Deployment Guide

Genesys Web Engagement 8.5.2

1/2/2022

Table of Contents

Genesys Web Engagement Deployment Guide

Welcome to the *Genesys Web Engagement Deployment Guide*. This document introduces you to the concepts, terminology, and procedures relevant to Genesys Web Engagement. See the summary of chapters below.

Getting Started

Find information to help plan your Genesys Web Engagement Deployment.

What is Genesys Web Engagement?

Deployment Scenarios

Prerequisites

Sizing

Installing GWE in a Lab

Find procedures to install and configure Genesys Web Engagement in a lab environment.

Lab Deployment Scenario

Installing Genesys Web Engagement

Automatic Provisioning

Installing the GWE Plug-ins

Find procedures to install and configure the Genesys Web Engagement plug-ins.

Installing the Plug-in for Workspace Desktop Edition

Deploying GWE in Production

Find procedures to deploy and configure Genesys Web Engagement in a production environment.

Production Deployment Scenario

Load Balancer Sample Configurations

External Cassandra Settings

Configuring Specific Features

You can configure these features to

enable their functionality:

A Pacing Algorithm

A Chat Channel (legacy, deprecated)

Access to GeoIP Information

What is Genesys Web Engagement?

If you're just getting started with Web Engagement, check out A First Look at Genesys Web Engagement.

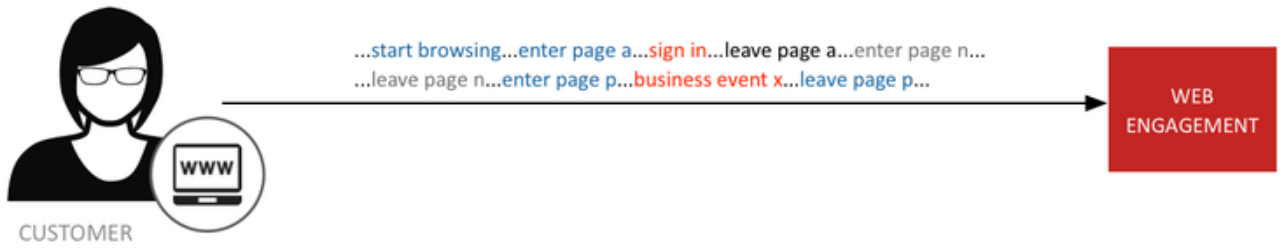
Overview

Genesys Web Engagement enhances customer experience by bringing together three related technologies:

- **Monitoring** records the customer's web browsing activity.
- **Notification** lets a customer know about an offer to engage via chat so they can accept it or decline it.
- **Engagement** provides the actual chat window to the customer's browser. (Starting with GWE 8.5.1, engagement technology is deprecated. Instead, you can integrate GWE with CX Widgets.)

Here's how they work together:

Monitoring

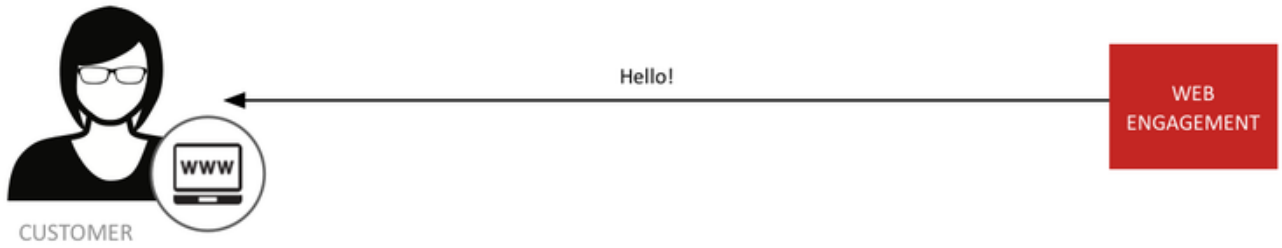


Monitoring allows you to gather data on customer browsing behavior so you can analyze it.

For example, if you've noticed that a lot of customers reach a certain point on your site and then abandon their transactions, you may want to offer a chat as soon as other people reach that point.

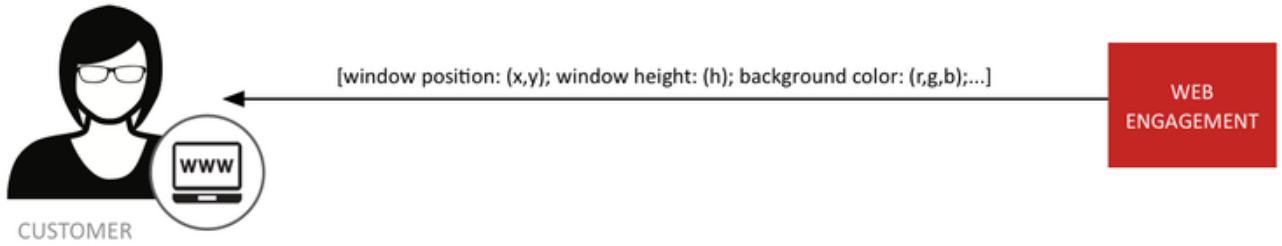
Another case might be where someone has been looking at a certain type of product, and you want your agents to know what it was, so they can respond to the customer's queries more intelligently.

Notification



While the concepts behind **notification** are simple, the power it gives you can lead to a lower abandon rate and higher levels of customer satisfaction—all because Web Engagement provides easy-to-use JavaScripts that allow you to reach out to those customers who need help.

Engagement



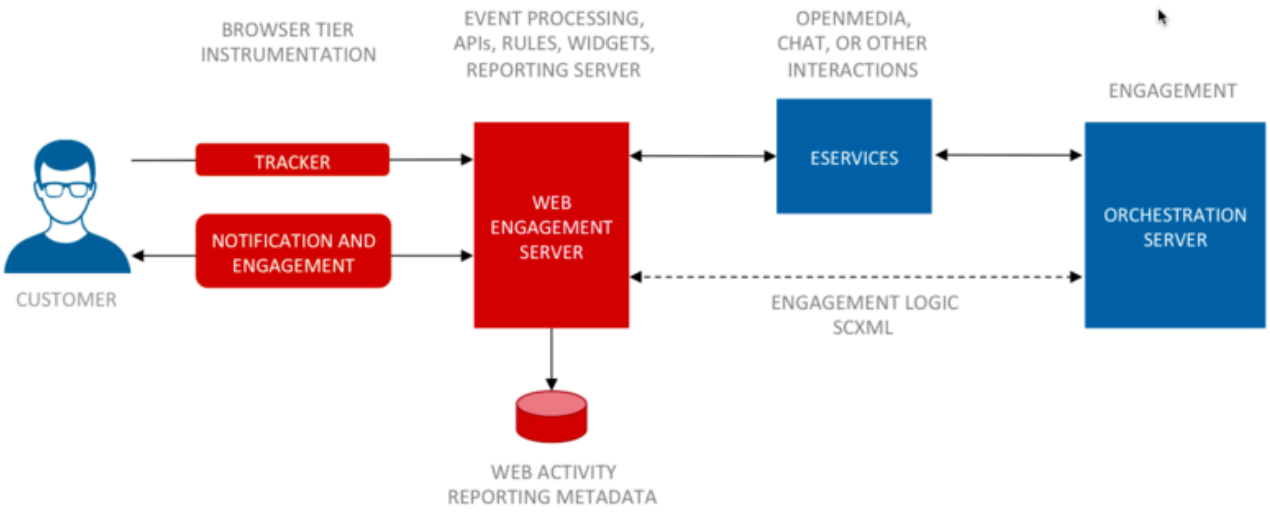
Engagement uses simple JavaScript technologies to display a chat invitation window to the appropriate customers.

Basic Architecture

Browser Tier

Each of these three functions has an agent in the browser tier, as shown in the diagram below.

The browser tier connects to a load balancer that in turn communicates with a cluster of Web Engagement Servers. This cluster is connected to the rest of the Genesys solution, as well as to the Web Engagement database, which is frequently located in the Enterprise Tier.

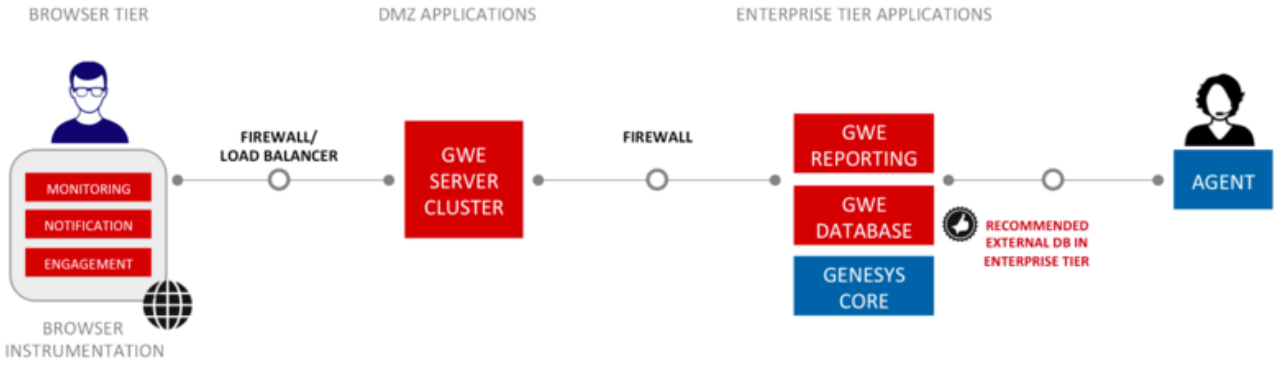


Enterprise Tier

Let's take a closer look at this architecture, with a somewhat sharper focus on the enterprise tier.

Web Engagement 8.5.2 uses Cassandra 3.11 for database storage and Elasticsearch 6.8 as an indexing engine. These two components work closely together to provide high-performance access to large amounts of data, both for operational needs and as input to your reporting. As mentioned, the database is often housed in the enterprise tier, but these components can live anywhere within reach of the Web Engagement Server cluster.

The rest of the enterprise tier consists of the various Genesys components that work together with Web Engagement, backing it up with the full power of the Genesys CX solution.



The Web Engagement Tier

The Web Engagement Cluster is built on top of an N + 1 architecture to support an easily extendable

set of Web Engagement Servers. These servers facilitate a complicated set of services that bring together the power of Web Engagement. Let's take a look at some of the most important ones.

Events, Categories, and Rules

System Events

The basic goal of Web Engagement is to help you decide when to invite a customer to engage with your agents. You can use Web Engagement's six **System Events** to help with this. These events are:

- **VisitStarted**—Includes the URL the customer first visited on your site and the time they started the visit
- **PageEntered**—Includes the URL and time when a customer entered a page
- **PageExited**—Includes the URL and time when a customer exited a page
- **UserInfo**—Reflects that a visitor is recognized, but not signed in
- **SignIn**—Reflects that the user has signed in
- **SignOut**—Reflects that the user has signed out

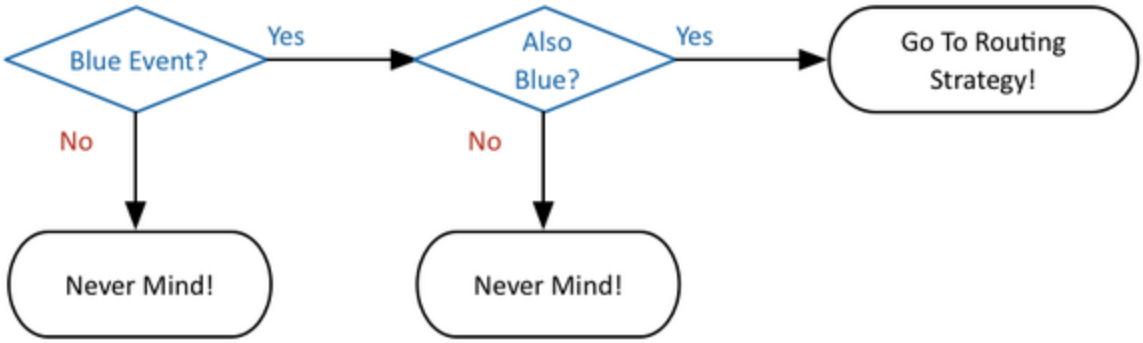
Sometimes you just need to know how long someone has been on your site. But in many situations, you are looking for more complicated patterns. These six events give you a lot to work with.

Categories

Electronics	ExpensivePurchase	ContactUs	None
PageEntered	AddToCart	PageEntered	
Timeout-30	RemoveFromCart	Timeout-30	
PageExited		LocationMapClick	
		PageExited	

And then there are the times when you need to pay attention to certain *types* of activity on your site. For example, you may be having a special on 40-inch HDTVs, so you want to keep track of customers who are visiting the pages for 40-inch models. You can set up a **Category** using the Web Engagement Management interface that allows you to tag the **PageEntered** events for these pages so you can track them.

Rules



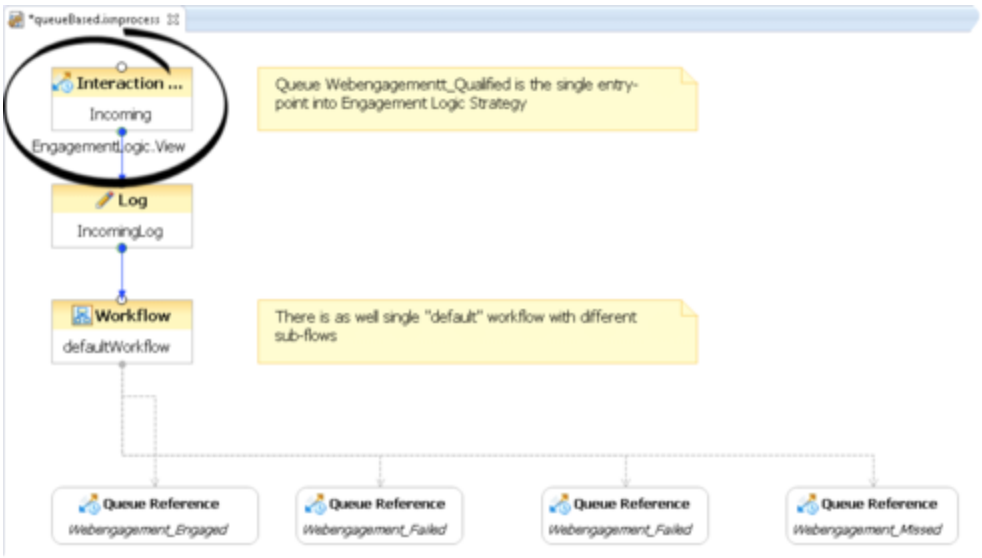
Once you know which system events and categories you're interested in, you need to write some logic that decides what to do with the information you have gathered about them. This logic is known as a **Rule**.

Business Events

In some cases, the system events and categories don't tell you everything you need to know. In those situations, you may want to define your own **Business Events**. As with system events, you can often get a lot more out of your business events by using them in conjunction with categories.

Routing and Pacing

Routing Strategies



Having identified your events and categories, and after writing your rules, you can finally decide who to engage. Interactions for these customers will be processed by the appropriate **Routing Strategy**.

Pacing

When you start sending out chat invitations, you need to make sure you don't send more than your agents can handle. Web Engagement includes intelligent **pacing** technology to help you optimize the number of invitations you send at any given time.

Web Engagement Applications

To make all of these features work together, you must develop a Web Engagement **application**, so that your website can start using the power of Genesys Web Engagement. The Web Engagement application is basically a set of site-specific resources designed to be deployed into the Genesys Web Engagement cluster.

Important: Before using Genesys Web Engagement, you must either create and deploy your own Web Engagement application or deploy the provided sample application. Otherwise, the product will not work.

Multi-Tenancy

You can create one Genesys Web Engagement cluster per tenant.

Deployment Scenarios

Genesys Web Engagement has two flavors of deployment: the simplest is appropriate for a lab environment, while a production environment requires solution that is a bit more complex. Select the appropriate tab below for details about each deployment scenario and the tasks to install and configure GWE.

> **Important**
>
> Genesys strongly recommends that you first install Genesys Web Engagement in a lab configuration, which will make it easier to start working with your GWE application.

Lab

Overview

This deployment is appropriate for a lab environment and consists of a single Web Engagement Server node.

Deployment Tasks

Complete the following tasks to deploy Genesys Web Engagement:

1. Review the prerequisites. Make sure your planned environment meets the requirements for Genesys Web Engagement and contains the right versions of the required Genesys components.
2. Install the Web Engagement server. Complete these procedures to configure and install the GWE Server.
3. Configure Genesys Rules System. You need to configure Genesys Rules Authoring Tool and Genesys Rules Development Tool to work with GWE.
4. Install the Plug-in for Workspace Desktop Edition. You will need this plug-in to enable chat engagement features in Workspace Desktop Edition.
5. Configure your required security. You can enable SSL, configure TLS for Genesys and GWE servers, and enable authentication for the Web Engagement Server, Workspace Desktop Edition, and the Engagement Strategy.
6. Configure your required features. You can set up the pacing algorithm, enable authentication and SSL, and more.
7. Develop an application. You're now ready to move on to the Developer's Guide, where you will learn how to develop and customize a Web Engagement application, not to mention being able to create

categories, business events, and rules.

Production

Overview

This deployment scenario is appropriate for a production environment and consists of multiple Web Engagement Server nodes. Ideally, these servers should be installed on separate hosts to provide the best performance and the best high availability.

> ### Important
> You could host your Web Engagement Servers either in DMZ or in a secure zone, along with your Chat Server(s), in order to protect your data. The Web Engagement Servers do require internet access for chat traffic, but this can be solved by using a reverse proxy, which is a standard function of most load balancers.

> ### Important
> Genesys recommends that you have a minimum of 3 Web Engagement Servers for each cluster installation. This prevents "split brain" issues in the cluster and also enables the cluster to continue operating if one of the nodes fails.

Deployment Tasks

Once you have thoroughly explored Web Engagement's features in your lab environment, you can complete the following tasks to deploy Genesys Web Engagement in production:

1. Review the prerequisites. Make sure your planned environment meets the requirements for Genesys Web Engagement and contains the right versions of the required Genesys components.
2. Install the Web Engagement servers. Complete these procedures to configure and install the nodes.
3. Configure Load Balancing. This guide includes sample configurations for Apache and Nginx, although your settings may be different depending on the architecture of your cluster.
4. Configure External Cassandra. Genesys Web Engagement requires an external Cassandra ring in your production environment. This article discusses how to do that.
5. Enable SSL. You need to use a certificate issued by a third-party Certificate Authority for a production environment.
6. Configure your other required security. You can also configure TLS for Genesys and GWE servers, and

enable authentication for the Web Engagement Server, Workspace Desktop Edition, and the Engagement Strategy.

7. Configure Genesys Rules System. You need to configure Genesys Rules Authoring Tool and Genesys Rules Development Tool to work with GWE.
8. Configure your required features. You can set up the pacing algorithm, enable authentication and SSL, and more.
9. Install the Plug-in for Workspace Desktop Edition. You will need this plug-in to enable chat engagement features in Workspace Desktop Edition.
10. Develop an application. You're now ready to develop and customize your production Web Engagement application.

Prerequisites

Hardware Requirements

See Sizing for details.

OS Requirements

See the Operating Systems section of the Genesys Web Engagement page in the *Genesys Supported Operating Environment Reference Guide* for more information on supported operating systems.

Browser Support

Web Browsers

See the Browsers section of the Genesys Web Engagement page in the *Genesys Supported Operating Environment Reference Guide* for supported browsers.

The following previously supported browsers, although tested, have been discontinued and are no longer supported by Genesys:

- Safari 3 and 5
- Firefox 3, 4, and all non-ESR versions
- Internet Explorer 7, 8, 9, and 10

Mobile Browsers

- iOS Safari
- Android Chrome

Java Requirements

For Java prerequisites, see the Prerequisites section of the Genesys Web Engagement page in the *Genesys Supported Operating Environment Reference Guide* for more detailed information and a list of all prerequisites.

DB Layer Components

Starting with GWE 8.5.2, all database layer components must be external: Cassandra, Elasticsearch, and Kibana (optionally, if you are using GWE reporting).

Component Name	Compliant Version
Cassandra	3.11
Elasticsearch	6.8
Kibana (optional, to support GWE reporting)	6.8.1

Genesys Environment

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

Mandatory Components

Component Name	Minimum Compliant Version
Orchestration Server	8.1.300.55
Stat Server	8.5.000.17
Interaction Server	8.5.100.18
Configuration Server (for UTF-8 support)	8.1.200.17
Genesys Widgets	9.0.014.04

Note: Web Engagement supports a connection to the Interaction Server Proxy as an alternative to a direct connection to Interaction Server.

Components for Application Development

The following components are mandatory to create customized Genesys Web Engagement applications:

Component Name	Minimum Compliant Version	Details
Genesys Rules Authoring Tool	8.5.303.16	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	8.1.400.05	This component must be deployed as Composer plug-in or independently in Eclipse; make

Component Name	Minimum Compliant Version	Details
		sure that settings are correct for Configuration Server and Repository Server, as detailed in Installing the GRDT Component in the Genesys Rules System Deployment Guide.
Composer	8.1.430.03	Mandatory to publish the Web Engagement rules template. This component can also be used to update or deploy routing and engagement strategies.
Genesys Administrator	8.1.300.02	Used when provisioning Web Engagement, particularly when defining roles.

Components for Interaction Management

Component Name	Workspace Desktop Edition Plug-in Version	Minimum Compliant Version
Workspace Desktop Edition	8.5.200.02	8.5.124.05

Additional Components (Optional)

Component Name	Minimum Compliant Version
CCPulse+	8.0.000.36
Pulse	9.0.000.02
Chat Server	8.1.001.41

Sizing Information

Before deploying the Genesys Web Engagement (GWE) solution to your production site, you must estimate the size of solution that will be able to handle the expected user load. Genesys recommends that you download the **GWE Sizing Calculator**, an Excel workbook that you can use to help calculate the number of Web Engagement Server nodes required for your production deployment.

Click HERE to download the **GWE Sizing Calculator**. See download tips if the download was not started automatically.

The process of estimation starts from input values, usually given in the terms of business operations; for example, daily visit rates, or page view rates. Using some math, and having in mind the workflow that is applied to the input traffic, you can then produce the expected load values in terms of requests per second. Applying these values to the experimentally produced measurements, you can estimate the size of the solution required to be deployed.

> **Important**
>
> The exact deployment architecture and solution size will vary depending on your hardware equipment, and that the deployed system can be fine-tuned to get the best performance on given equipment and with given user load. However, the estimation can give some basic ideas for the deployment.

Input Data for Load Estimation

To estimate the load, you need to know the following data:

- Average visits rate and page views rate (per hour)
- Maximum visits rate and page views rate (per hour)

To estimate your disk space consumption, it is helpful to have the following information about the configuration of the solution:

- Number of business events per page or visit
- Portion of categorized events

You can use these numbers as inputs into the **GWE Sizing Calculator**.

Basic Load Estimation

Visits Rate Estimation

Having the maximum or average visits count and page view count per hour (or day), you can get visits rate and visit depth:

- Visits per second = Visits per hour / 3600 = Visits per Day / 24*3600
- Visit depth = Page views per hour / Visits per hour = Page views per day / Visits per day

Events Rate Estimation

Having the following input data about solution configuration:

- Average number of business events (timeout, search, and so on) per page – page custom events
- Average number of user events (SignIn/SignOut/UserInfo) per visit – custom visit events
- The fact that every visit produces one system event (visit started) and every page view produces two system events (PageEntered and PageExited)

The calculator can now estimate the average event rate produced by user visits:

- Events per second = Visits per second * (custom visit events + 1) + (Visits per Seconds * Visit Depth) * (custom page events + 2)

Requests Per Second

Assuming that each visit also contains two requests for loading client scripts and DSL, and every page view also contains one request for categorization info:

- Requests per second = Visits per second * 1 + (Visits per second * Visit depth) * 2 + Events per second

Peak Load Estimation

Having only average values for visit and page views, you can only predict some average load, and, consequently, only an *average* solution size that will be able to handle such load. However, the user load is not evenly distributed during the day, so you must estimate possible peak load during busy hours.

To cover that scenario, you can take your Visits as the Poisson distributed flow. To estimate the possible peak load, use the following calculation:

- Maximum Visits per Second = Average Visits per Second + 4 * SQRT (Average Visits per Second)

Now, having the Maximum Visits per Second value, it is possible to estimate the Maximum Events per Second and the Maximum Requests per Second, using the same approach as for Basic Load, but replacing average visits rate with maximum visits rate.

Load Estimation Example

Visits Rate Estimation

Having the maximum visits and page view rates per hour, you're calculating visits rate and visit’s depth. For example, having the following values:

- maximum number of visits is 3000 per hour
- maximum number of page views is 32000 per hour

The estimated visit depth is 32000/3000 = 10.7 pages.

- Average load: 3000 visits per hour = 0.83 visits per second
- Maximum load: 0.83 + 4*sqrt(0.83) = 4.5 visits per second

Events Rate Estimation

Given two business events per page, and about 10 percent of visits producing additional events:

- Average events per second = 0.83*(1 + 0.1) + (0.83*10.7)*(2 + 2) = 37 events per second

Requests per Second

Assuming that every visit generates one request for script and another for data, and every page view generates one request for categories:

- Average requests per second = 37 events + 0.83*2 + (0.83*10.7)*1 = 47 requests per second

Peak Load Estimation

Using Maximum Visits per second instead of Average Visits per Second, the following values can be calculated:

- Maximum events per second = 4.5*(1 + 0.1) + (4.5*10.7)*(2 + 2) = 197 events per second
- Maximum requests per second = 197 + 4.5*2 + (4.5*10.7)*1 = 254 requests per second

Minimum Solution Size

The solution deployed should handle all user input, and have some N+1 redundancy. Recommended solution size for Web Engagement Server—three nodes (to support high availability and load balancing). To avoid "split brain" issues, keep an uneven number of nodes in the cluster. The cluster is considered as “ready-to-work” only after a majority of nodes have joined. A majority is N / 2 + 1, where N is the total number of nodes configured for the cluster.
The minimum solution size for Cassandra and disk space requirements are described in the sections below.

Cassandra

- Cassandra—three nodes (to provide data consistency and to allow a fault tolerance of one node). The consistency level must be `LOCAL_QUORUM`.

For more help calculating the number of Cassandra nodes you need to support data consistency in the cluster, see http://www.ecyrd.com/cassandracalculator/.

Disk Space Usage

Disk space usage directly depends on the following factors:

- The event rate in the solution
- The average count of the categories applied to each event
- The replication factor specified for Cassandra (and Elasticsearch)

Glossary

Visit—series of page views from the same uniquely identified browser. Note that pages opened in different tabs or windows of the same browser will be treated as part of the same visit.

Average visits per second—service load in terms of customer site.

Request—a single request to the service. A single page view produces a series of requests to the Web Engagement Server:

- Loading of monitoring script
- Loading of categorization information
- Sending information about **PageEntered**, **PageExited**, and business events back to the Web Engagement Server

Requests per second (RPS) – actual load in terms of service performance.

Useful Links

Refer to the following links for more information about planning an external Cassandra cluster:

> **Important**
> Genesys is not responsible for the content of external internet sites.

- For help understanding the Cassandra architecture, see Architecture in brief.

- For information about hardware considerations for Cassandra nodes, see Planning a cluster deployment.
- For details about Cassandra cluster configuration, refer to Initializing a multiple node cluster (single datacenter).
- For more about Cassandra clusters and memory, see Tuning Java resources.
- For more help calculating the number of Cassandra nodes you need to support data consistency in the cluster, see Cassandra Parameters for Dummies.

Installing the Genesys Web Engagement Server

Starting with Genesys Web Engagement release 8.5.1, the GAX plug-in is replaced with the Web Engagement Management interface. This means that Web Engagement is no longer dependent on a specific GAX version, allowing you to upgrade Web Engagement independently from other GAX-related products.

> **Important**
>
> Before using Genesys Web Engagement, you must either create and deploy your own Web Engagement application or deploy the provided sample application. Otherwise, the product will not work.
>
> After Genesys Web Engagement is ready to use, but before you use it, Genesys recommends that you verify the status of your Cassandra cluster and your Elasticsearch cluster.

About the Web Engagement Cluster

Genesys Web Engagement is built on the principles of N+1 architecture. This means that:

- The cluster combines 1 or more nodes. In other words, **you must create and configure at least one node in order to use Web Engagement.** And every time you add another node, you need to create and configure it using the same steps you used to create and configure the first one.
- All nodes are treated as equivalent to each other, so that almost all of the configuration is defined in the cluster application, while the nodes only contain the options required to connect to a particular host and their cluster.
- All server connections, such as Interaction Server and Stat Server, are defined for the cluster.
  Note: Web Engagement supports a connection to the Interaction Server Proxy as an alternative to a direct connection to Interaction Server.
- Connections to external clusters, such as a cluster of Chat Servers, must be configured in the cluster application, rather than adding connections to particular nodes. In other words, if you add nodes to the cluster, you don't have to reconfigure the existing nodes and cluster, whether you are migrating from one node to two nodes, or adding a thousand nodes to the system.

These features make it easy to maintain your cluster configuration and help you avoid faulty configuration scenarios.

Deploying Web Engagement

To deploy Web Engagement, follow these steps:

1. Importing the Web Engagement Cluster Template
2. Creating the Cluster Application
3. Configuring the Cluster Application
4. Configuring a Connection to a Cluster of Chat Servers (Optional)
5. Importing the Web Engagement Client Template
6. Creating a Client Application
7. Configuring a Client Application
8. Importing the Web Engagement Server Template
9. Creating a Node Application
10. Configuring a Node Application
11. Adding Nodes to a Cluster
12. Installing the Web Engagement Server
13. Configuring alarms

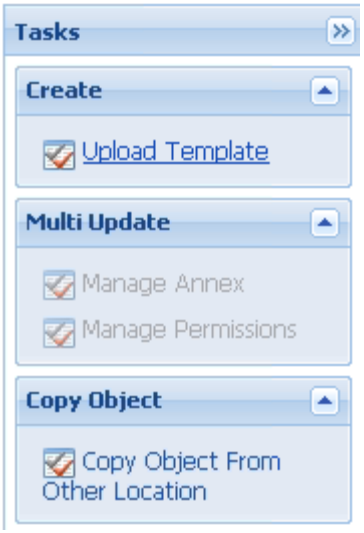
Note: For more information on how to work with templates and application objects in Genesys Administrator, consult Generic Configuration Procedures.

Importing the Web Engagement Cluster Template

Note: For more information on how to work with templates in Genesys Administrator, consult Generic Configuration Procedures.

Start

1. Open Genesys Administrator and navigate to **Provisioning > Environment > Application Templates**.
2. In the **Tasks** panel, click **Upload Template**.



Upload Template link in the Tasks panel

3. In the **Click 'Add' and choose application template (APD) file to import** window, click **Add**.
4. Browse to the **GWE_Server_Cluster.apd** file. The **New Application Template** panel opens.
5. Click **Save & Close**.

End

Creating the Cluster Application

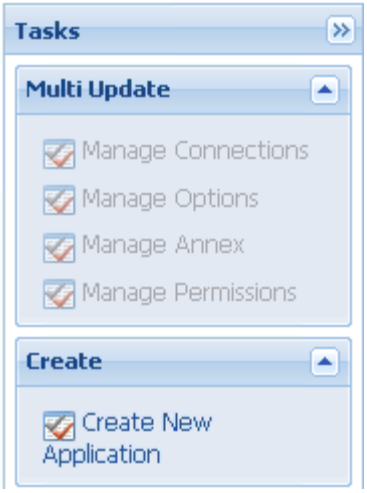
Note: For more information on how to work with application objects in Genesys Administrator, consult Generic Configuration Procedures.

Prerequisites

- You completed Importing the Web Engagement Cluster Template.

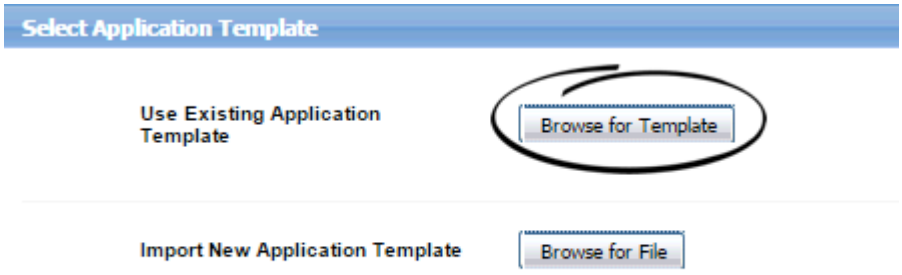
Start

1. Open Genesys Administrator and navigate to **Provisioning > Environment > Applications**.
2. In the **Tasks** panel, click **Create New Application**.



Create New Application link.

