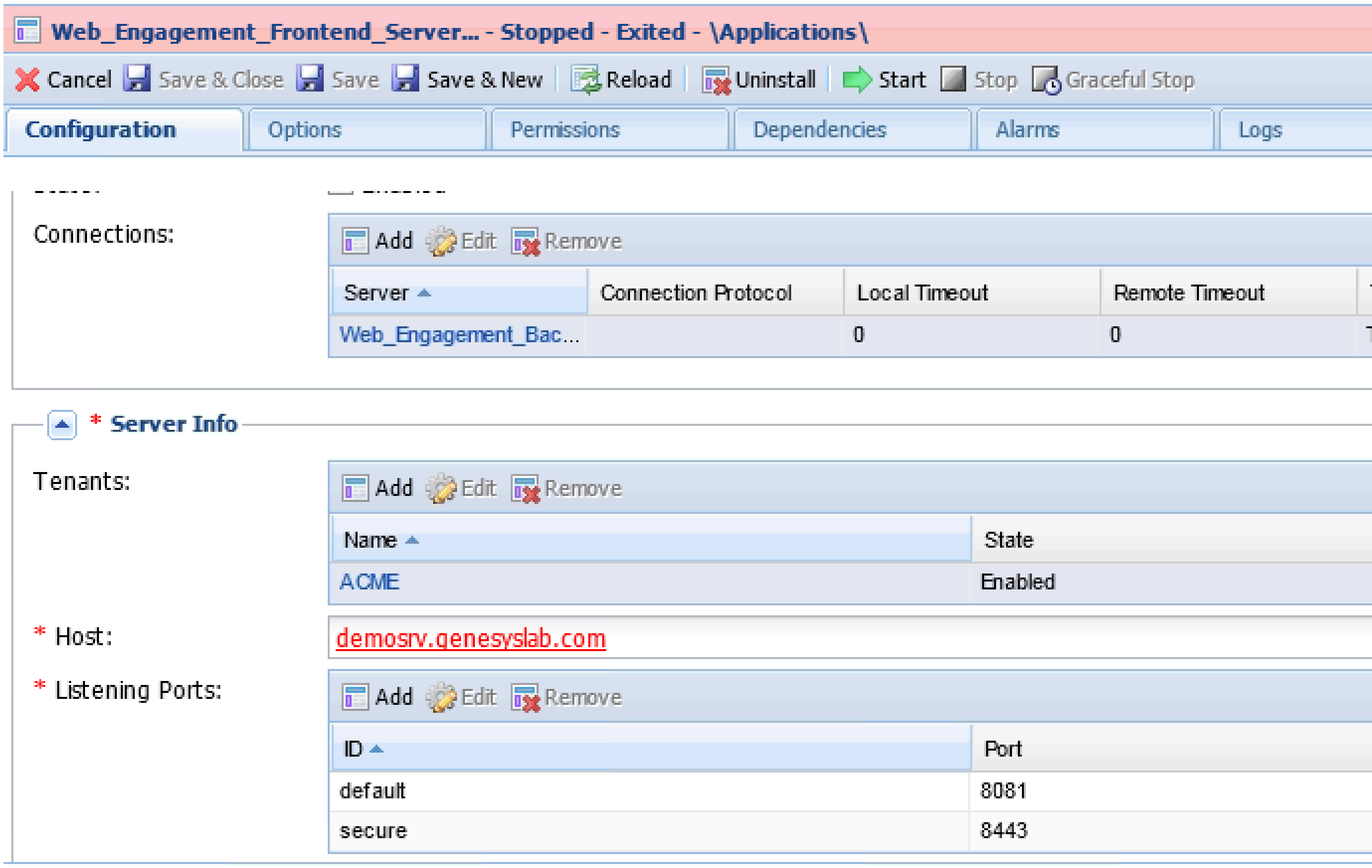
Purpose: To accomplish some higher-level business or configuration goal.

Prequisites

- You already created the configuration applications for the Load Balancing Frontend and Backend Servers.

Start

1. In Genesys Administrator, check your Load Balancing Backend Server configuration. Select the application and click on the `Edit...` button.
    - Open your application and select the options tab. Make sure that all the options contain the right IP addresses and port in their URLs. For instance, if the Load Balancing Backend server should be located at `http://135.225.54.24:9081`, make sure that this URL is used in all concerned options, such as for instance:
        - the `wmsg.connector.scxml.appUrl` option in section `service:wmsg` is set to the `http://135.225.54.24:9081/backend/resources /scxml/src-gen/IPD_default_FetchedWorkflow.scxml` value.
    - Check that the Connections section of the Configuration tab includes the following applications:
        - The Load Balancing Frontend Server;
        - The Interaction Workspace, to enable the display of pages and categories.
2. In Genesys Administrator, check your Load Balancing Frontend Server configuration. Select the application and click on the `Edit...` button.
    - Check that the `Connections` section of the Configuration tab includes the Load Balancing Backend Server.



Web_Engagement_Frontend_Server... - Stopped - Exited - \Applications\

Connections:

Server	Connection Protocol	Local Timeout	Remote Timeout
Web_Engagement_Bac...		0	0

* Server Info

Tenants:

Name	State
ACME	Enabled

* Host: demosrv.genesyslab.com

* Listening Ports:

ID	Port
default	8081
secure	8443

The configuration of the Load Balancing Frontend Server.

End

Next Steps

Create Server Nodes

Create Server Nodes

Purpose: Create applications for Backend and Frontend Servers used as node applications, in opposition to Load Balancing applications defined in previous steps. Note that each server node must use a hostname and a listening port value different from those used by the Load Balancing servers.

Prequisites

- You have created and configured the load balancing applications for the Frontend and Backend Servers;
- You have planned additional hosts for installing Backend and Frontend Servers used as application nodes.
- Each application node will use an IP address and a listening port different from the ones used by the Load Balancing Servers.