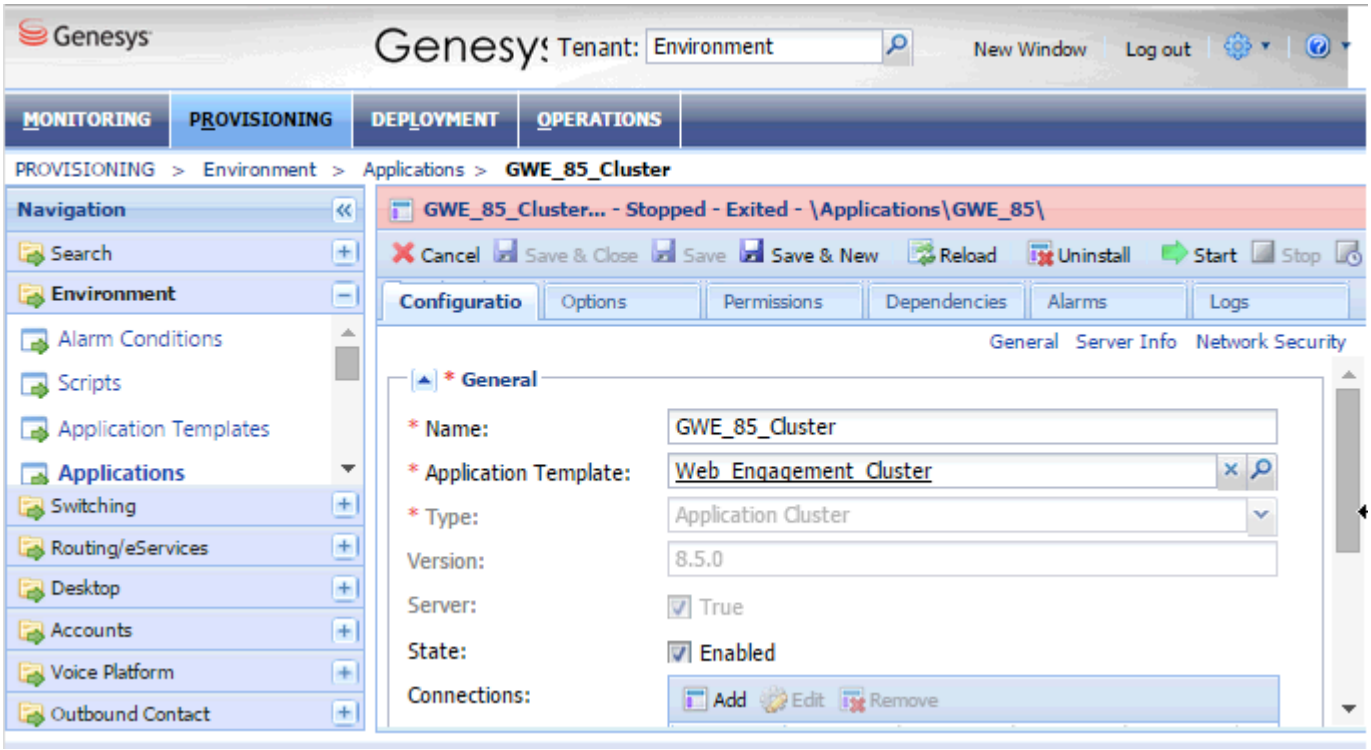
3. In the **Select Application Template** panel, click **Browse for Template**.



Browse for Template.

4. Select the Web Engagement Cluster template that you imported in Importing the Web Engagement Cluster Template. Click **OK**.
5. The template is added to the **Select Application Template** panel. Click **Next**.
6. In the **Select Metadata file** panel, click **Browse**, then click **Add**, and then select the **GWE_Server_Cluster.xml** file. Click **Open**.
7. The metadata file is added to the **Select Metadata file** panel. Click **Next**.
8. In the **Specify Application parameters** tab:
   - Enter a name for your application. For instance, `GWE_Cluster`.
   - Make sure that **State** is enabled.
   - Select the **Host** which will be used as an entry point into the Web Engagement Cluster. Typically, this is the Load Balancer host.
   - Click **Create**.
9. The **Results** panel opens.
10. Enable **Opens the Application details form after clicking 'Finish'** and click **Finish**. The Web Engagement Cluster application form opens and you can start configuring the Web Engagement Cluster

application.



Web Engagement Cluster app opened in Genesys Administrator.

End

Configuring the Cluster Application

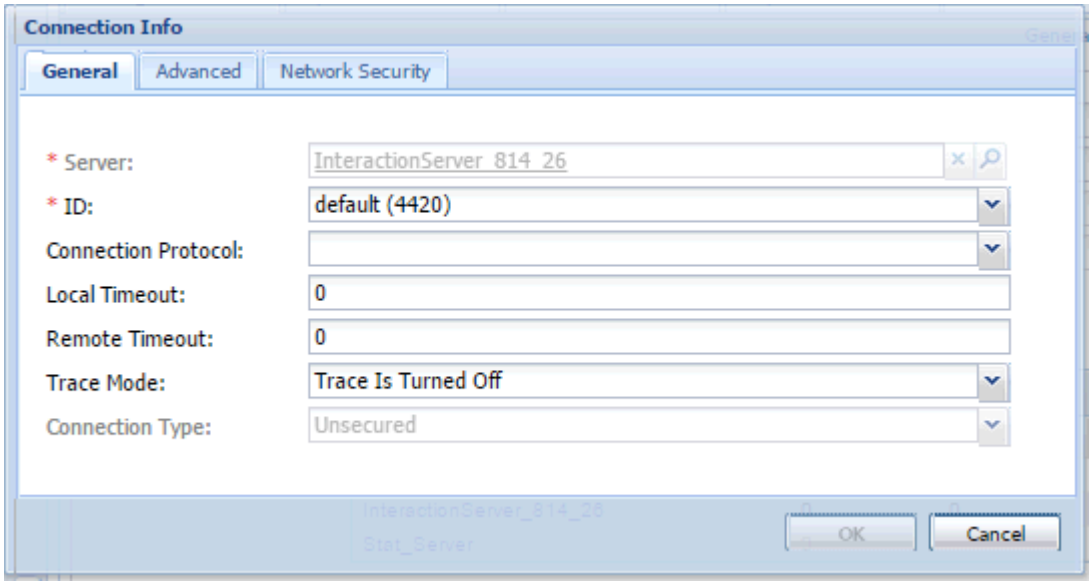
Note: For more information on how to work with application objects in Genesys Administrator, consult Generic Configuration Procedures.

Prerequisites

- You completed Creating the Cluster Application.

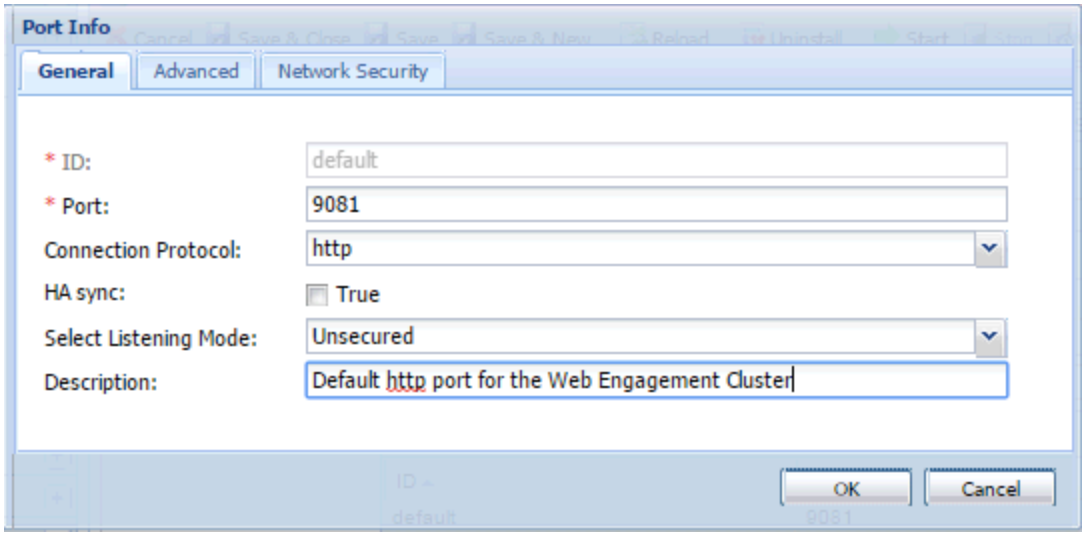
Start

1. If your Cluster application form is not open in Genesys Administrator, navigate to **Provisioning > Environment > Applications**. Select the application defined for the Web Engagement Cluster and click **Edit...**.
2. In the **Connections** section of the **Configuration** tab, click **Add**. The **Browse for applications** panel opens. Select the Genesys application defined for **Interaction Server**, then click **OK**.
   Note: Web Engagement supports a connection to the Interaction Server Proxy as an alternative to a direct connection to Interaction Server.



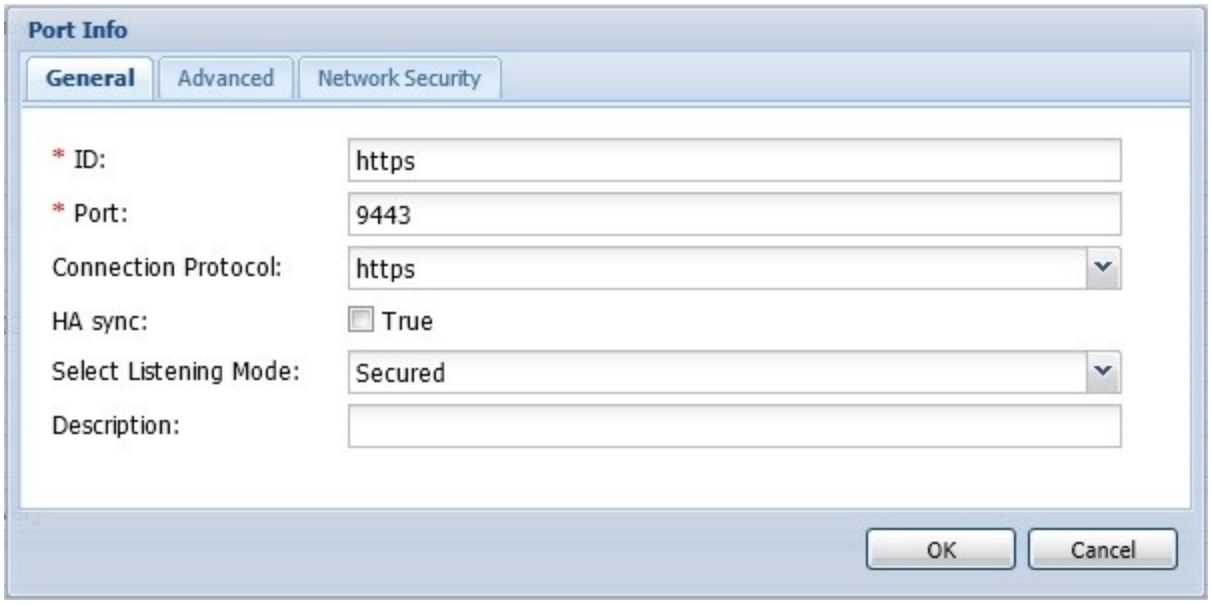
The port **ID** is set to the `default` port.

3. Repeat the previous step for **Stat Server**. Optionally, you can also add a connection to Message Server (to apply the **network** logging option).
4. If you are using legacy Web Engagement chat channel, configure a connection to the Chat Server or to a cluster of Chat Servers. If you are using a single Chat Server, you must set your port ID to `webapi` when specifying your connection to Chat Server. For information on how to connect a cluster of Chat Servers, see Configuring a Connection to a Cluster of Chat Servers (Optional).
5. Expand the **Server Info** pane.
6. In the Tenant section, click **Add** and select your tenant. For instance, `Environment`. Click **OK**.
7. If your **Host** is not defined, click the lookup icon to browse to the hostname of your application.
8. In the Listening Ports section, create the default port by clicking **Add**. The **Port Info** dialog opens.
   Note: You must specify the default port.
   - Enter the **Port**. For instance, `9081`.
   - Choose `http` for the **Connection Protocol**.
   - Click **OK**. The HTTP port with the default identifier appears in the list of **Listening ports**.



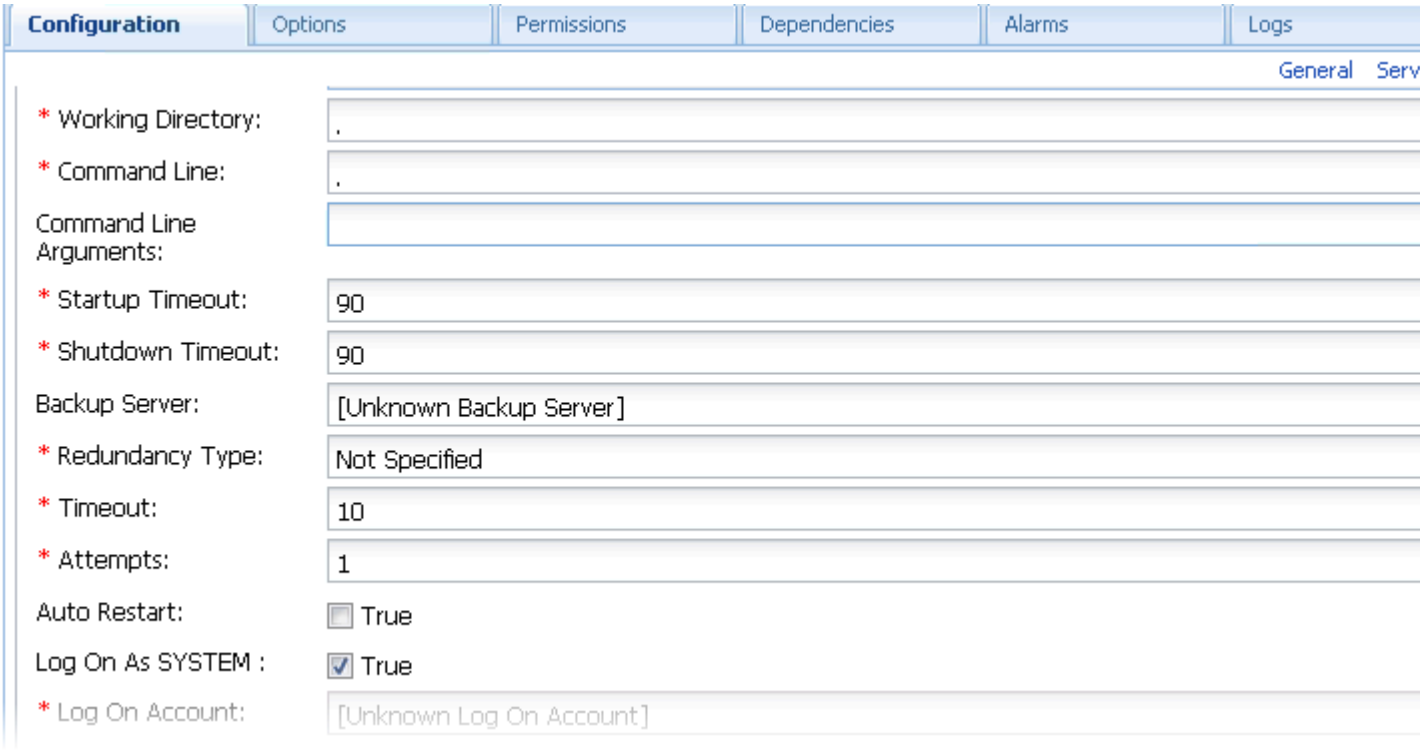
Default HTTP port

9. Create the port with the `server/data/rules/deploy` identifier by clicking **Add**.
   - Enter `server/data/rules/deploy` for the ID.
   - Enter a port value that is **different** from the one you chose for the default port. For instance, `9083`.
   - Select `http` for the **Connection Protocol**.
   - Click **OK**. The HTTP port with `server/data/rules/deploy` ID appears in the list of Listening ports.
10. Create the port with the `ui` identifier by clicking **Add**.
    - Enter `ui` for the ID.
    - Enter a port value that is different from the ports already assigned. For instance, `9082`.
    - Select `http` for the **Connection Protocol**.
    - Click **OK**. The HTTP port with the `ui` ID appears in the list of Listening ports.
11. Optionally, you can add a secure listening port for authenticated users, secured connections, and secure chat. Click **Add**.The **Port Info** dialog opens.
    - Enter `https` for the ID field. This specific ID is required in GWE.
    - Enter the **Port**. For instance, `9443`.
    - Enter `https` for the **Connection Protocol**.
    - Choose `Secured` for the **Listening Mode**.
    - Click **OK**.



Secure listening port

12. Ensure the **Working Directory** and **Command Line** fields contain "." (period).



Commands

13. Click **Save**.
14. The **Confirmation** dialog for changing the application's port opens. Click **Yes**.
15. Click **Save & Close**. If the **Confirmation** dialog opens, click **Yes**.

End

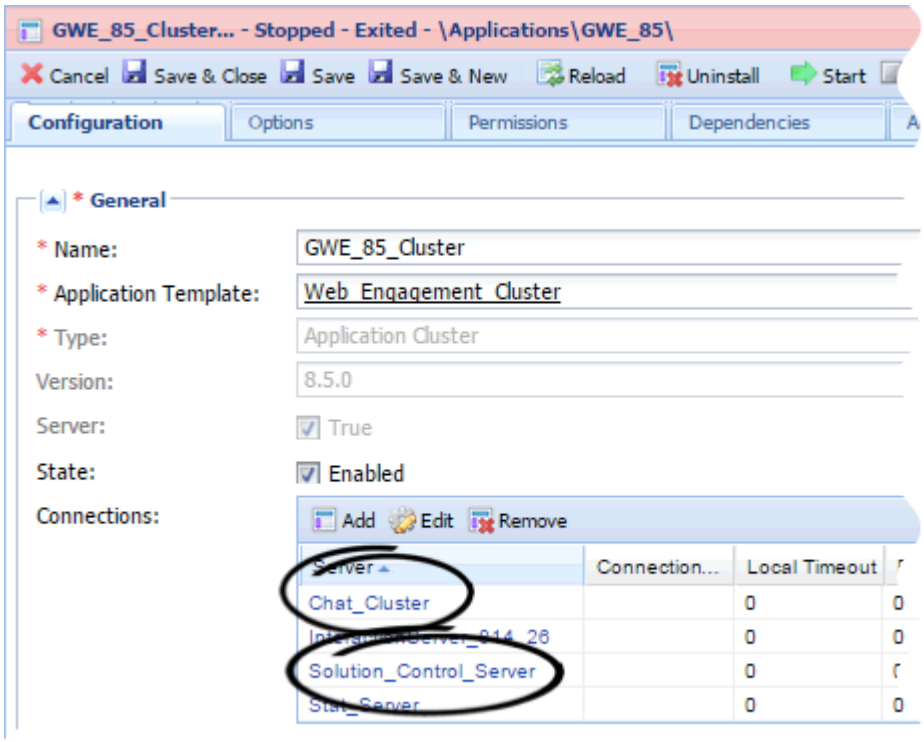
Configuring a Connection to a Cluster of Chat Servers (Optional)

Note: The Web Engagement "legacy" chat channel is deprecated as of version 8.5. Genesys recommends using the CX Widgets-based chat. **Note:** If you are using a legacy Web Engagement chat channel and running Web Engagement in a production environment, Genesys recommends connecting the Web Engagement Cluster to a cluster of Chat Servers rather than to a single instance of Chat Server.

Complete the steps below to configure a cluster of Chat servers for the Cluster.

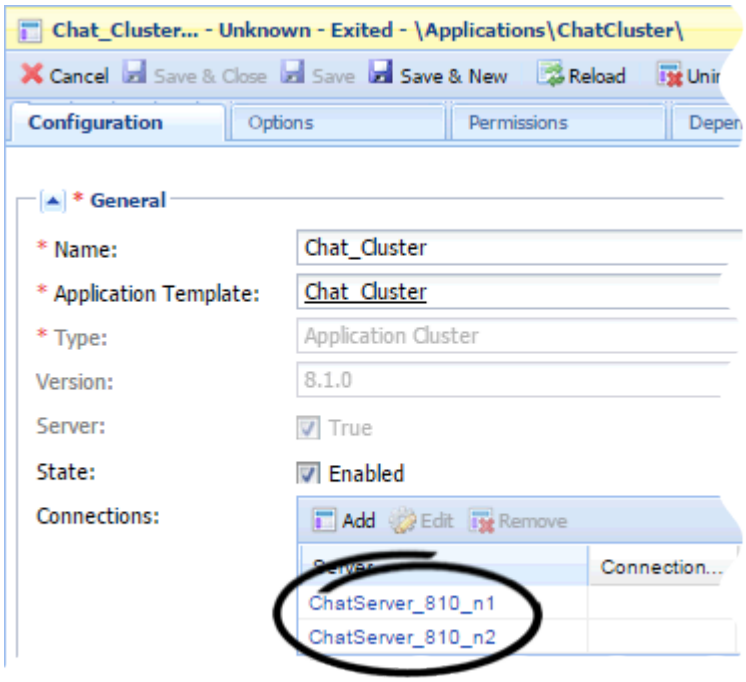
Start

1. In Genesys Administrator, create an application with a type of **Application Cluster**. This example uses an Application Cluster app called **Chat Cluster**.
2. Navigate to **Provisioning > Environment > Applications**, select your Cluster application, and click **Edit**.
3. In the **Connections** section, add a connection to Solution Control Server and an Application Cluster application that has connections to one or more Chat servers.



The Cluster has a connection to the **ChatCluster** Application Cluster application.

4. Click **Save & Close**
5. Open your Application Cluster application.
6. In the **Connections** section, add connections to one or more Chat Servers, using a port ID of **webapi** for each connection.



The **ChatCluster** application has connections to two Chat servers.

7. Click **Save & Close**.

End

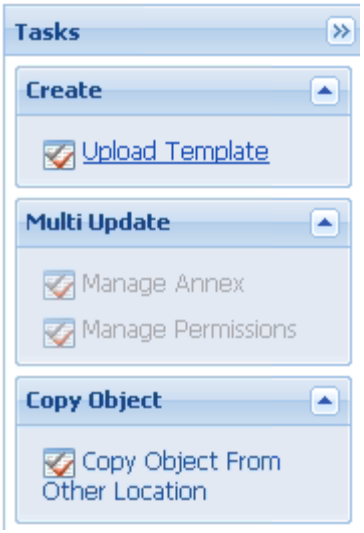
Importing the Web Engagement Client Template

Prerequisites

- You completed Configuring the Cluster Application.

Start

1. Open Genesys Administrator and navigate to **Provisioning > Environment > Application Templates**.
2. In the **Tasks** panel, click **Upload Template**.



Upload Template link in the Tasks panel

3. In the **Click 'Add' and choose application template (APD) file to import** window, click **Add**.
4. Browse to the **GWE_Client.apd** file, available in the **templates** directory of your installation CD. The **New Application Template** panel opens.
5. Click **Save & Close**.

End

Creating a Client Application

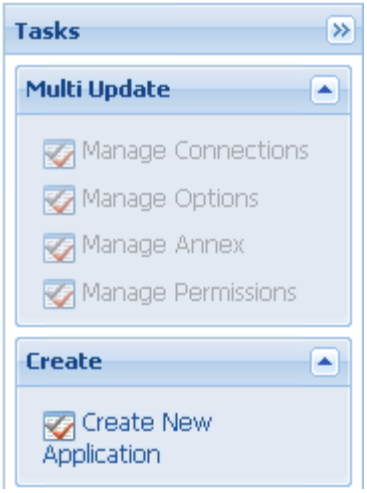
To support the Web Engagement Management UI, you must create and configure Web Engagement Client application using the instructions in configuring a client application. The client application will be used to provide write-access to the Configuration Server.

Prerequisites

- You completed Importing the Web Engagement Client Template.

Start

1. Open Genesys Administrator and navigate to **Provisioning > Environment > Applications**.
2. In the **Tasks** panel, click **Create New Application**.



Create New Application link.

3. In the **Select Application Template** panel, click **Browse for Template** and select the Web Engagement Client template that you imported in Importing the Web Engagement Client Template. Click **OK**.
4. The template is added to the **Select Application Template** panel. Click **Next**.
5. In **Specify Application parameters**:
   - Enter a name for your application. For instance, `GWE_Client`.
   - Make sure that **State** is enabled.
   - Select the **Host** on which the node will reside.
   - Click **Create**.
6. The **Results** panel opens. Click the **Finish** button. The Web Engagement Client application form opens.
7. Click **Save & Close**. If the **Confirmation** dialog opens, click **Yes**.
8. Enable **Opens the Application details form after clicking 'Finish'** and click **Finish**. The GWE_Client application form opens and you can start configuring the client application.

End

Configuring a Client Application

Prerequisites

- You completed Creating a Client Application.

Start

1. If your client application form is not open in Genesys Administrator, navigate to **Provisioning > Environment > Applications**. Select the application defined for the client and click **Edit...**.

2. In the **Connections** section of the **Configuration** tab, click **Add**. The **Browse for applications** panel opens. Select the Web Engagement Cluster application you defined above, then click **OK**.

End

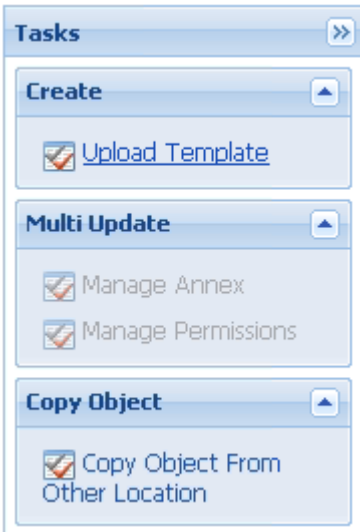
Importing the Web Engagement Server Template

Prerequisites

- You completed Configuring the Cluster Application.

Start

1. Open Genesys Administrator and navigate to **Provisioning > Environment > Application Templates**.
2. In the **Tasks** panel, click **Upload Template**.



Upload Template link in the Tasks panel

3. In the **Click 'Add' and choose application template (APD) file to import** window, click **Add**.
4. Browse to the **GWE_Server.apd** file available in the **templates** directory of your installation CD. The **New Application Template** panel opens.
5. Click **Save & Close**.

End

Creating a Node Application

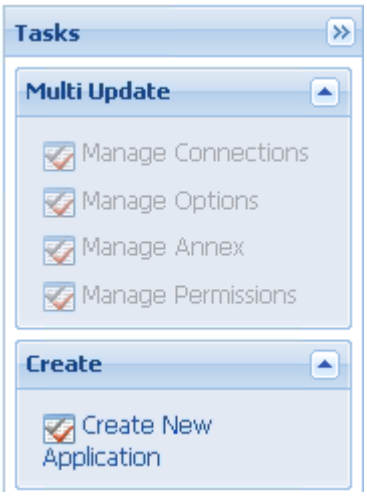
You must create and configure every node that you add to the cluster, using the instructions in this section and the next one.

Prerequisites

- You completed Importing the Web Engagement Server Template.

Start

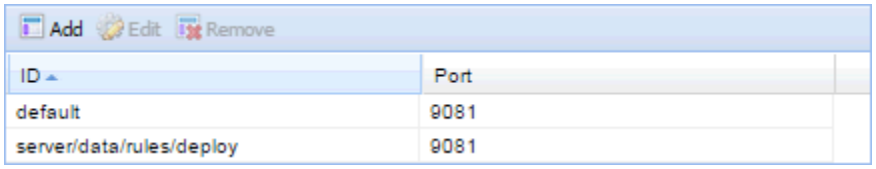
1. Open Genesys Administrator and navigate to **Provisioning > Environment > Applications**.
2. In the **Tasks** panel, click **Create New Application**.



Create New Application link.

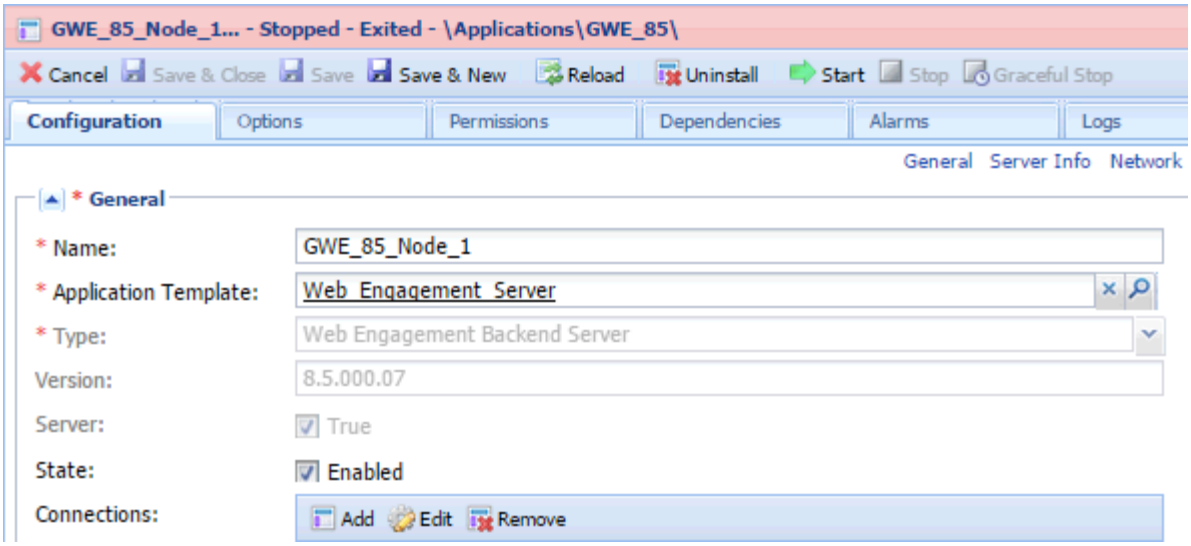
3. In the **Select Application Template** panel, click **Browse for Template** and select the Web Engagement Server template that you imported in Importing the Web Engagement Server Template. Click **OK**.
4. The template is added to the **Select Application Template** panel. Click **Next**.
5. In the **Select Metadata file** panel, click **Browse**, then click **Add**, and select the **GWE_Server.xml** file. Click **Open**.
6. The metadata file is added to the **Select Metadata file** panel. Click **Next**.
7. In **Specify Application parameters**:
   - Enter a name for your application. For instance, `GWE_Node_1`.
   - Make sure that **State** is enabled.
   - Select the **Host** on which the node will reside.
   - Click **Create**.
8. The **Results** panel opens. Click the **Finish** button. The Web Engagement Node application form opens and you can start configuring the Web Engagement Node application.
9. Expand the **Server Info** pane.

10. In the Listening Ports section, create the default port by clicking **Add**. The **Port Info** dialog opens.
    - Enter the **Port**. For instance, `9081`.
    - Choose `http` for the **Connection Protocol**.
    - Click **OK**. The HTTP port with the default identifier appears in the list of **Listening ports**.
11. Create the port with the `server/data/rules/deploy` identifier by clicking **Add**.
    - Enter `server/data/rules/deploy` for the ID.
    - Enter a port value that is **different** than the one you chose for the default port. For instance, `9083`.
    - Select `http` for the **Connection Protocol**.
    - Click **OK**. The HTTP port with `server/data/rules/deploy` ID appears in the list of Listening ports.



Node listening ports

12. Create the port with the `ui` identifier by clicking **Add**.
    - Enter `ui` for the ID.
    - Enter a port value that is different than the ports already assigned. For instance, `9082`.
    - Select `http` for the **Connection Protocol**.
    - Click **OK**. The HTTP port with `ui` ID appears in the list of Listening ports.
13. Click **Save**.
14. A **Confirmation** dialog opens. Click **Yes**.
15. Click **Save & Close**. If the **Confirmation** dialog opens, click **Yes**.
16. Enable **Opens the Application details form after clicking 'Finish'** and click **Finish**. The GWE_Node_1 application form opens and you can start configuring the node application.



Node app opened in Genesys Administrator.

End

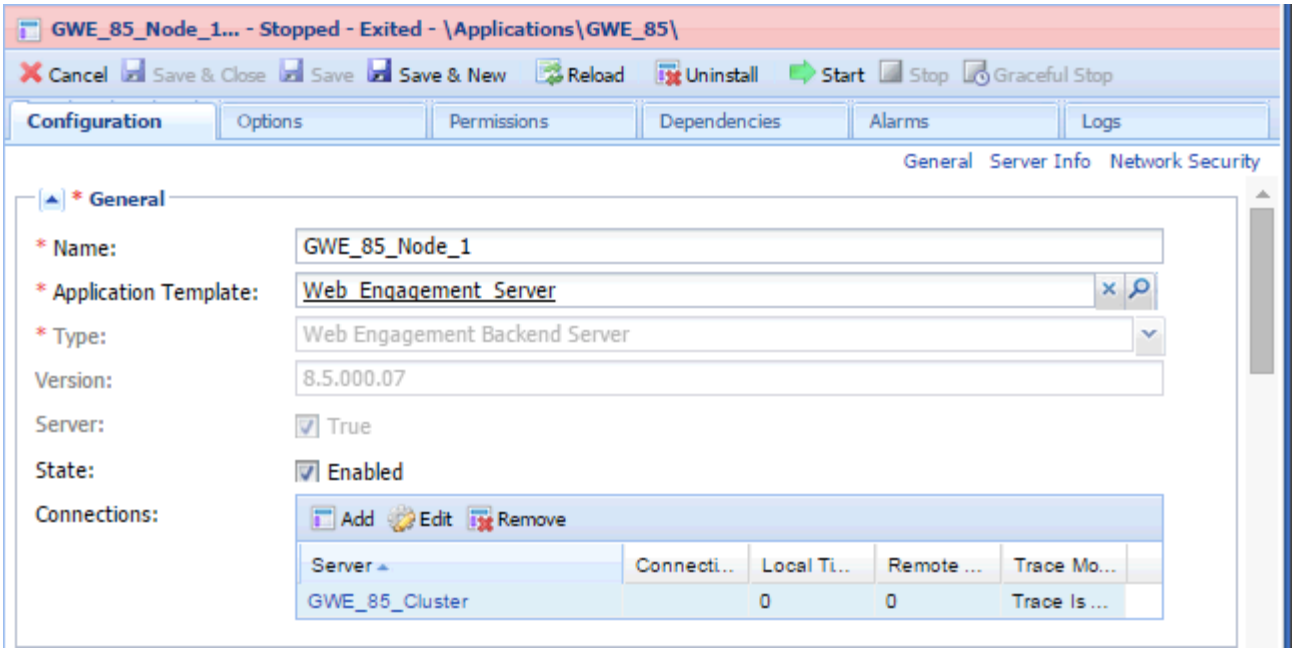
Configuring a Node Application

Prerequisites

- You completed Creating a Node Application.

Start

1. If your node application form is not open in Genesys Administrator, navigate to **Provisioning > Environment > Applications**. Select the application defined for the node and click **Edit...**.
2. In the **Connections** section of the **Configuration** tab, click **Add**. The **Browse for applications** panel opens. Select the Web Engagement Cluster application you defined above, then click **OK**.



Node connection to Cluster

3. Expand the **Server Info** pane.
4. In the Tenant section, click **Add** and select your tenant. For instance, `Environment`. Click **OK**.
5. If your **Host** is not defined, click the lookup icon to browse to the hostname of your application.
6. The **Confirmation** dialog for changing the application's port opens. Click **Yes**.
7. (Optional) 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\GWE_Node1` to force the system to write logs in the console and in a file called `GWE_Node1.log`.