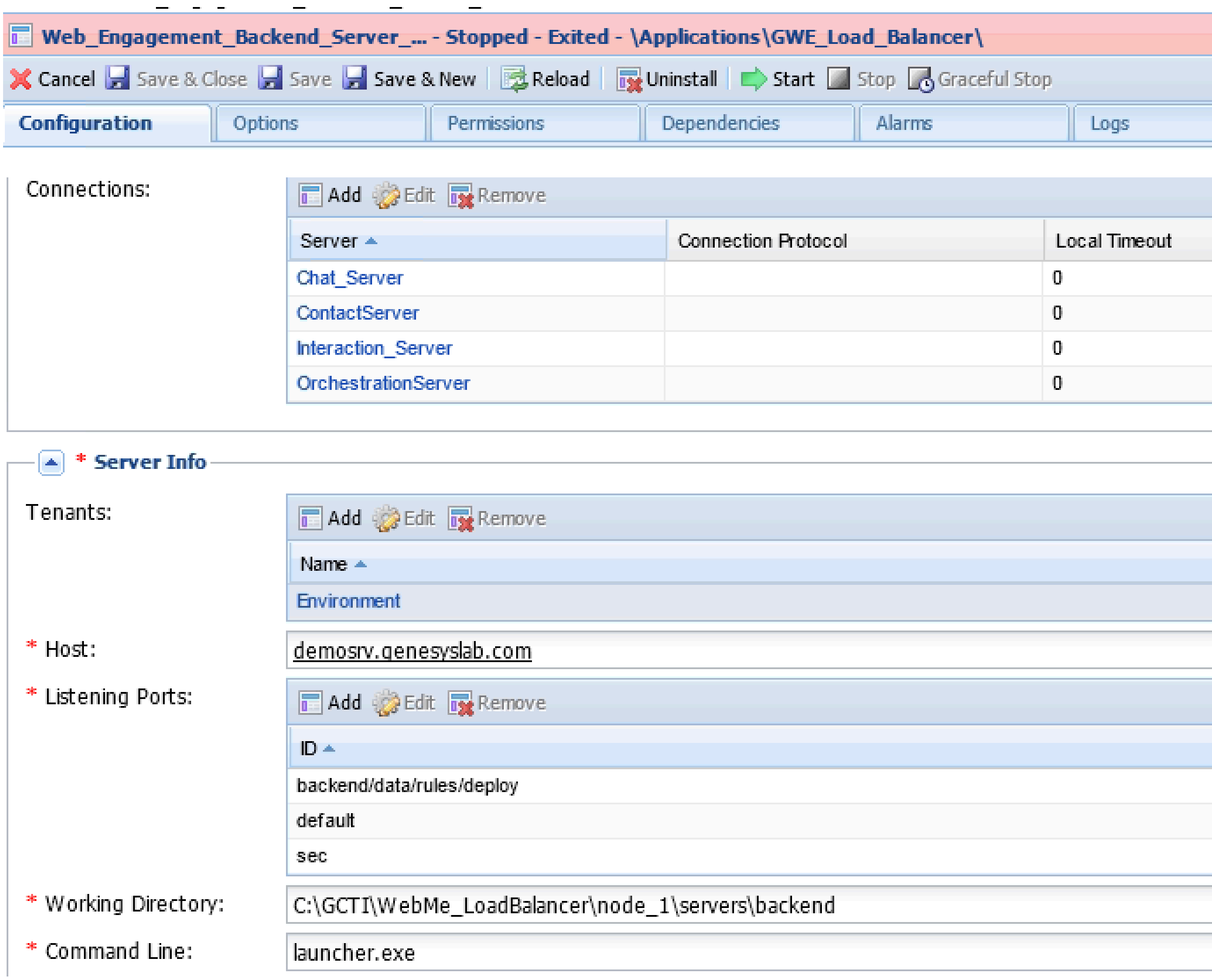
Start

1. For each new node, create a new Backend Server application, including provisioning, as detailed in the previous steps of the Installation chapter. See Configure the Backend Server:
    - Make sure that each server node is using a hostname and a listening port value different from those used by the Load Balancing applications. For instance, if the load balancing servers URL are `http://135.225.54.24:9081` for the Backend Server, you could use the following URLs for your nodes:

      ```
      Node 1 (backend): 192.168.3.103:9083

      Node 2 (backend): 135.225.51.237:9084
      ```

    - In the `Connections` section of your Configuration panel, do not add any connection to a Frontend Server node.



Web_Engagement_Backend_Server_... - Stopped - Exited - \Applications\GWE_Load_Balancer\

Connections:

Server	Connection Protocol	Local Timeout
Chat_Server		0
ContactServer		0
Interaction_Server		0
OrchestrationServer		0

* Server Info

Tenants:

Name
Environment

* Host: demosrv.genesyslab.com

* Listening Ports:

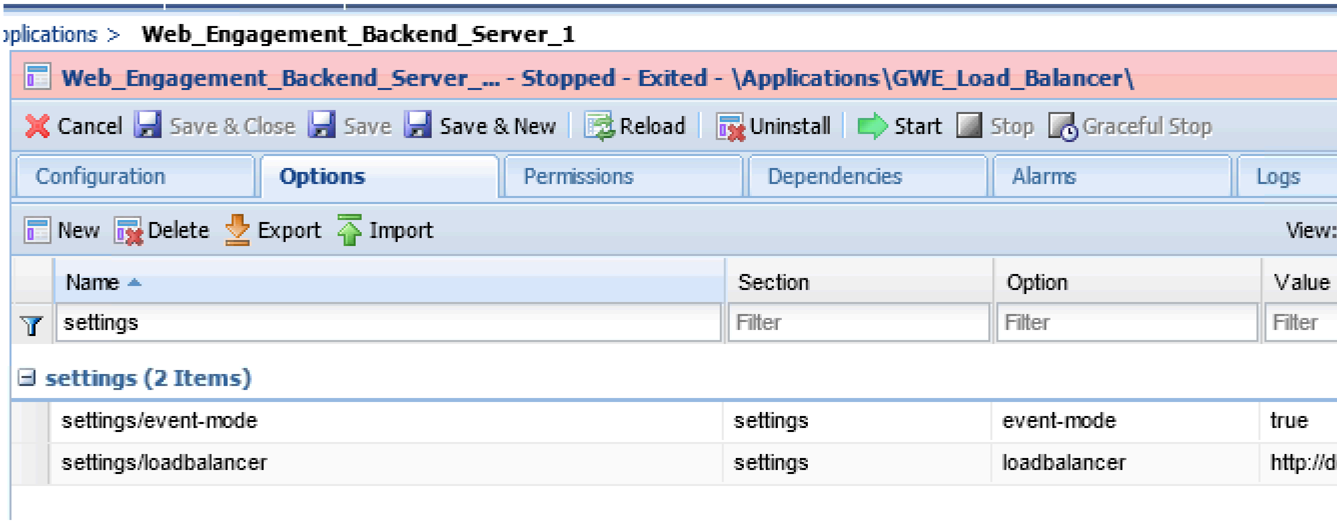
ID
backend/data/rules/deploy
default
sec

* Working Directory: C:\GCTI\WebMe_LoadBalancer\node_1\servers\backend

* Command Line: launcher.exe

Configuration sample for a node

- In the `settings` section of the `Options` tab, the value of the `loadbalancer` option must be set to: `http://<load balancing frontend host address>:<listening port>/frontend` where:
  - `<load balancing frontend host address>` is the host address for the Load Balancing Frontend Server;
  - `<listening port>`is the listening port of the Load Balancing Frontend Server.

- 

  pplications > **Web_Engagement_Backend_Server_1**

  Web_Engagement_Backend_Server_... - Stopped - Exited - \Applications\GWE_Load_Balancer\

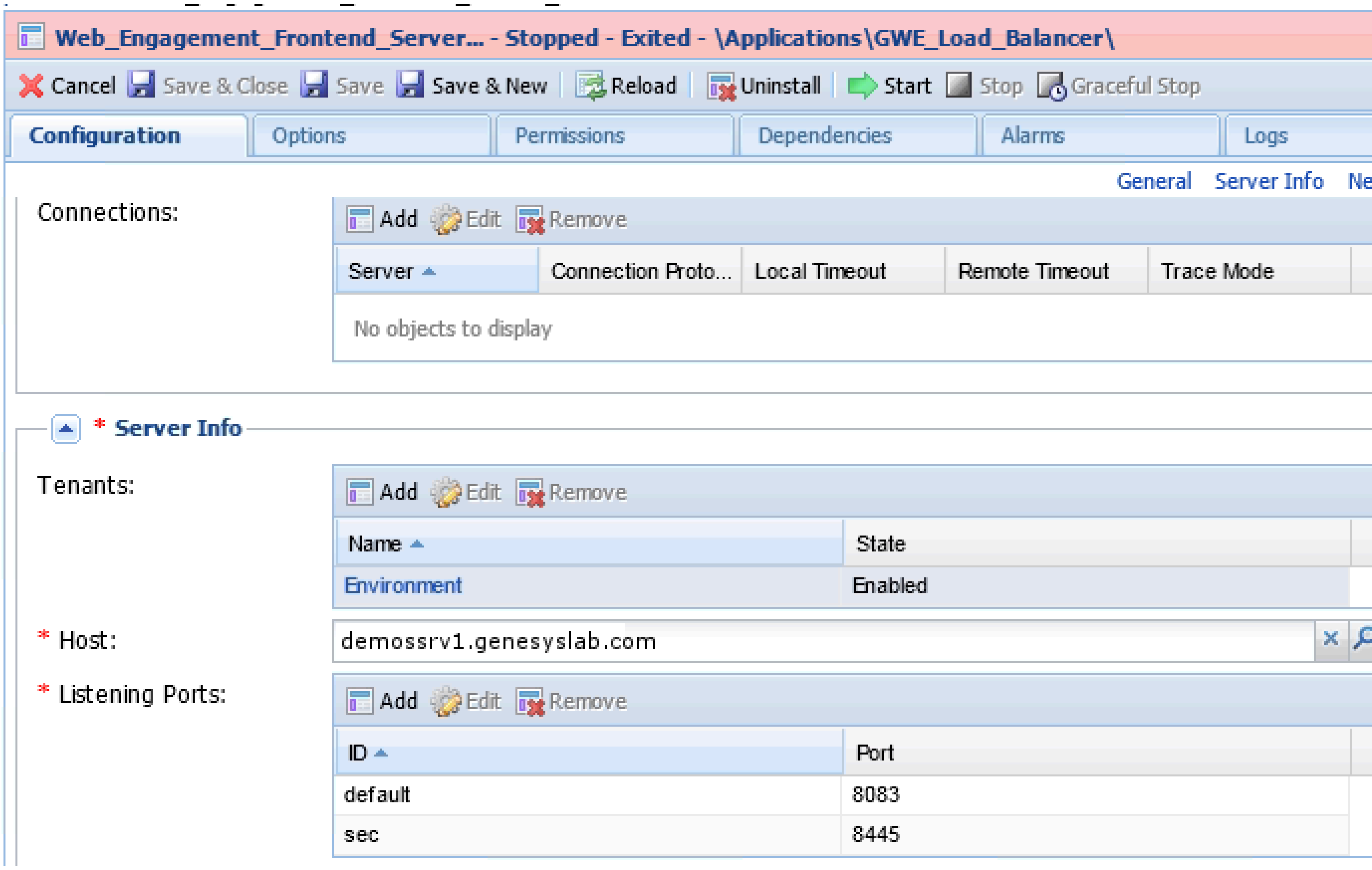
Name	Section	Option	Value
settings	Filter	Filter	Filter
settings (2 Items)			
settings/event-mode	settings	event-mode	true
settings/loadbalancer	settings	loadbalancer	http://d

2. For each new node, create a new Frontend Server application, including provisioning, as detailed in the previous steps of the Installation chapter. See Configure the Frontend Server.
   - Make sure that each application is using a hostname and a listening port value different from those used by the Load Balancing servers. For instance, if the Load Balancing server URL is `http://135.225.54.24:8081` for the Backend Server, you could use the following URLs for your nodes:

     ```
     Node 1 (Frontend): 192.168.3.103:8083
     ```

     ```
     Node 2 (Frontend): 135.225.51.237:8084
     ```

   - In the `Connections` section of your Configuration panel, do **not** add any connection to a Backend Server node.



Connections:

Server	Connection Proto...	Local Timeout	Remote Timeout	Trace Mode
No objects to display				

Tenants:

Name	State
Environment	Enabled

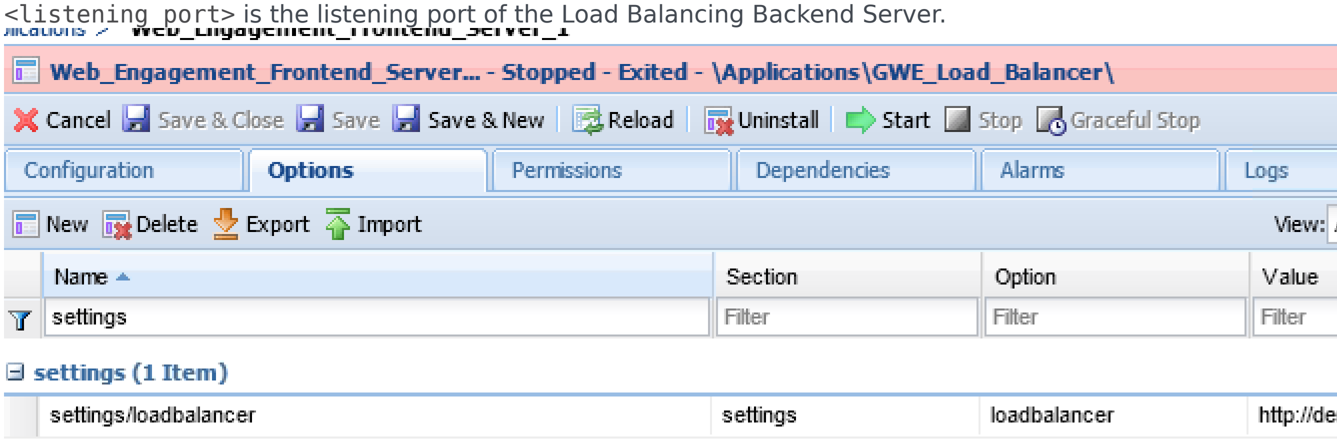
* Host: demossrv1.genesyslab.com

* Listening Ports:

ID	Port
default	8083
sec	8445

Configuration sample for a node

- In the `settings` section of the `Options` tab, the value of the `loadbalancer` option must be set to: `http://<load balancing backend host address>:<listening port>/frontend` where:
  - `<load balancing backendhost address>` is the host address for the Load Balancing Backend Server;
  - `<listening port>` is the listening port of the Load Balancing Backend Server.