End

Adding Nodes to a Cluster

Prerequisites

You completed:

- Creating the Cluster Application.
- Configuring the Cluster Application
- Creating a Node Application
- Configuring a Node Application

Note: A single-node configuration works well in a lab environment, but in production, you need more than one node, in order to provide high availability. And every time you establish a new node, you must complete the following steps in order to create and configure it.

Genesys recommends that your cluster contain at least three Web Engagement nodes so that it can survive the loss of one node. To avoid "split brain" issues, keep an uneven number of nodes in the cluster. The cluster is considered as “ready-to-work” only after a majority of nodes have joined. A majority is N / 2 + 1, where N is the total number of nodes configured for the cluster.

Start

1. Follow the instructions above for Creating a Node Application, but use a different name for the new node.
2. Configure the new node application, as shown above, but point to a different server address.

End

Important: If you use more than one node, you need to set up Load Balancing in your environment.

Installing the Web Engagement Server

Install the Web Engagement Server on Windows or Linux.

Note: For more information on how to install apps that you have configured in Genesys Administrator, consult Generic Installation Procedures.

Windows

Prerequisites

- Configuring a Node Application

Start

1. In your installation package, locate and double-click the **setup.exe** file. The Install Shield opens the welcome screen.
2. Click **Next**. The **Connection Parameters to the Configuration Server** screen appears.
3. Under **Host**, specify the host name and port number where Configuration Server is running. (This is the main "listening" port entered in the **Server Info** tab for Configuration Server.)
4. Under **User**, enter the user name and password for logging on to Configuration Server.
5. Click **Next**. The **Select Application** screen appears.
6. Select the Web Engagement Server Application—that is, the Node app you created above—that you are installing. The **Application Properties** area shows the **Type**, **Host**, **Working Directory**, **Command Line executable**, and **Command Line Argument**s information previously entered in the **Server Info** and **Start Info** tabs of the selected Application object.

7. Click **Next**. The **Choose Destination Location** screen appears.
8. Under **Destination Folder**, keep the default value or browse for the desired installation location.
9. Click **Next**. The **Backup Configuration Server Parameters** screen appears.
10. If you have a backup Configuration Server, enter the **Host name** and **Port**.
11. Click **Next**. The **Ready to Install** screen appears.
12. Click **Install**. The Genesys Installation Wizard indicates it is performing the requested operation for Web Engagement Server. When through, the **Installation Complete** screen appears.
13. Click **Finish** to complete your installation of the Web Engagement Server.

- **Before using Genesys Web Engagement, you must either create and deploy your own Web Engagement application or deploy the provided sample application.**

End

Linux

Prerequisites

- Configuring a Node Application

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 Server application—that is, the Node app you created above—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 you have a backup Configuration Server, enter the Host name and Port.
7. If the installation is successful, the console displays the following message:
   `Installation of Genesys Web Engagement Server, version 8.5.x has completed successfully.`

- **Before using Genesys Web Engagement, you must either create and deploy your own Web Engagement application or deploy the provided sample application.**

End

Configuring alarms

Genesys recommends that you tune the following Web Engagement alarms:

- Incorrect load balancer routing
- Event Duration
- VisitProfile cache size
- Events cache size
- DroolsSession cache size
- Heap Memory Usage
- GC Latency

Next Steps

- Before using Genesys Web Engagement, you must either create and deploy your own Web Engagement application or deploy the provided sample application.
- At that point, you can Configure Genesys Rules System to work with GWE.

Configuring Genesys Rules System

Complete the procedures in the tabs below to tune Genesys Rules System to work with Web Engagement.

Genesys Rules Authoring Tool

Configuring Genesys Rules Authoring Tool

Prerequisites

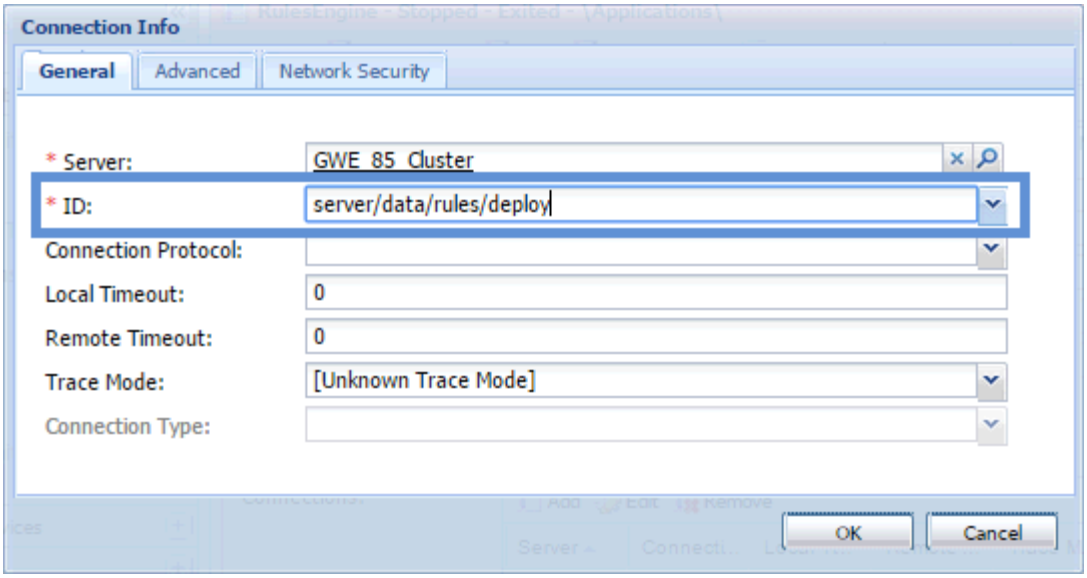
- Your environment includes Genesys Rules Authoring Tool (GRAT). See Genesys environment prerequisites for compliant versions. For more information about installing GRAT, refer to the Genesys Rules System Deployment Guide.

Start

1. In Genesys Administrator, navigate to **Provisioning > Environment > Applications**, select the GRAT Server application (the application with type `Business Rules Application Server`), and click **Edit**.
2. In the Connections section, click **Add...**.
3. Select the Genesys Web Engagement Cluster and click **OK**.

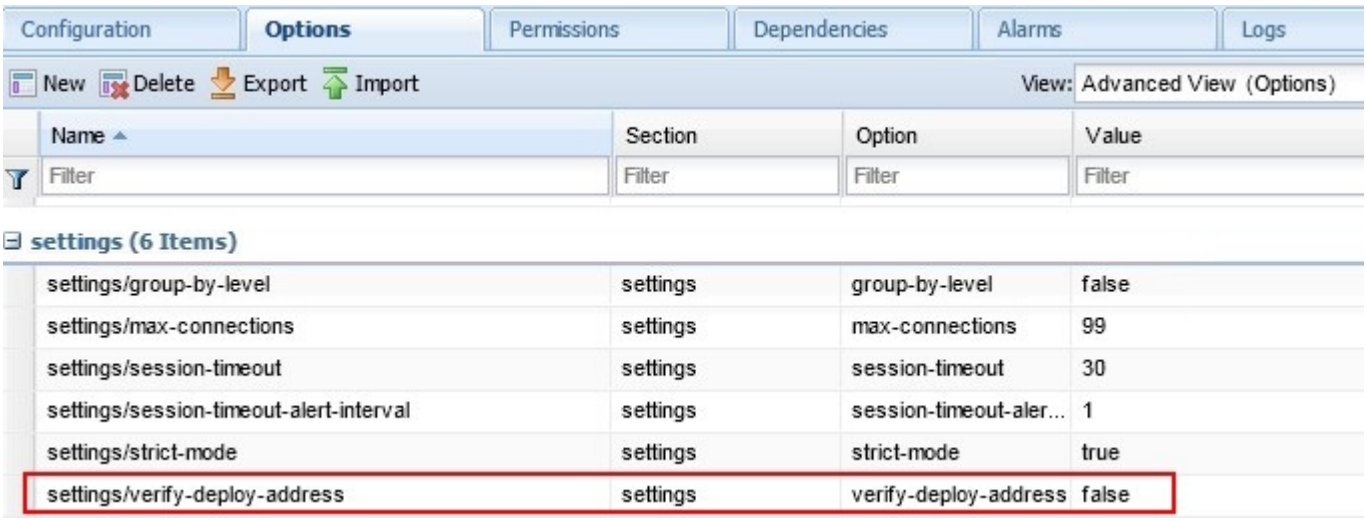
   Important: If your version of GRAT does not recognize connections with your Application Cluster object, you can connect to any of your Web Engagement Server node applications instead.

   You can determine whether your Application Cluster is recognized by seeing whether it shows up as a valid location in the rules package deployment scenario.
4. Select the Cluster in the list of connections and click **Edit**.
5. In the **Connection Info** window, select the correct port for the ID field, that is, `/server/data/rules/deploy`. Click **OK**.



ID is set to the correct port.

6. Select the **Options** tab.
7. In the **[settings]** section, set **verify-deploy-address** to `false`. You must set this option because in Genesys Web Engagement, rule packages are deployed to the Cluster, not to Genesys Rules Engine. When set to `false`, this option prevents GRS from trying to verify that the server deploying the rules is Genesys Rules Engine.



verify-deploy-address is set to `false`.

8. In the **Permissions** 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).
9. Click **Save & Close**.

End

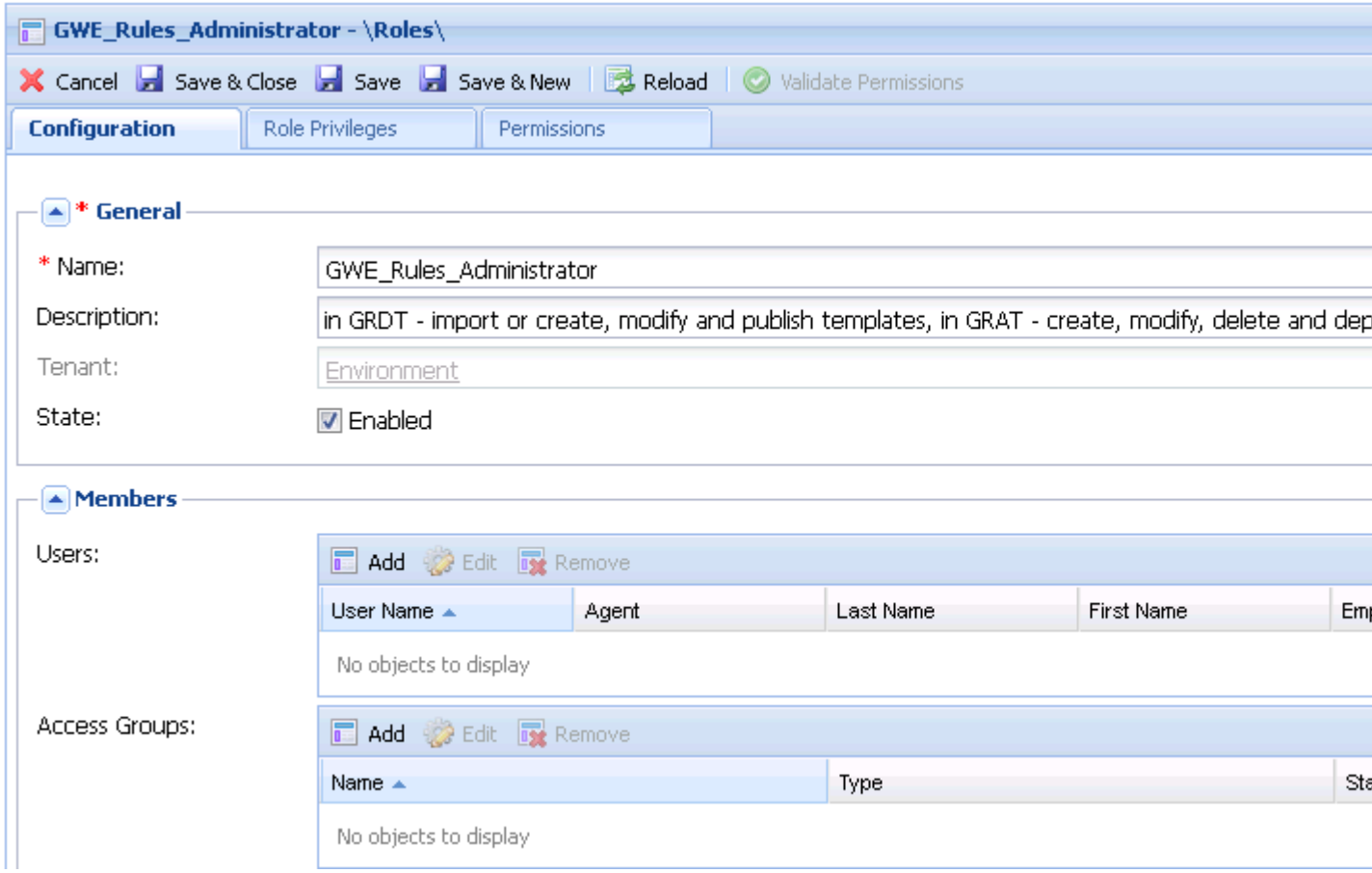
(Optional) Configuring Roles Settings for Rules Management

You should complete this procedure if you need to import permissions to enable a user to create rules for Genesys Web Engagement. Once roles are imported, you can assigned them to the user who publishes the rule templates and creates rules for GWE.

You should not complete this procedure if you have already created a user who has permissions to create rules packages in Genesys Rules Authoring (as described in the "Role-based Access Control" chapter of the Genesys Rules System Deployment Guide).

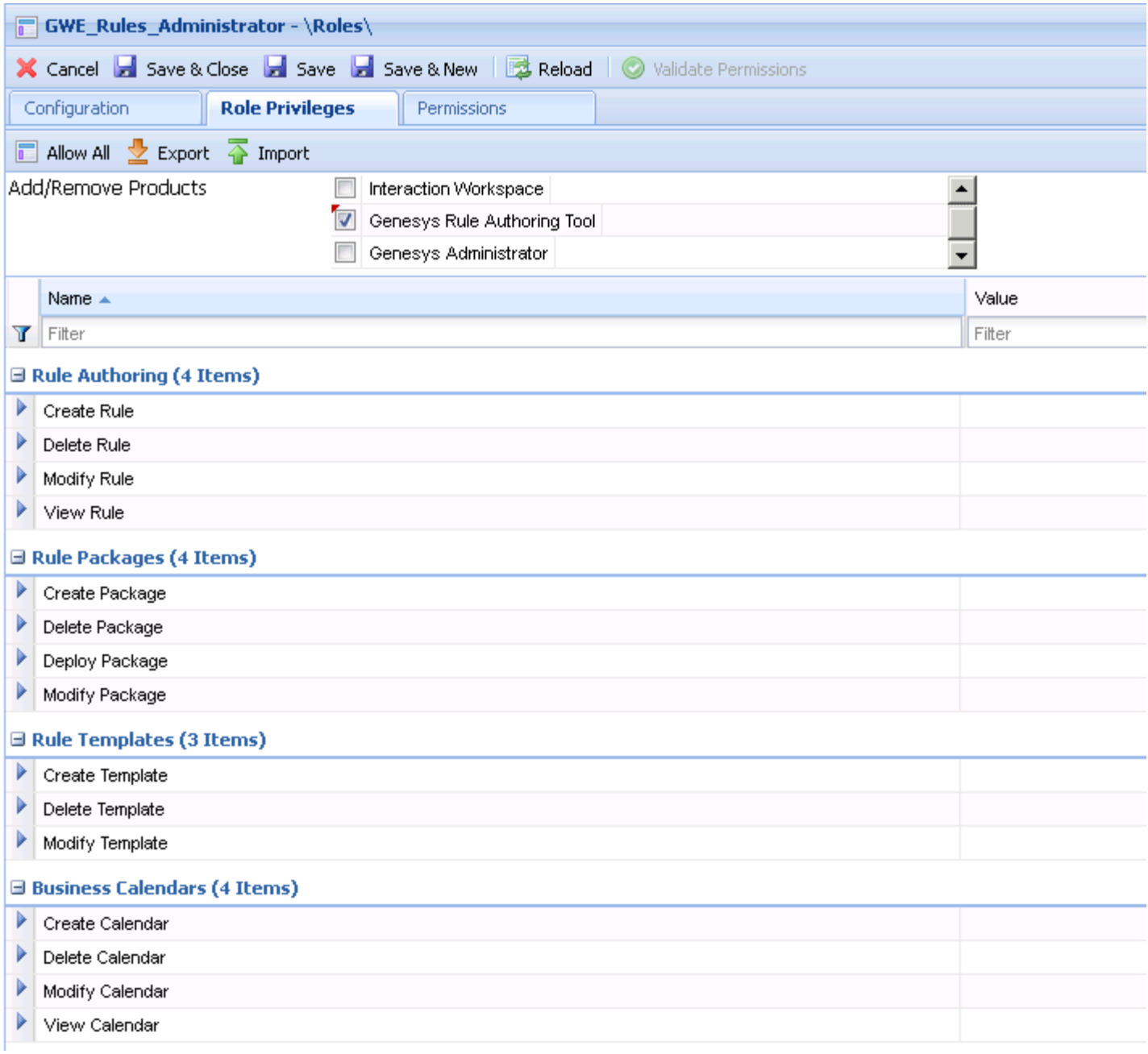
Start

1. In Genesys Administrator, navigate to **Provisioning > Accounts > Roles** and click **New...**.
2. Enter a name for the new role. For example, `GWE_Rules_Administrator`.
3. In the Members section, you can specify who should have this role. Click **Add** to add as many access groups or users as you need.



New Role

4. Select the **Role Privileges** tab and select `Genesys Rules Authoring Tool`. The privileges for GRAT are added to the GWE_Rules_Administrator role.



Imported Role Privileges

5. Click **Save & Close**.

End

Members of the GWE_Rules_Administrator role can now do the following:

- Create, modify, and delete rules
- Create, modify, delete, and deploy rule packages

- Create, modify, and publish CEP rule templates

Genesys Rules Development Tool

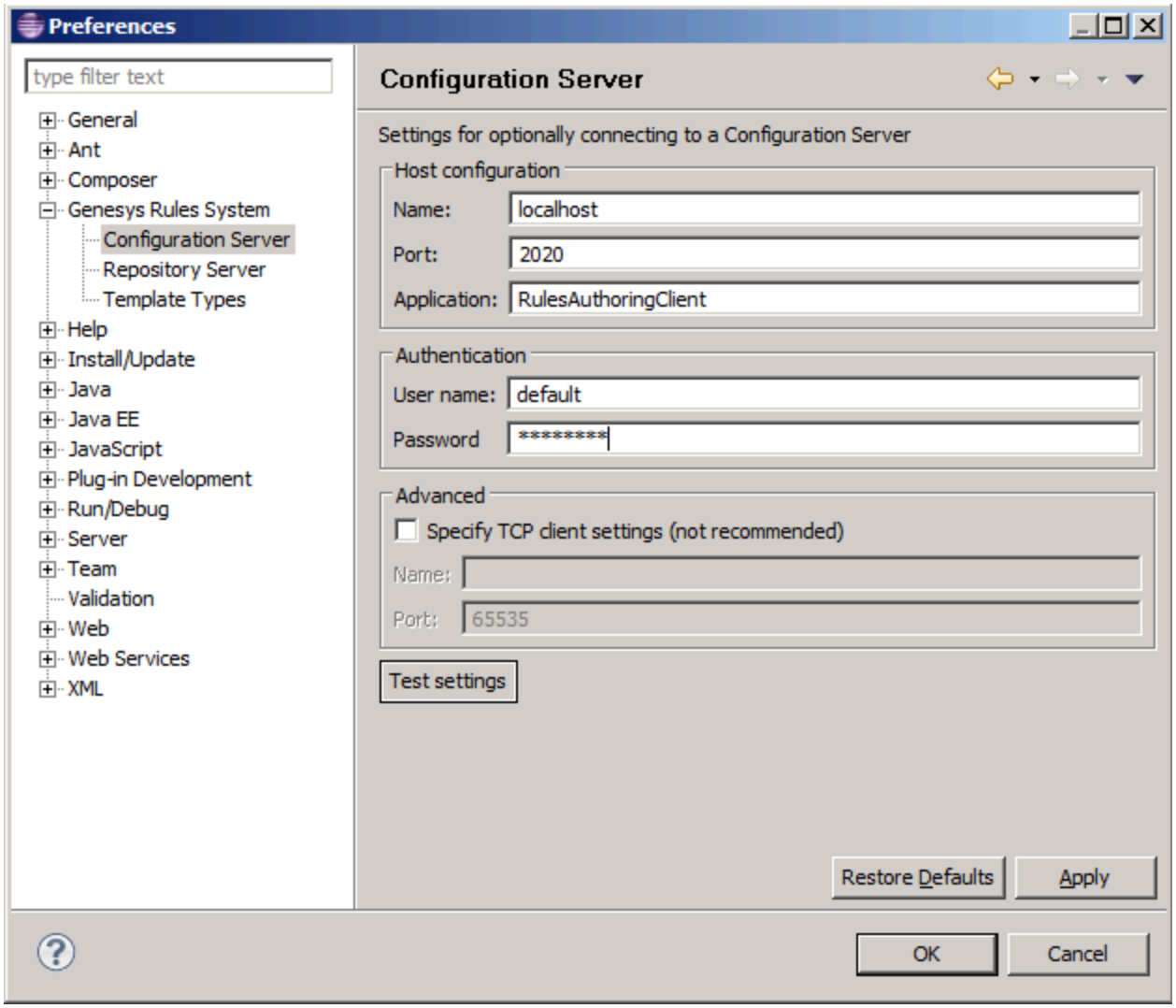
Configuring Genesys Rules Development Tool

Prerequisites

- Your environment includes Genesys Rules Development Tool (GRDT), deployed as a Composer plug-in. See Genesys environment prerequisites for compliant versions. For more information about installing GRDT, refer to Installing the GRDT Component in the the Genesys Rules System Deployment Guide.
- You enabled the Galileo update site in GRDT, as described in Installing the GRDT Component.

Start

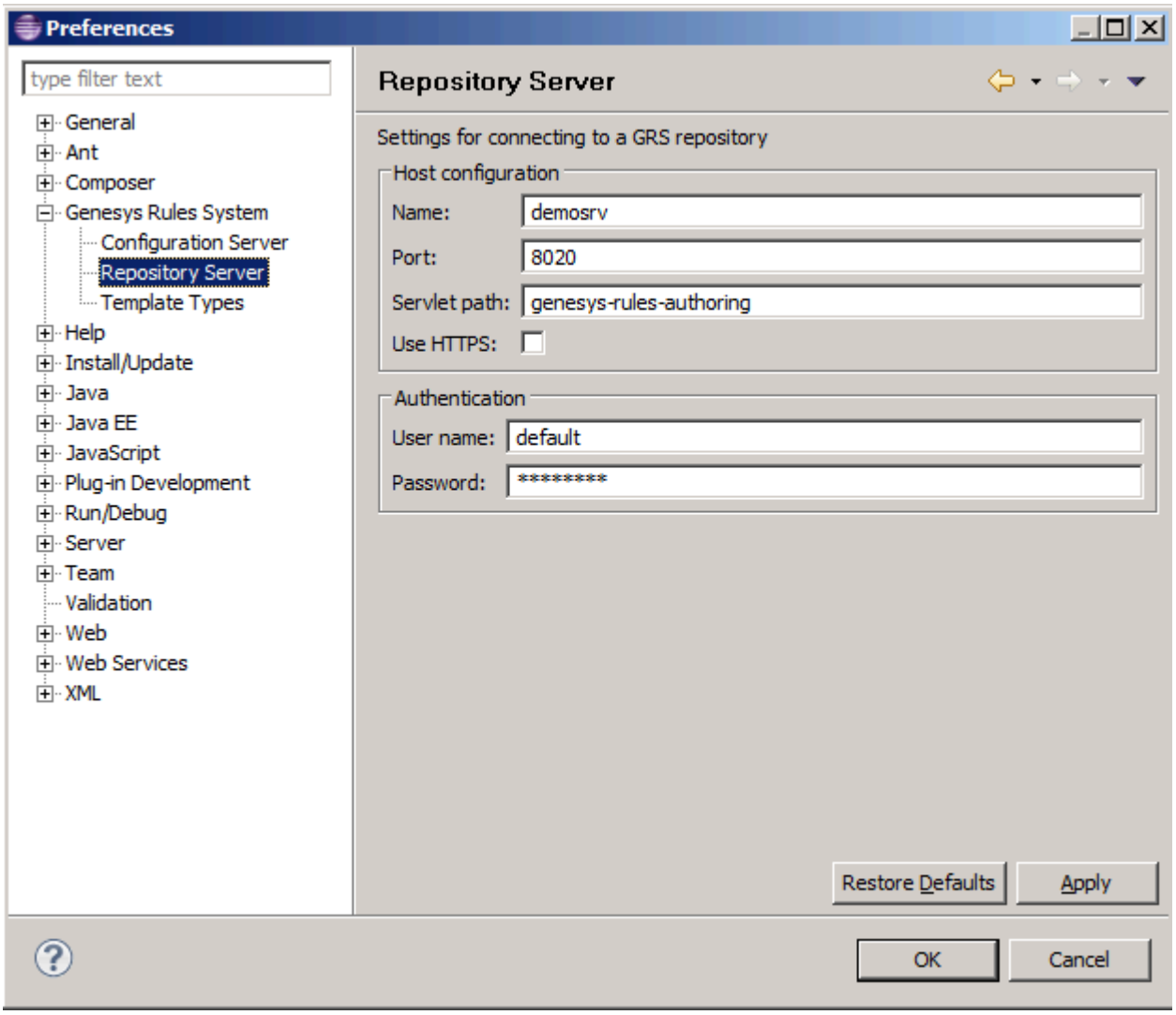
1. Open Genesys Rules Development Tool by starting Composer.
2. Navigate to **Window > Preferences**. The **Preferences** window opens.
3. Navigate to **Genesys Rules System > Configuration Server**. Edit the settings.
   - Enter the Configuration Server host name. For instance, `localhost`.
   - Enter the Configuration Server port. For instance, `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 Configuration Server and click **Apply**.



GRDT settings for Configuration Server.

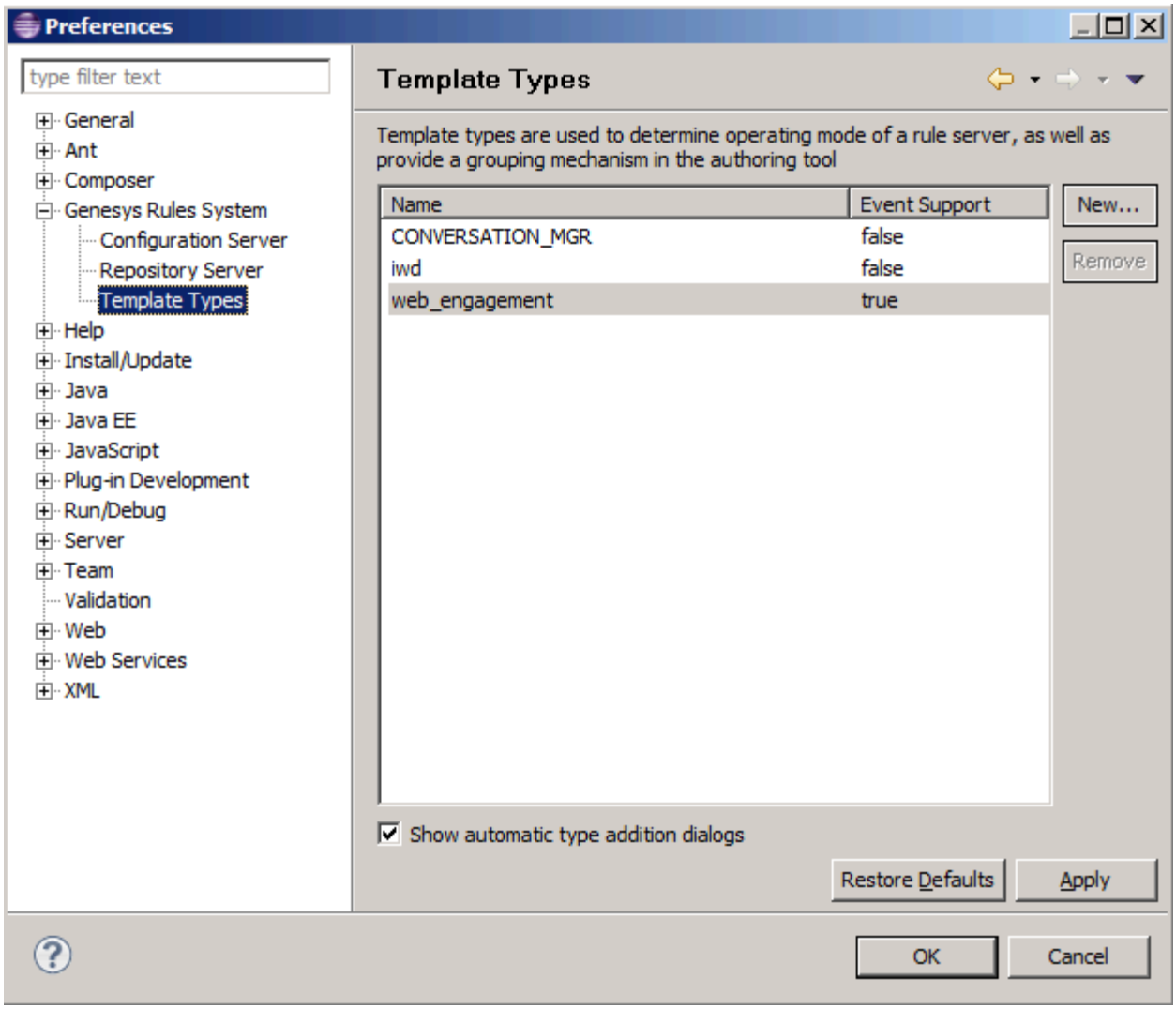
4. Navigate to **Genesys Rules System > Repository Server**. Edit the settings.
   - Enter the Repository Server host name. This is the name of the host specified for the GRS application (application with type Business Rules Execution Server) in the Genesys Configuration Layer. For instance, `localhost`.
   - Enter the Repository Server port. This is the port with the ID `genesys-rules-engine` that is specified for the GRS application (with type Business Rules Execution Server) in the Genesys Configuration Layer. For instance, `8020`.
   - Enter the Servlet path as `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 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.

5. Click **Apply**.



GRDT settings for the Repository Server.

6. Navigate to **Genesys Rules System > Template Types**. If it is not present, add the **web_engagement** template type and set **Event Support** to `true`. Click **Apply**.



Template types in Composer

7. Click **OK**.

End

Next Steps

- Configure the Generic Cassandra Settings for your Web Engagement Server.

Configuring External Cassandra and Elasticsearch

> **Important**
>
> Starting with version 8.5.2, Genesys Web Engagement supports only external Cassandra and Elasticsearch clusters.

To set up an external Cassandra and Elasticsearch clusters, follow these steps:

- Create and configure Cassandra access points in Genesys Administrator for each *seed* node of the external Cassandra cluster.
- Deploy Cassandra.
- Create and configure Elasticsearch access points in Genesys Administrator for Load Balancer, used as a gateway to Elasticsearch (if you configured Load Balancer) or for an Elasticsearch node, which will be considered as a *seed* node.
- Deploy Elasticsearch.

Note: Genesys recommends that you use Linux when deploying external Cassandra and Elasticsearch clusters. Cassandra nodes could be collocated on the same host with Elasticsearch nodes.

Note: You must synchronize the time on all hosts that contain Cassandra nodes. Failure to do this may lead to problems with removing data by TTL.

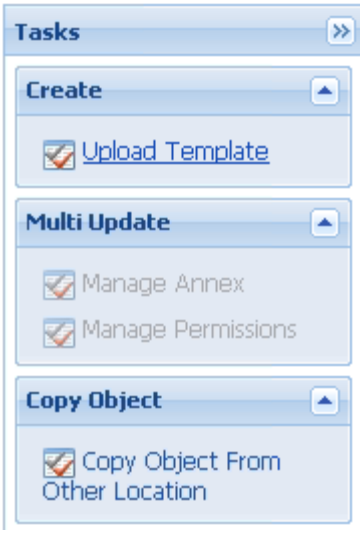
Note: If you plan to establish secure communications with your Cassandra cluster, Genesys recommends that you carefully evaluate the related security considerations.

Importing the Cassandra Resource Access Point Template

Note: For more information on how to work with templates in Genesys Administrator, consult Generic Configuration Procedures.

Start

1. Open Genesys Administrator and navigate to **Provisioning > Environment > Application Templates**.
2. In the **Tasks** panel, click **Upload Template**.



Upload Template link in the Tasks panel

3. In the **Click 'Add' and choose application template (APD) file to import** window, click **Add**.
4. Browse to the **Cassandra_RAP.apd** file. The **New Application Template** panel opens.
5. Click **Save & Close**.

End

Creating the Cassandra Resource Access Point Application

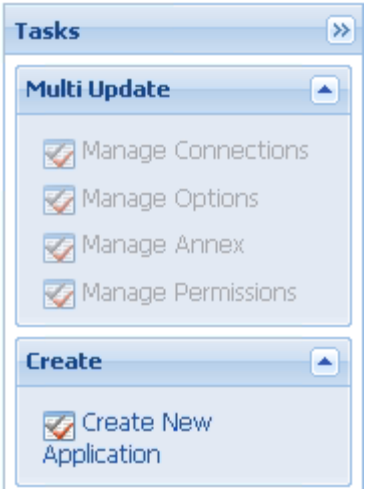
Note: For more information on how to work with application objects in Genesys Administrator, consult Generic Configuration Procedures.

Prerequisites

- You completed Importing the Cassandra Resource Access Point Template.

Start

1. Open Genesys Administrator and navigate to **Provisioning > Environment > Applications**.
2. In the **Tasks** panel, click **Create New Application**.



Create New Application link.

3. In the **Select Application Template** panel, click **Browse for Template** and select the Cassandra Resource Access Point template that you imported in Importing the Cassandra Resource Access Point Template. Click **OK**.
4. The template is added to the **Select Application Template** panel. Click **Next**.
5. In the **Select Metadata file** panel, click **Browse** and select the **Cassandra_RAP.xml' file. Click** Open**.**
6. The metadata file is added to the **Select Metadata file** panel. Click **Next**.
7. In **Specify Application parameters**:
   - Enter a name for your application. For instance, `GWE_Cassandra_Access_Point`.
   - Make sure that **State** is enabled.
   - Select the **Host** on which the Access Point will reside.
   - Click **Create**.
8. The **Results** panel opens.
9. Enable **Opens the Application details form after clicking 'Finish'** and click **Finish**. The Cassandra Resource Access Point application form opens and you can start configuring the Cassandra Resource Access Point application.

End

Configuring the Cassandra Resource Access Point Application

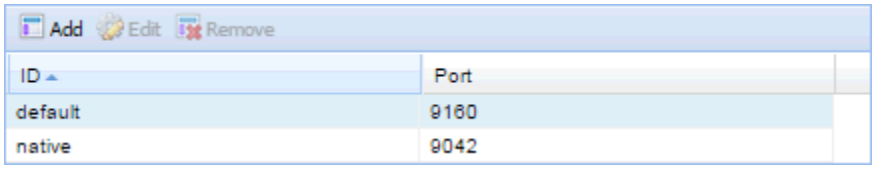
Note: For more information on how to work with application objects in Genesys Administrator, consult Generic Configuration Procedures.

Prerequisites

- You completed Creating the Cassandra Resource Access Point Application.

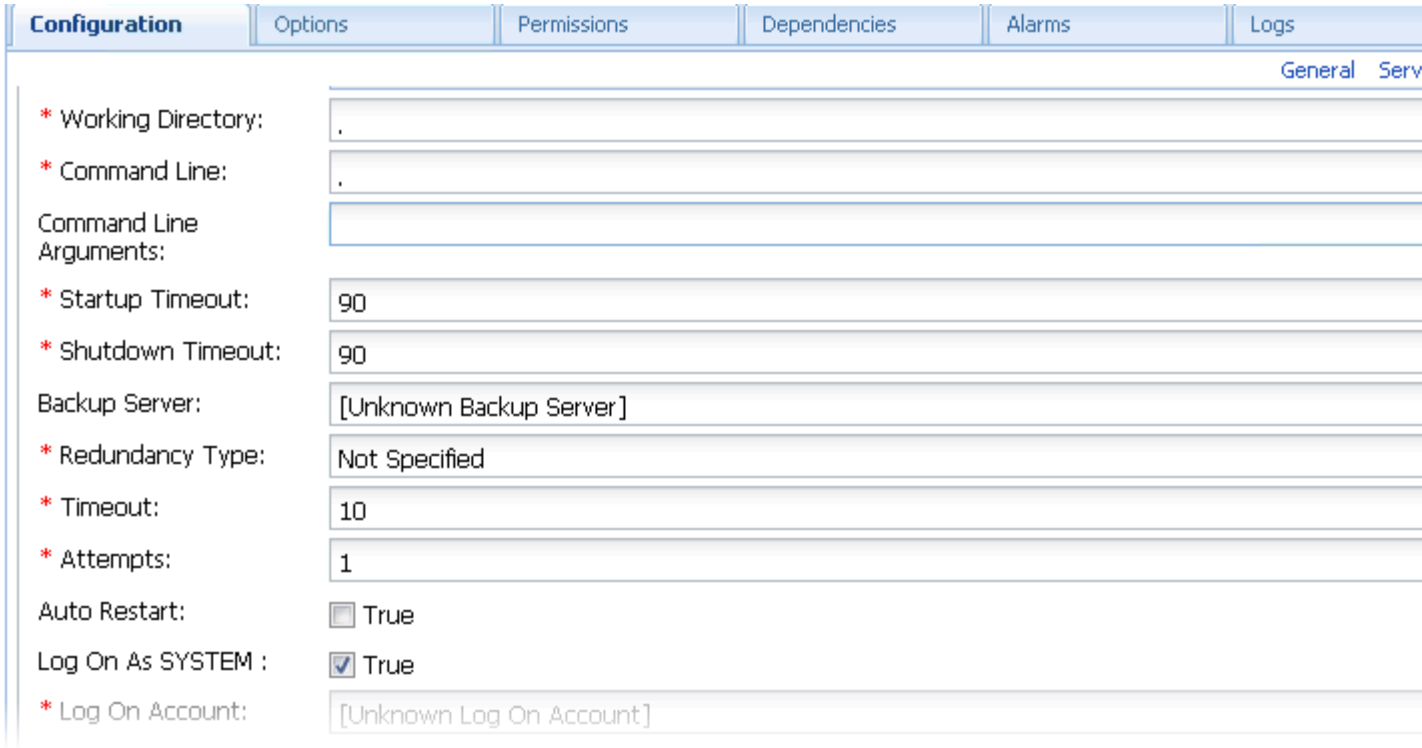
Start

1. If your Cassandra Resource Access Point application form is not open in Genesys Administrator, navigate to **Provisioning > Environment > Applications**. Select the application defined for the Cassandra Resource Access Point and click **Edit...**.
2. Expand the **Server Info** pane.
3. In the Tenant section, click **Add** and select your tenant. For instance, `Environment`. Click **OK**.
4. If your **Host** is not defined, click the lookup icon to browse to the hostname of your application, which should point to the host where you plan to locate your Cassandra node.
5. In the Listening Ports section, create the default port by clicking **Add**. The **Port Info** dialog opens.
   - Enter the **Port**. For instance, `9160`.
   - Click **OK**. The **default** port appears in the list of **Listening ports**.
6. Click **Add** again. The **Port Info** dialog opens.
   - In the **ID** field, enter `native`.
   - Enter the **Port**. For instance, `9042`.
   - Click **OK**. The **native** port appears in the list of **Listening ports**.



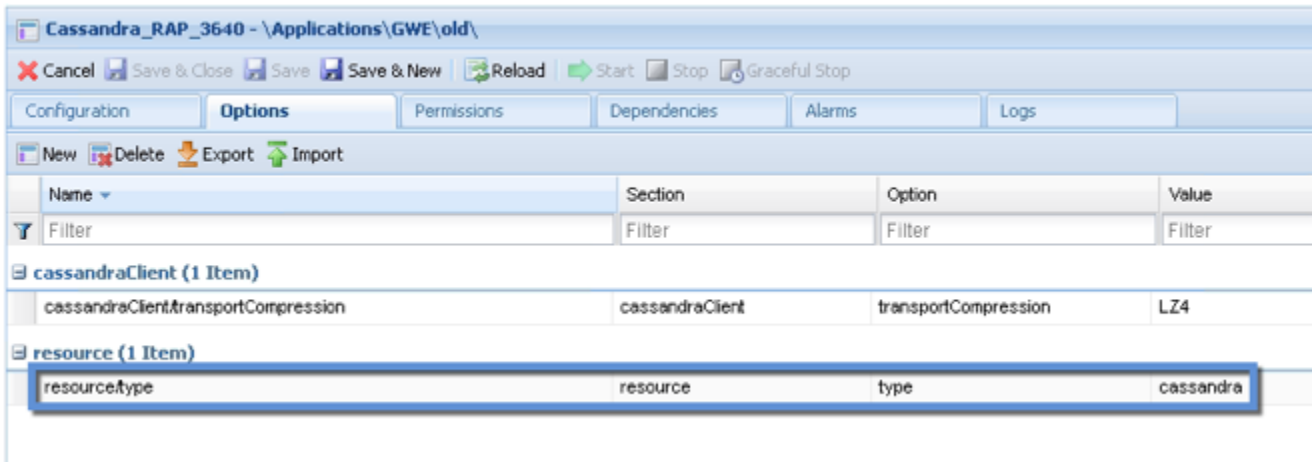
Cassandra port settings

7. Ensure the **Working Directory** and **Command Line** fields contain "." (period).



Commands

8. Click **Save**.
9. The **Confirmation** dialog for changing the application's port opens. Click **Yes**.
10. To configure External Cassandra, select the **Options** tab.
    - In the **[resource]** section, make sure **type** is set to `cassandra`.



Cassandra resource setting

11. Click **Save & Close**. If the **Confirmation** dialog opens, click **Yes**.

End

Note that you must execute this procedure and the previous one for each Cassandra seed**-node in your Cassandra cluster.**

Elasticsearch Resource Access Point Template

GWE installation does not provide a dedicated template for the Elasticsearch Resource Access Point. Instead, it is recommended to re-use the Cassandra Resource Access Point and correct the value of the "type" option.

Creating the Elasticsearch Resource Access Point Application

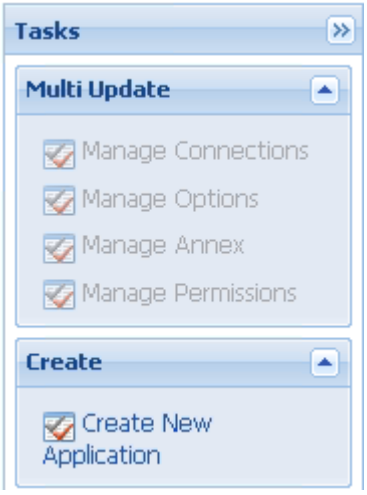
Note: For more information about working with application objects in Genesys Administrator, see Generic Configuration Procedures.

Prerequisites

- You completed Elasticsearch Resource Access Point Template.

Start

1. Open Genesys Administrator and navigate to **Provisioning > Environment > Applications**.
2. In the **Tasks** panel, click **Create New Application**.



Create New Application link.

3. In the **Select Application Template** panel, click **Browse for Template** and select the Cassandra Resource Access Point template that you imported in Importing the Cassandra Resource Access Point

Template. Click **OK**.

4. The template is added to the **Select Application Template** panel. Click **Next**.
5. In the **Select Metadata file** panel, click **Browse** and select the **Cassandra_RAP.xml** file. Click **Open**.
6. The metadata file is added to the **Select Metadata file** panel. Click **Next**.
7. In **Specify Application parameters**:
   - Enter a name for your application. For instance, `GWE_ES_Access_Point`.
   - Make sure that **State** is enabled.
   - Select the **Host** on which the Access Point will reside. It should be either Load Balancer (if one is configured) or one of the Elasticsearch nodes.
   - Click **Create**.
8. The **Results** panel opens.
9. Enable **Opens the Application details form after clicking Finish'** and click **Finish**. The Elasticsearch Resource Access Point application form opens and you can start configuring the Elasticsearch Resource Access Point application.

End

Configuring the Elasticsearch Resource Access Point Application

Note: For more information about working with application objects in Genesys Administrator, see Generic Configuration Procedures.

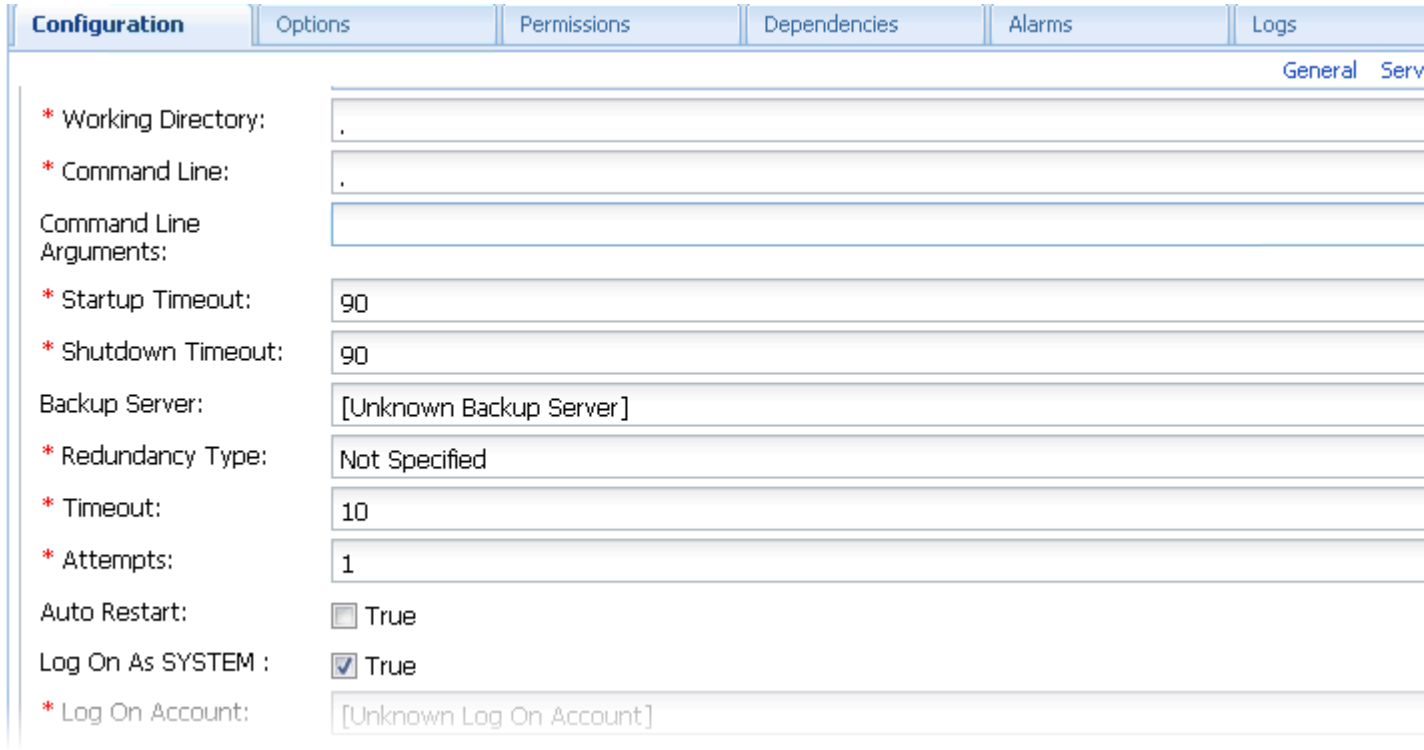
Prerequisites

- You completed Creating the Elasticsearch Resource Access Point Application.

Start

1. If your Elasticsearch Resource Access Point application form is not open in Genesys Administrator, navigate to **Provisioning > Environment > Applications**. Select the application defined for the Cassandra Resource Access Point and click **Edit...**.
2. Expand the **Server Info** pane.
3. In the Tenant section, click **Add** and select your tenant. For instance, `Environment`. Click **OK**.
4. If your **Host** is not defined, click the lookup icon to browse to the hostname of your application, which should point to the host where you plan to locate your *seed* Elasticsearch node or Load Balancer to all Elasticsearch nodes.
5. In the Listening Ports section, create the default port by clicking **Add**. The **Port Info** dialog opens.
   - Enter the **Port**. For instance, `9200`. This is the HTTP port configured in your Elasticsearch.
   - Click **OK**. The **default** port appears in the list of **Listening ports**.

6. Ensure the **Working Directory** and **Command Line** fields contain "." (period).



Commands

7. Click **Save**.
8. The **Confirmation** dialog for changing the application's port opens. Click **Yes**.
9. To confirm that the current application is an Elasticsearch Access Point, select the **Options** tab.
   - In the **[resource]** section, make sure **type** is set to `elasticsearch`.
10. Click **Save & Close**. If the **Confirmation** dialog opens, click **Yes**.

End

Configuring the Web Engagement Cluster for Use with External Cassandra and Elasticsearch

Prerequisites

- You completed Configuring the Cassandra Resource Access Point Application and Configuring the Elasticsearch Resource Access Point Application.

Start

1. Navigate to **Provisioning > Environment > Applications**. Select the application defined for the Web Engagement Cluster and click **Edit...**.
2. In the Connections section of the **Configuration** tab, click **Add**. The **Browse for applications** panel opens. Select a Genesys application defined as a Cassandra Resource Access Point, select the **native** connection port, and then click **OK**.
3. Repeat the previous step for all Cassandra Resource Access Point applications configured for external Cassandra nodes belonging to the same data center.
   Note: You must not connect Cassandra Resource Access Point applications belonging to different data centers to the same Web Engagement Cluster.
4. In the Connections section of the **Configuration** tab, click **Add**. The **Browse for applications** panel opens. Select a Genesys application defined as an Elasticsearch Resource Access Point, select the **default** connection port, and then click **OK**.

Note: Genesys recommends that you configure Load Balancer as an entry point to the Elasticsearch cluster. In this case, the Resource Access Point represents the Load Balancer address. You can also use a dedicated Elasticsearch node as an entry point. In this case, the Resource Access Point points to that node, so you must ensure that the node is available at the moment Web Engagement server is starting.

1. Click **Save & Close**. If the **Confirmation** dialog opens, click **Yes**.

End

Deploy an Elasticsearch Cluster Node

Installation

Installation procedure is described at Installing Elasticsearch

> ### Important
> Genesys is not responsible for the content of external internet sites.

Configuration

The following should be taken into account when configuring your Elasticsearch cluster:

1. The specified host should correspond to the one specified in the Elasticsearch Resource Access Point (unless you configured Load Balancer to access Elasticsearch nodes).
2. The specified **http.port** should correspond to the one specified in the Elasticsearch Resource Access Point (unless you configured Load Balancer to access Elasticsearch nodes).

3. Use unicast discovery mode to avoid potential issues when joining unexpected nodes to a cluster.

Genesys recommends that you configure Load Balancer to provide access to the HTTP port of the Elasticsearch cluster.

Deploy a Cassandra Cluster Node

Linux

Installation

1. Download version 3.11.4 (or higher) from the Cassandra 3.11 stream.
2. Unpack the archive into the installation directory. For example:

```
cd /genesys
tar xzf apache-cassandra-3.11.x-bin.tar.gz
```

> **Important**
> When installing Cassandra, do not use path names that contain spaces.

Configuration

1. Go to the directory where you installed your Cassandra node.
2. Edit **conf/cassandra.yaml**, using the following custom values:
   - **cluster_name**: *cluster name without spaces, for example* WebMe_Cassandra_Cluster
   - **seeds**: *<comma-separated list of fully qualified domain names (FQDN) or IP addresses of one or more Cassandra nodes>* **Note:** This value must be the same for all nodes. Here are two examples:
     - 192.168.0.1,192.168.0.3
     - host1.mydomain.com, host2.mydomain.com
   - **storage_port**: `7000` (default value)
   - **ssl_storage_port**: `7001` (default value)
   - **listen_address**: *<current node host name>* **Note:** This address is used for inter-node communication, so it must be available for use by other Cassandra nodes in your cluster.
   - **native_transport_port**: `9042` (default value)

- **rpc_address**: *<current node host name>* **Note:** This address is used by Web Engagement to connect to Cassandra, so it must be available to all Web Engagement hosts.
- **rpc_port**: `9160` (default value)
- **endpoint_snitch**: `GossipingPropertyFileSnitch`

Note: Make sure that each Cassandra node has access to the ports specified for the other nodes.

3. Edit **conf/cassandra-rackdc.properties**.
4. In order to set up indexing for Elasticsearch:
   1. Copy all jar files from ***Web Engagement installation directory*/tools/cassandra/libs** to ***Cassandra installation directory*/lib**.
   2. Copy the **es-index.properties** file from ***Web Engagement installation directory*/tools/cassandra** to ***Cassandra installation directory*/conf**.
   3. Edit the **es-index.properties** file.
5. Verify that the required communication ports are opened.

Setting Up a Cassandra Service

The sample script described in the following procedure should give you an idea of how to set up Cassandra as a service process.

1. Create the **/etc/init.d/cassandra** startup script.
2. Edit the contents of the file:

```
#!/bin/sh
#
# chkconfig: - 80 45
# description: Starts and stops Cassandra
# update daemon path to point to the cassandra executable
DAEMON=<Cassandra_installation_dir>/bin/cassandra
start() {
        echo -n "Starting Cassandra... "
        $DAEMON -p /var/run/cassandra.pid
        echo "OK"
        return 0
}
stop() {
        echo -n "Stopping Cassandra... "
        kill $(cat /var/run/cassandra.pid)
        echo "OK"
        return 0
}
case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  restart)
        stop
```

```
        start
        ;;
  *)
        echo $"Usage: $0 {start|stop|restart}"
        exit 1
esac
exit $?
```

3. Make the file executable: `sudo chmod +x /etc/init.d/cassandra`
4. Add the new service to the list: `sudo chkconfig --add cassandra`
5. Now you can manage the service from the command line:

- `sudo /etc/init.d/cassandra start`
- `sudo /etc/init.d/cassandra stop`

6. Configure the service to be started automatically together with the VM: `sudo chkconfig --level 2345 cassandra on`

Windows

Installation

1. Download version 3.11.4 or higher from the Cassandra 3.11 stream.
2. Unpack the archive into a path without spaces.

Configuration

1. Go to the directory where you installed your Cassandra node.
2. Edit **cassandra.yaml**, using the following custom values:
   - **cluster_name**: *cluster name without spaces, for example* WebMe_Cassandra_Cluster
   - **seeds**: *<comma-separated list of fully qualified domain names (FQDN) or IP addresses of one or more Cassandra nodes>* **Note:** This value must be the same for all nodes. Here are two examples:
     - 192.168.0.1,192.168.0.3
     - host1.mydomain.com, host2.mydomain.com
   - **storage_port**: `7000` (default value)
   - **ssl_storage_port**: `7001` (default value)
   - **listen_address**: *<current node host name>* **Note:** This address is used for inter-node communication, so it must be available for use by other Cassandra nodes in your cluster.
   - **native_transport_port**: `9042` (default value)
   - **rpc_address**: *<current node host name>* **Note:** This address is used by Web Engagement to connect to Cassandra, so it must be available to all Web Engagement hosts.

- **rpc_port**: 9160 (default value)
- **endpoint_snitch**: GossipingPropertyFileSnitch

Note: Make sure that each Cassandra node has access to the ports specified for the other nodes.

3. Edit **conf/cassandra-rackdc.properties**.
4. In order to set up indexing for Elasticsearch:
   1. Copy all jar files from ***Web Engagement installation directory*/tools/cassandra/libs** to ***Cassandra installation directory*/lib**.
   2. Copy the **es-index.properties** file from ***Web Engagement installation directory*/tools/cassandra** to ***Cassandra installation directory*/conf**.
   3. Edit the **es-index.properties** file.
5. Verify that the required communication ports are opened.
6. Start Cassandra.

Configuring cassandra-rackdc.properties

For a single data center, use the following as a guide:

```
dc=<Data Center name>
rack=<RACK ID>
```

For example,

```
dc=OperationalDC
rack=RAC1
```

Note: Genesys recommends that you use the same **rack** ID if you do not have a clear understanding of your servers' rack usage. For more information about `cassandra-rackdc.properties`, refer to GossipingPropertyFileSnitch.

Configuring es-index.properties

- For every node in a given data center, set the **discovery.zen.ping.unicast.hosts** property in the **es-index.properties** file for that node to a comma-separated list of the servers that host the Elasticsearch nodes belonged to that data center.

  If you have n data centers in your Cassandra cluster, then you should have n different versions of the **es-index.properties** file (one for each data center). The primary difference between these lists will be the values of the **discovery.zen.ping.unicast.hosts** property.

  For example:

  - The Cassandra nodes located on **host_DC1_A**, **host_DC1_B**, and **host_DC1_C** belong to data center **DC1**. Because of this, the **discovery.zen.ping.unicast.hosts** property in the **es-index.properties** files for nodes **host_DC1_A**, **host_DC1_B**, and **host_DC1_C** will be defined as `host_DC1_A,host_DC1_B,host_DC1_C`.
  - The Cassandra nodes located on **host_DC2_X**, **host_DC2_Y**, and **host_DC2_Z** belong to data center **DC2**. Because of this, the **discovery.zen.ping.unicast.hosts** property in the **es-index.properties** files for nodes **host_DC2_X**, **host_DC2_Y**, and **host_DC2_Z** will be defined as `host_DC2_X,host_DC2_Y,host_DC2_Z`.
- Set the **index.number_of_shards** property to three times the number of Cassandra nodes in the current data center. For example, if you have 3 Cassandra nodes in the current data center, then the number of shards for each index should be 9.
- If you prefer not to store your Elasticsearch data in the default directory, you can use the **path.data** property to specify another location. This property is commented out by default.

Communication Ports

Cassandra and Elasticsearch use the following ports for external and internode communication. **Note:** Either or both of them may not work as expected unless you ensure that these ports are opened for communication between all servers that host Cassandra nodes.

Port	Default	Where to Change the Value
Cassandra Storage port	7000	**storage_port** in **cassandra.yaml**
Cassandra SSL Storage port	7001	**ssl_storage_port** in **cassandra.yaml**
Cassandra Thrift port	9160	**rpc_port** in **cassandra.yaml**
Cassandra CQL port	9042	**native_transport_port** in **cassandra.yaml**
Elasticsearch REST request service port	9200	**http.port** property in the **es-index.properties** file
Elasticsearch transport port	9300	**transport.tcp.port** property in the **es-index.properties** file

Starting the Cassandra Cluster Nodes

Prior to starting Cassandra Cluster, you need to start the Elasticsearch cluster. There is no specific order when starting Elasticsearch nodes. However, your Cassandra nodes must be started in the following order:

1. Start the seed nodes.
2. Start the other non-seed nodes.

The seed node is one of the nodes specified in the **seeds** option.

Verifying Your Cassandra Cluster

After you have deployed your Cassandra Cluster, you may want to verify that all of the nodes can communicate with each other. To do this, execute the following command on any Database VM:

Linux

```
cd <Cassandra_installation_dir>/bin
./nodetool -h <hostname> status
```

Windows

```
cd <Cassandra_installation_dir >/bin
nodetool -h <hostname> status
```

Sample Output

This command should produce output that looks something like this:

```
Datacenter: DC1
==========================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--        Address                  Load                Tokens       Owns    Host
ID                                                                        Rack
UN     10.51.XX.XXX       106,36 KB       256               ?            380d02fb-
da6c-4f6a-820e-14538bd24a39     RAC1
UN     10.51.XX.XXX       108,22 KB       256               ?            601f05ac-
aa1d-417b-911f-22340ae62c38       RAC1
UN     10.51.XX.XXX       107,61 KB       256               ?            171a15cd-
fa4d-410e-431b-51297af13e96      RAC1

Datacenter: DC2
==========================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--        Address                  Load                Tokens       Owns    Host
ID                                                                        Rack
UN     10.51.XX.XXX       104,06 KB       256               ?
48ad4d08-555b-4526-8fab-d7ad021b14af     RAC1
UN     10.51.XX.XXX       109,56 KB       256               ?            8ca0fb45-aef7-4f0a-
ac4e-a324ceea90c9        RAC1
UN     10.51.XX.XXX       105,18 KB       256               ?
1c45e1fa-9f82-4bc4-a896-5575bad53808       RAC1
```

Verifying Your Elasticsearch Cluster

Each of your Cassandra data centers must have a dedicated Elasticsearch cluster. To verify the status of your Elasticsearch clusters, execute the following request. (You can use your browser for this.)

Prerequisites

- Your Cassandra cluster was initialized
- Your Cassandra cluster was started

- At least one Web Engagement Server instance is running in each of your Web Engagement Clusters

Start

```
http://<Elasticsearch_Node_OR_Elasticsearch_LoadBalancer>:9200/_cluster/state?pretty
```

End

Note: This example uses port 9200, which is the default Elasticsearch HTTP port. If you are not using the default Elasticsearch HTTP port, substitute the port number that you are using.

Output

The output from this request will describe your nodes, as shown here:

```
"nodes": {
    "xxx-xxxxxx-xxxxxxxxxxx": {
        "name": <node name>,
        "transport_address": <address for inter-node Elasticsearch communication>,
        "attributes": {}
    }
}
```

If your Elasticsearch clusters are correctly configured:

- Any requests executed on Elasticsearch nodes belonging to the *same* data center should result in *identical* lists of Elasticsearch nodes.
- Any requests executed on Elasticsearch nodes belonging to *different* data centers should result in *different* lists of Elasticsearch nodes.

Upgrading Cassandra Nodes

You can upgrade your Cassandra version without interrupting service if:

- The version you are upgrading to is in the same stream (for example, from one 3.11.x version to another)
- You are not changing your database schema

Use the following steps for this task:

1. Stop the first Cassandra seed node.
2. Preserve your database storage.
3. Upgrade your Cassandra version, following the instructions in the Release Notes for the new version.
4. Be sure that your database storage is in the preserved state (the same set of files).
5. Start the first Cassandra seed node.
6. Execute steps 1 through 5 for the other seed nodes.

7. Execute steps 1 through 5 for the other non-seed nodes.
8. Verify that the Cassandra cluster is working, as shown above in Verifying Your Cassandra Cluster.

If your upgrade plans include changing your database schema or changing Cassandra versions between streams, then you will have to interrupt service. Use the following steps for this task:

1. Stop all of your Cassandra nodes.
2. If your database schema has been changed since you installed the previous version, update the Cassandra database, following the instructions in the Release Notes for the new version.
3. Configure each node, following the instructions in the Release Notes for the new version.
4. Start the Cassandra seed nodes.
5. Start the other nodes.
6. Verify that the Cassandra cluster is working, as shown above in Verifying Your Cassandra Cluster.

Maintenance

Because Cassandra is a critical component of Web Engagement, it is essential to keep track of its health. The Datastax documentation provides some really good information about how to do this at https://docs.datastax.com/en/archived/cassandra/3.x/cassandra/tools/toolsNodetool.html.

Genesys recommends that you use the **nodetool** utility that is bundled with your Cassandra installation package and that you make a habit of using the following nodetool commands to monitor the state of your Cassandra cluster.

ring

Displays node status and information about the cluster, as determined by the node being queried. This can give you an idea of the load balance and whether any nodes are down. If your cluster is not properly configured, different nodes may show a different cluster; this is a good way to check that every node views the cluster the same way.

```
nodetool -h <HOST_NAME> -p <JMX_PORT> ring
```

status

Displays cluster information.

```
nodetool -h <HOST_NAME> -p <JMX_PORT> status
```

compactionstats

Displays compaction statistics.

```
nodetool -h <HOST_NAME> -p <JMX_PORT> compactionstats
```

getcompactionthroughput \ setcompactionthhroughput

Displays the compaction throughput on the selected Cassandra instance. By default it is 32 MB/s.

You can increase this parameter if you observe permanent growth of database size after the TTL and grace periods are passed. Note that increasing compaction throughput will affect memory and CPU consumption. Because of this, you need make sure to have sufficient hardware to support the rate that you have selected.

```
nodetool -h <HOST_NAME> -p <JMX_PORT> getcompactionthroughput
```

To increase compaction throughput to 64 MB/s, for example, use the following command:

```
nodetool -h <HOST_NAME> -p <JMX_PORT> setcompactionthroughput 64
```

Recovery

Depending on the replication factor and consistency levels of a Cassandra cluster configuration, the Web Engagement Cluster can handle the failure of one or more Cassandra nodes in the data center without any special recovery procedures and without interrupting service or losing functionality. When the failed node is back up, the Web Engagement Cluster automatically reconnects to it.

- Therefore, if an eligible number of nodes have failed, you should just restart them.

However, if too many of the Cassandra nodes in your cluster have failed or stopped, you will lose functionality. To ensure a successful recover from failure of multiple nodes, Genesys recommends that you:

- Stop every node, one at a time, with at least two minutes between operations.
- Then restart the nodes one at a time, with at least two minutes between operations.

Automatic Provisioning

Overview

You can create all the configuration information related to Genesys Web Engagement in Configuration Server by running the Provisioning Tool, located in the *Web Engagement installation directory***tools****provisioning** directory. The 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. You don't usually need to do this, but it can be necessary when you change things like the application name or your Configuration Server parameters.

Note that you can run the tool under Windows or Linux.

Created (or Corrected) Objects

The Provisioning Tool connects to Configuration Server and reads the configuration for the Web Engagement applications. It creates Genesys objects used by the Web Engagement Servers and edits the configuration files required to launch the Web Engagement Servers.

The following objects are created or corrected when you run the Provisioning Tool:

Agent Groups

Provisioning creates a "default" Agent Group: Web Engagement Chat. This group is used in the default engagement and chat routing strategies provided by Genesys Web Engagement. It is also used to provide input for the pacing algorithm.

Location in Genesys Administrator: **Provisioning > Accounts > Agent Groups**

> **Important**
> Provisioning does not create Agent objects, so make sure you add agents to your new Agent Groups after running the tool.

Categories

Genesys Web Engagement includes an out-of-the-box sample application **playground**. Provisioning creates category objects to support this application:

- **playground** sample application — categories start with the **PlayGround-** prefix.

Location in Genesys Administrator: **Provisioning > Routing/eServices > Business Attributes > Web Engagement Categories > Attributes Values**

Interaction Queues

Provisioning creates Interaction Queue objects that serve as the entry points for the Engagement Logic SCXML strategy and the Chat Routing SCXML strategy. Provisioning also creates Interaction Queues to support real-time reporting using CCPulse and Pulse templates. Provisioning creates the following queues:

- Engagement Logic SCXML strategy (and also for real-time reporting): Webengagement_Qualified
- Chat Routing SCXML strategy: Webengagement_Chat
- Real-time reporting:
  - Webengagement_Qualified (also used as the entry point for the Engagement Logic SCXML strategy)
  - Webengagement_Engaged
  - Webengagement_Accepted
  - Webengagement_Missed
  - Webengagement_Rejected
  - Webengagement_Timeout
  - Webengagement_Failed

Location in Genesys Administrator: **Provisioning > Routing/eServices > Interaction Queues (or Provisioning > Environment > Scripts)**

Interaction Queue Views

For each Interaction Queue object the Provisioning Tool creates, it also creates an Interaction Queue View object. Provisioning creates the following Interaction Queue View objects:

- Webengagement_Qualified.EngagementLogic.View
- Webengagement_Chat.ChatRouting.View
- Webengagement_Engaged.Clean
- Webengagement_Accepted.Clean
- Webengagement_Missed.Clean
- Webengagement_Rejected.Clean
- Webengagement_Timeout.Clean
- Webengagement_Failed.Clean

Location in Genesys Administrator: **Provisioning > Environment > Scripts**

Enhanced Routing objects

Provisioning creates a set of Enhanced Routing objects to work with the previously created Interaction Queues.

- The Webengagement_Qualified.Routing object is used to provide the Engagement Logic SCXML strategy for Orchestration.
- The Webengagement_Chat.Routing object is used to provide the Chat Routing SCXML strategy for Orchestration.