Name	Section	Option	Value
settings	Filter	Filter	Filter
settings (1 Item)			
settings/loadbalancer	settings	loadbalancer	http://de

3. Make sure that you created and deployed a Web Engagement application, as detailed in step 3 of Install the Load Balancing Frontend and Backend Servers. Copy the application's server directories, available in the `apps/servers` folder of the Web Engagement Installation Directory;
4. For each node:
   - Paste the server to the corresponding Web Engagement server directory of your node.
   - Edit the launcher.xml file (or launcher_32.xml or launcher_64.xml for Linux) and configure the `webs_host`, `http_port`, `https_port` variables to match the host and port that you specified previously in the node configuration.

End
Next Steps

Configure Apache for the Load Balancing Frontend Server

Configure Apache for the Load Balancing Frontend Server

Purpose: Configure the default Load Balancing server used for the Web Engagement Frontend Server.

Prerequisites

- You must instal `apache-2.2.xx-win32-x86.msi` on the Load Balancing host for the Frontend Server (Apache 2.2 or higher), which enables the use of the extension `mod_proxy_balancer`.
- Make sure that the following modules are present in your Apache `modules\` folder; upload them if they are missing:
  - `mod_proxy.so`
  - `mod_proxy_balancer.so`
  - `mod_proxy_connect.so`
  - `mod_proxy_http.so`

Start

1. Open the Apache installation directory and edit the `httpd.conf` file with a text editor.
2. Add the following text to your Apache configuration in the load modules section:
```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_balancer_module modules/mod_proxy_balancer.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
LoadModule proxy_http_module modules/mod_proxy_http.so
```
3. To implement the node-based configuration, add a balancer member for each node, as following:

```
   ‡‡‡‡‡‡loadbalancer configuration for GWE Frontend (sticky sessions)‡‡‡‡‡
```

```
    ProxyPass / balancer://my_cluster/

    <Proxy balancer://my_cluster>

       BalancerMember http://<Node1 IP address>:<listeningPort1> route=NodeName1

       BalancerMember http://<Node2 IP address>:<listeningPort2> route=NodeName2

     ...

       BalancerMember http://<NodeN IP address>:<listeningPortN> route=NodeNameN

    ProxySet stickysession=alias

    </Proxy>
```

where:

<Nodei IP address> is the IP address of your node, and <listeningPortN>, the associated listening port of your frontend application.

query parameter alias for Frontend cluster sticky session.

NodeName1 is to configuration application name

4. Then, add the following lines to enable the balancer manager and make it visible at the /balancer URL:

```
    <Location /balancer>

    SetHandler balancer-manager

    Order Deny,Allow

    Deny from all

    Allow from all

    </Location>
```

5. Start the Load Balancing server by running httpd.exe.

End

For example, let's consider two Frontend Server nodes, such as GWE_Frontend_103 and GWE_Frontend_52, defined in the Configuration Server:

```
ProxyRequests Off
<Proxy balancer://mycluster/>
   BalancerMember http://192.168.3.103:8083/frontend route=GWE_Frontend_103
   BalancerMember http://135.225.51.148:8084/frontend route=GWE_Frontend_52
ProxySet stickysession=alias
</Proxy>
ProxyPass /frontend/ balancer://mycluster/
‡‡‡‡‡‡‡loadbalancer UI for checking status‡‡‡‡‡‡‡
<Location /balancer>
SetHandler balancer-manager
Order Deny,Allow
Deny from all
Allow from all
</Location>
```

Next Steps

Configure Apache for the Load Balancing Backend Server

Configure Apache for the Load Balancing Backend Server

Purpose: Configure the default Load Balancing Server used for the Web Engagement Backend Server.

Prerequisites

- You must install `apache-2.2.xx-win32-x86.msi` on the Load Balancer host for the Backend Server (Apache 2.2 or higher), which enables the use of the extension `mod_proxy_balancer`.
- Make sure that the following modules are present in your Apache `modules\` folder; upload them if they are missing:
  - `mod_proxy.so`
  - `mod_proxy_balancer.so`
  - `mod_proxy_connect.so`
  - `mod_proxy_http.so`
  - `mod_headers.so`

Start

1. Open the Apache installation directory and edit the `httpd.conf` file with a text editor.
2. Add the following text to your Apache configuration in the load modules section:
   ```
   LoadModule proxy_module modules/mod_proxy.so
   LoadModule proxy_balancer_module modules/mod_proxy_balancer.so
   LoadModule proxy_connect_module modules/mod_proxy_connect.so
   LoadModule proxy_http_module modules/mod_proxy_http.so
   LoadModule headers_module modules/mod_headers.so
   ```
3. To implement the node-based configuration, add a balancer member for each node, as following:

```
‡‡‡‡‡‡loadbalancer configuration for GWE backend ‡‡‡‡‡‡

ProxyRequests Off

Header add Set-Cookie "ROUTEID=.%{BALANCER_WORKER_ROUTE}e; path=/"
 env=BALANCER_ROUTE_CHANGED

<Proxy balancer://mycluster/>

   BalancerMember http://<Node1 IP address>:<listeningPort1> route=NodeName1

   BalancerMember http://<Node2 IP address>:<listeningPort2> route=NodeName2

...
```

```
   BalancerMember http://<NodeN IP address>:<listeningPortN> route=NodeNameN

ProxySet stickysession=ROUTEID

</Proxy>

ProxyPass /backend/ balancer://mycluster/

ProxyPassReverse /backend/ balancer://mycluster/
```

4. Start the load balancer. Run the `httpd.exe` file.

End

For testing your Load Balancing Server, you can send requests to the host/port of the Load Balancing Server, turn off one of your nodes, and verify that your requests and responses delivered correctly. The following code sample shows the configuration for two Backend Server nodes.

```
ProxyRequests Off
Header add Set-Cookie "ROUTEID=.%{BALANCER_WORKER_ROUTE}e; path=/" env=BALANCER_ROUTE_CHANGED
<Proxy balancer://mycluster/>
    BalancerMember http://192.168.3.103:9083/backend route=1
    BalancerMember http://135.225.51.148:9084/backend route=2
        ProxySet stickysession=ROUTEID
</Proxy>
ProxyPass /backend/ balancer://mycluster/
ProxyPassReverse /backend/ balancer://mycluster/
```

Next Steps

Configure Cassandra Cluster

Configure Rules Deployment for the Cluster

Purpose: Configure the Rules Authoring Server and one of your Backend Servers to enable rules deployment over the cluster.