The tool also creates the following Enhanced Routing objects, which are used for cleaning purposes (for example, to clean interactions that for some reason were stuck in one of the statistical-related Interaction Queues):

- Webengagement_Engaged.Routing
- Webengagement_Accepted.Routing
- Webengagement_Missed.Routing
- Webengagement_Rejected.Routing
- Webengagement_Timeout.Routing
- Webengagement_Failed.Routing

Location in Genesys Administrator: **Provisioning > Routing/eServices > Orchestration (or Provisioning > Environment > Scripts)**

Case Data

Provisioning creates a pair of attributes under Case Data Business Attribute to support functionality in the Genesys Web Engagement Plug-in for Workspace Desktop Edition.

> ### Important
> Provisioning only adds attributes to existing Case Data Business Attribute, but does not create the attribute if it is absent. The Case Data Business Attribute should be created during the Workspace Desktop Edition installation process.

The tool creates the following attributes:

- WebEngagement.CurrentWebPage (display name is Current Web Page)
- WebEngagement.EngagementStartPage (display name is Engagement Start Page)

Location in Genesys Administrator: **Provisioning > Routing/eServices > Business Attributes > Case Data > Attributes Values**

Genesys Chat Server application

Provisioning corrects the Chat Server application connected to the Web Engagement Cluster application. It creates an option in the **[endpoints:*Web Engagement Server Tenant ID*]** section of the Chat Server application with the following value:

- Option name: **webme**
- Option value: `Webengagement_Chat`

> **Important**
>
> The connection to the Chat Server application does not have to be direct - it can be through a Genesys application with the type Application Cluster. See Configuring a Connection to a Cluster of Chat Servers (Optional) for details.

Location in Genesys Administrator: **Provisioning > Environment > Applications**

Genesys Stat Server application

Provisioning adds some statistics and filters into the options of the Stat Server application connected to the Web Engagement Cluster application.

If absent, the following statistics are added:

- Chat_Interactions_Abandoned
- Chat_Interactions_Accepted
- Chat_Interactions_Processed
- Chat_Interactions_Processing_Time
- Chat_Interactions_Reactive_Total_Entered
- Chat_Total_Entered_Queue
- Interactions_Average_Waiting_Time
- Virtual_Queue_Abandoned
- Webengagement_Total_Entered_Queue

If absent, the following filters are added:

- Webengagement_chat_filter
- Webengagement_filter
- Webengagement_rule_TEMPLATE

Location in Genesys Administrator: **Provisioning > Environment > Applications**

Genesys Web Engagement Server application

- The queueKey in the **[chat]** section is populated with the *Web Engagement Server Tenant ID*:webme value. For example, 1:webme. This change is paired with changes in the connected Chat Server.

Location in Genesys Administrator: **Provisioning > Environment > Applications**

Running the Provisioning Tool

Prerequisites

- The configuration applications for the Web Engagement servers are created in Configuration Server.
- The connections for the Web Engagement Server application include Interaction Server, and the Stat Server applications.
  See Creating a Node Application for details.

Start

1. Navigate to the Web Engagement installation directory and open the *Web Engagement installation directory***tools\provisioning** folder.
2. From the command line, run the following command: webengagement_provisioning.bat -host *Configuration Server host name or IP address* -port *Configuration Server port* -user *user ID login into Configuration Server* -password *password for the specified user ID* -app *Application name for Web Engagement Server*
   For Linux, use the same command line, but instead of webengagement_provisioning.bat, specify webengagement_provisioning.sh.

   > **Important**
   > User and password options may be optional, according to your Configuration Server settings.

   You can also run provisioning with the **overwrite** option. In overwrite mode, the provisioning tool replaces old objects with new objects. Already existing GWE-specific objects will be removed and new objects will be created instead. You will lose any changes you have made manually on GWE-specific objects. The command will look like this:

   webengagement_provisioning.bat -host *Configuration Server host name or IP address* -port *Configuration Server port* -user *user* -password *password* -app *Application name for Web Engagement Server* -overwrite

   > **Important**
   > User and password options may be optional, according to your Configuration Server settings.

3. The provisioning script starts. If the provisioning is successful, the following message is displayed: Provisioning script successfully finished his work

End

Note: The **webengagement_provisioning.bat** and **webengagement_provisioning.sh** files contain usage information that is printed when you run them. If you execute these files without parameters, this information will still be printed, but you will also receive an execution error.

Tuning Your JVM

Web Engagement Server allows you to tune your JVM parameters, which is especially important with respect to the virtual memory allocated to the JVM. You can modify your JVM parameters in the "'launcher.ini'" or **setenv.sh** file that is located at ***Web Engagement installation directory*\server**.

Tuning Your Memory Allocation

Windows

The out-of-box memory settings for the **launcher.ini** file are as follows:

```
[JavaArgs]
-Xmx4096m
-Xms4096m
```

Linux

The out-of-box memory settings for the **setenv.sh** file are as follows:

```
MEMORY_MIN=4096m
MEMORY_MAX=4096m
```

If you want to change these settings, update the appropriate values and restart the Web Engagement Server.

Enabling the JMX Port

You can monitor the JVM where your Web Engagement Server is running by using some of the standard Java tools distributed with the JDK, such as JConsole or Visual VM. Since these tools work through the JMX port, the Web Engagement Server needs to open this port on startup.

Windows

The **launcher.ini** file contains all of the settings you need to turn on JMX access:

```
; uncomment and configure the properties below to use JMX
;-Dcom.sun.management.jmxremote.local.only=false
;-Dcom.sun.management.jmxremote.port=5999
;-Dcom.sun.management.jmxremote.authenticate=false
;-Dcom.sun.management.jmxremote.ssl=false
```

Linux

The **setenv.sh** file contains all of the settings you need to turn on JMX access:

```
# Uncomment for enabling JMX Remote
#JMX_PORT=5999
#JAVA_OPTS="${JAVA_OPTS} \
#  -Dcom.sun.management.jmxremote.port=${JMX_PORT} \
#  -Dcom.sun.management.jmxremote.local.only=false \
#  -Dcom.sun.management.jmxremote.ssl=false \
#  -Dcom.sun.management.jmxremote.authenticate=false"
```

As you can see, the JMX parameters are turned off by default. To enable JMX access, you need to uncomment these settings.

Note: Opening the JMX port without adequate security can be risky. See the JMX monitoring and management documentation for information about how to tune your security settings when using JMX to monitor your JVM.

Installing the Plug-in for Workspace Desktop Edition

The Genesys Web Engagement Plug-in for Workspace Desktop Edition allows you to enable chat and web callback engagement features in Workspace Desktop Edition. See Genesys Web Engagement Plug-in for Workspace Desktop Edition Help for details.

To install this plug-in, complete the following procedures:

1. Installing the Plug-in for Workspace Desktop Edition
2. Importing the Plug-in for Workspace Desktop Edition Template
3. Adding a Connection to the Web Engagement Cluster.
4. Adding a Connection to the Web Engagement Cluster using a load balancer option (an alternative approach to Adding a Connection to the Web Engagement Cluster).
5. Genesys Web Engagement can also work with agents who are Team Leads. For details about how to configure Team Leads, see the following topics in the Workspace Desktop Edition Deployment Guide:
   - Procedure: Enabling agents to be Team Leads
   - Monitoring Chat Interactions

Installing the Plug-in for Workspace Desktop Edition

Prerequisites

- Your environment includes Workspace Desktop Edition. See Genesys environment prerequisites for compliant versions. For more information about installing Workspace Desktop Edition, refer to the Workspace Desktop Edition Deployment Guide.

Start

1. In your installation package, locate and double-click the **setup.exe** file.
2. Click **Next**. The **Select Installed Application** screen appears.
3. Select your Workspace Desktop Edition application.
4. Click **Next**. The **Ready to Install** screen appears.
5. Click **Install**. The Genesys Installation Wizard indicates it is performing the requested operation for the Genesys Web Engagement Plug-in for Workspace Desktop Edition. When through, the **Installation Complete** screen appears.
6. Click **Finish** to complete your installation. As a result of the installation, the following files are copied to the Workspace Desktop Edition installation directory:

- Genesyslab.Desktop.Modules.WebEngagement.dll
- Genesyslab.Desktop.Modules.WebEngagement.module-config
- Genesyslab.Desktop.Modules.WebEngagement.deployment-config
- Newtonsoft.Json.Net35.dll

End

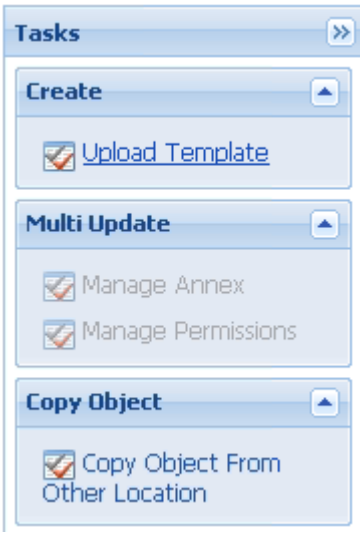
Importing the Plug-in for Workspace Desktop Edition Template

Prerequisites

- You completed Installing the Plug-in for Workspace Desktop Edition

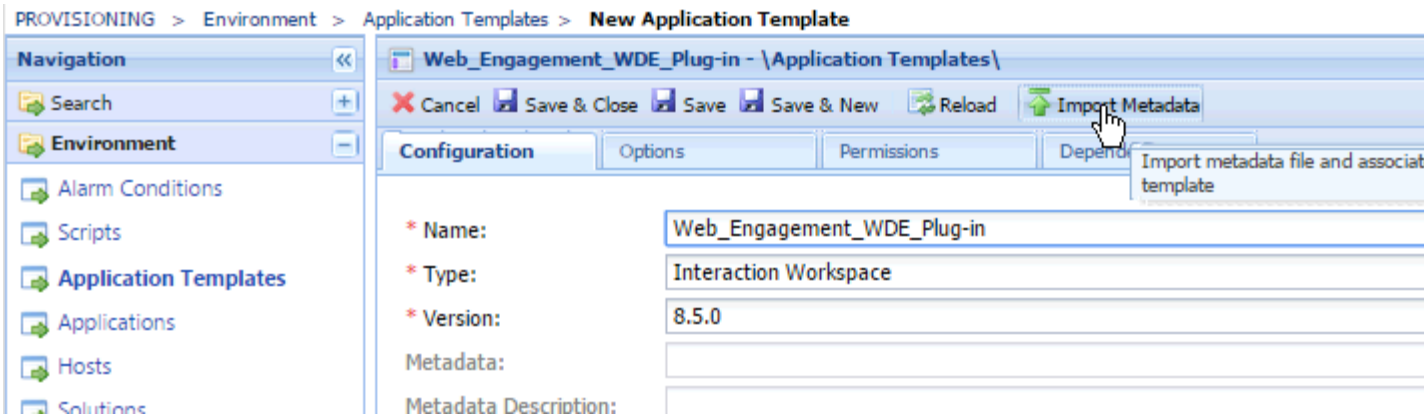
Start

1. In Genesys Administrator, navigate to **Provisioning > Environment > Application Templates**.
2. In the **Tasks** panel, click '**Upload Template**.



Upload Template link in the **Tasks** panel

3. In the **Click 'Add' and choose application template (APD) file to import** window, click **Add**.
4. Browse to the *GWE_WDE_Plug-in.apd* file located in the **Templates** folder in your installation package. The **Configuration** tab for the new template opens.
5. Click **Import Metadata**.



Click **Import Metadata**.

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

End

Adding a Connection to the Web Engagement Cluster

Prerequisites

- You completed Importing the Plug-in for Workspace Desktop Edition Template

Start

1. In Genesys Administrator, navigate to **Provisioning > Environment > Applications**, select the Workspace Desktop Edition application, and click **Edit...**.
2. In the Connections section, click **Add**. The **Browse Applications** window opens.
3. Select the Web Engagement Cluster application and click **OK**. The cluster is added to the list of Connections.
4. Click **Save & Close**.

End

Adding a Connection to the Web Engagement Cluster using a load balancer option

(This is an alternative approach to Adding a Connection to the Web Engagement Cluster.)

Prerequisites

- You completed Importing the Plug-in for Workspace Desktop Edition Template
- Your Workspace Desktop Edition application already has a connection to an application cluster other than the Web Engagement Cluster.

Start

1. In Genesys Administrator, navigate to **Provisioning > Environment > Applications**, select the Workspace Desktop Edition application, and click **Edit...**.
2. Select the **Options** tab and click **New**.
3. Set the following values:
   - **Location**: `Options`
   - **Section**: `settings`
   - **Name**: `loadbalancer`
   - **Value**: The address of your load balancer for the Web Engagement Cluster — for example, `http://198.51.100.12:8000`.
4. Click **OK**. The option is added to the **[settings]** section.
5. Click **Save & Close**.

End

Configuring Role-Based Access Control

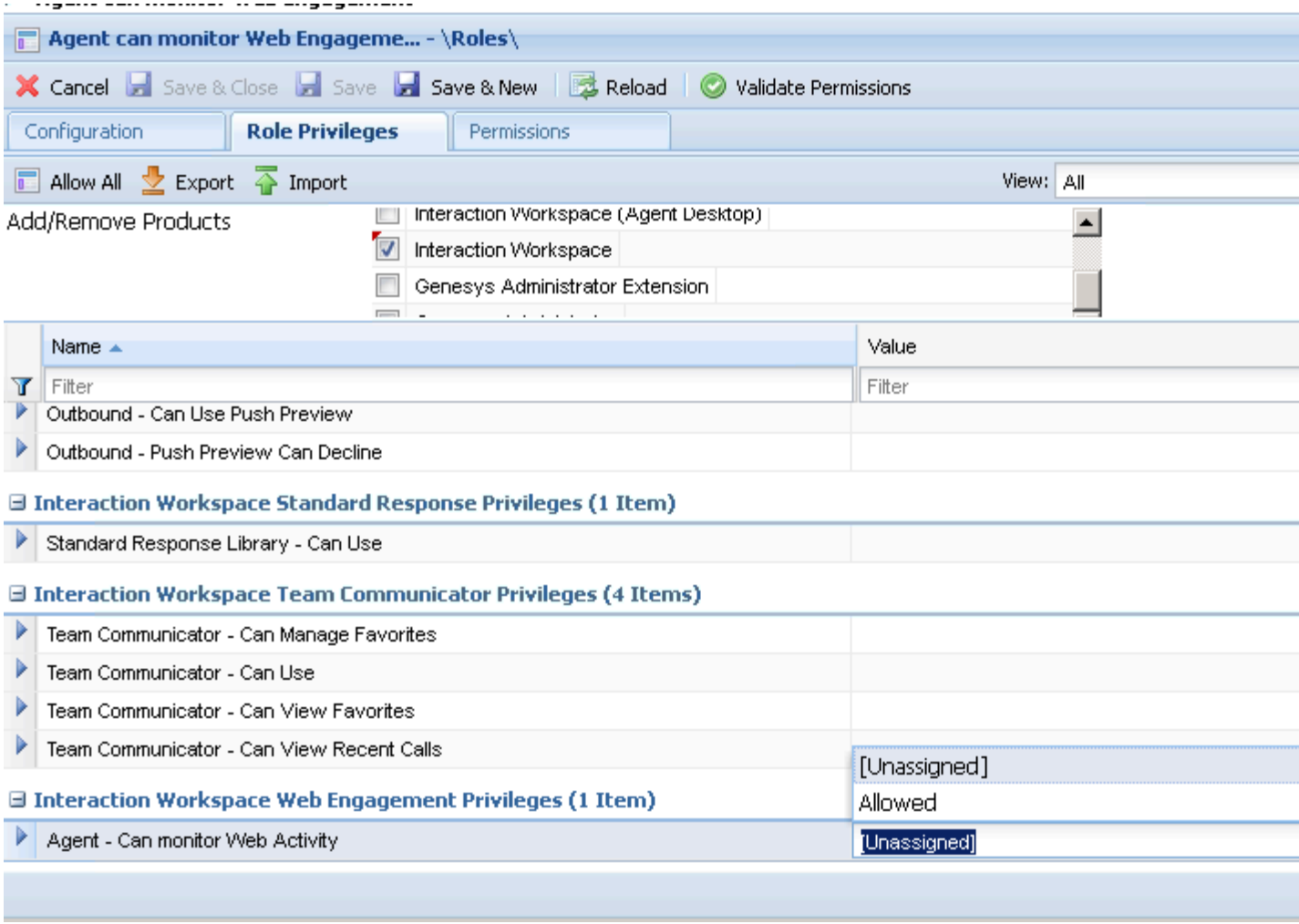
Complete this procedure to allow specific users or groups to manage Web Engagement in Workspace Desktop Edition.

Prerequisites

- You completed Importing the Plug-in for Workspace Desktop Edition Template

Start

1. In Genesys Administrator, navigate to **Provisioning > Accounts > Roles**.
2. Edit or create a Role responsible for managing Web Engagement in Workspace Desktop Edition. 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 Workspace Desktop Edition and expand the Workspace Desktop Edition Web Engagement Privileges section.
5. Set the `Allowed` value for the **Agent - Can Monitor Web Activity** option.



Select Allowed

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

End

Next Steps

- Tuning Role-Based Access in Genesys Administrator

Tuning Role-Based Access in Genesys Administrator

Web Engagement 8.5.2 supports role-based access for the following privileges:

- **Access to Web Engagement Categories**—allows users to create, read, update, or delete Web Engagement categories.
- **Access to Web Engagement Pacing Configuration**—allows users to configure Web Engagement pacing settings in Agent Group objects.

Note: Once you have enabled these privileges, as shown below, you must grant the appropriate privileges to any users who need to use the related functionality.

Here is how to set up these privileges:

Prerequisites

You have created at least one Genesys Web Engagement Server application in the Configuration Server (see Creating a Node Application) and imported the **metadata xml** file for the application.

1. Open Genesys Administrator.
2. Go to **Account/Roles** and open your role (or create a new one).
3. Create a new Role dedicated to the Genesys Web Engagement Management UI and name it accordingly. For example, GWE852_Admin.
4. Go to the **Role Privileges** tab. In the **Add/Remove Products** list, select the **Genesys Web Engagement** product. The privileges **Access to GWE Categories Management** and **Access to GWE Pacing Configuration** will appear.
5. Change the state of both privileges to **Allowed**.
6. Click the **Save & Close** button.

Reporting

Web Engagement release 8.5.2 does not use Genesys Data Processing Server for reporting purposes. Instead, it relies on Elasticsearch 6.8 capabilities.

Prerequisites

The following services must be installed, configured, and started:

- Cassandra 3.11.x
- ElasticSearch 6.8
- GWE Server 8.5.2, with cluster dispatcher enabled for reporting.
- Pulse 9.0.000.xx, with an appropriate user ID with permissions to create and modify pulse templates, widgets, etc.
- Pulse Pull Collector, which is located in the root folder of Pulse 9.0.000.xx

Configuration

Enable cluster dispatcher

To enable cluster dispatcher for reporting, do the following:

- Go to the GWE Cluster Application Options.
- In the **cluster-dispatchers** section, set the **startup-reporting** option to **reporting-dispatcher:reporting**.
- Restart the server.

Install and configure Kibana

- Download and unpack Kibana 6.8.1
- Read the official documentation for configuring Kibana
- In the **kibana.6.8.1/config/kibana.yml** file, set the **kibana.index** option to the following value:
  ```
  gpe.kibana
  ```
- Start the Kibana server and confirm that **gpe.kibana** is successfully created.
- Stop the Kibana server. (This is required to prevent the Kibana server from overriding the customization and configuration settings you'll make in the next steps.)

Kibana customization and configuration

- Copy and replace the customization files from **GWE/tools/kibana** to the root Kibana folder, **kibana.6.8.1**
- Import the visualization metadata:
  - Run **GWE/server/etc/reporting/addCC.bat** (Windows) or **addCC.sh** (Linux):
    ```
    addCC [gweUrl] [gweIndexName] [pulseUrl] [pulseUser] [kibanaUrl]
    ```

    Where :

    [gweUrl] = the GWE Server URL where the Reporting API is available (do not include a forward slash "/" at the end)

    [gweIndexName] = the GWE Elasticsearch index name; use the default gpe_events

    [pulseUrl] = the Pulse URL where Pulse was configured (do not include a forward slash "/" at the end)

    [pulseuser] = the username for importing templates. Make sure the user has appropriate permissions to write Pulse data

    [kibanaurl] = the Kibana Server URL (do not include a forward slash "/" at the end)

    For example: `addCC.bat http://localhost:9081 gpe_events http://localhost:8040 default http://localhost:5601`
- Start the Kibana server

Pulse Widgets

After completing the reporting configuration, the following should be listed in the Pulse widgets:

- GWE - Engagement Summary Today
- GWE - Rates Today
- GWE - Authenticated Visitors Today
- GWE - New Vs Returning Today
- GWE - Web Visit Summary
- GWE - Web Traffic Summary

For information about how to create and edit widgets, see Add a Widget in the Pulse documentation.

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 Web Engagement server.
- 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.

> **Important**
>
> Genesys performs security testing with OWASP Zed Attack Proxy (ZAProxy) to protect the Genesys Web Engagement solution against known vulnerabilities. For details, see Security Testing with ZAProxy.

Genesys Web Engagement includes additional security configurations that can be used with your GWE installation:

- Secure Sockets Layer (SSL) — Load SSL certificates and configure Jetty.
- Transport Layer Security (TLS) — Configure TLS for Genesys and Web Engagement servers.
- Authentication — Enable authentication for the Web Engagement Server, Interaction Workspace, and the Engagement Strategy.
- Cassandra Security — Establish a secure channel between your client and Cassandra coordinator notes.

Next Steps

After you configure security for Genesys Web Engagement, you can configure features to enable additional functionality.

Security Tips for Third-Party Components

Web Engagement can't do its job without the help of these third-party Java-based applications:

- Cassandra
- Elasticsearch

Because Genesys doesn’t create these products, we can't control their dependencies. This means that they are subject to vulnerabilities that would not otherwise affect Web Engagement. Please pay attention to the following recommendations as you are securing your Web Engagement environment.

Cassandra

Be sure that:

- This cluster is located in the secure zone.
- Access to Cassandra hosts (default ports 9042, 9160, 7000, 7001) is granted only for a limited set of client connections.

By default, Cassandra will not open a JMX port, but if you do need to open one, please consider that an open JMX connection is a security exposure, so pay attention to securing the JMX port by taking care of things like these:

- Limit access to specific IP addresses.
- Apply authentication parameters.

Elasticsearch

GWE 8.5.2 relies on the external Elasticsearch 6.8 that is installed by customers. However, X-Pack security is not currently supported by GWE. Be sure that:

- Elasticsearch cluster is located in the secure zone.
- Access to Elasticsearch hosts (default ports 9200, 9300) is granted only for a limited set of IP addresses.

Secure Connections to HTTP Clients

The Jetty web server supplied with the Genesys Web Engagement solution includes a pre-configured, self-signed certificate. This allows you to use HTTPS out of the box in a lab deployment.

For a production deployment, you should use a certificate issued by a third-party Certificate Authority. The procedures on this page provide examples of ways to load SSL certificates and configure Jetty. These examples may vary depending on your environment.

Loading an SSL Certificate and Private Key into a JSSE Keystore

> **Important**
> 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.

Start

1. 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 -importcert -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.

> **Important**
> 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 **launcher.ini** configuration file.

End

Next Steps

- Configuring Launcher

Configuring Launcher

Prerequisites

- You completed Loading an SSL Certificate and Private Key into a JSSE Keystore

Start

1. Modify configuration files:
   - Windows
   1. Open the configuration file, ***Web Engagement Root Directory*/server/launcher.ini**, in a text editor.
   2. Find the block of SSL-related parameters, starting from *-Dtrusted-ca*
   3. Fulfill parameters:
      - *-Dtrusted-ca* - Path to trusted CA PEM file or JKS truststore file or SHA-1 Thumbprint for MSCAPI storage.
      - *-Dprovider* - (optional) Type of security provider used. Supported values are PEM, JKS and

PKCS11. The provider type will be detected automatically if it is not specified.

- *-Dtruststore-password* - Password for trust store if trusted CA is in the JKS format.
- *-Dcertificate-key* - Unencrypted private key in PEM format or Certificate SHA-1 Thumbprint for MSCAPI storage. Ignored for JKS storage.
- *-Dkeystore-password* - Password for key store if key storage is in the JKS format.
- *-Dkey-entry-password* - Additional password if the private key is encrypted by its own password.
- *-Dcertificate* - Client certificate file in PEM format or JKS keystore file or SHA-1 Thumbprint for MSCAPI storage.

- Linux\CentOS

1. Open the configuration file, ***Web Engagement Root Directory*/server/setenv.sh**, in a text editor.
2. Find the block of SSL-related parameters, starting from *#PROVIDER*.
3. Uncomment and fulfill parameters:
   - *TRUSTED_CA* - Path to trusted CA PEM file or JKS truststore file or SHA-1 Thumbprint for MSCAPI storage.
   - *PROVIDER* - (optional) Type of security provider used. Supported values are PEM, JKS and PKCS11. The provider type will be detected automatically if it is not specified.
   - *TRUSTSTORE_PASSWORD*' **-** *Password for trust store if trusted CA is in the JKS format.*
   - *PRIVATE_KEY* - Unencrypted private key in PEM format or Certificate SHA-1 Thumbprint for MSCAPI storage. Ignored for JKS storage.
   - *KEYSTORE_PASSWORD* - Password for key store if key storage is in the JKS format.
   - *KEYENTRY_PASSWORD* - Additional password if the private key is encrypted by its own password.
   - *CERTIFICATE* - Client certificate file in PEM format or JKS keystore file or SHA-1 Thumbprint for MSCAPI storage.

- 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 the keystore has a password, 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://*your_IP*:8743/

Next Steps

- Return to the Genesys Web Engagement Security page.

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 the keystore has a password, 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://*your_IP*:8743/

Next Steps

- Return to the Genesys Web Engagement Security page.

Transport Layer Security (TLS) with Genesys Servers

Genesys Web Engagement supports the Transport Layer Security (TLS) protocol to secure data exchanged with other Genesys components. For details about TLS, see the Genesys Security Deployment Guide. You can configure TLS for Web Engagement by completing the procedures on this page.

Configuring TLS for Genesys Servers

To configure the TLS parameters for Genesys servers such as Configuration Server, Message Server, Interaction Server, and so on, see Configuring TLS Parameters in Configuration Manager.

Configuring TLS for Web Engagement Servers

To enable TLS support for the Genesys Web Engagement Server, you must do the following:

1. Have properly installed trusted certificates for the Genesys servers.
2. Configure TLS options for the Web Engagement Server application.
3. Configure the appropriate connections between the Web Engagement Server application and the necessary Genesys servers through secure ports.

Configuring TLS Options

The Genesys Web Engagement Server includes the following TLS-related configuration options in its [security] section.

Option	Default Value	Mandatory	Changes Take Effect	Description
provider	none	no	after restart	Type of trusted storage Valid values: MSCAPI, PEM or JKS. If empty, TLS support is disabled.
trusted-ca	none	no	after restart	Specifies the name of the trusted store file which holds the public certificate to verify the server.

Option	Default Value	Mandatory	Changes Take Effect	Description
				Applicable for PEM and JKS trusted storage types only. Valid values: valid file name (including path)
truststore-password	none	no	after restart	Password for the JKS trusted storage. Valid values: any string

See Configuring Trusted Stores below for details about configuration for a specific type of store (PEM, JKS, MSCAPI).

Configuring Trusted Stores

PEM Trusted Store

PEM stands for "Privacy Enhanced Mail", a 1993 IETF proposal for securing email using public-key cryptography. That proposal defined the PEM file format for certificates as one containing a Base64-encoded X.509 certificate in specific binary representation with additional metadata headers.

PEM certificate trusted store works with CA certificate from an X.509 PEM file. It is a recommended trusted store to work on Linux systems.

Complete the steps below to work with the PEM certificate trusted store:

Start

1. Configure TLS for Genesys servers to use certificates signed by CA certificate **certificateCA.crt**.
2. Place the trusted CA certificate in PEM format on the Genesys Web Engagement Server application host. To convert a certificate of another format to .pem format you can use the OpenSSL tool. For example:
   - Convert a DER file (.crt .cer .der) to PEM:
     ```
     openssl x509 -inform der -in certificateCA.crt -out certificateCA.pem
     ```
   - Convert a PKCS#12 file (.pfx .p12) containing a private key and certificates to PEM:
     ```
     openssl pkcs12 -in certificateCA.pfx -out certificateCA.pem -nodes
     ```
     You can add **-nocerts** to only output the private key or add **-nokeys** to only output the certificates.
3. In Genesys Administrator, navigate to **Provisioning > Environment > Applications** and open your Web Engagement Server application.
4. Click the **Options** tab and navigate to the [security] section.
5. Set the **provider** option to PEM.
6. Set the **trusted-ca** option to the path and file name for your trusted CA in PEM format on the Genesys Web Engagement Server application host.

7. Click **Save & Close**.

End

JKS Trusted Store

A Java KeyStore (JKS) is a repository of security certificates used, for instance, in SSL/TLS encryption. The Java Development Kit provides a tool named keytool to manipulate the keystore.

Complete the steps below to work with the JKS certificate trusted store:

Start

1. Configure TLS for Genesys servers to use certificates signed by CA certificate **certificateCA.crt**.
2. Import the CA certificate to an existing Java keystore using keytool:
   - Run the `keytool` command with option `-alias` set to `root`:
     ```
     keytool -import -trustcacerts -alias root -file certificateCa.crt -keystore
     /path/to/keysore/keystore.jks
     ```
   - Enter the keystore password in command line prompt - for example:
     ```
     Enter keystore password: somepassword
     ```
3. In Genesys Administrator, navigate to **Provisioning > Environment > Applications** and open your Web Engagement Server application.
4. Click the **Options** tab and navigate to the [security] section.
5. Set the **provider** option to JKS.
6. Set the **trusted-ca** option to the path and file name for your JKS trusted storage type on the Genesys Web Engagement Server application host.
7. Set the **trusted-pwd** option to the password defined for your keystore in Step 2.
8. Click **Save & Close**.