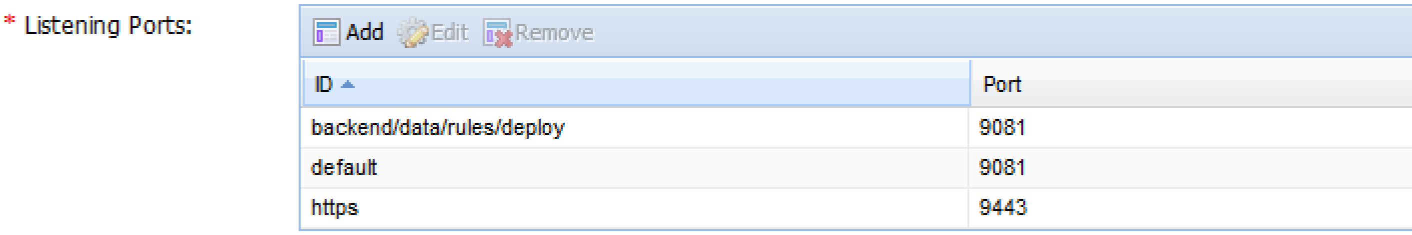
Prerequisites

- You must choose one of your Backend Servers (any node in the cluster) to be in charge of rules deployment over the cluster. After this configuration step, this Backend Server will be able to publish rules for all the Frontend Servers of the cluster.

Start

> ## Important
>
> In the following steps, *Web_Engagement_Backend_server_1* is used as the Backend Server in charge of rules deployment.

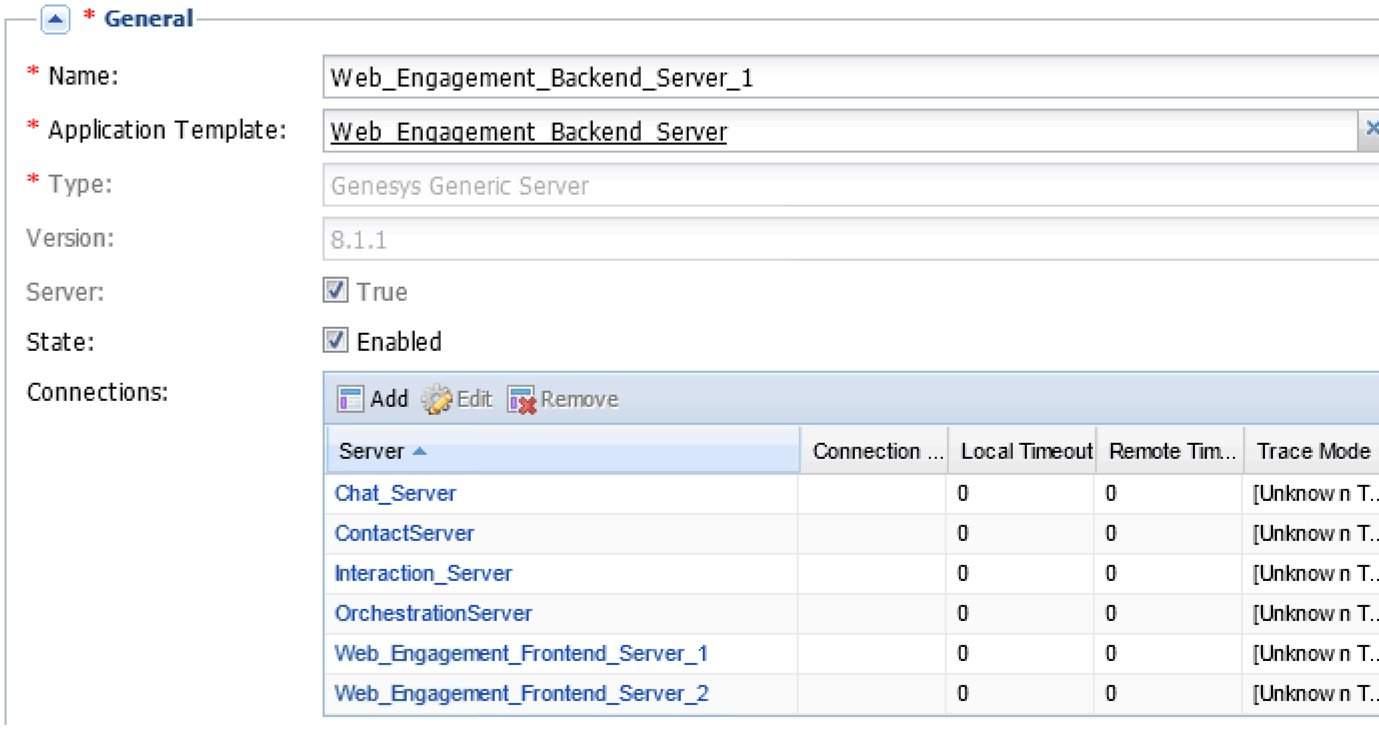
1. In Genesys Administrator, select the Rules Deployment Backend Server and click the `Edit...` button to check its configuration. Select the Configuration tab. Make sure that, in section Server Info, the listening ports include a port with the `backend/data/rules/deploy` id and the `http` protocol.



ID ▲	Port
backend/data/rules/deploy	9081
default	9081
https	9443

Listening ports

2. Add the Rules Deployment Backend Server application to the connections of the Genesys Rules Authoring Server, as detailed in Step 1 of the Configuring the Genesys Rules Authoring Tool procedure.
3. If you have more that one Frontend server in your cluster, add all the Frontend Servers to the Connections of the Rules Deployment Backend Server:
   - In the Connections Section, click the `Add...` button. Browse one of the Frontend Server. Click OK.
   - Redo this step as many time as needed.



* Name: Web_Engagement_Backend_Server_1

* Application Template: Web_Engagement_Backend_Server

* Type: Genesys Generic Server

Version: 8.1.1

Server: ☑ True

State: ☑ Enabled

Connections:

Server	Connection ...	Local Timeout	Remote Tim...	Trace Mode
Chat_Server		0	0	[Unknow n T..
ContactServer		0	0	[Unknow n T..
Interaction_Server		0	0	[Unknow n T..
OrchestrationServer		0	0	[Unknow n T..
Web_Engagement_Frontend_Server_1		0	0	[Unknow n T..
Web_Engagement_Frontend_Server_2		0	0	[Unknow n T..

Connections to all the Frontend Servers

End

- Rules deployment is now possible, as described in Create a rules package;
- Before you start deploying rules, make sure that at least the following servers are up and running:
  - The Backend Server chosen for rules deployment;
  - All the Frontend Servers.

Next Steps

Back to Task Table

Options Reference

	Purpose: Lists all the options available for the configuration of the Web Engagement solution.