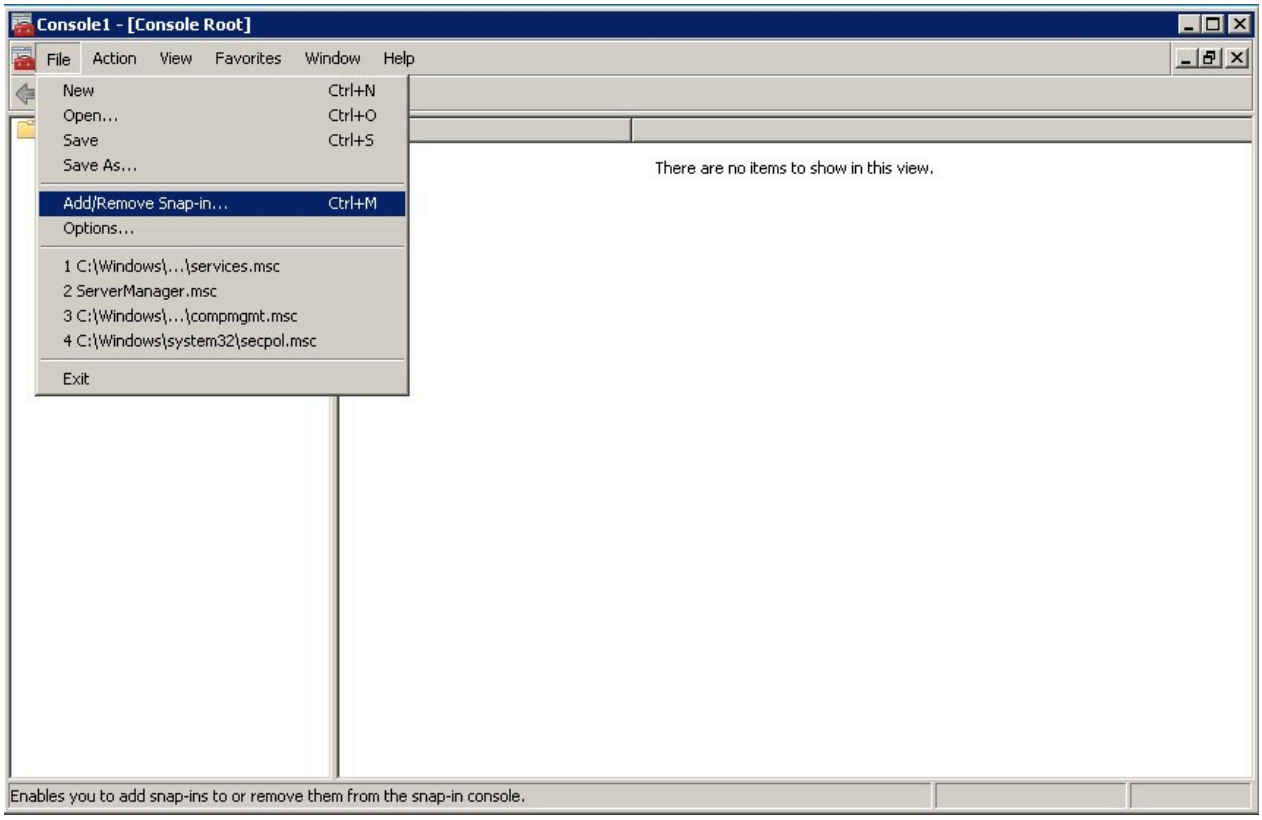
End

MSCAPI Trusted Store

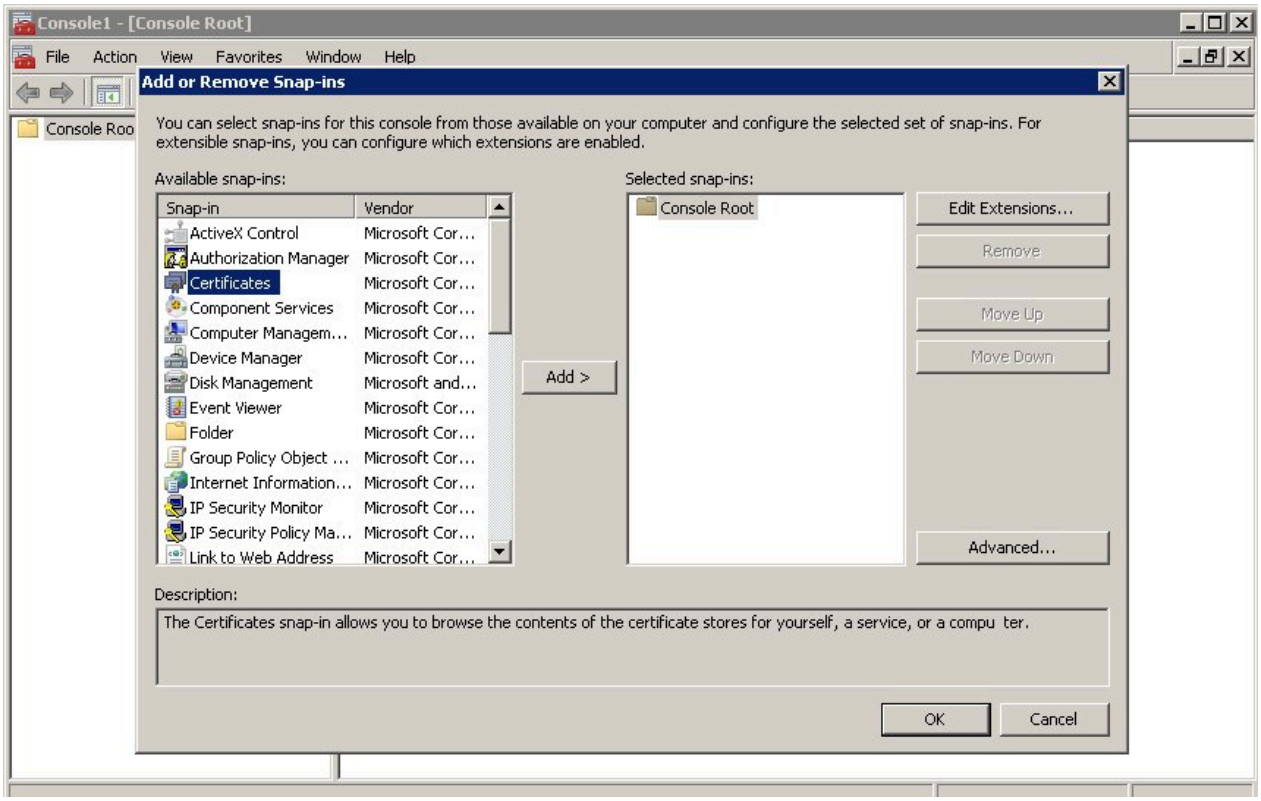
Complete the steps below to work with the MSCAPI certificate trusted store:

Start

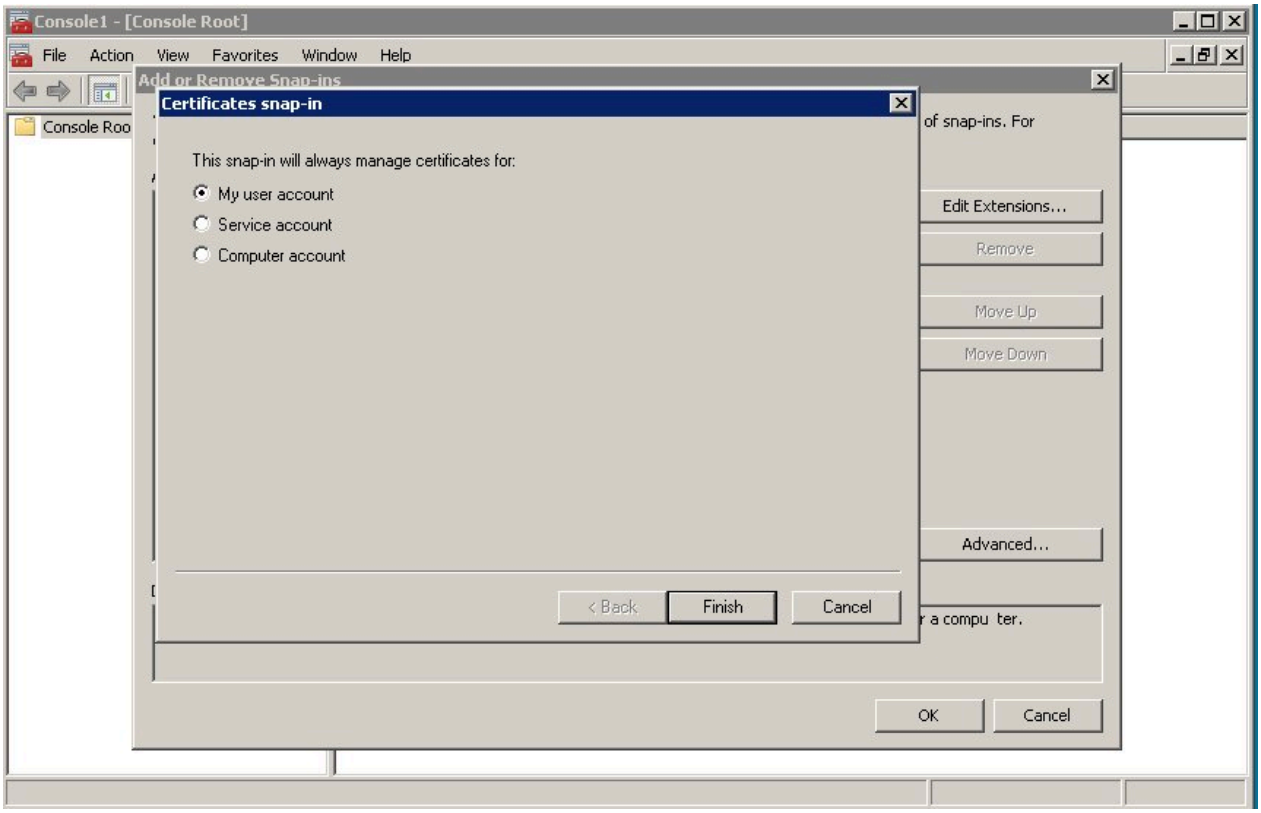
1. Configure and tune TLS for Genesys servers to use certificates signed by the same CA.
2. If the Web Engagement Server is running on a different host, copy the trusted CA certificate to this host.
3. Import the CA certificate to WCS via Certificates Snap-in on the Web Engagement Server host by launching the MMC console. Enter `mmc` at the command line.
4. Select **File > Add/Remove Snap-in...** from the main menu.



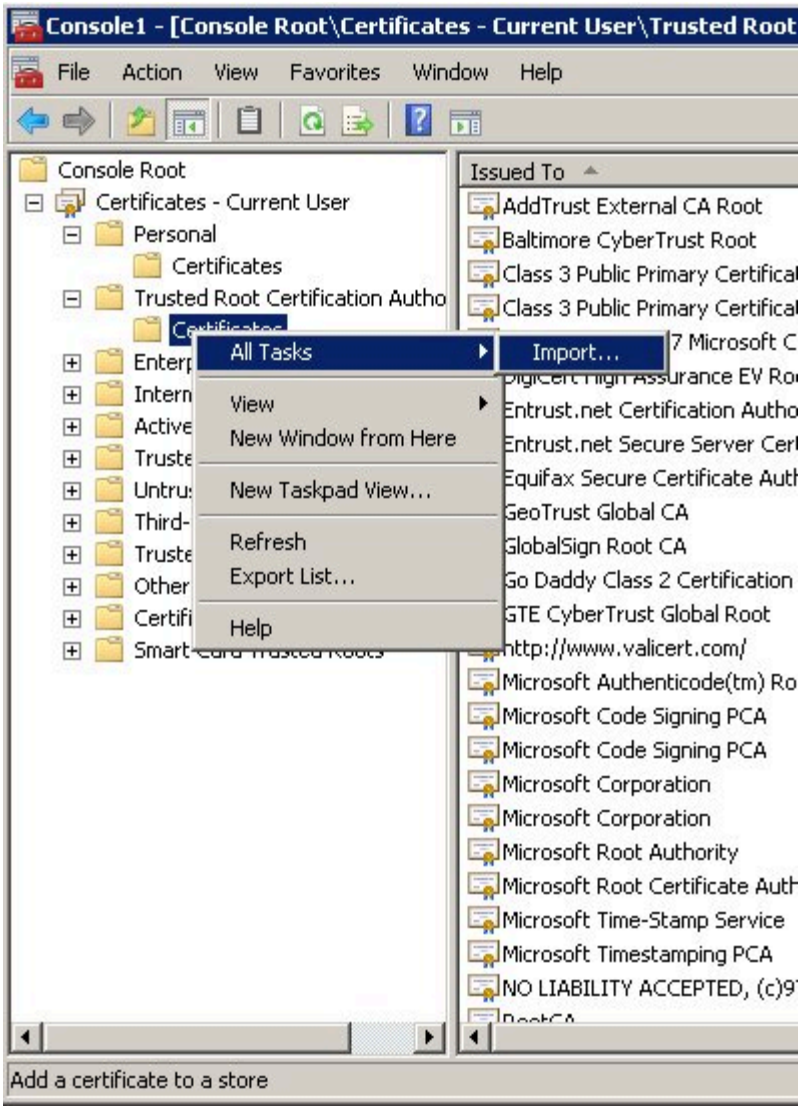
5. Select **Certificates** from the list of available snap-ins and click **Add**.



6. Select the account to manage certificates for and click **Finish**. It is important to place certificates under the correct Windows account. Some applications are run as services under the Local Service or System account, while others are run under user accounts. The account chosen in MMC must be the same as the account used by the application which certificates are configured for, otherwise the application will not be able to access this WCS storage.



7. Click **OK**.
8. Import a certificate. Right-click the "Trusted Root Certification Authorities/Certificates" folder and choose `All Tasks > Import...` from the context menu. Follow the steps presented by the Certificate Import Wizard. Oonce finished the imported certificate appears in the certificates list.



9. In Genesys Administrator, navigate to **Provisioning > Environment > Applications** and open your Web Engagement Server application.
10. Click the **Options** tab and navigate to the [security] section.
11. Set the **provider** option to `MSCAP`.
12. Click **Save & Close**.

End

Next Steps

- Return to the Genesys Web Engagement Security page.

Authentication

You can enable secure communications with the History REST API by completing the procedures below to implement authentication. If you do enable authentication, then all the clients of the API must use the authentication scheme and credentials. Two common clients of the API are the Genesys Web Engagement Plug-in for Interaction Workspace and the Engagement Strategy. See "Configuring Authentication in Interaction Workspace" and "Configuring Authentication in the Engagement Strategy" for details.

Configuring Authentication in the Web Engagement Server

Complete the steps below to enable authentication for the History REST API.

Start

1. In Genesys Administrator, navigate to **Provisioning > Environment > Applications**, select the Genesys Web Engagement Server application, and click **Edit...**.
2. Click the **Options** tab and scroll down to the **[security]** section.
3. Set the following options:
   - auth-scheme
   - user-id
   - password
4. Click **Save & Close**.

End

Configuring Authentication in Interaction Workspace

If you enable authentication for the History REST API and use the Genesys Web Engagement Plug-in for Interaction Workspace, then you must complete the steps below to enable authentication for the plug-in.

Prerequisites

- You completed "Configuring Authentication in the Web Engagement Server".

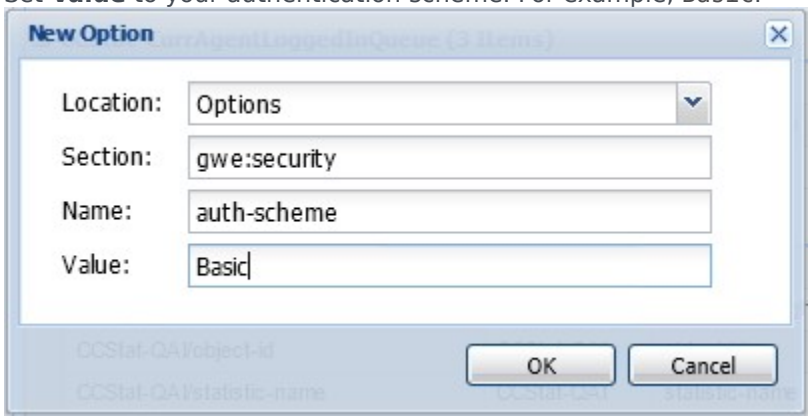
Start

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

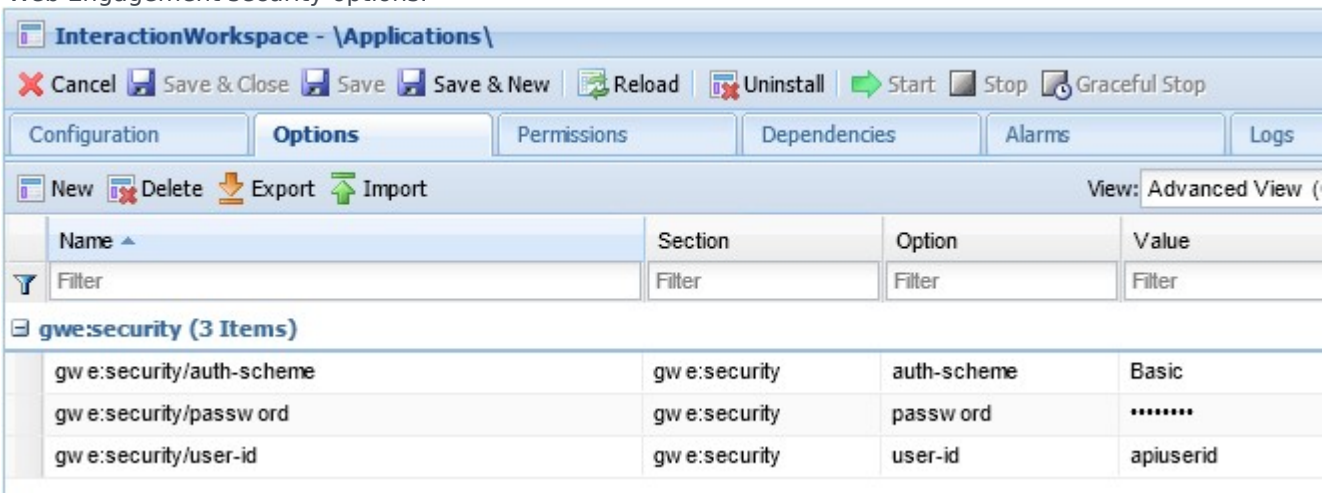


   d. Click **OK**

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:



New security options for Genesys Web Engagement.

5. Click **Save & Close**.

End

Configuring Authentication in the Default Engagement Strategy

Complete the steps below to add security credentials to the default SCXML strategy to support authentication for the REST API.

Prerequisites

- You completed "Configuring Authentication in the Web Engagement Server".
- Your SCXML strategy uses the REST API. See Customizing the Engagement Strategy for details.

Start

1. Open the Engagement Logic strategy in Composer.
2. Open **default.workflow**.
3. Find and set your user and password credentials in the list of properties at the Entry point (Start).
4. Regenerate the SCXML and follow the Web Engagement application deployment procedure.

End

Next Steps

- Return to the Genesys Web Engagement Security page.

Cassandra Security

Unauthorized access to Cassandra data is possible at several points:

- Direct access via "standard" interfaces: Thrift and CQL
- Access to data traveling through the network
- Access to data files that Cassandra stores on hard drives

Cassandra's default configuration provides mechanisms to secure direct interfaces (through authentication and authorization) and network traffic (through the use of TLS). The data stored on hard drives can be secured either by third-party commercial offerings or with some development investments.

The following sections describe how to provide secure access to:

- External Cassandra

External Cassandra

Securing access interfaces

You can secure your access interfaces based on an authentication and authorization scheme. In other words, Cassandra needs to know:

- Who is trying to access the system
- Whether they are allowed to access the system at all
- If so, which data they should have access to

With the default setup, anybody is allowed to access all the data.

Authentication

Authentication (who) is managed by the **authenticator** parameter in the **cassandra.yaml** file. By default, GWE 8.5 only provides a login/password authentication scheme with the use of **PasswordAuthenticator**.

Procedure

Start

1. Edit ***Cassandra installation directory*/conf/cassandra.yaml**:

1. Change the **authenticator** parameter to `PasswordAuthenticator`. Note that by default, the **authenticator** option is set to `AllowAllAuthenticator`.
2. Tune your **system_auth** keyspace replication according to the DataStax system_auth documentation.

2. Restart the Cassandra client.
   The default superuser name and password that you use to start the client is stored in Cassandra:

   ```
   <client startup string> -u cassandra -p cassandra
   ```

3. Start cqlsh using the superuser name and password (cassandra):
   ```
   ./cqlsh -u cassandra -p cassandra
   ```
4. Deactivate the default superuser, which is called `cassandra`, as mentioned above. This step is optional but highly recommended.
   1. Create a superuser with another name.
   2. Log in as the newly create superuser.
   3. Change the `cassandra` user password to something long and incomprehensible, and then forget about it. It won't be used again.
   4. Take away the `cassandra` user's superuser status.
5. Use the following CQL 3 statements to set up user accounts and then grant permissions to access the database objects:
   - CREATE USER
   - GRANT
6. Set the new user name and password to the values of the [cassandraKeyspace] userName and password options for the Web Engagement Cluster application.

End

Authorization

Authorization (which data) is managed by the authorizer in the GWE cluster configuration options. Cassandra offers a familiar relational database **GRANT/REVOKE** paradigm to grant or revoke permissions for accessing data.

Procedure

Start

Edit ***Cassandra installation directory*/conf/cassandra.yaml**:

1. Change the **authorizer** parameter to `CassandraAuthorizer`. Note that by default, the **authorizer** option is set to `AllowAllAuthorizer`.
2. Tune your **system_auth** keyspace replication according to the DataStax system_auth documentation. Note that the validity period for permissions caching is 2000 ms.

For more information about permissions see the DataStax Object permissions documentation.

End

CQL supports these authorization statements:

- GRANT
- LIST PERMISSIONS
- REVOKE

Resource Access Point Configuration

Prerequisites

- The Cassandra Resource Access Point applications are created and configured

Procedure

Start

For all Cassandra Resource Access Points:

1. Open the **cassandraClient** (previously **cluster**) configuration option section.
2. Set the **userName** option to the name of an already-created user.
3. Set the **password** option to the user's password.

End

Securing Network Traffic

The client-to-node and node-to-node traffic in your Cassandra deployment may require protection. They can both be secured by using SSL (Secure Sockets Layer) encryption.

Client-to-Node Encryption

Client-to-node encryption uses SSL to protect data that is traveling from client machines to a database cluster. It does this by establishing a secure channel between the client and the coordinator node.

Prerequisites

- You must install Java Cryptography Extension (to enable 256-bit encryption).
- All nodes must have all of the relevant SSL certificates. See Preparing server certificates.

Procedure

Start

On each node that belongs to the Cassandra cluster, edit ***Cassandra installation directory*/conf/cassandra.yaml**:

1. Set the client_encryption_options/enabled option to `true`.
2. Set the appropriate paths to your .keystore and .truststore files in the client_encryption_options/keystore and client_encryption_options/truststore options.
3. Provide the required passwords in client_encryption_options/keystore_password and client_encryption_options/truststore_password. The passwords must match the passwords used when generating the keystore and the truststore.
4. To enable client certificate authentication, set the client_encryption_options/require_client_auth option to `true`.

End

Node-to-Node Encryption

Node-to-node encryption uses SSL to protect data being transferred between cluster nodes. This includes node-to-node gossip communication.

Prerequisites

- You must install Java Cryptography Extension (to enable 256-bit encryption).
- All nodes must have all of the relevant SSL certificates. See Preparing server certificates.

Procedure

Start

On each node that belongs to the Cassandra cluster, edit ***Cassandra installation directory*/conf/cassandra.yaml**:

1. Set the server_encryption_options/internode_encryption option to one of the following:
   - **all**—Cassandra encrypts all internodal traffic
   - **dc**—Cassandra encrypts all traffic between datacenters
   - **rack**—Cassandra encrypts all traffic between racks
2. Set the appropriate paths to your .keystore and .truststore files in the server_encryption_options/keystore and server_encryption_options/truststore options.
3. Provide the required passwords in server_encryption_options/keystore_password and server_encryption_options/truststore_password. The passwords must match the passwords used when generating the keystore and the truststore.
4. To enable client certificate authentication, set the server_encryption_options/require_client_auth option to `true`.

End

Resource Access Point Configuration

Prerequisites

- The Cassandra Resource Access Point applications are created and configured.

Procedure

Start

For all Cassandra Resource Access Points:

1. Open the properties for the port with an ID of `native`.
2. Set this port to `secured`.
3. If your Cassandra cluster has its own properly configured security certificate, enable Mutual TLS for this port.

End

Web Engagement Application Configuration

Prerequisites

- The Web Engagement cluster applications are created and configured.
- The Web Engagement cluster applications have their own properly configured security certificates.

Procedure

Start

- For every Web Engagement application:
  1. Make sure its security certificate was correctly configured.
  2. For security fine tuning, read the description of the TLS options in Configuration Options Reference Manual.
- For every Web Engagement cluster application:
  - Set the Web Engagement application port to `secured`.

End

Using cqlsh

If you are planning to use the cqlsh standard utility with encryption, please consult the Datastax

documentation.

Securing access interfaces

You can secure your access interfaces based on an authentication and authorization scheme. In other words, Cassandra needs to know:

- Who is trying to access the system
- Whether they are allowed to access the system at all
- If so, which data they should have access to

With the default setup, anybody is allowed to access all the data.

Authentication

Authentication (who) is managed by the **authenticator** parameter in the **cassandra.yaml** file. By default, GWE 8.5 only provides a login/password authentication scheme with the use of **PasswordAuthenticator**.

Procedure

Start

1. Change the authenticator option to `PasswordAuthenticator`.
   By default, the **authenticator** option is set to `AllowAllAuthenticator`.
2. Tune your **system_auth** keyspace replication according to the DataStax system_auth documentation.
3. Restart the Cassandra client.
   The default superuser name and password that you use to start the client is stored in Cassandra:

   ```
   <client startup string> -u cassandra -p cassandra
   ```
4. Start cqlsh using the superuser name and password (cassandra):
   ```
   ./cqlsh -u cassandra -p cassandra
   ```
5. Deactivate the default superuser, which is called `cassandra`, as mentioned above. This step is optional but highly recommended.
   1. Create a superuser with another name.
   2. Log in as the newly create superuser.
   3. Change the `cassandra` user password to something long and incomprehensible, and then forget about it. It won't be used again.
   4. Take away the `cassandra` user's superuser status.
6. Use the following CQL 3 statements to set up user accounts and then grant permissions to access the database objects:
   - CREATE USER
   - GRANT

7. Set the new user name and password to the values of the [cassandraKeyspace] userName and password options for the Web Engagement Cluster application.

End

Authorization

Authorization (which data) is managed by the authorizer in the GWE cluster configuration options. Cassandra offers a familiar relational database **GRANT/REVOKE** paradigm to grant or revoke permissions for accessing data.

Procedure

Start

1. Change the authorizer|authorizer option in the cassandra.yaml file to `CassandraAuthorizer`.
   By default, the **authorizer** option is set to `AllowAllAuthorizer`.
2. Tune your **system_auth** keyspace replication according to the DataStax system_auth documentation.
   Note that the validity period for permissions caching is 2000 ms.
   For more information about permissions see the DataStax Object permissions documentation.

End

CQL supports these authorization statements:

- GRANT
- LIST PERMISSIONS
- REVOKE

Securing Network Traffic

The client-to-node and node-to-node traffic in your Cassandra deployment may require protection. They can both be secured by using SSL (Secure Sockets Layer) encryption.

Client-to-Node Encryption

Client-to-node encryption uses SSL to protect data that is traveling from client machines to a database cluster. It does this by establishing a secure channel between the client and the coordinator node.

Prerequisites

- You must install Java Cryptography Extension (to enable 256-bit encryption).
- All nodes must have all of the relevant SSL certificates. See Preparing server certificates.
- To enable client-to-node SSL, you must set the *client_encryption_options* options in the cassandra.yaml file.

- All Web Engagement node applications must have correctly configured security certificates. To learn more about how to configure your node application certificates, read the description of the TLS options in Configuration Options Reference Manual.

Procedure

Start

For each node in the Cassandra cluster, open the cassandra.yaml file and:

1. Set the client_encryption_options\enabled option to `true`.
2. Set the appropriate paths to your .keystore and .truststore files in the client_encryption_options\keystore and client_encryption_options\truststore options.
3. Provide the required passwords in client_encryption_options\keystorePassword and client_encryption_options\truststorePassword. The passwords must match the passwords used when generating the keystore and the truststore.
4. To enable client certificate authentication, set the client_encryption_options\require_client_auth option to `true`.
5. For security fine tuning, read the description of the TLS options in Configuration Options Reference Manual.

End

Node-to-Node Encryption

Node-to-node encryption uses SSL to protect data being transferred between cluster nodes. This includes node-to-node gossip communication.

Prerequisites

- You must install Java Cryptography Extension (to enable 256-bit encryption).
- All nodes must have all of the relevant SSL certificates. See Preparing server certificates.
- To enable node-to-node SSL, you must set the *server_encryption_options* options in the cassandra.yaml file.

Procedure

Start

On each node that belongs to the Cassandra cluster:

1. Set the server_encryption_options\internode_encryption option to one of the following:
   - **all**—Cassandra encrypts all internodal traffic
   - **dc**—Cassandra encrypts all traffic between datacenters

- **rack**—Cassandra encrypts all traffic between racks

2. Set the appropriate paths to your .keystore and .truststore files in the server_encryption_options\keystore and server_encryption_options\truststore options.
3. Provide the required passwords in server_encryption_options\keystore_password and server_encryption_options\truststore_password. The passwords must match the passwords used when generating the keystore and the truststore.
4. To enable client certificate authentication, set the server_encryption_options\require_client_auth option to `true`.

End

Using cqlsh

If you are planning to use the cqlsh standard utility with encryption, please consult the Datastax documentation.

Configuring Specific Features

Genesys Web Engagement includes additional features that you can configure to enable the following functionality:

- Pacing Algorithm—Configure this algorithm to help keep a balanced workflow in your contact center by aligning the generated Web Engagement invites with the agents you have available for both proactive and reactive traffic.
- Chat Channel—Manually configure the Web Engagement Server and Chat Server to support a chat channel or specify chat as the default channel of engagement.

> **Important**
>
> The Genesys Web Engagement chat channel is now legacy.

- GeoIP Information—You can configure Web Engagement Server to collect information about a visitor's IP address, as well as converting this information into the visitor's geolocation coordinates.

Pacing

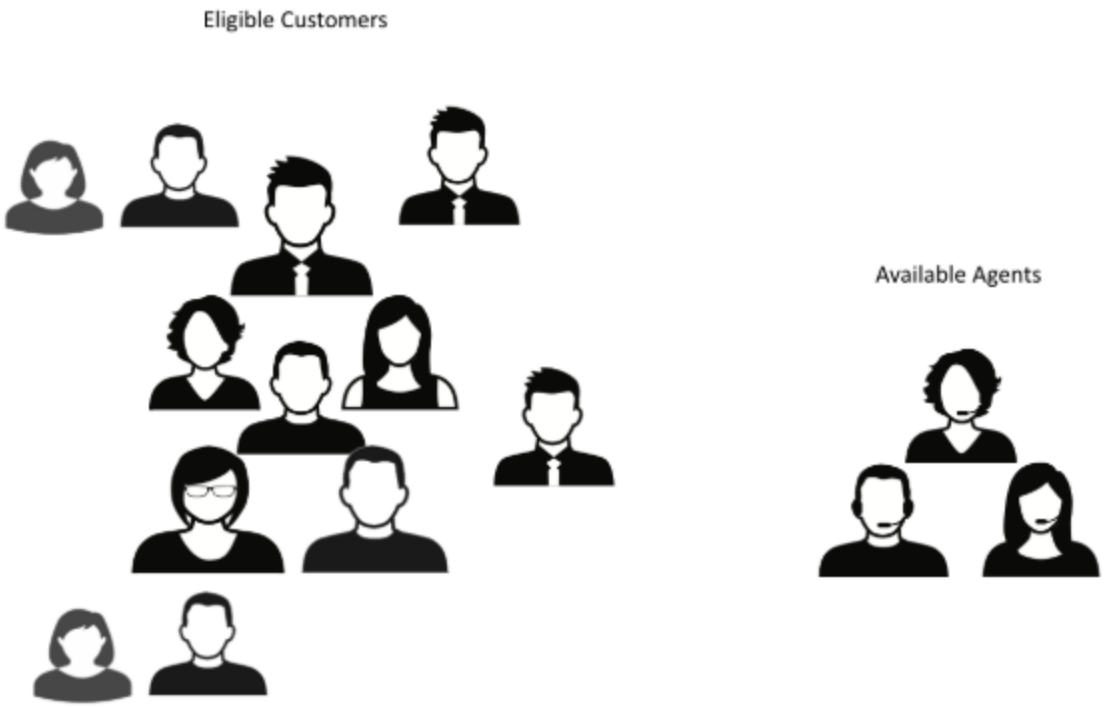
About Pacing

Many products allow you to focus on *reactive* engagement sessions, in which a customer initiates the connection with your contact center by responding to a *static* request, such as a **Chat Now** button. Genesys Web Engagement goes a step further, by predicting when it's a good time for your contact center to take the initiative—to reach out to a customer proactively. For example, Genesys Web Engagement might see that a customer is likely to abandon a purchase, so it proactively sends them a pop-up that says "Let's chat!" At that point, the customer can click an **Accept** button and your agents have an opportunity to help them complete their purchase.

Most of the information that Genesys Web Engagement works with is focused on your customers. This data helps you pinpoint the ones who are most likely to respond well to a proactive invitation.

This works great when you have enough agents:

But what if a customer accepts an engagement offer—and your agents are too busy to answer?



That's what pacing is for. A *pacing algorithm* helps you manage your agent load by comparing it to your traffic and predicting how many proactive invitations you can send out—without making your customers wait too long for a response from your agents.

For more information

- For a more in-depth look at the pacing algorithm and the usage methodology behind it, click HERE to download the "Pacing Algorithm Usage Methodology" Word document. See download tips
- If you want to combine proactive (outbound) and reactive (inbound) traffic, find out more about how Web Engagement handles dual pacing.
- You can also customize the way the pacing algorithm is used in the Engagement Logic Strategy. For details, see Accessing Pacing Information from the Engagement Logic Strategy.

To enable the pacing algorithm, complete the procedures below.

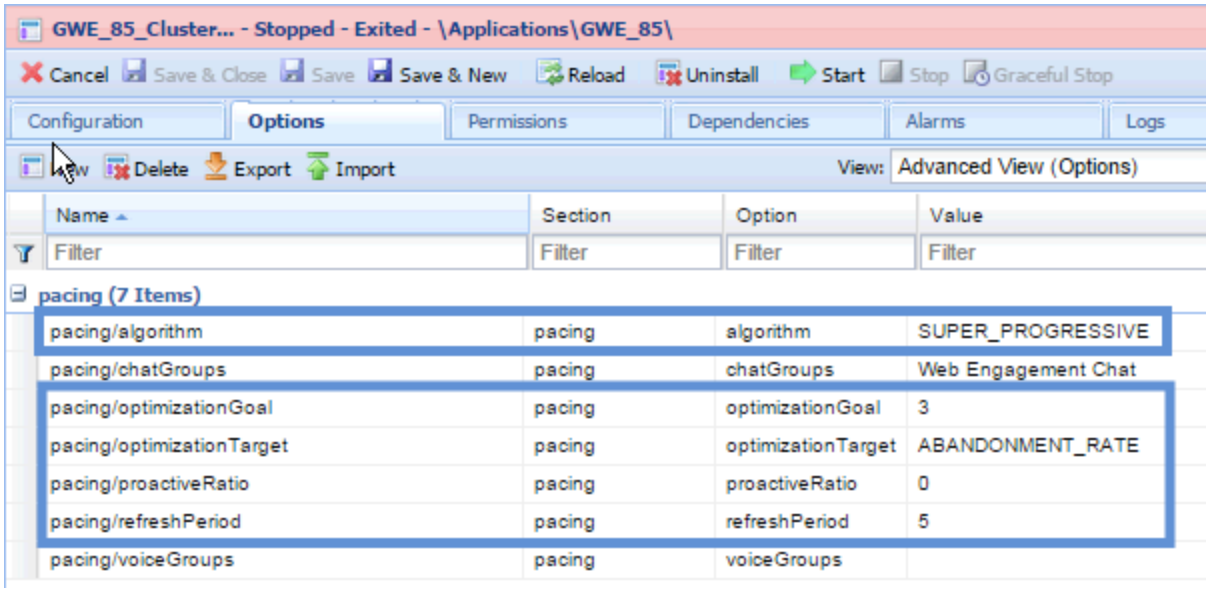
Configuring your optimization model

Note: Starting with version 8.5.000.33, Web Engagement allows you to use Agent Group objects as the primary way to configure pacing settings. You can still configure these settings in the Web Engagement Cluster or Web Engagement Server application objects—as shown in this section and the following section—but settings configured in the Agent Group objects have a higher priority.

Complete this procedure to enable the pacing algorithm to predict traffic for your Web Engagement solution.

Start

1. Open Genesys Administrator and navigate to **Provisioning > Environment > Applications**.
2. Edit the Web Engagement Cluster application and click the **Options** tab.
3. In the [pacing] section, set the algorithm option. The supported values are:
   - SUPER_PROGRESSIVE — Recommended for small agent groups (1-30 agents); only proactive traffic is predicted.
   - SUPER_PROGRESSIVE_DUAL — Recommended for small agent groups (1-30 agents); both proactive and reactive traffic are predicted.
   - PREDICTIVE_B — Recommended for bigger agent groups (start from 30 agents); only proactive traffic is predicted
   - PREDICTIVE_B_DUAL — Recommended for bigger agent groups (start from 30 agents); both proactive and reactive traffic are predicted.
4. Set the optimizationTarget option. The supported values are:
   - ABANDONMENT_RATE — Percentage of interactions that will be abandoned.
   - BUSY_FACTOR — Percentage of time during which an agent plans to be busy with an interaction-related activity.
5. Set the optimizationGoal option. The value you set for this option depends on the value you set for **optimizationTarget**:
   - If your optimization target is ABANDONMENT_RATE, Genesys recommends that you use small values. For example, from 3 to 5.
   - If your optimization target is BUSY_FACTOR, Genesys recommends that you use big values. For example, from 70 to 85.
6. If you chose a dual algorithm in Step 3 (SUPER_PROGRESSIVE_DUAL or PREDICTIVE_B_DUAL), specify a value for the proactiveRatio option.
   This option basically controls the minimal percentage of agents that will be reserved for processing proactive engagement interactions.
7. Set the refreshPeriod option. This option controls how frequently the pacing algorithm provides predictions. Genesys recommends that you use values from 1 to 5.



Pacing algorithm settings

End

Next Steps

- Configuring the Agent Groups

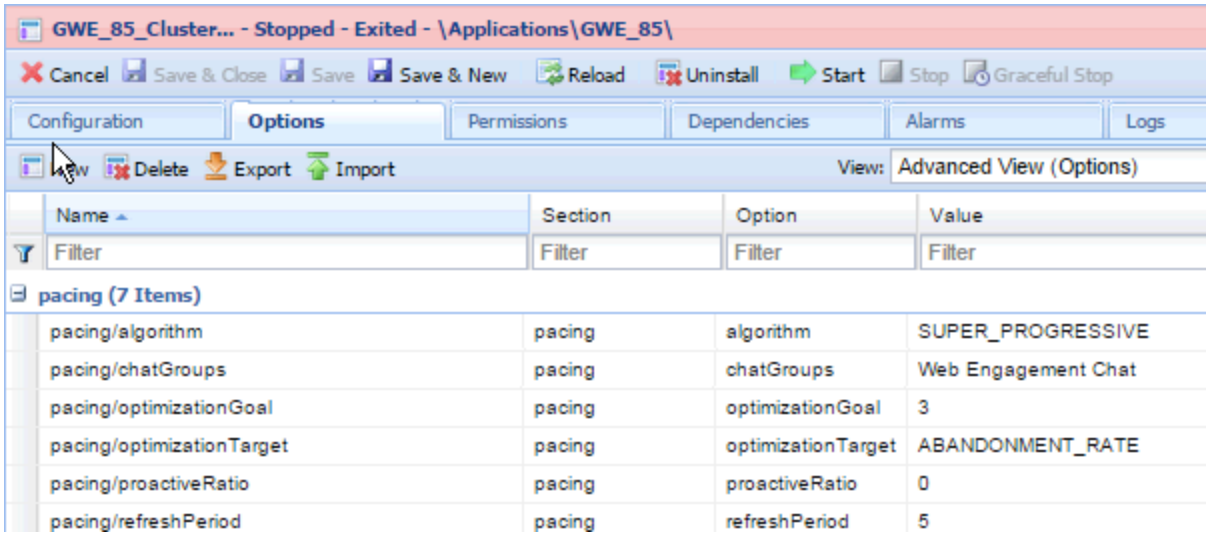
Configuring the agent groups

In the previous procedure, you set the **algorithm** option according to the size of your agent group and the type of traffic the algorithm should handle. In this procedure, you configure your agent groups for the pacing algorithm to use.

When you install Genesys Web Engagement, the Provisioning Tool automatically creates the following agent group:

- Web Engagement Chat

In the steps below, you'll confirm that the agent group was created and then add agents to it.



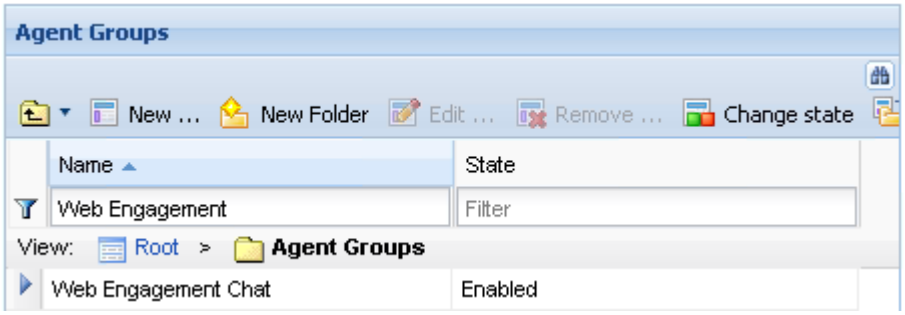
Configure new agent groups

Important

You can use your own groups instead by changing the value of the chatGroups option to the names of other groups you configured.

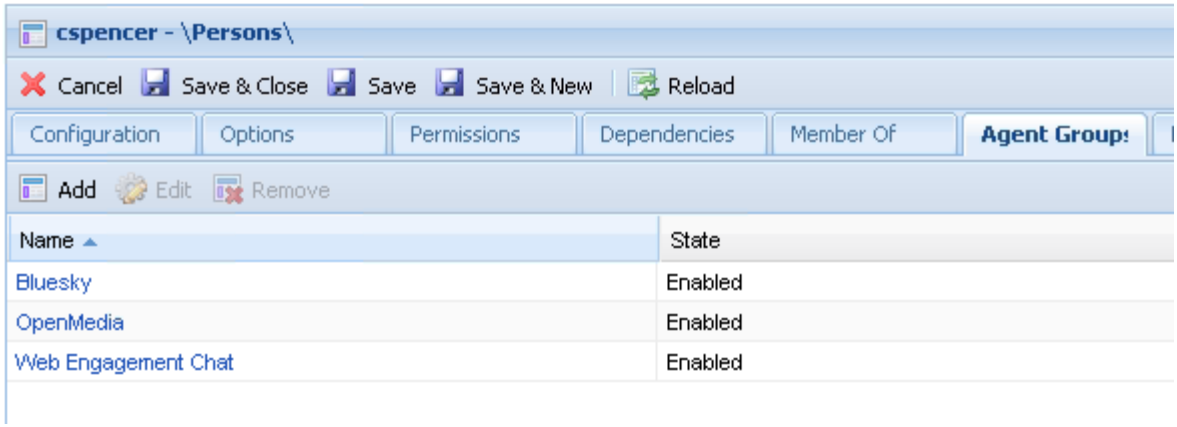
Start

1. Open Genesys Administrator and navigate to **Provisioning > Accounts > Agent Groups**. Make sure that the Web Engagement Chat group is created. You can use the filter to display the groups.



The default agent groups

2. Navigate to **Provisioning > Accounts > Users.**
3. Select an agent that should manage Web Engagement interactions and click **Edit...**.
4. Select the Agent Group tab and click **Add**. The Browse dialog opens.
5. Select one of the groups and click **OK**.



The agent named Spencer now belongs to the Web Engagement Chat group.

6. Repeat Steps 3-5 for each agent you want to add to the chat agent group.

End

Using Agent Groups to configure pacing settings

Starting with version 8.5.000.33, Web Engagement allows you to use Agent Group objects as the primary way to configure pacing settings. You can still configure these settings in the Web Engagement Cluster or Web Engagement Server application objects, but settings configured in the Agent Group objects have a higher priority.

Note: You must enable the **Access to Web Engagement Pacing Configuration** privilege for any users who need to control pacing settings.

To view and manage pacing configuration settings for Agent Groups, open the Genesys Web Engagement Management interface and select **Pacing**. To access the Genesys Web Engagement Management interface, open a browser and go to:

`http(s)://<GWE Load Balancer host>:<GWE Load Balancer port>/server`

The load balancer will redirect requests from *<GWE Load Balancer port>* to the port with the ID **ui**, as defined in the Genesys application (see Configuring the Cluster Application).

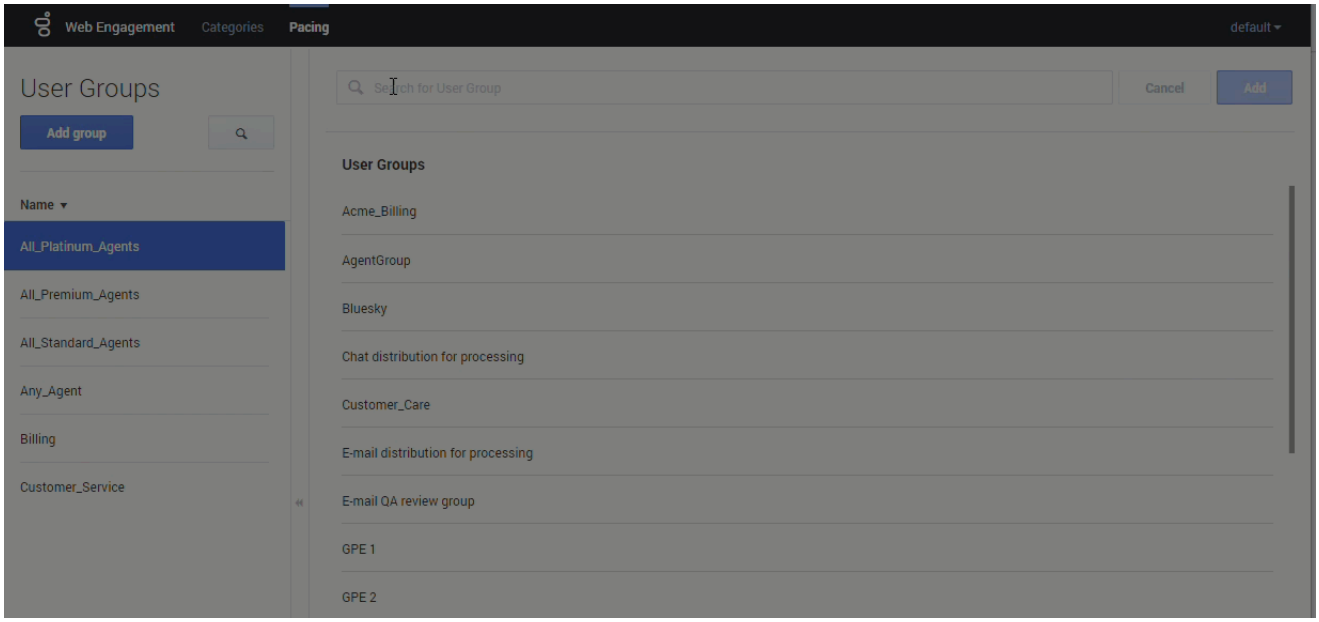
You'll be prompted for a **user ID** and **password**. After you log in, you can select **Pacing** to view and manage the pacing configuration settings. From the **Pacing** page, you can manage Agent Groups that have been added to the list, or add new groups. **Note:** You can still configure these options in the Web Engagement Cluster or Web Engagement Server application objects, but options configured on the **Pacing** management page have a higher priority.

> **Important**

To manage pacing configuration settings using the Web Engagement Management tool, your user ID must be assigned to a role with the required permissions. If you do not see any menu options, your user ID is not authorized to manage pacing configuration settings using this tool.

Adding a group

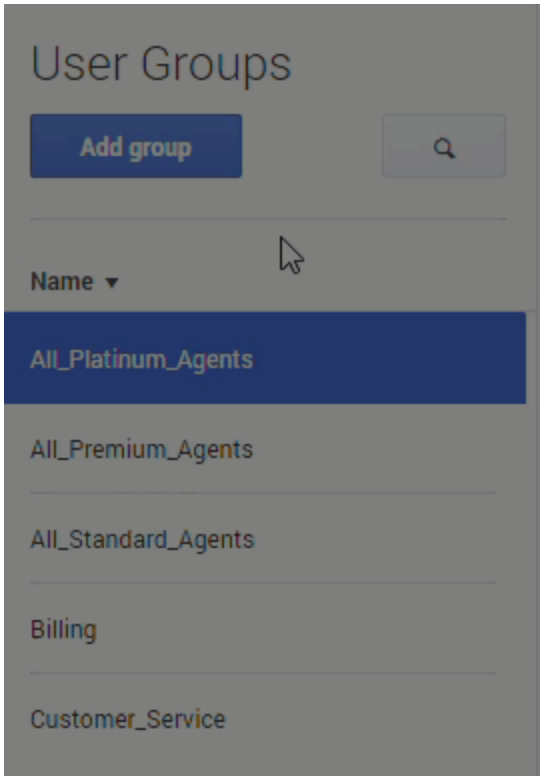
To add a new agent group, click **Add group**, and use the search bar (or browse the list) to find the group you want to add. Click **Add** to add the group to the Agent Groups list:



After a group is added to the Agent Groups list, it no longer appears in the list of available groups that can be selected when you click **Add group**.

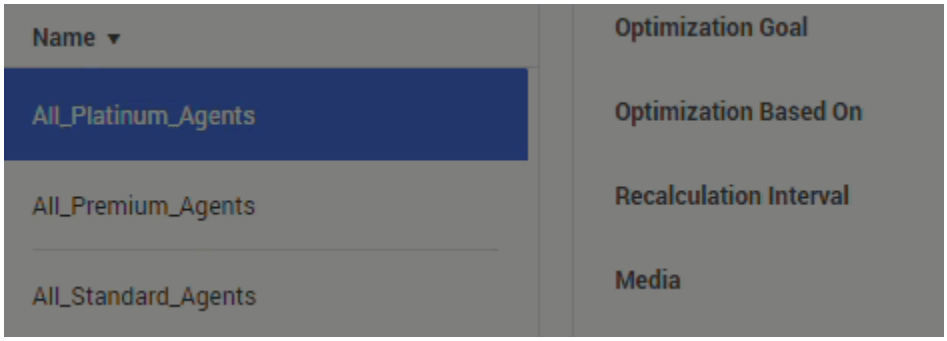
Sorting and filtering agent groups

You can sort the Agent Groups list by **Name**, or use the **search** button to filter the list or locate a specific group:



Deleting a group

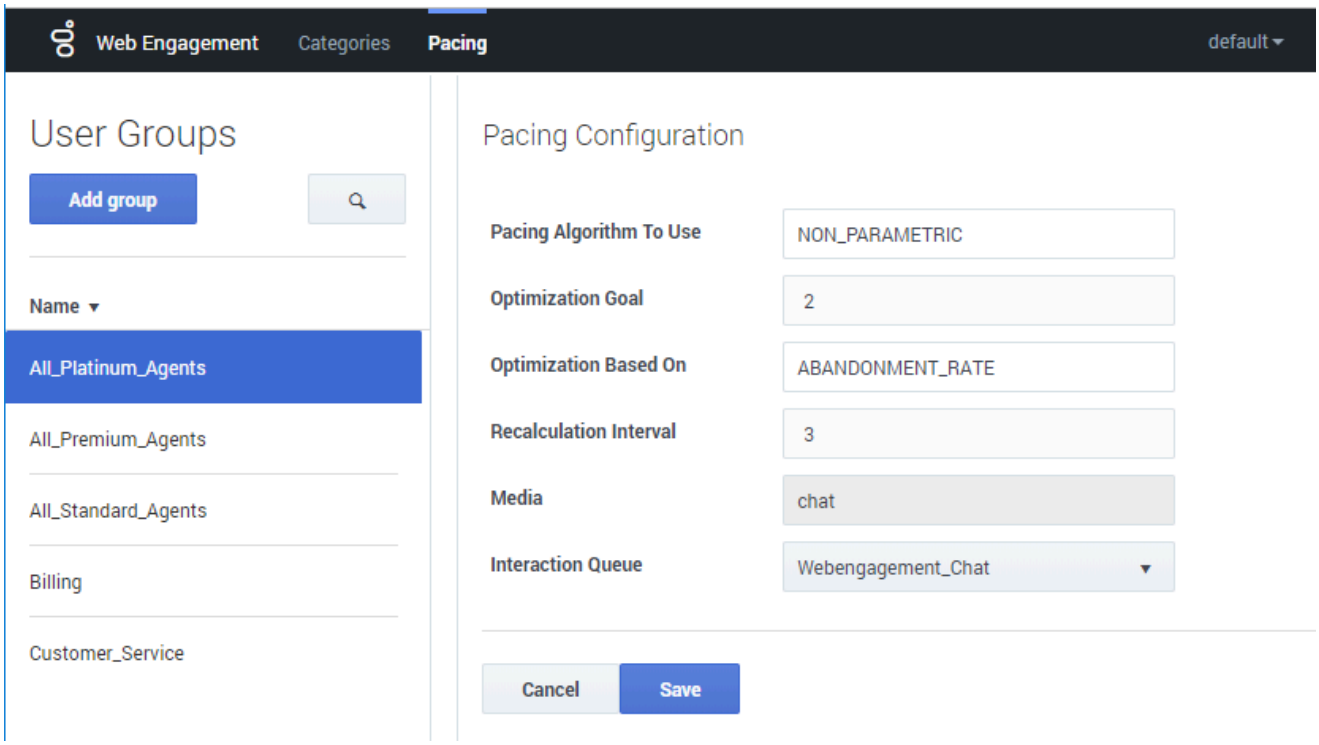
If you hover your mouse over a group, a trash icon appears, which you can click to delete the group:



After you click the trash icon, you'll be asked to confirm the deletion. Click **Delete** to remove the selected group and its configuration settings.

Editing the pacing configuration options for a group

Selecting a group will display its options on the right:



You can then select the desired pacing configuration options. (If you make an invalid entry or selection, the management tool will prompt you to correct the error.)

Pacing Algorithm To Use	Specifies the type of pacing algorithm to be used by the system. Changes take effect after server restart.
Optimization Goal	Specifies the optimal percentage value for your chosen optimization target, which can be either ABANDONMENT_RATE or BUSY_FACTOR, as specified in the **Optimization Based On** field that appears just below this field.
Optimization Based On	Specifies whether to optimize based on ABANDONMENT_RATE or BUSY_FACTOR. Changes take effect after server restart.
Percentage Of Proactive Traffic	Specifies the minimum percentage of agent resources that are reserved for handling proactive interactions. If 0 is specified, no resources are specifically allocated to handle proactive interactions, but proactive traffic is still allowed. If 100 is specified, all resources are allocated to handle proactive interactions and no reactive interactions are allowed.
Recalculation Interval	Specifies the frequency, in seconds, of predictions produced by the pacing algorithm. Changes take effect after server restart.
Media	Indicates the interaction type (for example, **chat**). This value cannot be changed.

Interaction Queues	A list of the channels and queues used for tracking engagements. **Note:** • You must have at least one queue in this list. • This list can only contain interaction queues. Virtual queues are not supported.

When you are finished making changes, click **Save**. You can also click **Cancel** to roll any changes back to their original settings.

Next Steps

- Return to the Genesys Web Engagement Features page.

Chat Channel

When you install Genesys Web Engagement, the Provisioning Tool automatically configures the Web Engagement Cluster and Chat Server to support a chat channel for routing chat interactions.

If you need to, you can configure this manually by completing the "Configuring the Web Engagement Cluster and Chat Server to Support a Chat Channel" procedure.

> **Important**
> The Genesys Web Engagement chat channel is legacy and deprecated.

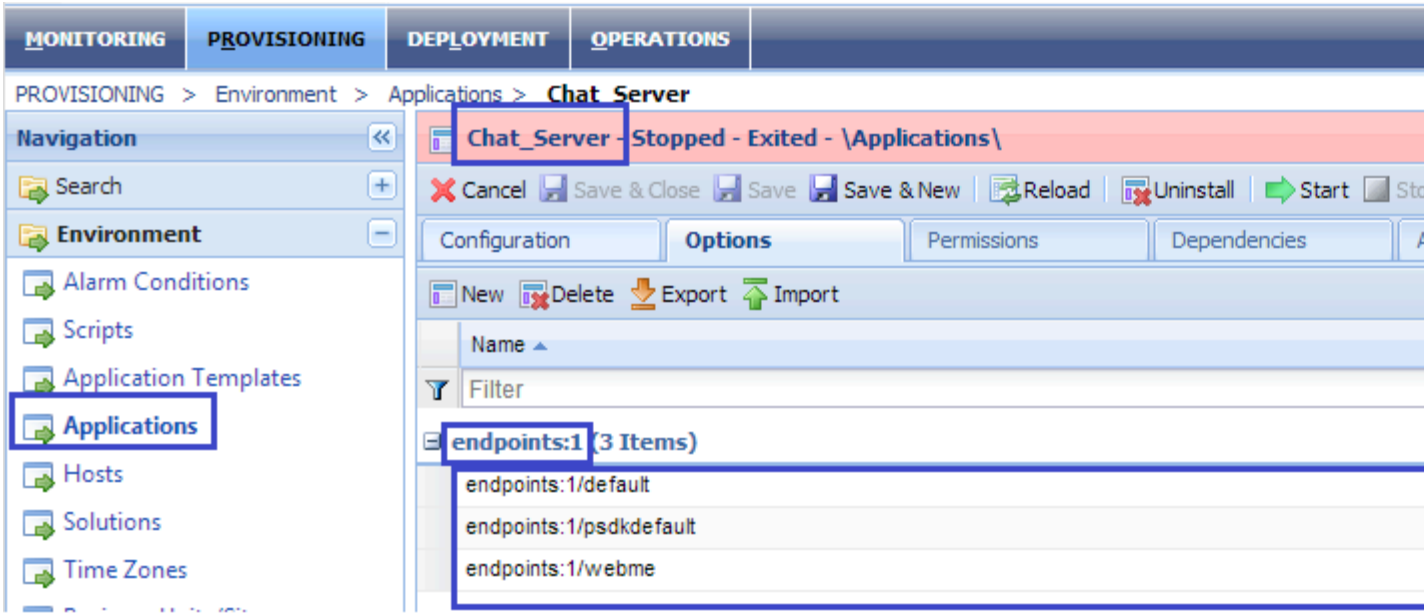
Configuring the Web Engagement Cluster and Chat Server to Support a Chat Channel

Prerequisites

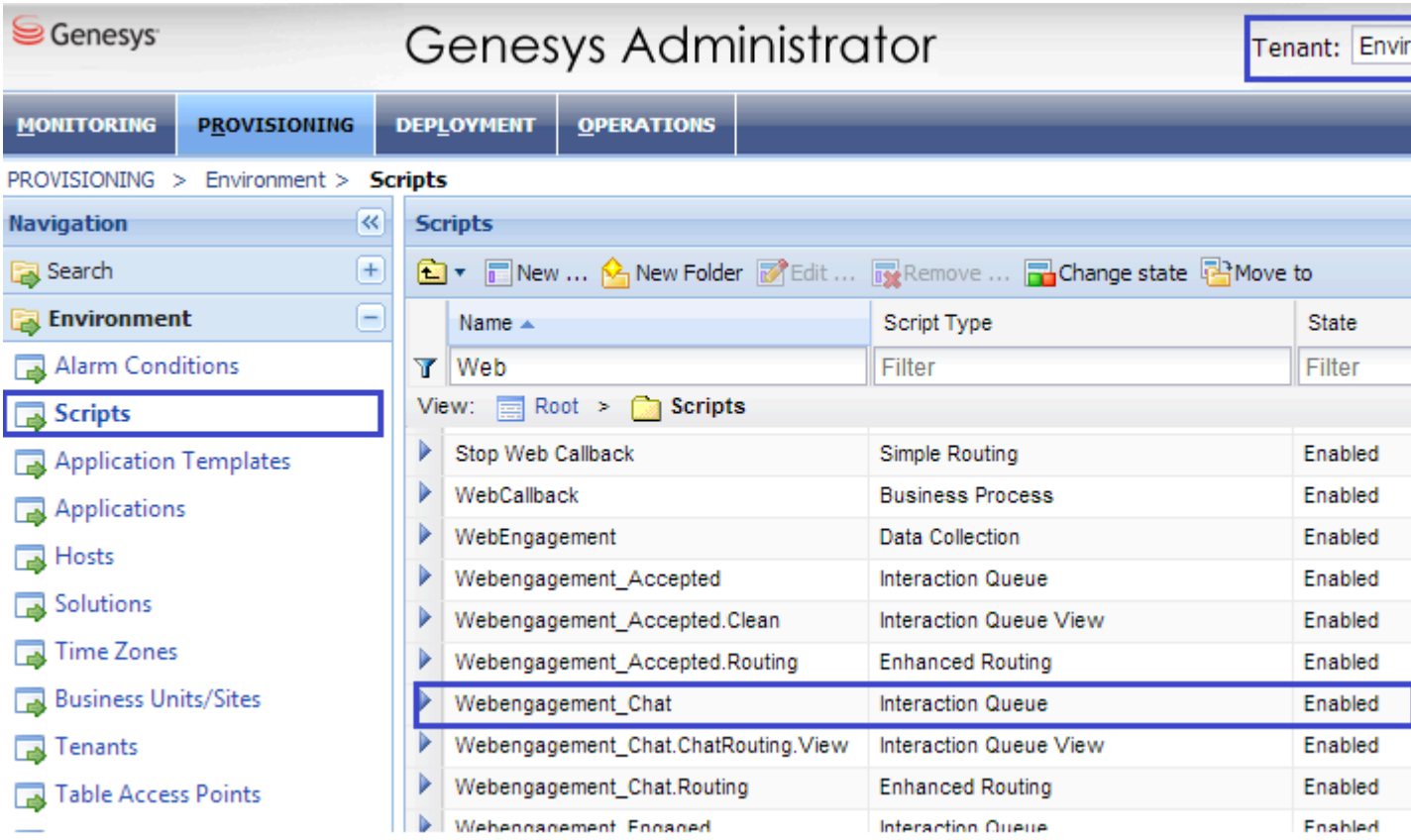
- On your Web Engagement Cluster application, you have a connection to one of following:
  - Chat Server — See Configuring the Cluster Application for details.
  - A cluster of Chat Servers — See Configuring a Connection to a Cluster of Chat Servers (Optional) for details.

Start

1. In Genesys Administrator, open your Chat Server application - either the one you connected to directly on the Web Engagement Cluster, or the Chat Server on your Application Cluster (you must complete the following steps for each Chat Server application on your Application Cluster).
2. Select the **Options** tab and find the endpoints section for your tenant: **[endpoints:*tenant ID*]**. For example, if Chat Server works with the Environment tenant, there should be a section called **[endpoints:1]**.
3. Set the endpoint value for the **endpoints:*tenant ID*/webme** option to the name of the Interaction Queue where the chat interaction should be placed.
   Note: Each Interaction Queue can be related to one routing strategy, either Orchestration Server or Universal Routing Server.



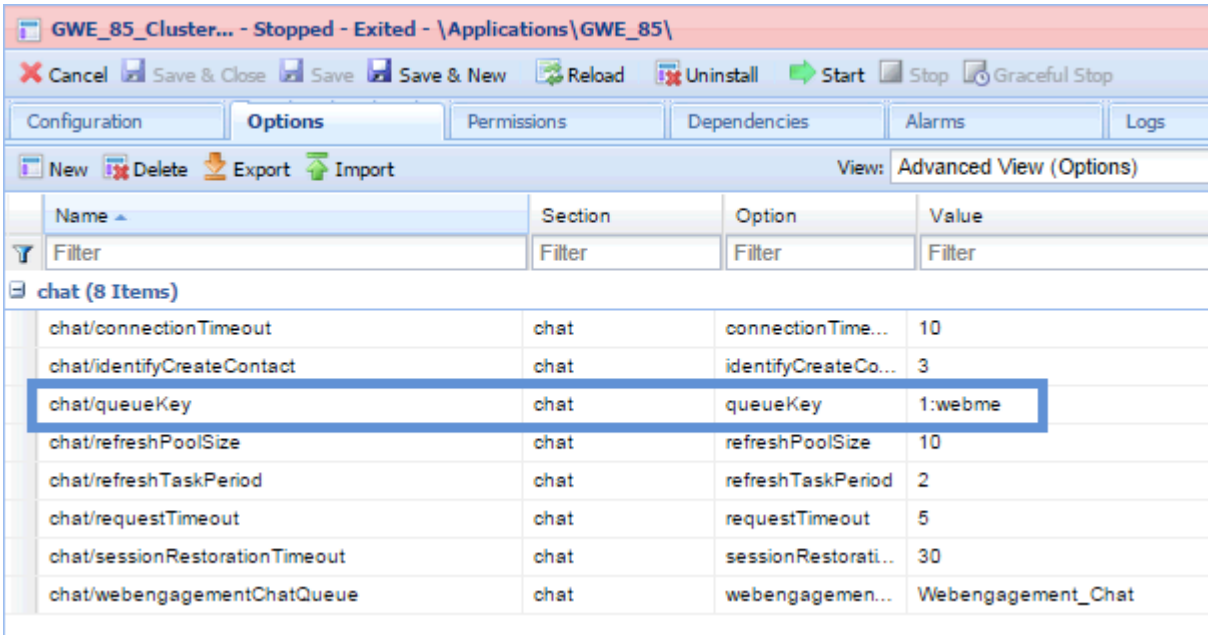
The **webme** option is set to Webengagement_Chat



The Webengagement_Chat Interaction Queue

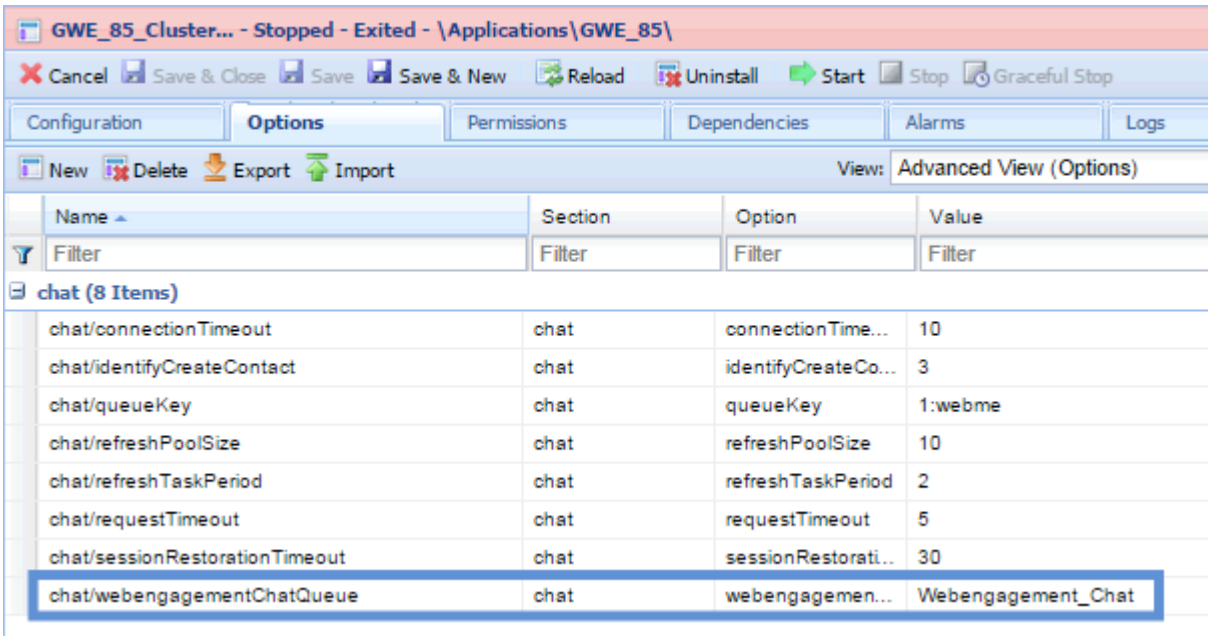
4. Configure the Chat Server endpoint for the Web Engagement Web Engagement Cluster application by opening the Web Engagement Cluster application and select the **Options** tab. In the [chat] section, set the value of the queueKey option to the name of the endpoint you specified in the Chat Server

application option in Step 3. The format is *tenant ID:endpoint name*.



The **queueKey** option is set to 1:webme

5. Specify the Interaction Queue that is used as a starting point to route chat interactions. In the [chat] section, set the value of the webengagementChatQueue option to the same queue you specified for the Chat Server endpoint in Step 3.

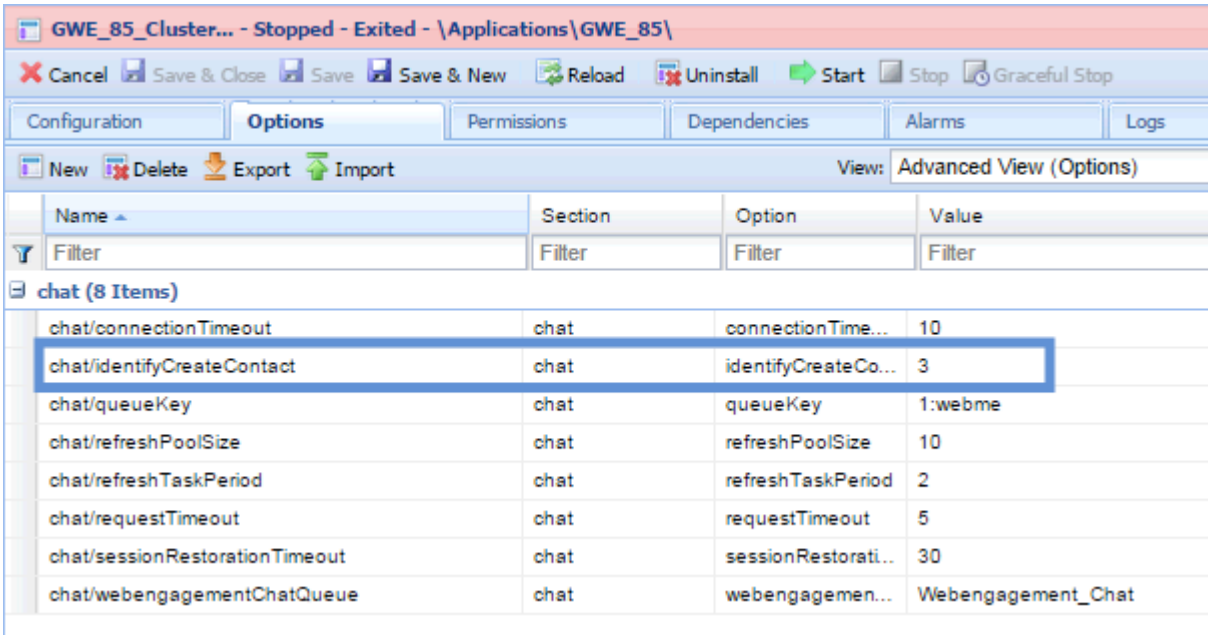


The **webengagementChatQueue** option is set to Webengagement_Chat

6. Configure how contact management will behave when a chat session is instantiated. In the [chat] section, set the identifyCreateContact option to one of the following values:

   - **1** — Do not identify and do not create a contact

- **2** — Identify, but do not create a contact
- **3** — Identify and create a contact (if absent).