The Configuration Options, covered in this sections, concern the following components:

- **Genesys Web Engagement Backend Server Template**

Click to display Backend Server Options

Section Name	Options
log *Configure the logs generated by the Backend Server*	all buffering expire segment standard trace verbose
security *Settings for authentication for the REST API.*	auth-scheme user-id password
service:wes *Settings for the web engagement service*	wes.connector.chatServer.queueKey wes.connector.chatServer.queueWebengagement wes.connector.interaction.copyUserData wes.connector.interactionServer.wcb.queueSubmit
service:wmdb *Settings to enable data retention.*	wmdb.retention.entity.all wmdb.retention.entity.<object> wmdb.retention.time-unit
service:wmsg *Settings for the monitoring service*	wmsg.connector.defaultEngagementChannel wmsg.connector.engagementExpirationTime wmsg.connector.interactionServer.queueAccepted wmsg.connector.interactionServer.queueEngaged wmsg.connector.interactionServer.queueFailed wmsg.connector.interactionServer.queueQualified wmsg.connector.interactionServer.queueRejected wmsg.connector.ors.cmsPassword wmsg.connector.ors.cmsUser wmsg.connector.registrationFormExpirationTime wmsg.connector.secureChat wmsg.connector.scxml.appUrl wmsg.connector.scxml.incomingInteractionQueue wmsg.connector.wmdb.serverUrl wmsg.connector.wns.showRegistrationForm
settings *General settings*	event-mode loadbalancer ors

Section Name	Options
service:pacing *Settings for the pacing algorithm*	pacing.connector.algorithm pacing.connector.chatGroup pacing.connector.optimizationGoal pacing.connector.optimizationTarget pacing.connector.refreshPeriod pacing.connector.voiceGroup

- **Genesys Web Engagement Frontend Server Template**

Click to display Frontend Server Options

Section Name	Options
log *Configure the logs generated by the Frontend Server*	all buffering expire segment standard trace verbose
settings *General settings*	loadbalancer

Backend Server Reference

	Purpose: Lists all the options available for the configuration of the Web Engagement Backend Server application.

Backend Server Options List

Section Name	Options
log *Configure the logs generated by the Backend Server*	all buffering expire segment standard trace verbose
security *Settings for authentication for the REST API.*	auth-scheme user-id password
service:wes *Settings for the web engagement service*	wes.connector.chatServer.queueKey wes.connector.chatServer.queueWebengagement wes.connector.interaction.copyUserData wes.connector.interactionServer.wcb.queueSubmit
service:wmdb *Settings to enable data retention.*	wmdb.retention.entity.all wmdb.retention.entity.<object> wmdb.retention.time-unit
service:wmsg *Settings for the monitoring service*	wmsg.connector.defaultEngagementChannel wmsg.connector.engagementExpirationTime wmsg.connector.interactionServer.queueAccepted wmsg.connector.interactionServer.queueEngaged wmsg.connector.interactionServer.queueFailed wmsg.connector.interactionServer.queueQualified wmsg.connector.interactionServer.queueRejected wmsg.connector.ors.cmsPassword wmsg.connector.ors.cmsUser wmsg.connector.registrationFormExpirationTime wmsg.connector.secureChat wmsg.connector.scxml.appUrl wmsg.connector.scxml.incomingInteractionQueue wmsg.connector.wmdb.serverUrl wmsg.connector.wns.showRegistrationForm
settings *General settings*	event-mode loadbalancer ors
service:pacing *Settings for the pacing algorithm*	pacing.connector.algorithm pacing.connector.chatGroup pacing.connector.optimizationGoal pacing.connector.optimizationTarget

Section Name	Options
	pacing.connector.refreshPeriod pacing.connector.voiceGroup

Backend Server log

	Purpose: Lists the options available for the **log** section of the Genesys Web Engagement Backend Server.

all

- Default Value: `stdout`
- Valid Values (log output types):

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

- Changes take effect: When the application is started or restarted.
- Description: Specifies the outputs to which an application sends all log events. The log output types must be separated by a comma when more than one output is configured. For example:

```
all = stdout, logfile
```

buffering

- Default Value: `true`
- Valid Values:

`true`	Enables buffering.
`false`	Disables buffering.

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

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

expire

- Default Value: 3
- Valid Values:

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

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

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

segment

- Default Value: `1000`
- Valid Values:

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

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

standard

- Default Value: `stdout`
- Valid Values:

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

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

trace

- Default Value: `stdout`
- Valid Values:

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

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

verbose

- Default Value: `info`
- Valid Values:

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

- Changes Take Effect: Immediately
- Description: Determines whether a log output is created. If it is, specifies the minimum level of log events generated. The log events levels, starting with the highest priority level, are Standard, Interaction, Trace, and Debug.

Backend Server security

Purpose: Lists the options available for the **security** section of the Genesys Web Engagement Backend Server. This section defines the authentication used for the REST API and its clients. See Authentication for details.

auth-scheme

- Default Value: `none`
- Valid Values: `none` or `Basic`
- Changes take effect:
- Description: Specifies the HTTP authentication scheme used to secure the History REST API. With the `Basic` scheme, clients must be authenticated with a user ID and password.

user-id

- Default Value: No default value
- Valid Values: Any string
- Changes take effect:
- Description: The user identifier (login) used in authentication for the History REST API. See auth-scheme.

password

- Default Value: No default value
- Valid Values: Any string
- Changes take effect:
- Description: The user password used in authentication for the HISTORY REST API. See auth-scheme.

Backend Server service:wes

Purpose: Lists the options available for the **service:wes** section of the Genesys Web Engagement Backend Server. This section defines the server-side engagement functionality; it covers the creation of media interaction and the messages used for the transfer from the client-side application to the Genesys media servers.

Note: Some default values are filled by the Provisioning Tool.

wes.connector.chatServer.identifyCreateContact

- Default Value: 3
- Valid Values: `1, 2` or 3
- Changes take effect: When the application is started or restarted.
- Description: Controls whether a contact for a created chat interaction should be identified and/or created in the Universal Contact Server database (transferred as attached data to Chat Server, which is responsible for details of processing). The valid values have the following effect:
  - 1 — Do not identify and do not create contact
  - 2 — Identify, but do no create contact
  - 3 — Identify and create contact (if absent)

  The default value is applied if the option is missed or specified incorrectly.

wes.connector.chatServer.requestTimeout

- Default Value: 5
- Valid Values: `1 – 30`
- Changes take effect: When the application is started or restarted.
- Description: Specifies the timeout (in seconds) for the `sendRequest` operation in Chat Server. The default value is used if the option is absent or specified incorrectly.

wes.connector.chatServer.queueKey

- Default Value: `1:webme`
- Valid Values: Key defined in the section `endpoints:<tenantID>` of the ChatServer application.
- Changes take effect: When the application is started or restarted.
- Description: Specify the key of a queue used by the Chat Server. For instance, for the Environment tenant, this would be a key defined in the `endpoints:1` section of the ChatServer application.

wes.connector.chatServer.queueWebengagement

- Default Value: `WebEngagement_ChatRouting.webme_chat.WebME_ChatQueue`
- Valid Values: Letters A to Z and a to z. Numbers 0 through 9. The minus, dot, underscore characters. Space characters are allowed inside the string only (not at the beginning or at the end).
- Changes take effect: When the application is started or restarted.
- Description: Name of the Interaction Queue object used to place chat interactions for routing with the Web Engagement Chat Routing strategy.

wes.connector.interaction.copyUserData

- Default Value: `no`
- Valid Values: `all, no, <key_name1;key_name2;...key_nameN>`
- Changes take effect: When the application is started or restarted.
- Description: Specifies the mode Web Engagement uses to copy UserData from the Open Media webengagement interaction to the chat or webcallback engagement interaction. The following explains the results for each valid value:
  - `all` — All keys are copied.
  - `<key_name1;key_name2;...key_nameN>` — Only the keys in the list are copied.
  - `no` — No UserData is copied
  - Option is omitted, or has a blank or empty value — No UserData is copied.

> **Important**
>
> Trailing or leading spaces are considered part of the key name.

wes.connector.interactionServer.wcb.queueSubmit

- Default Value: `New`
- Valid Values: Letters A to Z and a to z. Numbers 0 through 9. The minus, dot, underscore characters. Space characters are allowed inside the string only (not at the beginning or at the end).
- Changes take effect: When the application is started or restarted.
- Description: Message for the queue used to submit newwebcallback interactions. Standard for the Web API Server.

Backend Server service:wmdb

wmdb.retention.entity.all

- Default Value: 14
- Valid Values: Any integer
- Changes take effect: When the application is started or restarted.
- Description: The default expiration period for all entities, in the time units set in the wmdb.retention.time-unit option. The entities affected by this option are: `event, page, session, visit, engagementattempt, useragent`, and `identity`. This option is mandatory.

wmdb.retention.entity.<object>

- Default Value: The value of the wmdb.retention.entity.all option.
- Valid Values: Any integer
- Changes take effect: When the application is started or restarted.
- Description: Defines the default expiration period for the selected entity, in the time units set in the wmdb.retention.time-unit option. Possible values for `<object>` are: `event, page, session, visit, engagementattempt, useragent`, and `identity`. Setting `wmdb.retention.entity.<object>` is optional.

wmdb.retention.time-unit

- Default Value: `day`
- Valid Values: `sec, min, hour, day, month`
- Changes take effect: When the application is started or restarted.
- Description: Defines the time units for the expiration period set in the wmdb.retention.entity.all and wmdb.retention.entity.<object> options. This option is mandatory.

Backend Server service:wmsg

Purpose: Lists the options available for the **service:wmsg** section of the Genesys Web Engagement Backend Server.

Note: Some default values are filled by the Provision Tool.

wmsg.connector.defaultEngagementChannel

- Default Value: N/A
- Valid Values: proactiveChat, proactiveCallback
- Changes take effect: Immediately.
- Description: A valid name for the default Engagement Channel. When specified, this option turns off detection of an agent's availability by the pacing service.

> **Important**
>
> The `wmsg.connector.defaultEngagementChannel` option is intended for development purposes only and should not be used in a production environment.

wmsg.connector.engagementExpirationTime

- Default Value: `60`
- Valid Values: A positive integer or a double value.
- Changes take effect: Immediately.
- Description: The time at which the engagement attempt is considered to be expired.

wmsg.connector.interactionServer.queueAccepted

- Default Value: `Webengagement_Accepted`
- Valid Values: Letters A to Z and a to z. Numbers 0 through 9. The underscore and space characters.

Note: Be sure that the specified value uses the name of the Interaction Queue object as provisioned in Configuration Manager.

- Changes take effect: Immediately.
- Description: A valid name of a queue used for the accepted interactions. An interaction is placed in this queue if the customer accepts the engagement proposal (the disposition code is set to `acceptCall`; Genesys Web Engagement then stops the interaction.

wmsg.connector.interactionServer.queueEngaged

- Default Value: `Webengagement_Engaged`
- Valid Values: Letters A to Z and a to z. Numbers 0 through 9. The underscore and space characters.
- Changes take effect: Immediately.
- Description: A valid name of a queue used for the engagement interactions. An interaction is placed in this queue if Orchestration Server decides to engage the customer. The interaction usually is not stopped till it located in this queue.

wmsg.connector.interactionServer.queueFailed

- Default Value: `Webengagement_Failed`
- Valid Values: Letters A to Z and a to z. Numbers 0 through 9. The underscore and space characters.
- Changes take effect: Immediately.
- Description: A valid name of a queue used for the failed engagement interactions. An interaction is placed in this queue in two cases:
  - If its disposition code is set to timeout.
  - If the interaction was cleaned from another queue.
  - If the Orchestration Server strategy notified the Backend server that interaction should not be processed to the real engagement.
    Web Engagement Backend server will stop the interaction as soon as it placed in the Failed queue.

wmsg.connector.interactionServer.queueQualified

- Default Value: `Webengagement_Qualified`
- Valid Values: Letters A to Z and a to z. Numbers 0 through 9. The underscore and space characters.
- Changes take effect: Immediately.
- Description: A valid name of a queue used for the creation of engagement interactions. Each time Genesys Web Engagement creates an interaction, the new interaction is added to this queue. The interaction is not stopped until it is located in this queue.

wmsg.connector.interactionServer.queueRejected

- Default Value: `Webengagement_Rejected`
- Valid Values: Letters A to Z and a to z. Numbers 0 through 9. The underscore and space
- Changes take effect: Immediately.
- Description: A valid name of a queue used for the rejected engagement interactions. An interaction is placed in this queue if the customer rejects the engagement proposal (the disposition code is set to reject); Genesys Web Engagement then stops the interaction.

wmsg.connector.ors.cmsPassword

- Default Value: N/A

- Valid Values: Letters A to Z and a to z. Numbers 0 through 9. The underscore and space characters.
- Changes take effect: When the application is started or restarted.
- Description: Password for the account used to connect to the Context Management Server.

wmsg.connector.ors.cmsUser

- Default Value: `N/A`
- Valid Values: Letters A to Z and a to z. Numbers 0 through 9. The underscore and space characters.
- Changes take effect: When the application is started or restarted.
- Description: A valid name of an account used to connect to the Context Management Server.

wmsg.connector.registrationFormExpirationTime

- Default Value: `120`
- Valid Values: `30—1800`
- Changes Take Effect: Immediately
- Description: Specifies the time, in seconds, during which the registration form must be completed. If the registration form is not completed before the expiration time, the engagement attempt is considered invalid.

wmsg.connector.secureChat

- Default Value: `false`
- Valid Values: `true, false`
- Changes take effect: When the application is started or restarted.
- Description: If true and a secure load balancer or secure port is not configured, the application will throw an exception and will not work. All cometD communication should be in the https protocol with a secure chat connection.

wmsg.connector.scxml.appUrl

- Default Value: `http://<$BACKEND_HOST$>:<$BACKEND_PORT$>/backend/resources/scxml/src-gen/IPD_default_FetchedWorkflow.scxml`
- Valid Values: Letters A to Z and a to z. Numbers 0 through 9. The underscore and space characters.
- Changes take effect: When the application is started or restarted.
- Description: The URL under which the engagement Orchestration Server strategy is stored. Used in the REST-based approach of work with Orchestration Server.

wmsg.connector.scxml.incomingInteractionQueue

- Default Value: `WebEngagement_EngagementLogic.queueBased.Incoming`
- Valid Values: Letters A to Z and a to z. Numbers 0 through 9. The underscore and space characters.