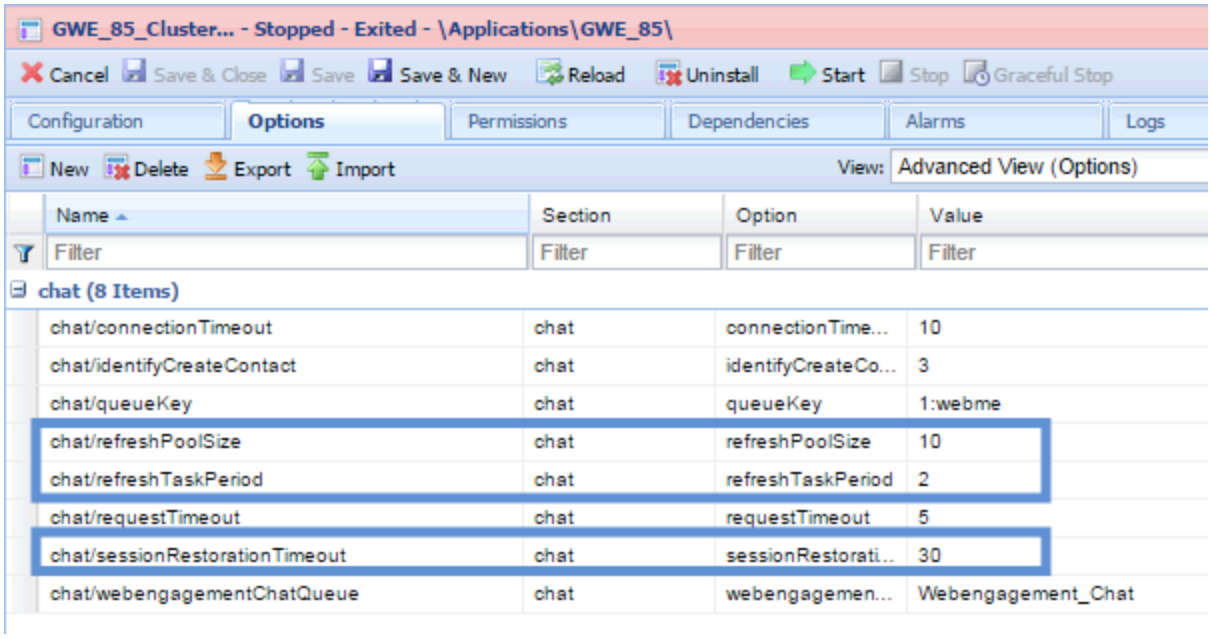


The **identifyCreateContact** option is set to 3

The default value (3) is applied if the option is absent or specified incorrectly.

Note: Your Chat Server must have a connection to Universal Contact Server in order to control contact management through the chat session.

7. Configure the chat session behavior by setting the following three options in the [chat] section:
   - refreshTaskPeriod — Specifies the frequency (in seconds) of chat session updates in the chat widget. The allowed range is from 1 to 5 seconds.
   - refreshPoolSize — Specifies the count of threads that serve the communication between the chat widgets and Chat Server(s)
   - sessionRestorationTimeout — Specifies the timeout (in seconds) during which Genesys Web Engagement tries to restore a broken chat session if the Chat Server becomes unavailable.



Chat-related options in the chat section

End

GeoIP Information

Turning on Geolocation

Web Engagement Server can collect information about a visitor's IP address, as well as converting this information into the visitor's geolocation coordinates. This capability is turned off by default.

To convert IP addresses into geolocation coordinates, Web Engagement Server uses a library and a GeoDB file (**GeoLite2-City.mmdb**) provided by MaxMind.

Web Engagement installs **GeoLite2-City.mmdb** automatically into ***Web Engagement installation dir*\server\etc**. However, you may need to upgrade it to the most recent version. To do that, follow these steps:

1. Download the **GeoLite2-City.mmdb** file from MaxMind.
2. Place the downloaded file into ***Web Engagement installation dir*\server\etc**.
3. Restart Genesys Web Engagement

Before you start your Web Engagement Server, make sure that the following options from the [privacy] section have the appropriate values:

- **collectActualIPs** = `true`
- **collectForwardedIPs** = `true`

Verifying Geolocation Coordinates

There are two ways to determine whether your geolocation coordinates are being calculated properly:

- Use the Advanced Reporting Dashboard and view the results in the map widget
- Inspect the Cassandra database and find one or more **VisitStarted** events

If the transformation is working, the **data** object of your **VisitStarted** events will contain one of the following fields, depending on the value of the **geoMode** option: **location**, **city**, or **countryCode**.

Monitoring Web Engagement Server

Web Engagement provides access to **metrics** and other key performance indicators (KPIs).

It also gives you the ability to configure Message Server **alarms** when a KPI passes its threshold value.

Web Engagement Metrics

Web Engagement 8.5.1 integrates with the third-party Metrics Java library to keep track of several Web Engagement metrics. The Metrics toolkit includes counters, timers, histograms, and gauges.

You will probably want to use Java Management Extensions (JMX) as your main way of reporting on these metrics. We show how to do that here. Or you may want to check out some of the other tools that are available.

You can also use REST—which is helpful for performance testing—or write your metrics to a log file or to the console.

Available Metrics

Web Engagement Server generates the following kinds of metrics:

- Cache usage statistics
- Event processing time statistics
- A flag for incorrect load balancing
- A flag for interactions that are running too long

Metric name	Description
<CacheName>.CacheSize	Returns the approximate number of entries in this cache
<CacheName>.EvictionCount	Returns the number of times an entry has been evicted
<CacheName>.HitCount	Returns the number of times that cache lookup methods have returned a cached value
<CacheName>.LoadExceptionCount	Returns the number of times that cache lookup methods have thrown an exception while loading a new value
<CacheName>.LoadSuccessCount	Returns the number of times that cache lookup methods have successfully loaded a new value

Metric name	Description
<CacheName>.MissCount	Returns the number of times that cache lookup methods have returned an uncached (newly loaded) value
<CacheName>.TotalLoadTime	Returns the total number of nanoseconds the cache has spent loading new values
<CacheName>.isFull	Returns `true` if the cache is full; otherwise returns `false`
monitoring.event.timer	Provide event-processing statistics
lb.routing.correct	Returns `true` if a sticky cookie has a valid application name; otherwise returns `false`
interaction.duration.exceeded	Returns `true` if a media interaction has been processed for too long; otherwise returns `false`

The following cache names are available:

- DroolsSessionCache
- EventsCache
- IxnProfileCache
- VisitProfileCache

For example, the cache size for the events cache will be available as a metric called **EventsCache.CacheSize**.

Web Engagement Alarms

Web Engagement lets you use tools from the Genesys Management Layer for monitoring and controlling your applications. These tools can be an important factor in improving performance—especially alarms, which let you set performance thresholds for these key metrics:

- Event duration
- Incorrect load balancer routing
- Interaction duration
- **VisitProfile** cache size
- **IxnProfile** cache size
- **Events** cache size
- **DroolsSession** cache size
- Garbage collection latency
- Heap memory usage

Alarm Configuration

Alarm name	Alarm description	Alarm Condition object					Related configuration option
Threshold type	**Selection mode**	**Application type**	**Detect Event ID**	**Cancel Event ID**			
Event Duration	Indicates that the time spent processing the event was greater than the specified limit.	predefined	Select by Application Type	Web Engagement Backend Server	100601	100602	EventDuration.threshold (**metrics** section)
Incorrect load balancer routing	Indicates that a request from a specific client has come to a node that is different from the nodes to which previously served requests from that client were routed.				100401	100402	N/A
Interaction duration	Indicates that a "short-lived" media interaction was detected. This may mean there is malicious traffic that can corrupt the results of the pacing algorithms.				100501	100502	interactionMinProcessingTime (**pacing** section)
VisitProfile cache size	Indicates that the **VisitProfile**				100305	100306	N/A

Alarm name	Alarm description	Alarm Condition object					Related configuration option
	object cache is full. This usually means that the server needs more memory						
IxnProfile cache size	Indicates that the **IxnProfile** object cache is full. This usually means that the server needs more memory				100303	100304	N/A
Events cache size	Indicates that the **Event** object cache is full. This usually means that the server needs more memory				100301	100302	N/A
DroolsSession cache size	Indicates that the **Drools** session object cache is full. This usually means that the server needs more memory				100201	100202	N/A
Heap Memory Usage	Defines the heap memory usage threshold value. This is the ratio of used heap memory to				10001	10002	HeapMemoryUsage.threshold (**metrics** section)

Alarm name	Alarm description	Alarm Condition object					Related configuration option
	maximum heap memory.						
GC Latency	Defines the garbage collection latency threshold value, in milliseconds, in relation to the last time the garbage was collected within the configured time interval.				10005	10006	GcLatency.threshold (**metrics** section)

Alarm Actions

Alarm name	Detect alarm message example	Problem description	Solution	Cancel alarm message
Event Duration	[ERROR] Average event duration is too long %s	The average event processing time is too long.	• Resolve performance issues • Correct the solution configuration	[INFO] Average event duration is back to normal
Incorrect load balancer routing	[ERROR] Incorrect routing for request %s	Invalid sticky cookie in request	• Correct invalid StickyCookieFilter configuration • Correct invalid Load Balancer configuration	[INFO] Routing is back to normal
Interaction duration	[ERROR] Interaction duration %s is too short	The media interaction processing time is too short. Potential DDoS attack, due to an excessively large number of chat requests.	Resolve network security issues	[INFO] Interaction duration is back to normal
VisitProfile cache size	[ERROR] VisitProfile Cache is full	The cache is not large enough to effectively process the actual number of visit profiles.	• Increase cache size • Resolve performance issues	[INFO] Cache size is back to normal
IxnProfile cache size	[ERROR] IxnProfile Cache is full	The cache is not large enough to effectively process the actual number of engagement profiles.	• Increase cache size • Resolve performance issues	[INFO] Cache size is back to normal
Events cache size	[ERROR] Events Cache is full	The cache is not large enough to effectively process the actual number of events.	• Increase cache size • Resolve performance issues	[INFO] Cache size is back to normal

Alarm name	Detect alarm message example	Problem description	Solution	Cancel alarm message
DroolsSession cache size	[ERROR] DroolsSession Cache is full	The cache is not large enough to effectively process the actual number of drools sessions.	• Increase cache size • Resolve performance issues	[INFO] Cache size is back to normal

View Metrics with JMX

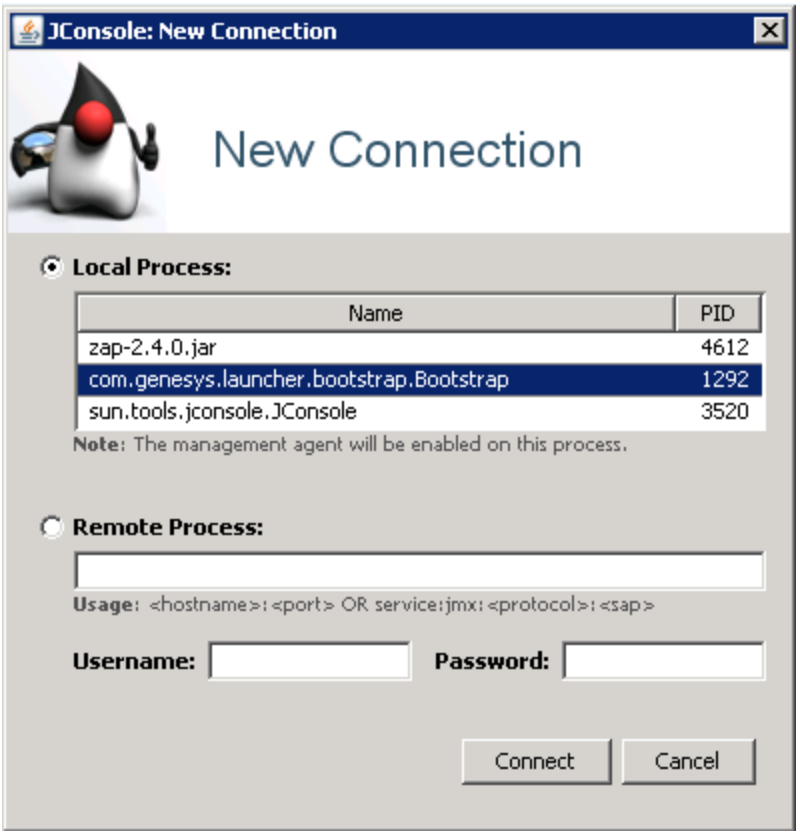
You can use JConsole to view metrics provided by your Web Engagement Server. To do this, you can start Web Engagement Server as a:

- Local java process
- Server on a remote host
- Windows service

Once you have connected, you can view your metrics in a JConsole JMX panel.

You may also want to look into some of the other tools that are available for viewing your Web Engagement metrics.

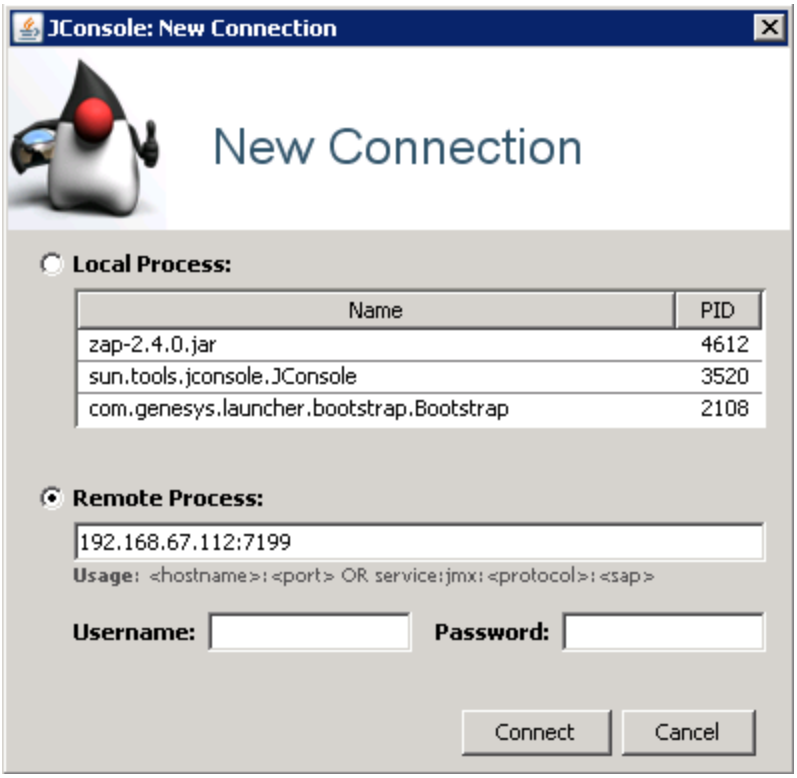
Connect to Web Engagement started as a **local java process**.



1. Run **jconsole.exe** from the ***jdk*/bin** directory.
2. In the **New Connection** dialog, specify the Web Engagement launcher java process.

   If the Web Engagement Server was started via a BAT file in the same host where the JMX console is opened, this launcher process is the **com.genesys.launcher.bootstrap.Bootstrap** process from the **Local Process** list.

Connect to Web Engagement Server started on a **remote host**.

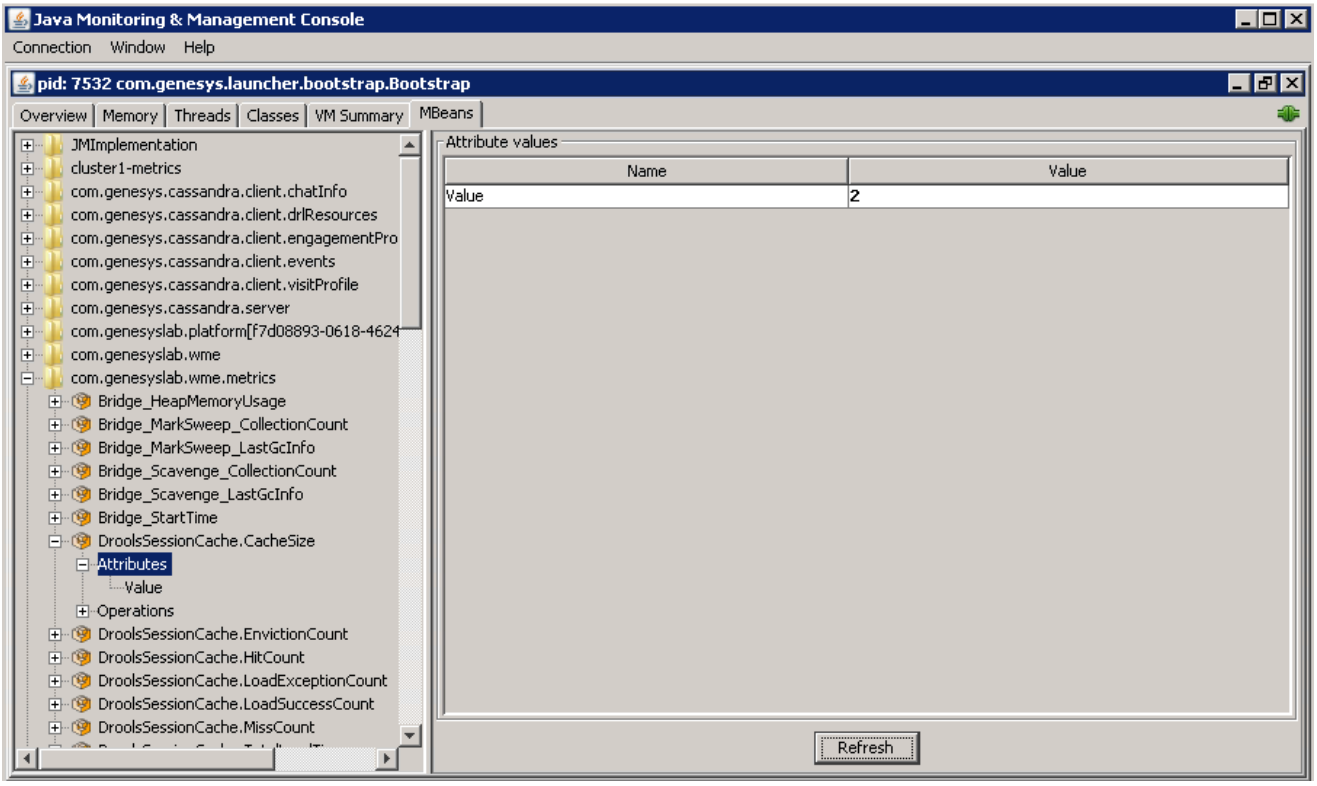


If the Web Engagement Server was started remotely as a server, follow these steps:

1. Run **jconsole.exe** from the ***jdk*/bin** directory.
2. Open the **launcher.ini** file and uncomment all of the lines that appear under the following:

   ```
   ; uncomment and configure the properties below to use JMX
   ```
3. Save your changes.
4. Restart the Web Engagement Server application.
5. Specify *host*:*JMX port* in the **Remote Process** section, as shown in the screenshot on the left.

Open the JMX panel to view the metrics.



1. Click **Connect** in the **New Connection** dialog. The JMX panel opens.
2. Open the **MBeans** tab and expand **com.genesyslab.gemc.metrics**. All of the Web Engagement metrics are there.
3. To refresh the metrics, click **Refresh**.

Other Tools

We have just explained how to use the JConsole tool bundled with Oracle Java (TM) to view your metrics, but there are several other tools you can use to do this:

- The EJTools JMX Browser
- Panoptes
- jManage
- MC4J
- Zabbix

Load Balancing

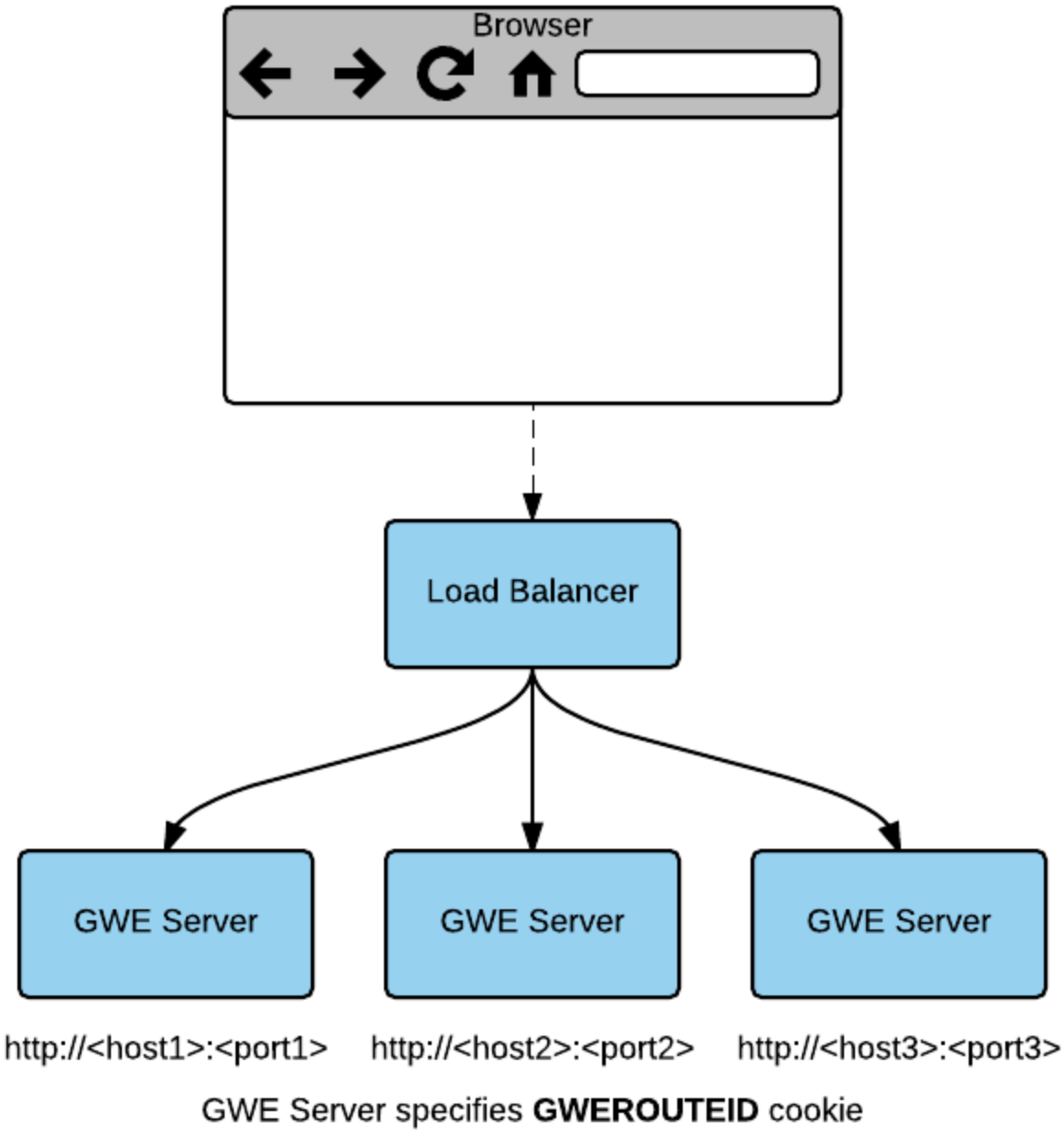
Genesys Web Engagement supports any third-party load balancer as long as the load balancing features include cookie support.

The following points are important for you to consider when setting up load balancing:

- Due to Safari's strict cookie policy, Genesys recommends that your load balancer is hosted under the same domain as the website (or its subdomain). Otherwise, chat "stickiness" cookies might be rejected as "third-party", and the solution will not work (users won't be able to start chat).
- Apache does not support WebSockets load balancing by default. If you want to enable this option, you must use the mod_proxy_wstunnel module. **Note:** This module requires Apache version 2.4+ and is only available for Linux.
- If your load balancer does not support WebSockets, make sure that you disable WebSockets on the client side. See Chat Application - disableWebSockets and Tracker Application - disableWebSockets for details. You can also control the usage of WebSockets in CometD on the server side. See the transports option for details.

Architecture

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



Sample of Deployment for Load Balancing

In the above example, the load balancer implements sticky sessions to GWE Servers based on the **GWEROUTEID** cookie. The GWE Server is responsible for specifying this cookie.

In the Web Engagement 8.5 architecture, the load balancer can be associated with the Web Engagement cluster application. Usually, this means that the host and the default port specified in the cluster application should correspond to the host and port of the load balancer. For more information about the cluster application see the section on Creating the Cluster Application in Installing the Genesys Web Engagement Server.

Sticky Sessions

The load balancer implements sticky sessions to GWE Servers based on the **GWEROUTID** cookie specified by the GWE Server instance. Therefore, it must support the following feature:

- Cookie-based stickiness to enable engagement.

> **Important**
> The GWEROUTEID cookies are created by the Genesys Web Engagement servers.

GWE requires sticky sessions not only for performance reasons, but also to switch over ongoing transactions if a node fails.

Cookie Name	Cookie Value
GWEROUTEID	"." + *<GWE Server application name in Genesys Administrator>*

Sample Configurations

Genesys provides sample load balancing configurations for two common load balancers: Apache and Nginx. For details, select a tab below:

Apache

The following procedures provide a sample load balancing configuration for Apache. Before you begin, make sure you have completed the following prerequisites:

- You already deployed your Web Engagement application into a production (or production-like) environment and have at least two nodes configured to work in the cluster (see Installing the Genesys Web Engagement Server for details).
- For this configuration example, you installed and configured an instance of Apache, version 2.2.

Configuring the Apache Load Balancer

Start

1. Confirm that the following modules are present in your Apache load balancer:
   - mod_proxy.so
   - mod_proxy_balancer.so

- mod_proxy_connect.so
- mod_proxy_http.so
- mod_headers.so

2. Edit the **./conf/httpd.conf** file and confirm that the modules are loaded:
   - LoadModule proxy_module modules/mod_proxy.so
   - LoadModule proxy_module modules/mod_proxy_balancer.so
   - LoadModule proxy_module modules/mod_proxy_connect.so
   - LoadModule proxy_module modules/mod_proxy_http.so
   - LoadModule proxy_module modules/mod_headers.so
3. Add the following configuration script to the end of the **httpd.conf** file:

```
<VirtualHost *:80>
ProxyRequests Off
<Proxy *>
order allow,deny
Allow from All
</Proxy>
ProxyPass /server http://<GWE_cluster_app_host_IP_or_FQDN>:<GWE_cluster_app_port>/server
ProxyPassReverse /server
http://<GWE_cluster_app_host_IP_or_FQDN>:<GWE_cluster_app_port>/server
</VirtualHost>
Listen <GWE_cluster_app_port>
<VirtualHost *:<GWE_cluster_app_port>>
<Proxy balancer://cluster>
   #BalancerMember route parameter is the same as name of GWE Server node application in
the Genesys Configuration Layer, for example GWE_Node_1
   BalancerMember http://<GWE_node1_app_host_IP_or_FQDN>:<GWE_node1_app_port>/server
route=GWE_Node_1
   BalancerMember http://<GWE_nodeN_app_host_IP_or_FQDN>:<GWE_nodeM_app_port>/server
route=GWE_Node_N
ProxySet stickysession=GWEROUTEID
</Proxy>
ProxyPass /server balancer://cluster
<Location /balancer-manager>
SetHandler balancer-manager
Order Deny,Allow
Allow from all
</Location>
</VirtualHost>
```

4. Save your changes. The load balancer is now configured.
5. Your cluster is healthy if the load balancer receives a successful response on requests to
   - http(s)://*Web Engagement Server Host*:*Web Engagement Server Port (secured port for https)*/server/about

   or

   - http(s)://*Web Engagement Server Host*:*Web Engagement Server Port (secured port for https)*/server/isAlive

End

Nginx

The following procedures provide a sample load balancing configuration for Nginx. Before you begin, make sure you have completed the following prerequisites:

- You already deployed your Web Engagement application into a production (or production-like) environment and have at least two nodes configured to work in the cluster (see Installing the Genesys Web Engagement Server for details).
- For this configuration sample, you installed and configured an instance of Nginx.

To configure your Nginx load balancer, edit the **./conf/nginx.conf** file and modify the configuration according to the samples provided below. For details about the configuration, consult the Nginx documentation.

Configuration Sample: Nginx Load Balancer

```
events {
            worker_connections 1024;
}
http {
        include mime.types;
         default_type application/octet-stream;
        map_hash_bucket_size 64;

        log_format main 'balancing_cookie: $cookie_GWEROUTEID --> $remote_addr - $remote_user
[$time_local] "$request" ';

        access_log logs/nginx_access.log main;
        error_log logs/nginx_error.log debug;

# Select node on top of existing cookie GWEROUTEID
map $cookie_GWEROUTEID $http_sticky {
        .GWE_Node_1 192.168.1.1:9081; # GWE_Node_1 is running on 192.168.1.1:9081
        .GWE_Node_2 192.168.1.2:9081; # GWE_Node_2 is running on 192.168.1.2:9081
 }

# Select node (round-robin) if cookie GQWEROUTEID is absent
upstream http_cluster {
        server 192.168.1.1:9081 fail_timeout=30s; # GWE_Node_1 is running on 192.168.1.1:9081
        server 192.168.1.2:9081 fail_timeout=30s; # GWE_Node_2 is running on 192.168.1.2:9081
 }

 map $http_upgrade $connection_upgrade {
        default upgrade;
        '' close;
 }
 server {
        listen <GWE_cluster_app_port>;

        location @fallback {
               proxy_pass http://http_cluster;
        }

        location /server {
               # Allow websockets, see http://nginx.org/en/docs/http/websocket.html
```

```
                proxy_http_version 1.1;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header Upgrade $http_upgrade;
                proxy_set_header Connection $connection_upgrade;
                # Increase buffer sizes to find room for DOM and CSS messages
                proxy_buffers 8 2m;
                proxy_buffer_size 10m;
                proxy_busy_buffers_size 10m;
                proxy_connect_timeout 5s;
                # Fall back if server responds incorrectly
                error_page 502 = @fallback;
                # or if doesn't respond at all.
                error_page 504 = @fallback;
                # Create a map of choices
                # see https://gist.github.com/jrom/1760790
                if ($scheme = ‘http’) {
                    set $ftest HTTP;
                 }
                if ($http_sticky) {
                    #echo 'HTTP-STICKY scheme';
                    set $ftest "${ftest}-STICKY";
                }
                if ($ftest = HTTP-STICKY) {
                    #echo 'Pass to stickyness ';
                    proxy_pass http://$http_sticky$uri?$args;
                    break;
                }
                if ($ftest = HTTP) {
                    proxy_pass http://http_cluster;
                    break;
                }
                return 500 "Misconfiguration";
        }
    }
}
```

Configuration Options

Web Engagement configuration options are maintained in the Genesys Configuration Options guide. You can use this chapter as a quick reference for options that are available for Web Engagement Server.

Note: Configuration options fall into three categories:

- Those options which should be specified in the Web Engagement Cluster application only (cluster-only options). It is critical to guarantee that these options are the same for all cluster nodes.
- Those options which should only be specified in Web Engagement Server applications (node-only options). These options are tightly coupled with a particular instance of Web Engagement Server. For example, the name of the log file that will be produced by a particular node.
- Those options which can be specified either in the Web Engagement Cluster or in Web Engagement Server applications. If one of these options is specified in both cluster and server applications, then the option for the particular server will be used. The verbosity level is a good example of this kind of option, as it is specified in the Cluster application, but can be redefined in a particular node for troubleshooting purposes.

You can configure the behavior of Genesys Web Engagement by using the configuration options listed below.

cassandraKeyspace Section

Click an option to view its properties:

dataCompression

name

readConsistencyLevel

replicationStrategy

replicationStrategyParams

retention.entity.all

retention.entity.<object>

writeConsistencyLevel

privacy Section

Click an option to view its properties:

collectActualIPs

collectForwardedIPs

geoMode

pathToGeoDB

cep Section

Settings for Complex Event Processing (CEP).

Click an option to view its properties:

domainSeparation

chat Section

Click an option to view its properties:

connectionTimeout

refreshPoolSize

queueKey

webengagementChatQueue

identifyCreateContact

refreshTaskPeriod

requestTimeout

sessionRestorationTimeout

cluster-dispatchers Section

Click an option to view its properties.

startup-reporting

cometd Section

This section includes options which correspond to the native CometD options described at https://docs.cometd.org/current/reference/#_java_server_configuration_bayeux.

Click an option to view its properties:

heartBeatTimeout

interval

maxInterval

timeout

transports

engagement Section

Click an option to view its properties:

engagementExpirationTime

registrationFormExpirationTime

strictEngagementMode

visitExpirationTime

esArea Section

Click an option to view its properties:

name

metrics Section

Click an option to view its properties:

reporter.jmx.enabled

reporter.log.enabled

reporter.log.logFrequency

reporter.messageServer.enabled

reporter.messageServer.logFrequency

reporter.console.enabled

reporter.console.logFrequency

HeapMemoryUsage.threshold

GcFrequency.threshold

GcLatency.threshold

EventDuration.threshold

monitoring.event.timer.slidingWindowSize

pacing Section

Click an option to view its properties:

algorithm

chatGroups

interactionMinProcessingTime

optimizationGoal

optimizationTarget

proactiveRatio

refreshPeriod

queues Section

Click an option to view its properties:

queueAccepted

queueEngaged

queueFailed

queueMissed

queueQualified

queueRejected

queueTimeout

userData Section

Click an option to view its properties:

keysToPropagate

eventType.ACTIONABLE

eventName.SignIn

eventName.UserInfo

eventName.VisitStarted

log Section

Click an option to view its properties:

all

standard

trace

debug

message-format

outputPattern

suppress-data

verbose

segment

expire

affectedLoggers

time_format

time_convert

security Section

Click an option to view its properties:

auth-scheme

user-id

password

web Section

These settings are used for configuring web access to GWE Server.

Click an option to view its properties:

cors.allowedHeaders

cors.allowedMethods

cors.allowedOrigins

cors.urlMapping

jsonp.whiteList

staticResourcesCacheControl

staticResourcesCacheControlPattern