- Changes take effect: Immediately.
- Description: The name of interaction Queue into which Web Engagement will place interactions that should be consumed by engagement strategy. If specified, turned on queue-based approach of work with ORS. If not specified (or empty) - REST-based approach of work with ORS will be used.

wmsg.connector.wmdb.serverUrl

- Default Value: `http://<$BACKEND_HOST$>:<$BACKEND_PORT$>/backend`.
- Valid Values: URL of this backend server application + "/backend"
- Changes take effect: When the application is started or restarted.
- Description: The URL used for accessing the Cassandra Database of the Backend Server application (or cluster).

wmsg.connector.wns.showRegistrationForm

- Default Value: `all`
- Valid Values: all, anonymous, recognized
- Changes take effect: Immediately.
- Description: Specifies the type of users for which the registration form should be shown.

Backend Server settings

	Purpose: Lists the options available for the **settings** section of the Genesys Web Engagement Backend Server.

event-mode

- Default Value: `true`
- Valid Values: `true`, `false`
- Changes take effect: When the application is started or restarted.
- Description: Enables or disables the event mode for the collaboration with Genesys Rules System.

loadbalancer

- Default Value: ""
- Valid Values: `<IP_address>:<listening_port>/frontend` or empty
- Changes take effect: When the application is started or restarted.
- Description: Sets the location of the Load Balancer Frontend Server. This option must be set only for Backend Servers used as nodes in a load balancing deployment. If this option is set, the configuration information in the Connections tab of the Frontend Server will be ignored. The Connections section of the Backend Sever node must not include any connection to a Frontend Server application.

ors

- Default Value: ""
- Valid Values: `<IP_address>:<listening_port>`
- Changes take effect: When the application is started or restarted.
- Description: Specifies host and port of Orchestration Servers' cluster. If specified, it will be used for interactions with ORS through REST. If not specified or specified as empty, the information set in the Connections of the Backend Server is used.

Backend Server service:pacing

pacing.connector.algorithm

- Default Value: `progressive`
- Valid Values: `progressive`
- Changes take effect: When the application is started or restarted.
- Description: Defines the optimization model. Currently, progressive is the only supported model.

pacing.connector.chatGroup

- Default Value: `Web Engagement Chat`
- Valid Values: One or more chat group names. You must separate multiple groups with a semi-colon (;). For example, `Chat Group 1;Chat Group 2`.
- Changes take effect: When the application is started or restarted.
- Description: Defines the agent group managing chat engagements.
  Note: Any agent that participates in the Web Engagement project should belong to one (exactly) of the groups defined for the pacing algorithm (chatGroup or voiceGroup). If an agent belongs to more than one group, the pacing algorithm will produce incorrect results.

> **Important**
>
> Leading or trailing spaces are considered part of the group name. For example, `My Group 1; My Group 2` and `My Group 1;My Group 2` are different sets because the first set has a space after the semi-colon.

pacing.connector.optimizationGoal

- Default Value: 3
- Valid Values: A positive integer between 1 and 99.
- Changes take effect: When the application is started or restarted.
- Description: Defines the goal percent for the specified optimization target. Recommendations are the following:
  - `ABANDONEMENT_RATE`: Set optimizationGoal from 3 to 5;
  - `BUSY_FACTOR`: Set optimizationGoal from 70 to 85.

pacing.connector.optimizationTarget

- Default Value: `ABANDONEMENT_RATE`

- Valid Values: `ABANDONEMENT_RATE`, `BUSY_FACTOR`
- Changes take effect: When the application is started or restarted.
- Description: Defines the optimization type for the agent targets. ABANDONEMENT_RATE is the ratio of abandoned calls among the total number of calls transferred to the queue, while BUSY_FACTOR is the agent occupancy. This option is associated with the optimizationGoal option, which defines the percentage to use for the current optimization target. Recommendations for optimizationGoal are the following:
  - `ABANDONEMENT_RATE`: Set optimizationGoal from 3 to 5;
  - `BUSY_FACTOR`: Set optimizationGoal from 70 to 85.

pacing.connector.refreshPeriod

- Default Value: 1
- Valid Values: A positive integer between 1 and 5.
- Changes take effect: When the application is started or restarted.
- Description: Defines the period, in seconds, to refresh the data retrieved from the Statistic Server. The Pacing Algorithm is executed as soon as the data are refreshed.

pacing.connector.voiceGroup

- Default Value: `Web Engagement Voice`
- Valid Values: One or more voice group names. You must separate multiple groups with a semi-colon (;). For example, `Voice Group 1;Voice Group 2`.
- Description: Defines the agent group managing voice engagements.
  Note: Any agent that participates in the Web Engagement project should belong to one (exactly) of the groups defined for the pacing algorithm (chatGroup or voiceGroup). If an agent belongs to more than one group, the pacing algorithm will produce incorrect results.

> ### Important
>
> Leading or trailing spaces are considered part of the group name. For example, `My Group 1; My Group 2` and `My Group 1;My Group 2` are different sets because the first set has a space after the semi-colon.

Frontend Server Reference

	Purpose: Lists all the options available for the configuration of the Web Engagement Frontend Server application.

Frontend Server Options List

Section Name	Options
log *Configure the logs generated by the Frontend Server*	all buffering expire segment standard trace verbose
settings *General settings*	loadbalancer

Frontend Server log

	Purpose: Lists the options available for the **log** section of the Genesys Web Engagement Frontend Server.

all

- Default Value: `stdout`
- Valid Values (log output types):

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

- Changes take effect: When the application is started or restarted.
- Description: Specifies the outputs to which an application sends all log events. The log output types must be separated by a comma when more than one output is configured. For example:

```
all = stdout, logfile
```

buffering

- Default Value: `true`
- Valid Values:

`true`	Enables buffering.
`false`	Disables buffering.

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

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

expire

- Default Value: 3
- Valid Values:

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

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

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

segment

- Default Value: `1000`
- Valid Values:

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

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

standard

- Default Value: stdout
- Valid Values (log output types):

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

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

```
standard = stderr, network
```

trace

- Default Value: `stdout`
- Valid Values (log output types):

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

- Changes Take Effect: Immediately
- Specifies the outputs to which an application sends the log events of the `Trace` level and higher (that is,

log events of the `Standard`, `Interaction`, and `Trace` levels). The log outputs must be separated by a comma when more than one output is configured. For example:

```
trace = stderr, network
```

verbose

- Default Value: `standard`
- Valid Values:

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

- Changes Take Effect: Immediately
- Description: Determines whether a log output is created. If it is, specifies the minimum level of log events generated. The log events levels, starting with the highest priority level, are Standard, Interaction, Trace, and Debug.

Frontend Server settings

Purpose: Lists the options available for the **settings** section of the Genesys Web Engagement Frontend Server.

Note: Some default values are filled by the Provision Tool.

loadbalancer

- Default Value:
- Valid Values: `<IP address>:<listening port>/backend`
- Changes take effect: When the application is started or restarted.
- Description: Sets the location of the Load Balancer Backend server. This option must be set only for Frontend Servers used as nodes in a load balancing deployment. If this option is set, information about connection to Backend Server from Connections tabe will be ignored.